

В. В. Козлов, А. І. Сбітнєв,
А. Ю. Пашковська, Т. В. Томашевська

# ІНФОРМАТИКА

**НАЦІОНАЛЬНА АКАДЕМІЯ СТАТИСТИКИ,
ОБЛІКУ ТА АУДИТУ**

**В. В. Козлов, А. І. Сбітнєв,
А. Ю. Пашковська, Т. В. Томашевська**

# ІНФОРМАТИКА

**Навчальний посібник**

**Київ 2016**

УДК 004(075.8)
ББК 32.81я7
А 59

Рецензенти:
*доктор технічних наук, професор Ю. В. Кравченко* (Державний університет телекомунікацій, завідувач кафедри комп'ютерних систем і мереж);
*кандидат технічних наук, старший науковий співробітник М. І. Кузнецов* (Національний університет державної податкової служби України, доцент кафедри інтелектуальних управляючих та обчислювальних систем факультету інформаційних технологій та менеджменту).

*Схвалено Вченою радою Національної академії статистики, обліку та аудиту (протокол № 10 від 26.05.2015 р.).*

**Інформатика** : навч. посіб. / В. В. Козлов, А. І. Сбітнєв, А. Ю. Пашковська., Т. В. Томашевська; Нац. акад. статистики, обліку та аудиту. – Київ : ДП "Інформ.-аналіт. агентство", 2016. – 727 с.

ISBN 978-617-571-131-6

Навчальний посібник складається з чотирьох розділів. У першому розділі розкривається поняття "структура інформатики". Підкреслюється пріоритетність розвитку інформаційного суспільства країни. Велику увагу приділено системам числення і способам кодування інформації. Описуються апаратні та програмні засоби реалізації інформаційних технологій, інформаційні системи та їх класифікація. Другий розділ присвячений особливостям функціонування комп'ютерних мереж. Детально викладено питання організації глобальної мережі Інтернет і її основних сервісів. Також у цьому розділі висвітлено питання захисту інформації та відомостей, що становлять державну таємницю, складові елементи інформаційної безпеки. У третьому розділі розкриваються основні прийоми в роботі з текстовим процесором, табличним процесором. Розглядаються питання, що стосуються алгоритмізації та мов програмування, Web-дизайну. Четвертий розділ присвячений сучасним інформаційним технологіям, а саме вивченню баз даних, їх різновидів і систем управління базами даних. Опрацьовано питання щодо використання штучного інтелекту в сучасних інформаційних системах, подано структуру, приклади застосування експертних систем.

Навчальній посібник буде корисним для студентів різних спеціальностей, які вивчають курс "Інформатика", "Інформатика і комп'ютерна техніка", "Інформаційні системи та технології", "Економічна інформатика", "Інформаційни технології в економіці" тощо.

ISBN 978-617-571-131-6

УДК 004(075.8)
ББК 32.81я7

©© В. В. Козлов, А. І. Сбітнєв, А. Ю. Пашковська, Т. В. Томашевська, 2016
©© Національна академія статистики, обліку та аудиту (НАСОА), 2016

# Зміст

Перелік умовних позначень, символів, одиниць, скорочень і термінів .....9
**Вступ** .....11

**РОЗДІЛ 1.**
**ОСНОВНІ ПОНЯТТЯ ІНФОРМАЦІЙНИХ СИСТЕМ**
**І ТЕХНОЛОГІЙ В ЕКОНОМІЦІ** .....14

1.1 Сутність інформатики як галузі людської діяльності .....14
1.1.1 Предмет і задачі інформатики .....14
1.1.2 Уявлення про інформаційне суспільство .....19
1.2 Основні відомості про комп'ютер .....24
1.2.1 Історія розвитку обчислювальної техніки .....24
1.2.1.1 Еволюція засобів обчислювальної техніки .....24
1.2.1.2 Покоління сучасних комп'ютерів .....29
1.2.2 Загальні відомості про персональний комп'ютер .....42
1.2.2.1 Склад персонального комп'ютера .....42
1.2.2.2 Загальні відомості про пристрої, що входять до складу персонального комп'ютера .....45
1.2.2.3 Пристрої введення-виведення інформації .....57
1.3 Основні поняття економічної інформатики .....78
1.3.1 Поняття економічної інформації, її особливості .....78
1.3.2 Класифікація й кодування економічної інформації .....86
1.3.2.1 Система класифікації .....86
1.3.2.2 Система кодування .....93
1.3.3 Класифікація економічної інформації за різними ознаками .....96
1.4 Арифметичні основи цифрових ЕОМ .....99
1.4.1 Загальні відомості про системи числення .....99
1 4.2 Системи кодування інформації в ЕОМ .....102
1.4.2.1 Кодування символьних (текстових) даних .....103
1.4.2.2 Кодування графічних даних .....111
1.4.2.3 Кодування звукової та відео інформації .....115
1.4.3 Традиційні системи числення .....117
1.4.4 Нетрадиційні системи числення .....122
1.4.5 Способи переведення чисел з однієї системи числення в іншу .....123
1.4.6 Одиниці вимірювання даних .....130
1.4.7 Форми подання числових даних в комп'ютер .....131
1.4.7.1 Природна форма подання чисел .....132
1.4.7.2 Нормальна форма подання чисел .....135
1.5 Сучасні інформаційні системи та технології .....138
1.5.1 Поняття інформаційної технології .....138
1.5.2 Класифікація інформаційних технологій .....140
1.5.3 Інформаційні процеси і системи .....144

1.5.4 Класифікація ІС ....146
1.5.4.1 Класифікація ІС за функціональною ознакою ....150
1.5.4.2 Класифікація ІС за характером обробки інформації ....153
1.5.5 Нова інформаційна технологія ....154
1.5.6 Складові частини інформаційних технологій ....157
1.5.6.1 Інструментарій інформаційної технології і його види. Переваги, що забезпечує комп'ютерна технологія ....159
1.5.6.2 Приклади інформаційних технологій в економічній діяльності ....160
1.6 Поняття та призначення програмного забезпечення ....164
1.6.1 Поняття та призначення програмного забезпечення, класифікація ...164
1.6.2 Загальні основи операційних систем ....167
1.6.2.1 Призначення і функції ОС ....167
1.6.2.2 Класифікація ОС ....181
1.6.2.3 Операційні системи Windows ....183
1.6.3 Мережні операційні системи ....188
1.6.3.1 Мережні й розподілені ОС ....189
1.6.3.2 Функціональні компоненти мережної ОС ....190
1.6.4 Вимоги до ОС ....196
1.7 Організація файлової системи. Захист від збоїв і несанкціонованого доступу ....197
1.7.1 Логічна організація файлової системи ....197
1.7.2 Фізична організація файлової системи ....208
КОНТРОЛЬНІ ПИТАННЯ ....230
ТЕСТИ ДЛЯ САМОКОНТРОЛЮ ....233

**РОЗДІЛ 2. МЕРЕЖНІ ТЕХНОЛОГІЇ ТА ОРГАНІЗАЦІЯ КОМП'ЮТЕРНОЇ БЕЗПЕКИ, ЗАХИСТУ ІНФОРМАЦІЇ В ЕКОНОМІЦІ** ....236

2.1 Основні поняття комп'ютерних мереж ....236
2.1.1 Поняття та призначення комп'ютерної мережі ....236
2.1.2 Класифікація комп'ютерних мереж ....239
2.1.3 Архітектура комп'ютерних мереж ....241
2.1.4 Топологія обчислювальних мереж ....242
2.2 Основи мереж передачі даних ....245
2.2.1 Способи комутації. Класифікація способів комутації даних ....245
2.2.2 Протоколи мережі. Базова еталонна модель взаємозв'язку відкритих систем ....247
2.2.2.1 Протоколи мережі ....247
2.2.2.2 Еталонна модель взаємозв'язку відкритих систем ....248
2.2.3 Апаратне забезпечення комп'ютерних мереж ....254
2.2.3.1 Комп'ютери ....255
2.2.3.2 Комунікаційне встаткування ....256
2.2.4 Програмне забезпечення мережі ....260

2.2.5 Пропозиції зі створення та організації роботи в локальних комп'ютерних мережах ....262
2.3 Загальні відомості та характеристика мережі Internet ....265
2.3.1 Історія створення Інтернету ....265
2.3.2 Структура Internet. Способи доступу до Internet ....272
2.3.2.1 Структура Internet ....272
2.3.2.2 Способи доступу до Internet ....274
2.3.3 Системи адресації в internet. Поняття IP-адреси та доменної адреси, протоколу DNS ....276
2.4 Основні послуги Internet ....282
2.4.1 Електронна пошта ....282
2.4.2 Мережеві новини Usenet ....283
2.4.3 Протокол передачі файлів (FTP) ....285
2.4.4 Мурашка (Archie) ....285
2.4.5 Telnet ....286
2.4.6 Гіпертекстова система Gopher ....287
2.4.7 WAIS – інформаційна система широкого користування ....287
2.4.8 IRC (Internet Relay Chat) – "Базікання по Internet" ....287
2.4.9 "Всесвітня павутина" (World Wide Web) ....287
2.5 Введення в захист інформації. Джерела, класифікація погроз і форми атак на інформацію ....290
2.5.1 Захист комп'ютерної інформації: основні поняття й визначення ....291
2.5.2 Класифікація погроз безпеки інформації ....297
2.5.3 Форми атак на об'єкти інформаційних систем ....304
2.5.3.1 Форми атак ....304
2.5.3.2 Програмні закладки ....308
2.5.3.3 Моделі впливу програмних закладок на комп'ютери ....310
2.5.3.4 Троянські програми ....313
2.5.3.5 Клавіатурні шпигуни ....314
2.5.4 Аналіз погроз і каналів просочування інформації ....319
2.6 Підходи до створення комплексної системи захисту інформації ....322
2.6.1 Підходи до створення комплексної системи захисту інформації ....322
2.6.1.1 Структура системи захисту інформації ....322
2.6.1.2 Політика інформаційної безпеки ....324
2.6.2 Методи й засоби захисту інформації в інформаційно-обчислювальних мережах ....326
2.6.2.1 Класифікація методів і способів захисту інформації ....326
2.6.2.2 Комп'ютерна інформація – об'єкт правовідносин ....327
2.6.3 Основні відомості про криптографію. Підсистема криптографічного захисту ....331
2.6.3.1 Основні поняття ....331
2.6.3.2 З історії криптографії ....333
2.6.3.3 Підсистема криптографічного захисту ....337
2.6.3.4 Класифікація криптографічних методів ....337

2.7 Комп'ютерні віруси та антивірусні програми ....343
2.7.1 Віруси. Види й класифікації ....343
2.7.2 Проблеми антивірусного захисту інформації ....360
2.7.3 Загальна характеристика антивірусних програм для захисту робочої станції ....363
2.8 Захист від збоїв і несанкціонованого доступу в ОС ....378
2.8.1 Базові технології безпеки ОС ....378
2.8.2 Ядро безпеки ОС ....382
2.8.3 Механізм ідентифікації й аутентифікації в ОС Windows NT ....384
2.8.4 Механізми захисту в ОС UNIX ....386
КОНТРОЛЬНІ ПИТАННЯ ....391
ТЕСТИ ДЛЯ САМОКОНТРОЛЮ ....392

**РОЗДІЛ 3.**
**ПРОГРАМНІ ЗАСОБИ РОБОТИ ЗІ СТРУКТУРОВАНИМИ ДОКУМЕНТАМИ. ОСНОВИ ОФІСНОГО ПРОГРАМУВАННЯ ТА WEB-ДИЗАЙНУ** ....396

3.1 Основи web-дизайну ....396
3.1.1. Основні питання web-дизайну ....396
3.1.1.1 Визначення web-дизайну ....396
3.1.1.2 Введення в web-стандарти ....400
3.1.1.3 Характеристика сайтів ....406
3.1.2 Типи й архітектури сайтів ....411
3.1.2.1 Типи сайтів ....413
3.1.2.2 Структура сайту ....423
3.1.3 Базові Web-технології ....440
3.1.3.1 Web-браузери ....440
3.1.3.2 Комп'ютерні методи пошуку ....443
3.1.3.3 Мови розмітки ....447
3.1.3.4 Технології програмування ....447
3.2 Основи офісного програмування ....448
3.2.1 Основні поняття алгоритмізації та програмування ....448
3.2.1.1 Поняття та основні властивості алгоритмів ....448
3.2.1.2 Блок-схеми та основні алгоритмічні конструкції ....451
3.2.1.3 Основні етапи проектування програми ....460
3.2.2 Розвиток мов програмування ....460
3.2.2.1 Мови програмування ....460
3.2.2.2 Сучасні системи програмування ....466
3.2.2.3 Архітектура програмних систем ....469
3.2.3 Автоматизація завдань в інтегрованих середовищах ....471
3.2.3.1 Мова макросів – засіб автоматизації роботи з додатками ....471
3.2.3.2 Мова Visual Basic for Applications (VBA) ....475
3.3 Програмні засоби роботи зі структурованими документами ....483
3.3.1 Призначення й класифікація текстових редакторів ....484
3.3.2 Базові можливості текстових редакторів ....485

3.3.2.1 Типова структура інтерфейсу ....485
3.3.2.2 Керування курсором ....487
3.3.2.3 Перегляд тексту ....487
3.3.2.4 Режими роботи текстового процесора "вставка" й "заміна" .. 488
3.3.2.5 Богатовіконий режим роботи ....488
3.3.3 Підготовка текстового документа ....488
3.3.3.1 Набір тексту ....490
3.3.3.2 Редагування тексту ....491
3.3.3.3 Форматування тексту ....494
3.3.3.4 Друк тексту ....496
3.3.3.5 Ведення архіву текстів ....497
3.3.4 Робота з текстом ....497
3.3.4.1 Мінімальний набір типових операцій ....497
3.3.4.2 Розширений набір типових операцій ....499
3.3.5 Призначення й області застосування табличних процесорів ....507
3.3.5.1 Поняття та призначення процесора електронних таблиць Microsoft Excel ....508
3.3.5.2 Історія й тенденції розвитку ....510
3.3.6 Основні поняття. Типова структура інтерфейсу ....511
3.3.6.1 Основні поняття ....511
3.3.6.2 Типова структура інтерфейсу ....513
3.3.6.3 Інтерфейс MS Excel ....514
3.3.6.4 Дані, збережені в осередках електронної таблиці. Типи вхідних даних ....516
КОНТРОЛЬНІ ПИТАННЯ ....521
ТЕСТИ ДЛЯ САМОКОНТРОЛЮ ....522

**РОЗДІЛ 4.**
**БАЗИ ДАНИХ, ПРЕДСТАВЛЕННЯ ЗНАНЬ ТА ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ** ....525

4.1 Програмні засоби роботи з базами та сховищами даних ....525
4.1.1 Основні поняття теорії систем управління базами даних. Реляційні бази ....525
4.1.1.1 Поняття бази даних, об'єкта, класу об'єктів, атрибутів, доменів, кортежів ....525
4.1.1.2 Поняття моделі даних, типи моделей даних ....533
4.1.1.3 Характеристики баз даних ....536
4.1.1.4 Реляційні бази даних. Поняття та вимоги до реляційної бази даних. Типи зв'язків у базах даних, їх характеристика ....542
4.1.2 Системи управління базами даних. Основні поняття СУБД ....554
4.1.2.1 Типи СУБД їх коротка характеристика ....554
4.1.2.2 Призначення СУБД Microsoft Access та її характеристика. Об'єкти СУБД Microsoft Access ....562
4.1.3 Етапи проектування бази даних ....570

4.1.3.1 Мета та підходи до проектування бази даних ....570
4.1.3.2 Етапи проектування баз даних ....571
4.1.3.3 Поняття предметної області й технологія її аналізу ....574
4.1.3.4 Логічне проектування ....582
4.1.4 Поняття сховищ даних та передумови їх створення ....586
4.1.4.1 Поняття, передумови створення та основні характеристики сховищ даних ....586
4.1.4.2 Основні компоненти сховища даних, архітектура сховищ даних ....588
4.1.4.3 Архітектура сховищ даних ....590
4.1.4.4 Основні елементи сховища даних та підходи до проектування сховищ ....592
4.2 Експертні і навчальні системи ....596
4.2.1 Штучний інтелект – вершина розвитку інформаційних технологій ..596
4.2.1.1 Поняття "штучний інтелект", терміні й визначення ....596
4.2.1.2 Сучасні напрями досліджень у області штучного інтелекту ..599
4.2.1.3 Категорія знання, класифікація. Проблема представлення знань ....606
4.2.2 Моделювання знань про наочні області як основа інтелектуальних автоматизованих систем ....612
4.2.2.1 Моделі представлення знань ....612
4.2.2.2 Логічна модель для представлення знань ....612
4.2.2.3 Продукційна модель для представлення знань ....617
4.2.2.4 Фрейми для представлення знань ....620
4.2.2.5 Семантичні мережі для представлення знань ....624
4.2.2.6 Нові моделі представлення знань ....630
4.2.3 Характеристика й призначення інформаційної технології експертних систем ....640
4.2.3.1 Поняття та призначення експертних систем ....640
4.2.3.2 Характеристики та базові функції експертних систем ....644
4.2.3.3 Класифікація експертних систем ....650
4.2.3.4. Структура експертної системи ....657
4.2.3.5 Приклади реалізації експертних систем ....662
4.3 Перспективи розвитку інформаційних технологій ....666
4.3.1 Сучасний стан і напрямки розвитку IT ....666
4.3.1.1 Введення в нові інформаційні технології ....669
4.3.1.2 Нові технології проектування і аналізу систем, тенденції розвитку ....670
КОНТРОЛЬНІ ПИТАННЯ ....694
ТЕСТИ ДЛЯ САМОКОНТРОЛЮ ....695

**ДОДАТКИ** ....699
Додаток А Логічні основи цифрових ЕОМ ....699
Додаток Б Основні вбудовані статистичні функції MS Excel ....717
Додаток В Список фінансових функцій Excel 2003 ....720
Список джерел ....724

## Перелік умовних позначень, символів, одиниць, скорочень і термінів

АВ – антивірус;
АВПЗ – антивірусне програмне забезпечення;
АІС – автоматизована інформаційна система;
АІТ – автоматизована інформаційна технологія;
АРМ – автоматизоване робоче місце;
АС – автоматизована система;
АСУ – автоматизована система управління;
БД – база даних;
БЗ – база знань;
ВІС – великі інтегральні схеми;
ЕІС – економічна інформаційна система;
ЕМВВС – еталонна модель взаємозв'язку відкритих систем;
ЕОМ – електронно-обчислювальна машина;
ЕПТ – електронно-променева трубка;
ЕС – експертні системи;
ЕЦП – електронний цифровий підпис;
ЄАСУ – єдина автоматизована система управління;
ЗІ – захист інформації;
ІЕ – інформаційний елемент;
ІР – інформаційний ресурс;
ІС – інформаційна система;
ІТ – інформаційна технологія;
КВІ – канал витоку інформації;
КСА – комплекси засобів автоматизації;
КСАЗ – комп'ютерні системи антивірусного захисту;
КСЗІ – комплексна система захисту інформації;
КУД – крайове устаткування даних;
ЛОМ – локальна обчислювальна мережа;
МА – мережевий адаптер;
МВР – мова високого рівня;
МД – модель даних;
ММД – мова маніпулювання даними;
МОД – мова опису даних;
МП – материнська плата;

МПЗ – мережеве програмне забезпечення;
НВІС – невеликі інтегральні схеми;
НДР – науково-дослідна робота;
НІТ – нова інформаційна технологія;
НМ – нейронні мережі;
НСД – несанкціонований доступ;
ОЗП – оперативно запам'ятовуючий пристрій;
ОП – оперативна пам'ять;
ОПР – особа, яка приймає рішення;
ОС – операційна система;
ОСМ – операційна система мережі;
ОУ – орган управління;
ПЕОМ – персональна електронно-обчислювальна машина;
ПЗ – програмне забезпечення;
ПЗП – постійно запам'ятовуючий пристрій;
ПІБ – політика інформаційної безпеки;
ПК – персональний комп'ютер;
ППП – пакет прикладних програм;
ПрО – предметна область;
РБД – розподілена база даних;
РК – рідкокристалічний монітор;
РС – робоча станція;
САЗ – системи антивірусного захисту;
СД – сховище даних;
СЕ – сенсорний екран;
СЗІ – система захисту інформації;
СІТ – сучасні інформаційні технології;
СОК – система обчислення останніх класів;
СППР – система підтримки прийняття рішень;
СУБД – система управління базою даних;
СУБЗ – система управління базами знань;
СУРБД – система управління розподіленою базою даних;
ФС – файлова система;
ЦП – центральний процесор;
ШІ – штучний інтелект;
ЯД – ядро даних.

# ВСТУП

Історію цивілізації можна представити як безперервний процес розвитку стосунків, пов'язаних зі створенням, розподілом і споживанням ресурсів. Характер використовуваних ресурсів упродовж віків неодноразово змінювався.

Обмін ресурсами є однією з основ громадського розвитку точно так, як і в основі руху матерії лежить енергетичний обмін, а в основі розвитку живої природи – процеси обміну речовин.

Традиційною для XX століття була характеристика суспільства розвинених держав як індустріального. Вона безпосередньо пов'язана з визначальною роллю промислових і енергетичних ресурсів у матеріальній складовій громадських процесів. Вступаючи в XXI століття, людство відкриває новий етап розвитку – інформаційний, для якого характерним є домінуюча роль інформаційних ресурсів.

Інформація – це не лише сукупність відомостей, фактів або даних, як це представлялося раніше (у тому числі і в законодавчій базі). Інформація – це продукт динамічної взаємодії об'єктивних і суб'єктивних даних. Такий підхід лежить в основі адекватного розуміння суті інформаційного ресурсообміну. Це не просто обмін відомостями у формі їх публікації (оголошення, мовлення). Це ще і сучасні засоби зв'язку, і засоби обчислювальної техніки, і програмні засоби інформаційних технологій.

Інформатика в XXI столітті стає природною наукою, що займає положення між іншими природничими, технічними і громадськими науками. Її предмет складають інформаційні процеси, що протікають у природі, суспільстві та технічних системах. Її методи у своїй більшості засновані на взаємодії програмних і апаратних засобів обчислювальної техніки з іншими технічними системами, з людиною і суспільством. Її мета – наукове обгрунтування ефективних прийомів створення, розподілу і споживання усіх типів інформаційних ресурсів і методологічне забезпечення розробки нових інформаційних систем. Її центральна роль полягає в наданні свого апарату і понятійної бази іншим природничим, громадським і технічним дисциплінам.

*Інформаційна сфера і рівень розвитку інформаційних технологій стають вирішальними показниками розвиненості держави.*

Як сьогодні очевидно, що масова комп'ютеризація, розвиток і впровадження новітніх інформаційних технологій привели до вражаючого ривка провідних країн світу в сферах освіти, наукових досліджень, економіки,

соціального життя. З'явився новий вид ресурсу – інформаційний, характерною особливістю якого є внутрішній взаємозв'язок. Завдання розподілу інформаційних ресурсів набули нового вигляду.

Розглянуті вище особливості, безпосередньо пов'язані з питаннями ухвалення рішень в організаціях в умовах дефіциту часу на їх прийняття, з оптимальним використанням усіх ресурсів, у першу чергу інформаційних. Виникає проблема оптимального використання мереж АРМ, розподілу між ними ряду оперативно вирішуваних завдань.

Особливо складним і проблемним питанням залишається інформатизація інтелектуальної, творчої діяльності посадовців органів управління, відповідальних за прийняття рішень.

Кінцевим завданням інформатизації управлінської діяльності є інформаційно-аналітичне забезпечення завдань управління, яке розглядається як складова інформаційного забезпечення і підтримується інформаційно-аналітичними системами. Такі автоматизовані системи здійснюють на основі обробки даних і знань алгоритмічне розвязання різноманітних завдань щодо синтезу нових відомостей, які в них не присутні у явному вигляді.

Сучасна інформатика – це одна з фундаментальних галузей наукового знання. Вона вивчає інформацію та інформаційні взаємодії в техніці, природі, суспільстві, має сферу застосування, що постійно розширюється.

Пропонований навчальний посібник охоплює всі розділи відповідної дисципліни, яка викладена у стандартах підготовки фахівців вищої освіти економіко-управлінського напряму.

Мета навчального посібника:

– дати цілісне уявлення про інформатику та її роль у розвитку суспільства;
– розкрити суть і можливості технічних та програмних засобів інформатики;
– сформувати розуміння того, з якою метою та яким чином можна використовувати інформаційні системи і технології.

Викладений матеріал направлений на формування знань про принципи побудови та функціонування обчислювальних машин, організації обчислювальних процесів на персональних комп'ютерах та їх алгоритмізації, програмне забезпечення персональних комп'ютерів і комп'ютерних мереж, а також ефективне застосування сучасних інформаційно-комунікаційних технологій у професійній діяльності.

Навчальний посібник складається з чотирьох розділів.

У першому розділі висвітлені загальні поняття інформатики, інформаційних технологій, інформаційних систем, описуються арифметичні основи побудови сучасних комп'ютерів.

У другому розділі розглядаються базові принципи функціонування

комп'ютерних мереж, розкриваються питання про види комп'ютерних мереж, про особливості передачі сигналу різними каналами сполуки, розглядаються основні топології локальних мереж, описуються принципи організації глобальної мережі Інтернет. Крім того, у цьому розділі викладені питання захисту інформації, зокрема методи і засоби, а також складові елементи інформаційної безпеки, подано класифікацію та характеристику комп'ютерних вірусів.

У третьому розділі розкриваються основні прийоми роботи з офісними додатками для створення складних структурних документів. Окремо розглядаються поняття "алгоритм", мови програмування, застосування офісного програмування в професійній діяльності. Викладені сучасні підходи до Web-дизайну, описуються принципи організації Web-сайтів.

Четвертий розділ висвітлює інформаційні технології баз та сховищ даних. Наводяться поняття "бази даних", способи організації моделей даних, етапи проектування баз даних. Розглядаються питання щодо використання штучного інтелекту в сучасних інформаційних системах, подається структура, приклади застосування експертних систем.

Сподіваємося, що пропонований підручник допоможе Вам освоїти одну з найбільш передових областей людської діяльності та стати інформаційно грамотним фахівцем прийдешнього століття.

Прохання всі зауваження щодо змісту підручника та можливих помилок у ньому направляти на електрону адресу авторів: IsiT_15@ukr.net

# Розділ 1.
# ОСНОВНІ ПОНЯТТЯ ІНФОРМАЦІЙНИХ СИСТЕМ І ТЕХНОЛОГІЙ В ЕКОНОМІЦІ

## 1.1 Сутність інформатики як галузі людської діяльності

**26 листопада – Всесвітній день інформації**
Проводиться за ініціативою Міжнародної академії інформатизації (МАІ), що має генеральний консультативний статус в Економічній і Соціальній радах ООН.

### 1.1.1 Предмет і задачі інформатики

Інформатика – це наука, що утворилася порівняно недавно. Її розвиток пов'язаний з появою в середині XX століття електронно-обчислювальних машин, які стали універсальними засобами для зберігання, обробки й передачі інформації. Поява комп'ютерів дозволила передавати й обробляти інформацію зі швидкістю, що у мільйони разів перевищує швидкість обробки й передачі інформації людиною.

Поняття "інформатика" походить від французького слова "informatique", утвореного в результаті об'єднання термінів "information" (інформація) і "automatique" (автоматика), що виражає її суть як науки про автоматичну обробку інформації. Крім Франції, термін "інформатика" використовується в ряді країн Східної Європи. Воточас у більшості країн Західної Європи й США застосовується інший термін – Computer Science (наука про засоби обчислювальної техніки).

Джерелами інформатики звичайно називають дві науки – документалістику й кібернетику. Документалістика сформувалася наприкінці XI століття у зв'язку з бурхливим розвитком виробничих відносин. Її розквіт прийшовся на 30-і роки XX століття, а основним предметом стало вивчення раціональних засобів і методів підвищення ефективності документообігу.

Основи, близької до інформатики технічної науки – кібернетики, були закладені в працях з математичної логіки американським математиком Норбертом Вінером, опублікованими в 1948 році, а сама назва, походить від грецького слова "kyberneticos" (вправний у керуванні).

Уперше термін кібернетика ввів французький фізик Андре Марі Ампер у першій половині XI столітті. Він займався розробкою єдиної системи класифікації всіх наук і позначив цим терміном гіпотетичну науку

про керування, якої у той час не існувало, але яка, на його думку, повинна була існувати.

Сьогодні предметом кібернетики є принципи побудови й функціонування систем автоматичного керування, а основними завданнями – методи моделювання процесу прийняття рішень технічними засобами, зв'язок між психологією людини й математичною логікою, зв'язок між інформаційним процесом окремого індивідуума й інформаційних процесів у суспільстві, розробка принципів і методів штучного інтелекту. На практиці кібернетика, в багатьох випадках, опирається на ті ж програмні й апаратні засоби обчислювальної техніки, що й інформатика, а інформатика, у свою чергу, запозичила у кібернетики математичну й логічну базу для розвитку цих засобів.

Центральною категорією кібернетики є **інформація** – будь-який вид відомостей про предмети, факти, поняття предметної області, невідомих до їхнього одержання і що є об'єктом зберігання, передачі й обробки.

**Визначення інформації**

Існує безліч визначень інформації. Так, один з основоположників сучасної теорії інформації, Норберт Вінер, визначав інформацію так: "Інформація є інформація, а не матерія або енергія".

Згідно з імовірнісно-статистичним (або ентропійним) підходом, інформація – зменшення ступеня невизначеності знання про якийсь об'єкт, систему, процес або явище чи зміна невизначеності стану самого об'єкта, системи, явища, процесу.

Загальнонаукове (філософське) трактування: інформація – зміна обсягу та структури знань сприймаючої системи.

Нормативно-правове трактування: інформація – відомості про осіб, предмети, факти, події і процеси незалежно від способу подання цих відомостей.

Базове визначення інформації. ***Інформація*** – зміна обсягу і структури знання про деяку предметну галузь (осіб, предмети, факти, події, явища, процеси) сприймаючою системою (людина, організаційна структура, автоматизована інформаційна система) незалежно від форми і способу подання знання.

Під **інформацією** в кібернетиці розуміють будь-яку сукупність сигналів, впливів або відомостей, що деякою системою сприймаються від навколишнього середовища (вхідна інформація X), видаються в навколишнє середовище (вихідна інформація Y), а також зберігаються в системі (внутрішня, внутрішньосистемна інформація Z).

Галуззю науки, що вивчає структуру й загальні властивості наукової інформації, а також питання, пов'язані з її збиранням, зберіганням, пошуком, переробкою, перетворенням і поширенням, є **інформатика**.




Рисунок 1.1. Поняття інформації в кібернетиці

**Інформатика** – це технічна наука, що систематизує прийоми створення, зберігання, відтворення, обробки й передачі даних засобами обчислювальної техніки, а також принципи функціонування цих засобів і методи керування ними.

Предмет інформатики становлять наступні поняття:

- апаратне забезпечення засобів обчислювальної техніки;
- програмне забезпечення засобів обчислювальної техніки;
- засоби взаємодії апаратного й програмного забезпечення;
- засоби взаємодії людини з апаратними й програмними засобами.

Інформатика не тільки "чиста наука". Важлива особливість інформатики – найширші додатки, що охоплюють майже всі види людської діяльності: виробництво, керування, науку, освіту, проектні розробки, торгівлю, фінансову сферу, медицину, криміналістику, охорону навколишнього середовища та ін.

Як наука, інформатика вивчає загальні закономірності, властиві інформаційним процесам. При розробці нових носіїв інформації, канали зв'язку, прийоми кодування, візуального відображення інформації, тощо, конкретна природа даної інформації майже не має значення. Для розробника системи керування базами даних (СУБД) важливі загальні принципи організації й ефективність пошуку даних, а не те, які конкретно дані будуть потім закладені в базу численними користувачами. Ці загальні закономірності є предметом інформатики як науки.

Інформатика як наука складається з декількох частин і кожна може розглядатися як самостійна дисципліна.

*Теоретична інформатика* – частина інформатики, що включає ряд математичних розділів. Вона опирається на математичну логіку й включає такі розділи, як теорія алгоритмів і автоматів, теорія інформації й теорія кодування, теорія формальних мов і граматик, дослідження операцій та ін. Цей розділ використовує математичні методи для загального вивчення процесів обробки інформації.

*Обчислювальна техніка* – розділ, у якому розробляються загальні принципи побудови обчислювальних систем. Це не технічні деталі й електронні схеми, а принципові рішення на рівні так званої архітектури обчислювальних систем, що визначає склад, призначення, функціональні можливості й принципи взаємодії пристроїв.

*Програмування* – діяльність, пов'язана з розробкою систем програмного забезпечення.

*Інформаційні системи* – розділ інформатики, пов'язаний з рішенням питань із аналізу потоків інформації в різних складних системах, їхньої оптимізації, структурування, про принципи зберігання й пошуку інформації. Без теоретичного обґрунтування принципових рішень в океані інформації можна просто захлинутися. Відомим прикладом рішення проблеми на глобальному рівні може служити гіпертекстова пошукова система WWW.

*Штучний інтелект* – галузь інформатики, в якій вирішуються найскладніші проблеми, що перебувають на перетині із психологією, фізіологією, лінгвістикою й іншими науками. Основні напрямки розробок, що відносяться до цієї області – моделювання міркувань, комп'ютерна лінгвістика, машинний переклад, створення експертних систем, розпізнавання образів та ін.

Об'єктом додатків інформатики є ряд наук й областей практичної діяльності, для яких вона стала безперервним джерелом найсучасніших технологій, так званих "нових інформаційних технологій". Інформатика – практична наука. Її досягнення повинні бути підтверджені практикою й прийматися в тих випадках, коли вони відповідають критерію підвищення ефективності.

Застосування методів і засобів інформатики можливо у всіх тих областях людської діяльності, в яких існує принципова можливість (і необхідність) реєстрації й обробки інформації. Нижче перераховані приклади найбільш значних областей застосування засобів обробки даних.

- Автоматизовані системи планування й керування виробництвом, починаючи з окремих підприємств і завершуючи керуванням цілими галузями (залізничний транспорт, авіація та ін.).
- Автоматизовані робочі місця (АРМ) фахівця, наприклад АРМ бухгалтера, керівника, лікаря та ін.
- Банківські й біржові комп'ютерні системи.
- Бібліографічні комп'ютерні системи.
- Підготовка різних документів, звітів та інших друкованих матеріалів, рекламна справа.
- Керування роботою окремих верстатів (верстати із числовим програмним керуванням), роботи, робото-технічні лінії, цехи й заводи-автомати.
- Обробка конкретних експериментальних даних при проведенні економічних, соціологічних, математичних, фізичних, хімічних, біологічних, історичних та інших досліджень.
- Системи автоматизованого проектування, що забезпечують підтримку інженера-конструктора, істотно підвищують продуктивність його праці і скорочують терміни розробок. Широко застосовуються при проектуванні таких виробів, як

космічних човнів “Буран”, “Шаттл”, сучасні надзвукові літаки та ін.

- Військова справа, наприклад системи протиракетної оборони, космічні системи.
- Моделювання фізичних явищ і дослідження моделей за допомогою комп’ютера, наприклад, завдання термоядерного синтезу, космогонічні моделі. Моделювання найчастіше застосовується в тих випадках, коли проведення прямого фізичного експерименту або занадто дороге, або в принципі неможливе.
- Рішення завдань метеопрогнозу.
- Одержання зображень внутрішніх частин непрозорих тіл, у тому числі в медицині – комп’ютерна томографія, на виробництві – неруйнівний контроль якості виробів.
- Аранжування музичних продуктів, світломузика.
- Скульптура й архітектура.
- Комп’ютерний дизайн приміщень, технічних пристроїв, одягу.
- Машинний переклад з різних природних мов.
- Комп’ютерна мультиплікація й анімація.
- Криптографія – шифрування документів, доступ до яких повинен бути обмежений.
- Навчальні, програми тестування й контролюючі програми.
- Цифровий аудіо- і відео запис.
- Ігрові програми.
- Нові засоби зв’язку, що базуються на локальних і глобальних мережах.

Якщо підбити підсумок, то завданнями інформатики є:

– розробка й виробництво сучасних засобів обчислювальної техніки;
– проектування й впровадження прогресивних технологій обробки інформації;
– інформатизація суспільства й підвищення рівня його інформаційної культури.

**17 травня**
**Всесвітній день інформаційного співтовариства**
17 травня 1865 р. в Парижі був заснований International Telegraph Union (з 1932 р. – International Telecommunications Union). Після того, як людство стало передавати інформацію по дротах на гігантські відстані практично миттєво змінився весь світ.
27 березня 2006 р. Генеральна Асамблея ООН ухвалила Резолюцію, в якій проголосила 17 травня Всесвітнім Днем інформаційного співтовариства.

### 1.1.2 Уявлення про інформаційне суспільство

В історії розвитку цивілізації відбулося кілька інформаційних революцій – перетворень суспільних відносин через кардинальні зміни в сфері обробки інформації. Наслідком подібних перетворень було придбання людським суспільством нової якості.

*Перша інформаційна революція* пов'язана з появою мови і членороздільної людської мови. Адже саме розвиток мови зробив колосальний вплив на розвиток свідомості людей, а його використання в їх практичній діяльності стало інформаційною основою появи перших технологій, тобто знання і навичок раціональної організації цієї діяльності.

У первісному суспільстві використовувалися і поширювалися тільки "живі знання", носіями яких були живі люди – старійшини, жерці, шамани. У цих умовах процеси накопичення і поширення знання в суспільстві здійснювалися надзвичайно повільно, а збереження вже накопичених знань було недостатньо надійним. Із смертю їх носіїв багато знань втрачалися і повинні були формуватися наново. На це йшли століття.

Ситуація докорінно змінилася, коли люди навчилися відчужувати знання і фіксувати їх на матеріальних носіях у вигляді малюнків, креслень, умовних знаків, багато хто з яких зберігся до теперішнього часу. Це і привело до другої інформаційної революції.

*Друга революція* (біля трьох тисяч років до нашої ери) пов'язана з винаходом писемності, що привело до гігантського якісного й кількісного стрибка. З'явилася можливість передачі знань від покоління до покоління.

Істотним чином змінилося і *інформаційне середовище суспільства*, стали можливими нові види інформаційних комунікацій між людьми за допомогою обміну письмовими повідомленнями. З'явилися історичні літописи, поезія і література, зародилися елементи того нового і своєрідного явища, яке ми сьогодні називаємо *інформаційною культурою*. Нового сенсу набуло і поняття освіти.

*Третя* (середина XVI ст., епоха Відродження) викликана винаходом друкарства, яке слід визнати однією з перших ефективних *інформаційних*

*технологій*, що радикально змінило індустріальне суспільство, культуру, організацію діяльності. Друкарська книга стала головним хранителем і джерелом знань.

*Четверта* (кінець XIX ст.) обумовлена винаходом електрики, завдяки якому з'явилися телеграф, телефон, радіо, які дозволяють оперативно передавати й накопичувати інформацію в будь-якому обсязі. Сьогодні вони підключені до загального інформаційного простору не лише своєї країни, але і значної частини нашої планети. Ми вже не мислимо свого існування у відриві від *світового інформаційного простору*.

*П'ята* (70-і рр. XX ст.) пов'язана з винаходом мікропроцесорної технології й появою персонального комп'ютера. На мікропроцесорах і інтегральних схемах створюються комп'ютери, комп'ютерні мережі, системи передачі даних (інформаційні комунікації). Застосування цих засобів для обробки наукової, економічній і соціальній інформації кардинальним чином змінило можливості людини по активізації і ефективному використанню *інформаційних ресурсів*.

Цей період характеризують три фундаментальні інновації:

– перехід від механічних і електричних засобів перетворення інформації до електроннихмініатюризація всіх вузлів, пристроїв, приладів, машин;
– створення програмно-керованих пристроїв і процесів (рис. 1.2, 1.3).




Рисунок 1.2. Еволюція носіїв інформації

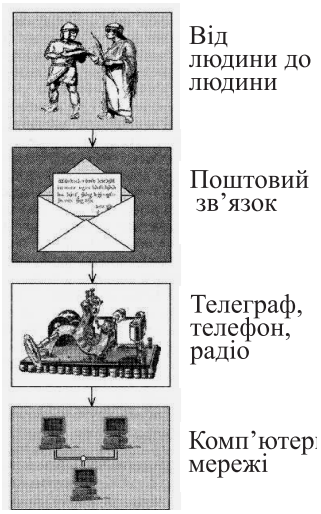



Рисунок 1.3. Еволюція засобі передачі інформації

Для отримання більше цілісного уявлення про цей період доцільно познайомитися зі зміною поколінь електронно-обчислювальних машин (ЕОМ) і зіставити ці відомості з етапами в області обробки і передачі інформації. В основу періодизації ЕОМ спочатку був покладений фізико-технологічний принцип: машину відносять до того або іншого покоління в залежності від використовуваних у ній фізичних елементів або технології їхнього виготовлення. Межі поколінь у часі розмиті, тому що одночасно випускалися машини різного рівня. Коли наводять дати, що відносяться до поколінь, то швидше за все мають на увазі період промислового виробництва; проектування велося значно раніше, а зустріти в експлуатації досить екзотичні пристрої можна і сьогодні.

За час існування змінилося п'ять поколінь комп'ютерів. Кожне покоління характеризується елементною базою та програмним забезпеченням (табл.1).

Таблиця 1.1

**Покоління електронно-обчислювальних машин**

| Показник | Покоління ЕОМ | | | | | |
|---|---|---|---|---|---|---|
| | Перше | Друге | Третє | Четверте | | П'яте теперішний час |
| | 1951 – 1954 | 1958 – 1960 | 1965 – 1966 | А 1976 – 1979 | Б 1980 – 1985 | |
| Елементна база процесора | Електронні лампи | Транзистори | Інтегральні схеми | Великі інтегральні схеми (ВІС) | Надвеликі інтегральні схеми (НВІС) | НВІС Оптоелектроніка кріоелектроніка |
| Елементна база ОЗП | Електронно-променеві трубки | Феритові сердечники | Феритові сердечники | ВІС | НВІС | НВІС |
| Максимальна місткість ОЗУ, байт | $10^2$ | $10^3$ | $10^4$ | $10^5$ | $10^7$ | $10^8$ |
| Максимальна швидкодія процесора, оп/с | $10^4$ | $10^6$ | $10^7$ | $10^8$ | $10^9$ Богато процессорність | $10^{12}$ Богато проце сорність |
| Мови програмування | Машинний код | Асемблер | Процедурні мови високого рівня (МВР) | Нові процедурні (МВР) | Непроцедурні (МВР) | Нові непроцедурні (МВР) |
| Засоби зв'язку користувача з ЕОМ | Пульт управління і перфокарти | Перфокарти і перфострічки | Алфавітно-цифровий термінал | Монохромний графічний дисплей, клавіатури | Кольоровий графічний дисплей, клавіатура, миша та ін. | Пристрої голосового зв'язку з ЕОМ |

*1-е покоління* (початок 50-х років XX століття). Елементна база – електронні лампи. ЕОМ характеризуються великими габаритами та обсягами споживання енергії, малою швидкодією, низькою надійністю, програмуванням у кодах.

*2-е покоління* (з кінця 50-х років XX століття). Елементна база – напівпровідникові елементи. Покращилися порівняно з ЕОМ попереднього покоління всі технічні характеристики. Для програмування використовуються алгоритмічні мови.

*3-е покоління* (початок 60-х років XX століття). Елементна база – інтегральні схеми, багатошаровий друкований монтаж. Різке зниження габаритів ЕОМ, підвищення їхньої надійності, збільшення продуктивності. Доступ з віддалених терміналів.

*4-е покоління* (із середини 70-х років XX століття). Елементна база – мікропроцесори, великі інтегральні схеми. Покращилися технічні характеристики. Масовий випуск персональних комп'ютерів. Напрямки розвитку: потужні багатопроцесорні обчислювальні системи з високою продуктивністю, створення дешевих мікро-ЕОМ.

*5-е покоління* (із середини 80-х рр.). Почалася розробка інтелектуальних комп'ютерів. Впровадження в усі сфери комп'ютерних мереж і їхнє об'єднання, використання розподіленої обробки даних, повсюдне застосування комп'ютерних інформаційних технологій.

Остання інформаційна революція висуває на перший план нову галузь – *інформаційну індустрію*, пов'язану з виробництвом технічних засобів, методів, технологій для виробництва нових знань. Найважливішими складовими інформаційної індустрії стають усі види інформаційних технологій, особливо телекомунікації. Сучасна інформаційна технологія опирається на досягнення в області комп'ютерної техніки й засобів зв'язку. Ускладнення індустріального виробництва, соціального, економічного й політичного життя, зміна динаміки процесів у всіх сферах діяльності людини привели, з одного боку, до росту потреб у знаннях, а з іншого боку – до створення нових засобів і способів задоволення цих потреб. Бурхливий розвиток комп'ютерної техніки й інформаційних технологій став поштовхом до розвитку суспільства, побудованого на використанні різної інформації і яке отримало назву інформаційного суспільства.

Японські вчені вважають, що в інформаційному суспільстві процес комп'ютеризації дасть людям доступ до надійних джерел інформації, позбавить їх від рутинної роботи, забезпечить високий рівень автоматизації обробки інформації у виробничій і соціальній сферах. Рушійною силою розвитку суспільства повинно стати виробництво інформаційного, а не матеріального продукту.

*Інформаційне суспільство* – суспільство, у якому більшість працюючих зайнята виробництвом, зберіганням, переробкою й реалізацією

інформації, особливо вищої її форми – знань. У реальній практиці розвитку науки й техніки передових країн наприкінці XX ст. – початку XXI ст. поступово здобуває зримі обриси створена теоретиками картина інформаційного суспільства. Прогнозується перетворення всього світового простору в єдине комп'ютеризоване й інформаційне співтовариство людей, які проживають у квартирах і котеджах, оснащених різними електронними приладами й комп'ютеризованими пристроями. У подальшому, діяльність людей буде зосереджена головним чином на обробці інформації, а матеріальне виробництво й виробництво енергії буде покладено на машини.

Пріоритетність розвитку інформаційного суспільства на найвищому рівні регламентовано з 2007 року прийняттям Закону України "Про Основні засади розвитку інформаційного суспільства в Україні на 2007-2015 роки" від 09.01.2007 року № 537-V.

1. Закон. Про Основні засади розвитку інформаційного суспільства в Україні на 2007-2015 роки: Закон України від 09.01.2007 року № 537-V: Законодавство України [Електронний ресурс] – Режим доступу: http://zakon.rada.gov.ua / cgi-bin / laws / main. cgi? nreg=537-16.

В інформаційному суспільстві, за деякими визначеннями, економіка, політика і культура залежать від створення, збереження і доступності інформації в національному і світовому масштабах. Відповідно, цінним документом, який відображає результуючі аспекти впливу інформаційного суспільства на розвиток країни, є Доктрина інформаційної безпеки України, затверджена Указом Президента України від 8.07.2009 року № 514 / 2009. Даний документ визначає основні засади інформаційної безпеки України, життєво важливі інтереси в інформаційній сфері для особи, суспільства, держави, основні реальні та потенційні загрози інформаційній безпеці України, а також напрями державної політики у сфері інформаційної безпеки України.

Нині сфери ІКТ та розвитку інформаційного суспільства регулюються також такими базовими законами України: "Про Концепцію Національної програми інформатизації"; "Про Національну програму інформатизації"; "Про електронні документи та електронний документообіг"; "Про електронний цифровий підпис"; "Про захист персональних даних".

З 10 травня 2011вступив в силу Закон України "Про доступ до публічної інформації".

2. Доктрина інформаційної безпеки України, затверджена Указом Президента України від 8.07.2009 року № 514 / 2009: Законодавство України [Електронний ресурс] – Режим доступу: http://zakon1.rada.gov.ua / cgi-bin / laws / main. cgi? nreg=514 %2F2009.

Цілком очевидним є те, що становлення інформаційного суспільства є процесом практично невідворотнім. Кількість користувачів інтернетом

нині вже перетнула межу в 2 мільярди осіб (при тому, що перший мільярд був "освоєний" лише наприкінці 2008 року). З них приблизно 1,2 млрд припадає на країни, що розвиваються, і 0,8 млрд – на розвинені країни. Світ динамічно переходить з мобільного зв'язку другого покоління (платформа 2G) на третє (відповідно – 3G): протягом 2007-2010 рр. кількість країн, що здійснили такий перехід, збільшилася з 95 до 143. Тим часом прискорюється поширення нової швидкісної платформи 4G (серед хедлайнерів цього процесу і Україна, що було навіть відмічено міжнародними організаціями). Водночас не можна не відмітити, що у сучасному розумінні "інформаційне суспільство" – це передусім гуманітарна категорія, яка описує якісні суспільні трансформації, зміщення акцентів із виробничої до невиробничої сфер, зміну характеру інформаційних потоків, групових та індивідуальних ідентичностей.

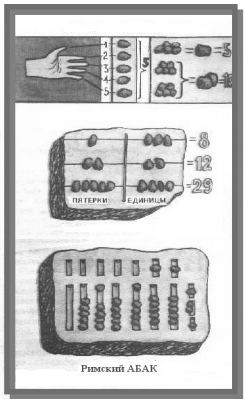
## 1.2 Основні відомості про комп'ютер

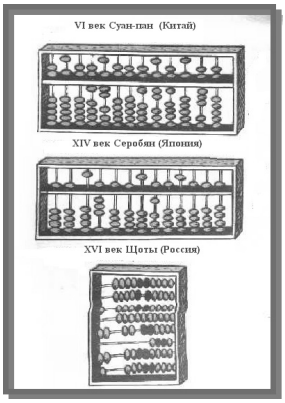
**16 листопада**
**День працівників радіо, телебачення і зв'язку України**
День працівників радіо, телебачення і зв'язку (День працівників радіо, телебачення та зв'язку) встановлений Указом Президента № 667/94 від 11 листопада 1994 року на підтримку ініціативи працівників радіо, телебачення і телекомунікаційної галузі України. Девізом працівників зв'язку і сфери інформатизації сталі слова: "Ми працюємо, щоб Україна стала розвиненою інформаційною державою".

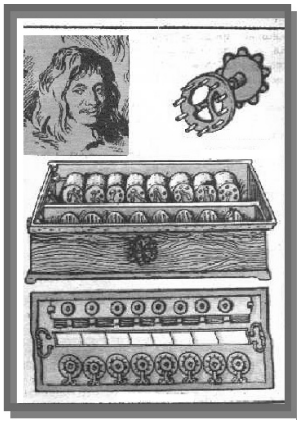
### 1.2.1 Історія розвитку обчислювальної техніки

1.2.1.1 Еволюція засобів обчислювальної техніки

Сучасним комп'ютерам передували механічні і електромеханічні пристрої. Аж до VIIX ст. абак і його модифікація – рахівниці (у китайців – *"суан-пан"*, у японців – *"серобян"*, в Росії – *"щоты"*.) – залишалися єдиним інструментом для проведення обчислень.



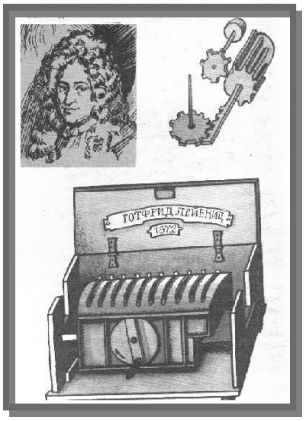





У 1642 році французький математик і філософ Блез Паскаль у віці 18 років створює першу модель обчислювальної машини *"Паскаліну"* или *"Паскалеве колесо"*.

Машина Паскаля складалася з восьми дисків, які рухалися, з прорізами і могла підсумовувати числа до восьми знаків. Для своєї машини Паскаль використовував десяткову систему числення. Наприклад, якщо перший диск зміщувався на десять прорізів, що складало його повний оберт, він переміщав наступний диск на одну позицію і, таким чином, збільшував кількість десятків на один. Коли диск, що представляє десятки, робив повний оберт, він зміщував наступний диск, збільшуючи кількість сотень, і т. д.

Відомі і більш ранні спроби створення механічних підсумовуючих машин. Опис підсумовуючої машини, що нагадує за характеристиками машину Паскаля, у 1967 році було виявлено в записках, що належать Леонардо да Вінчі. Подібний пристрій також був описаний у 1623 році Вільгельмом Шикардом. До наших днів дійшли тільки креслення Шикарда, виявлені в 1956 році.




У 1694 році німецький математик і філософ Готфрід Вільгельм Лей-бніц, використовуючи креслення і малюнки Паскаля, поліпшив машину Паскаля, додавши можливість перемножувати числа. Замість звичайних шестерінок Лейбніц використовував покроковий барабан.

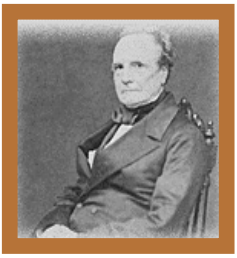
Проте широкого поширення обчислювальні апарати набули тільки в 1820 році, коли француз Чарльз Калмар винайшов машину, яка могла виконувати чотири основні арифметичні дії. Машину Калмара назвали арифмометр. Завдяки своїй універсальності, арифмометри використовувалися досить тривалий час. Багато учених і винахідників удосконалювали ці пристрої. Так, швед, що жив в Росії, Вільгодт Однер в 1880 році створив арифмометр, в якому використовувалося змінне число зубців. Пізніше на основі арифмометра Однера був створений арифмометр "Фелікс", що випускався в СРСР аж до 70-х років XX ст.

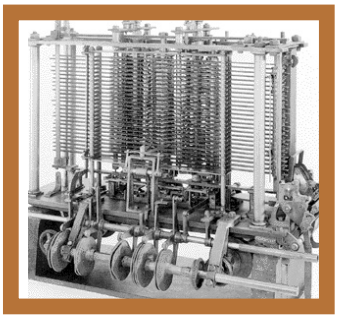
Початок ери комп'ютерів у тому вигляді, в якому вони існують зараз, пов'язаний з ім'ям англійського математика Чарльза Бебіджа, який

у 30-х роках XIX століття запропонував ідею обчислювальної машини, здійснену лише в середині XX століття. Бебідж звернув увагу на те, що машина може без помилок виконувати обчислення великих математичних таблиць за допомогою простого повторення кроків. Працюючи над цією проблемою, Бебідж у 1822 році запропонував проект машини для розв'язування диференціальних рівнянь. Для повторення операцій у машині Бебіджа повинна була використовуватися енергія пари. Таким чином, сам процес обчислень дійсно був автоматизований, тобто проходив без участі людини. Надалі Бебідж вирішив створити модель універсальної обчислювальної машини, здатної виконувати широке коло завдань. Він назвав її аналітичною машиною. В аналітичної машини Бебіджа були усі основні риси сучасного комп'ютера (для цієї машини була написана перша у світі програма. Її автор – Августа Ада Байрон (1815–1852), віконтеса де Лавлейс, дочка досить відомого поета Джорджа Гордона Байрона. Саме їй належала ідея використання перфорованих карт для програмування обчислювальних операцій (1843). Її можна з впевненістю назвати найпершим у світі програмістом. Сьогодні її ім'ям названа одна з відомих мов програмування). Аналітична машина, що складалася більш ніж з 50000 компонентів, включала пристрій введення інформації, блок управління, запам'ятовуючий пристрій і пристрій виведення результатів. Аналітична машина могла виконувати певний набір інструкцій, які записувалися на перфокартах. Перфокарти були прямокутними картками з картону. Кожній інструкції аналітичної машини відповідала певна послідовність дірочок, які пробивалися на перфокартах, а потім за допомогою пристрою введення поступали в блок управління. Бебиджу так і не вдалося закінчити свій проект, а незавершений варіант його "комп'ютера", що прозвав сучасниками "дивацтвом Беббиджа", був зданий на зберігання в музей королівського коледжу в Лондоні. Хоча аналітична машина в тому вигляді, в якому її задумав Бебідж, так і не була створена, ідеї, закладені Бебіджем, зробили величезний вплив на розвиток обчислювальної техніки. Автоматизація обчислень, універсальність обчислювальної машини, набір внутрішніх інструкцій, загальна конструктивна схема, організація введення і виведення інформації – усі ці елементи згодом були використані при створенні комп'ютера.

Повністю "Різницева машина" Ч. Бебіджа була добудована тільки у 1991 р. двома інженерами Р. Криком і Б. Холловеєм у Лондонському науковому музеї до 200-річчя з дня народження її автора. Вона складається з 4000 деталей и може обчислювати різниці 7-го порядку.

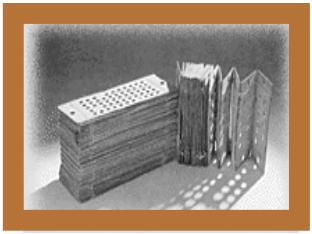
У 1889 році американський винахідник Герман Холлерит сконструював перфокартковий пристрій для розв'язування статистичних задач. На відміну від ідеї Бебіджа, зберігати на перфокартах інструкції, Холлерит використовував перфокарти для зберігання даних. Крім того, для робо-







ти перфокарткового пристрою використовувалася електрика. Цифри на перфокарті зображалися одинарними отворами, а букви алфавіту – подвійними. Спеціальний електричний прилад розпізнавав отвори на перфокартах і посилав сигнали в оброблювальний пристрій. Обчислювальна машина Холлерита виявилася на ті часи дуже швидким пристроєм обробки даних, а перфокарти – зручним способом зберігання даних.



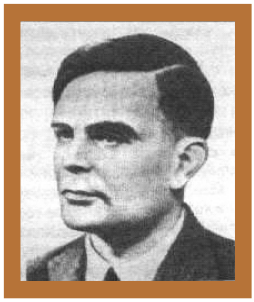
Машина Холлерита була використана для обробки результатів перепису населення США. Обробка результатів попереднього перепису 1880 року зайняла близько 10 років. За цей час встигло вирости нове покоління американців. За допомогою машини Холлерита ті ж дані були оброблені всього за шість тижнів. У 1896 році Холлерит заснував компанію по виробництву перфоруючих пристроїв – Tabulating Machine Company, яка в 1924 році після серії злиттів і поглинань перетворилася на знамениту компанію по виробництву комп'ютерів – IBM (International Business Machines).

Окрім механічних і електромеханічних обчислювальних машин з'явилися також аналогові обчислювальні машини, в яких обробка інформації відбувалася за допомогою спеціально підібраного фізичного процесу, що моделює обчислювану закономірність. Простою аналоговою обчислювальною машиною є годинник. Першими аналоговими машинами були пристрої, в яких головними елементами були інтегруючі і диференціюючі пристрої, що дозволяють швидко обчислювати інтеграл і похідну заданої функції, відстежуючи її зміну в часі.

Корисною властивістю аналогової обчислювальної машини є практично миттєве отримання рішення після задання необхідних параметрів задачі і встановлення моделюючого фізичного процесу. Проте коло задач, які може вирішувати аналогова машина, обмежений тими фізичними процесами, які вона в змозі моделювати. Крім того, точність рішення аналогової

машини часто недостатня для певного кола завдань, а підвищення точності пов'язане із значним зростанням вартості обчислень.

З іншого боку, механічні та електромеханічні обчислювальні машини, призначені для вирішення складних задач, вимагають наявності великої кількості елементів для представлення чисел і зв'язків між ними, що істотно ускладнює їхню роботу.



У 1936 році англійський математик Алан Тьюрінг опублікував роботу "Про обчислювані числа", заклавши теоретичні основи теорії алгоритмів. Концепція Т'юрінга виникла в результаті проведеного ним аналізу дій людини, що виконує відповідно до попередньо розробленого плану ті або інші обчислення, тобто послідовні перетворення знакових комплексів. Цей аналіз, у свою чергу, був здійснений ним з метою вирішення проблеми пошуку точного математичного еквіваленту для загального інтуїтивного уявлення про алгоритм. Робота Т'юрінга стимулювала виникнення абстрактної теорії автоматів і багато в чому визначила її особливості.

У своїй роботі Тьюрінг описав абстрактну обчислювальну машину, яка дістала назву машини Тьюрінга. Машина Тьюрінга є автоматичним пристроєм, здатним знаходитися в кінцевому числі внутрішніх станів і забезпеченим нескінченною зовнішньою пам'яттю – стрічкою. Серед станів виділяються два – початковий і кінцевий. Стрічка поділена на клітинки. У кожну клітинку може бути записана будь-яка з букв деякого алфавіту, у порожню клітинку – "порожня буква". У кожен момент часу машина Тьюрінга знаходиться в одному із своїх станів і, розглядаючи одну з клітинок стрічки, сприймає записаний в ній символ.

У некінцевому стані машина Тьюрінга здійснює крок, який визначається її поточним станом і символом на стрічці, що сприймається в даний момент. Крок машини Т'юрінга полягає в наступному:

1. У даній клітинці записується символ, співпадаючий із старим, або порожній.
2. Машина переходить у новий стан, що збігається зі старим, або кінцевим.
3. Стрічка зсувається на одну клітинку або залишається на місці.

Перерахування усіх можливих кроків машини Т'юрінга, залежно від поточної комбінації некінцевого стану і сприйманого символу, називається програмою машини Т'юрінга. Конфігурація машини Т'юрінга визначається конкретним заповненням клітинок стрічки символами і внутрішнім станом, в якому машина знаходиться. Якщо зафіксувати яку-небудь некінцеву конфігурацію машини в якості початкової, то робота машини полягатиме

в послідовному перетворенні початкової конфігурації відповідно до програми машини до тих пір, поки не буде досягнутий кінцевий стан.

Т'юрінг не переслідував мети винайти комп'ютер. Проте, описана ним абстрактна машина визначила деякі характеристики сучасних комп'ютерів. Так, наприклад, нескінченна стрічка є аналогом оперативної пам'яті сучасного комп'ютера. Уперше подібна модель пам'яті була використана в комп'ютері Атанасова і Беррі.

Вирішуючи цю проблему, американці Джон Атанасов і Кліффорд Беррі у 1940 році розробили модель повністю електронного комп'ютера, що використовує єдину систему представлення чисел і зв'язків між ними, – булеву алгебру. Їхній підхід базувався на роботах англійського математика XIX ст. Джорджа Буля, присвячених апарату символьної логіки. В основі булевої алгебри лежить інтерпретація елементів булевої алгебри як висловлювань, що набувають значення "істина" або "брехня". Атанасов і Беррі застосували цю концепцію для електронних пристроїв. Істині відповідало проходження електричного струму, а брехні – його відсутність. Для представлення чисел Атанасов і Беррі запропонували використовувати двійкову систему числення.

**14 лютого**
**День комп'ютерника**
Цього дня в 1946 р. був запущений комп'ютер ENIAC. Це був хоч і не перший комп'ютер, але найвідоміший і саме цього дня з'явилася офіційна професія програміста.

### 1.2.1.2 Покоління сучасних комп'ютерів

Розвиток обчислювальної техніки прийнято розглядати з точки зору зміни поколінь комп'ютерів. Кожне покоління комп'ютерів на початковий момент розвитку характеризується якісним стрибком у зростанні основних характеристик комп'ютера, викликаним, зазвичай, переходом на нову елементну базу, а також відносною стабільністю архітектурних і технологічних рішень.

Розбиття поколінь комп'ютерів за роками дуже умовне. В той час коли починалося активне використання комп'ютерів одного покоління, створювалися передумови для виникнення наступного. Окрім елементної бази і часового інтервалу використовуються наступні показники розвитку комп'ютерів одного покоління: швидкодія, архітектура, програмне забезпечення, рівень розвитку зовнішніх пристроїв. Іншим важливим якісним показником є широта сфери застосування комп'ютерів.

**Перше покоління комп'ютерів (1945–1954 рр.).** З початком другої світової війни усвідомлюючи стратегічну роль обчислювальних машин

у веденні війни, урядами різних країн було збільшено. фінансування відповідних розробок, що значною мірою стимулювало розвиток обчислювальної техніки. У 1941 році німецький інженер Конрад Цузе розробив обчислювальну машину Z2, що виконувала розрахунки, необхідні при проектуванні літаків і балістичних снарядів. У 1943 році англійські інженери завершили створення обчислювальної машини "Колос II" для дешифрування повідомлень німецької армії. Проте, ці пристрої не були універсальними обчислювальними машинами і призначалися лише для вирішення конкретних завдань.

У 1944 році американським інженером Говардом Ейкеном, за підтримки фірми IBM, було сконструйовано комп'ютер, що використовувався для балістичних розрахунків. Цей комп'ютер, названий "Марк I", за площею займав приблизно половину футбольного поля і включав більше 600 км кабелю. У ньому використовувався принцип електромеханічного реле, принцип якого полягає в тому, щоб електромагнітні сигнали переміщували механічні частини. "Марк I" був досить повільною машиною: виконання одного обчислення займало 3–5 с. Керування пристроєм "Марк I" здійснювалося за допомогою програми, яка вводилася з перфострічки, що дало можливість вирішувати досить широкий клас математичних завдань.

Проте, незважаючи на величезні розміри і повільність, "Марк I" став більш універсальним обчислювальним пристроєм, ніж машина Цузе або "Колос II".

У 1946 році американські учені Джон Мокнулі і Дж. Преспер Еккерт сконструювали електронний обчислювальний інтегратор і калькулятор (ЕНІАК) – комп'ютер, в якому електромеханічні реле були замінені на електронні вакуумні лампи. Застосування вакуумних ламп дозволило збільшити швидкість роботи ЕНІАК у 1000 разів порівняно з "Марк I".

ЕНІАК складався з 1800 вакуумних ламп, 70000 резисторів, 5 мільйонів сполучних спайок і споживав 160 кВт електричної енергії, що на ті часи було достатньо для освітлення великого міста. Між тим, ЕНІАК став працюючим прообразом сучасного комп'ютера. По-перше, ЕНІАК був заснований на принципі повністю цифрової обробки інформації. По-друге, ЕНІАК став дійсно універсальною обчислювальною машиною, він використовувався для розрахунку балістичних таблиць, прогнозу погоди, розрахунків в області атомної енергетики, аеродинаміки, вивчення космосу.

Наступний важливий крок у вдосконаленні обчислювальної техніки зробив американський математик Джон фон Нейман (рис.1.4). Попередні обчислювальні машини виконували тільки ті команди, що надходять ззовні та лише послідовно. Хоча використання перфокарт дозволяло спростити процес введення команд, проте, іноді процес налаштування обчислюваль-

ної машини та введення команд займали більше часу, ніж власне розв'язування поставленої задачі. Фон Нейман запропонував включити до складу комп'ютера для зберігання послідовності команд і даних спеціальний пристрій – пам'ять. Крім того, Джон фон Нейман запропонував реалізувати в комп'ютері можливість передачі управління від однієї програми до іншої.



Рисунок 1.4.
Джон фон Нейман

Зберігання в пам'яті комп'ютера різних наборів команд (програми), припиняти виконання однієї програми і передавати управління інший, а потім можливість повернення до початкової – значно розширювала можливості програмування для обчислювальних машин. Другою ключовою ідеєю, запропонованою фон Нейманом, став процесор (центральний оброблювальний пристрій), який повинен був управляти всіма функціями комп'ютера.

У 1945 році Джон фон Нейман підготував звіт, в якому визначив наступні основні принципи роботи і елементи архітектури комп'ютера (рис. 1.5):

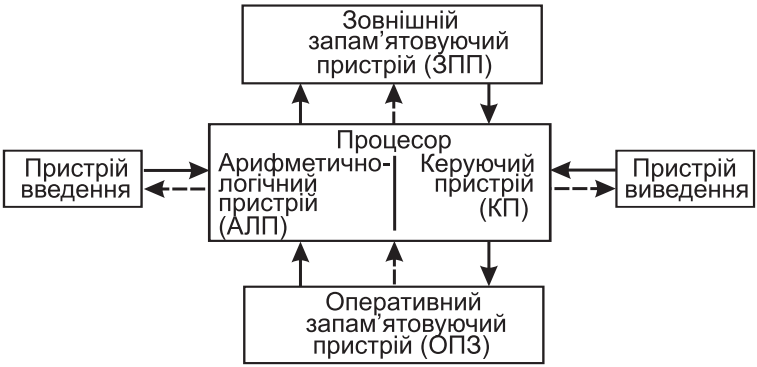
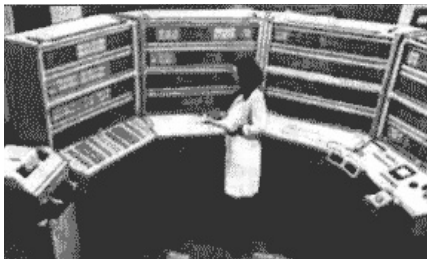



Рисунок 1.5. Архітектура ЕОМ, побудованої на принципах фон Неймана.
(Суцільні лінії стрілками вказують напрями потоків інформації,
пунктирні – керуючих сигналів від процесора до інших вузлів ЕОМ)

1. Комп'ютер складається з процесора (центрального оброблювального пристрою), пам'яті і зовнішніх пристроїв.
2. Єдиним джерелом активності (не враховуючи стартового або аварійного втручання людини) в комп'ютері є процесор, який, у свою чергу, керується програмою, що знаходиться в пам'яті комп'ютера.

3. Пам'ять комп'ютера складається з чарунок, кожна з яких має свою унікальну адресу. Кожна чарунка зберігає команду програми або одиницю оброблюваної інформації. Причому і команда, і інформація мають однакове представлення.
4. У будь-який момент процесор виконує одну команду програми, адреса якої знаходиться в спеціальному регістрі процесора – лічильнику команд.
5. Обробка інформації відбувається тільки в регістрах процесора. Інформація в процесор потрапляє з пам'яті або від зовнішнього пристрою.
6. У кожній команді програми зашифровані наступні приписи: з яких чарунок узяти інформацію для обробки; які операції провести з цією інформацією; у які елементи пам'яті направити результат; як змінити вміст лічильника команд, щоб дізнатися, звідки узяти наступну команду для виконання.
7. Процесор виконує програму команда за командою відповідно до зміни утримуваного лічильника команд до тих пір, поки не отримає команду зупинитися.

Надалі архітектура фон Неймана трохи змінювалася і доповнювалася, але початкові принципи управління роботою комп'ютера за допомогою програм, що зберігаються в пам'яті, залишилися незайманими. Переважна більшість сучасних комп'ютерів побудована саме за архітектурою фон Неймана. Так, у 1951 році був створений перший комп'ютер, призначений для комерційного використання – УНІАК (універсальний автоматичний комп'ютер), в якому були реалізовані усі принципи архітектури фон Неймана. У 1952 році за допомогою УНІАК був передбачений результат виборів президента США.



В Академії наук України у 1948-1951 рр. під керівництвом академіка Сергія Олексійовича Лебедєва була створена перша вітчизняна і перша на європейському континенті ЕОМ (мала електронно-рахункова машина – МЕРМ). Принципи побудови МЕРМ були розроблені Сергієм Олексійовичем Лебедєвим незалежно від проведених аналогічних робіт на Заході. У 1952–1953 роках "МЕРМ" залишалася самою швидкодіючою (50 операцій/с) обчислювальною машиною в Європі. До середини 50-х років швидкодію "МЕРМ" даний показник вдалося підвищити до кількох десятків тисяч операцій.

Колектив співробітників, який створив МЕРМ, став ядром організованого на базі лабораторії С.О. Лебедєва Обчислювального центру

НАН України, а з часом – Інституту кібернетики ім. В.М. Глушкова НАН України. У 1952-1953 рр. МЕРМ була єдиною ЕОМ в Європі, яка постійно експлуатувалася. У той період за допомогою цієї машини вирішувалися найважливіші завдання у сфері ядерної енергетики, космічних досліджень, далеких електропередач, механіки, математичної статистики та ін. Це був безумовний зліт науки в Україні у сфері обчислювальної техніки і її застосування.

Машинна мова, що застосовувалася в першому поколінні комп'ютерів, була вкрай незручна для сприйняття людиною. Числове кодування операцій, адрес чарунок і оброблюваної інформації, залежність виду програми від її місця в пам'яті не давали можливості стежити за змістом програми. Для подолання цих незручностей була запроваджена мова асемблер. Для запису кодів операцій і оброблюваної інформації в асемблері використовуються стандартні позначення, що дозволяють записувати числа і текст в загальноприйнятій формі, а для кодів команд – прийняті мнемонічні позначення. Для позначення величин, що розміщуються в пам'яті, можна застосовувати будь-які імена, що відповідають змісту програми. Після введення програми асемблер сам замінює символічні імена на адреси пам'яті, а символічні коди команд на числові. Використання асемблера зробило процес написання програм наочнішим.

У кінці 50-х – початку 60-х років комп'ютери другого покоління стали інтенсивно використовуватися державними організаціями і великими компаніями для вирішення різних завдань. До 1965 року велика частина великих компаній обробляла фінансову інформацію за допомогою комп'ютерів. Поступово вони набували риси сучасного комп'ютера. Так, в цей період були сконструйовані такі пристрої, як графічний пристрій і принтер, носії інформації на магнітній стрічці і магнітних дисках та ін.

Розширення сфери застосування комп'ютерів вимагало створення нових технологій програмування. Програмне забезпечення, написане на мові асемблер для одного комп'ютера, було непридатне для роботи на іншому. З цієї причини, зокрема, не вдавалося створити стандартну операційну систему – основну управляючу програму комп'ютера, оскільки кожен виробник комп'ютерів розробляв свою операційну систему на своєму асемблері.

У комп'ютерах першого покоління використовувалася машинна мова – спосіб запису програм, що уможливлює безпосереднє виконання їх на комп'ютері. Програма на машинній мові є послідовністю машинних команд, допустимих для даного комп'ютера. Процесор безпосередньо сприймає і виконує команди, виражені у вигляді двійкових кодів. Для кожного комп'ютера існувала своя власна машинна мова. Це також обмежувало сферу застосування комп'ютерів першого покоління.

Отже, поява першого покоління комп'ютерів стала можлива завдяки трьом технічним нововведенням: електронним вакуумним лампам, циф-

ровому кодуванню інформації і створенню пристроїв штучної пам'яті на електростатичних трубках. В комп'ютерах першого покоління використовувалася архітектура фон Неймана. Засоби програмування і програмне забезпечення не були досить розвинені, використовувалася низькорівнева машинна мова, а таким чином комп'ютери мали невисоку продуктивність: до декількох тисяч операцій в секунду, а сфера їх застосування була обмежена.

**4 грудня**
**День інформатики**
4 грудня 1948 р. – день народження російської інформатики. У серпні 1948 р. чл.-корр. АН СРСР И. С. Брук спільно з інженером Б.И. Рамеевым представив проект автоматичної обчислювальної машини. 4 грудня 1948 р. Державний комітет Ради Міністрів СРСР з впровадження передової техніки в народне господарство зареєстрував за номером 10475 винахід цифрової ЕОМ.

## Друге покоління комп'ютерів (1958–1960 роки)



Електронні вакуумні лампи виділяли велику кількість тепла, поглинали багато електричної енергії, були громіздкими, дорогими і ненадійними. Як наслідок, комп'ютери першого покоління, побудовані на вакуумних лампах, мали низьку швидкодію і невисоку надійність. У 1947 році співробітники американської компанії "Белл" Уільям Шоклі, Джон Бардін і Уолтер Бреттейн винайшли транзистор. Транзистори виконували ті ж функції, що і електронні лампи, але використовували електричні властивості напівпровідників. Порівняно з вакуумними трубками транзистори займали в 200 разів менше місця і споживали в 100 разів менше електроенергії. У той же час з'являються нові пристрої для організації пам'яті комп'ютерів – феритові сердечники. З винаходом транзистора і використанням нових технологій зберігання даних у пам'яті з'явилася можливість значно зменшити розміри комп'ютерів, зробити їх швидшими і надійнішими, а також значно збільшити місткість пам'яті комп'ютерів.

У 1954 році компанія Texas Instruments оголосила про початок серійного виробництва транзисторів, а в 1956 році вчені Массачусетського технологічного інституту створили перший, повністю побудований на транзисторах, комп'ютер TX-O.

У 50-60-х рр. в Обчислювальному центрі НАН України створили першу в Україні ЕОМ з асинхронним управлінням "Київ"; була розроблена і введена у виробництво серія машин для інженерних розрахунків (МИР1, МИР2, МИР3), попередників персональних ЕОМ. У багатьох промислових і експериментальних системах використовувалася перша вітчизняна

напівпровідникова ЕОМ широкого призначення, що управляє, під назвою "Дніпро". Кібернетик В. М. Глушков став відомий на заході не менше, ніж засновник цієї науки Н. Винер (США). На жаль, талановитий учений пішов з життя в п'ятдесят дев'ять років (1913-72 рр.).

Одночасно з Інститутом кібернетики в Україні активно працювали ряд інших подібних організацій. У Сєвєродонецькому інституті машин, що управляють, ще в 50-х роках був спроектований і успішно випробуваний на промисловому об'єкті перший в Радянському Союзі і Європі цифровий регулятор "Автооператор". Там же, була розроблена феррит-диодная ЕОМ "Автодиспетчер", а потім – серійна машина, що управляє, РОЗУМ-1. У 60-70-х рр. Київ і Сєвєродонецьк стали центрами комп'ютерної науки і виробництва, які забезпечили розробку і промислове виробництво машин, що управляють, для більшості систем, що управляють, створених у той час в Радянському Союзі.

Під керівництвом Н. П. Брусенцова (народився в Дніпродзержинську) в 1958 році була введена у виробництво перша і єдина у світі трійкова ЕОМ "Сетунь".

І. Я. Акушский, уродженець Дніпропетровська, став основоположником нетрадиційної арифметики, заснованої на системі числення в залишках. На її основі в 1960 році була розроблена спеціалізована ЕОМ, що виконувала більше мільйона операцій на секунду, – у той час це був світовий рекорд швидкості обчислення.

Киянин за походженням М. К. Судім був заступником С. О. Лебедєва, коли той працював над однією з перших ламповых ЕОМ М20, а з часом – керівником робіт із створення напівпровідникової ЕОМ М220.

Особливо необхідно відмітити заслуги киянина М.О. Карцева. Він спроектував ЕОМ М2, яка почала працювати майже одночасно з МЕРМ, а пізніше – серію СУПЕРЕОМ для систем спостереження за космічним простором.

Фахівці, що використовують у своїй діяльності комп'ютери, незабаром відчули потребу в природніших мовах, які б спрощували процес програмування, а також дозволяли переносити програми з одного комп'ютера на інший. Подібні мови програмування дістали назву мов високого рівня. Для їхнього використання необхідно мати компілятор (чи інтерпретатор), тобто програму, яка перетворить оператори мови в машинну мову цього комп'ютера.

Серед найбільш поширених мов того часу слід виділити:

- ***ALGOL (ALGOritmic Language*** – алгоритмічна мова**)**, що застосовувався для опису алгоритмів рішення складних математичних завдань;
- ***FORTRAN (FORmula TRANslator*** – формульный транслятор) служив для інженерно-технічних завдань;

- ***COBOL (COmputer Business Oriented Language)*** вживався при рішенні економічних завдань.

Однією з перших мов програмування стала мова Фортран (FORTRAN), яка призначалась для природного вираження математичних алгоритмів і стала надзвичайно поширеною серед учених. На Фортрані можна писати об'ємні програми, розбиваючи завдання на декілька частин (підпрограм), які програмуються окремо, а потім об'єднуються в єдине ціле. Оскільки Фортран призначений в основному для обчислень, у ньому були відсутні розвинені засоби роботи із структурами даних. Цей недолік був виправлений у мові Кобол (COBOL). Кобол спеціально призначався для обробки фінансово-економічних даних. Крім того, розробники доклали зусиль зробити Кобол максимально схожим на природну англійську мову, що дозволило писати програми цією мовою навіть неспеціалістам у програмуванні. З другим поколінням комп'ютерів почався розвиток індустрії програмного забезпечення.

У основі практично усіх мов програмування, розроблених для ЕОМ другого покоління, полягала ідея процедурної структуризації програм. Тобто, програміст повинен був вирішити, які саме процедури він використовуватиме у своїй програмі, а потім обере найбільш ефективні алгоритми для їх реалізації. Досить часто при послідовному застосуванні цього принципу, підпрограми розбивалися на процедури все більш і більш низького рівня до тих пір, поки алгоритми, що реалізовувалися ними, не стануть настільки простими, щоб їх можна було легко запрограмувати.

З приходом ЕОМ другого покоління з'явилися люди ще однієї комп'ютерної професії: експлуатаційники, що займалися рішенням завдань за допомогою програм, написаних програмістами.

Отже, другий період розвитку обчислювальної техніки характеризується застосуванням для створення комп'ютерів транзисторів і пам'яті на феритових сердечниках, збільшенням швидкодії комп'ютерів до декількох сотень тисяч операцій в секунду, виникненням нових технологій програмування, мов програмування високого рівня, операційних систем. Комп'ютери другого покоління набули широкого поширення, вони використовувалися для наукових, інженерних і фінансових розрахунків, для обробки великих обсягів даних на підприємствах, в банках, державних організаціях.

**Третє покоління комп'ютерів (1965–1971 роки).** У 1958 інженер компанії Texas Instruments Джек Кілбі запропонував ідею інтегральної мікросхеми – кремнієвого кристалу, на який монтуються мініатюрні транзистори та інші елементи. У тому ж році Кілбі представив перший зразок інтегральної мікросхеми, що утримує п'ять транзисторних елементів на кристалі германію. Мікросхема Кілбі займала трохи більше сантиметра площі і була декілька міліметрів завтовшки. Рік потому, незалежно від

Кілбі, Роберт Нойс розробив інтегральну мікросхему на основі кристалу кремнію. Згодом Роберт Нойс заснував компанію "Intel" з виробництва інтегральних мікросхем. Мікросхеми працювали значно швидше за транзистори і споживали значно менше енергії.



Перші інтегральні мікросхеми складалися всього з декількох еле-ментів. Проте, використовуючи напівпровідникову технологію, учені досить швидко навчилися розміщувати на одній інтегральній мікросхемі спочатку десятки, а потім сотні і більше транзисторних елементів.

Оскільки до ЕОМ третього покоління могло бути підключено досить значна кількість периферійних пристроїв (кожне з яких працювало зі своєю швидкістю), то забезпечити їх ефективну спільну роботу з комп'ютером було дуже складно. Тому управління обчислювальним процесом було покладено на саму машину.

Зрозуміло, для вирішення такого завдання був потрібний певний комплекс програм, що управляють. Він дістав назву операційної системи (ОС). Перша ОС була створена для машини IBM - 704 в 1953-1955 роках.

Поява ОС розділила програмістів на два табори: системщиків і прикладників.

Перші розробляли ОС і утиліти до неї, а другі писали програми для вирішення завдань, поставлених замовником.

У 1964 році компанія IBM випустила комп'ютер IBM System 360, побудований на основі інтегральних мікросхем. Сімейство комп'ютерів IBM System 360 найчисленніше сімейство комп'ютерів третього покоління і одне з найвдаліших в історії обчислювальної техніки. Випуск цих комп'ютерів можна вважати початком масового виробництва обчислювальної техніки. Усього було випущено більше 20 000 екземплярів System 360.

Тільки у 1972 році, через вісім років після США, зусиллями країн-членів СЕВ (СРСР, Болгарія, Угорщина, Польща, Чехословаччина, ГДР) розробляється Єдина Серія (ЄС) – універсальні ЕОМ третього покоління, повністю сумісні з американською системою IBM/360. Це машини ЄС-1010, ЄС-1020, ЄС-1022, ЄС-1030, ЄС-1040, ЄС-1050, ЄС-1060. У восьмидесятих випускається ряд моделей ЄС-1015, ЄС-1025, ЄС-1035, ЄС-1045, ЄС-1055, ЄС-1065, а також Сімейство Малих ЕОМ – СМ. Кращі вітчизняні зразки розвивали швидкість виконання до двох мільйонів операцій в секунду. Система адресації дозволяє адресувати об'єми пам'яті до 16 Мб.

IBM System 360 відноситься до класу так званих мейнфреймів. Компанія DEC (Digital Equipment Corporation) представила модель мінікомп'ютера PDP-8. Міні-комп'ютери або комп'ютери середньої продуктивності характеризуються високою надійністю і порівняно низькою вартістю, що дозволила почати застосовувати їх в невеликих

організаціях – дослідницьких лабораторіях, офісах, на невеликих промислових підприємствах. У той же час відбувалося вдосконалення програмного забезпечення. Операційні системи будувалися так, щоб підтримувати більшу кількість зовнішніх пристроїв, з'явилися перші комерційні операційні системи і нові прикладні програми.

У 1968 році на одній з конференцій Дуглас Енгельбарт із Стенфордського інституту продемонстрував створену ним систему взаємодії комп'ютера з користувачем, що складається з клавіатури, покажчика "миша" і графічного інтерфейсу, а також деякі програми, зокрема текстовий процесор і систему гіпертексту.

Серед універсальних мов високого рівня, створених в 60-70 роки для ЕОМ третього і четвертого покоління, виділимо наступні:

- ***BASIC (Beginner's All – purpose Symbolic Instruction Code).*** Проста для вивчення і застосування мова. Сьогодні багато непрофесійних програмістів використовують сучасні діалекти цієї мови для написання невеликих прикладних програм.
- ***Pascal*** (названий на честь конструктора одного з перших арифмометрів Блеза Паскаля). Мова розроблена Ніколасом Виртом для навчання студентів програмуванню. Чіткість і ясність його смислових конструкцій, велика кількість різноманітних типів даних, а також суворість контролю їх використання зацікавили багатьох програмістів-професіоналів. Дана мова дала можливість зручно реалізовувати складні алгоритми, що оперують різними типами даних. Простота трансляції програм в машинні команди забезпечила поширеність Pascal.
- ***C.*** Розроблений в 1972 р. Деннисом Ритчи, одним з авторів операційної системи UNIX. У подальшому Сі використовувався при створенні цієї системи і написанні бібліотек обслуговуючих програм. Будучи універсальною мовою загального призначення, дуже зручний для програмування системних (що відносяться до операційної системи) завдань.

У 1964 році з'явилася мова програмування Бейсік (BASIC – Beginner's All – Purpose Symbolic Instruction Code), призначена для навчання програмістів-початківців. Бейсік забезпечувала швидке введення і перевірку програм. Бейсик не дуже підходила для написання серйозних програм, проте вона давала загальне уявлення про програмування і дозволяла багатьом людям, які не мали відповідної освіти, швидко опанувати основні принципи програмування.

У 1970 році швейцарець Никлас Вірт (рис. 1.6) розробив мову програмування Паскаль, також призначену для навчання принципам програмування. Мова Паскаль, що створювалася як мова для навчання, виявилася дуже зручною для вирішення багатьох прикладних завдань. Вона прекрасно

забезпечувала застосування методів структурного програмування, що стало необхідно при створенні великих програмних систем.



Рисунок 1.6.
Проф. Н. Вірт

Отже, основою для комп'ютерів третього покоління стали інтегральні мікросхеми, що дозволило значно зменшити вартість і розміри комп'ютерів, почалося масове виробництво комп'ютерів. У цей період розвитку обчислювальної техніки тривало збільшення швидкості обробки інформації. (до одного мільйона операцій/с). З'явилися нові зовнішні пристрої, що полегшують взаємодію людини з комп'ютером. Збільшення швидкодії комп'ютерів і сфери їхнього застосування призвело до розробки нових методів створення програмного забезпечення. З'явилися перші комерційні операційні системи реального часу, спеціально розроблені для них мови програмування високого рівня. Сфера застосування комп'ютерів третього покоління надзвичайно широка: системи обробки даних, управління, проектування, розв'язування різних комерційних задач.

**Четверте покоління комп'ютерів (з 1976 –1985 рокі).** У 1965 році голова ради директорів компанії "Intel" Гордон Мур припустив, що кількість елементів на інтегральних мікросхемах повинна подвоюватися кожні 18 місяців. Надалі це правило, відоме як закон Мура, було застосовано до швидкості мікропроцесорів і досі не порушувалося.

У 1969 році компанія "Intel" випустила ще один важливий для розвитку обчислювальної техніки пристрій – мікропроцесор. Мікропроцесор є інтегральною мікросхемою, на якій зосереджений оброблювальний пристрій з власною системою команд. Конструкція мікропроцесора дозволяє застосовувати його для вирішення широкого кола завдань, створюючи при цьому різні функціональні пристрої. Використання мікропроцесорів значно спростило конструкцію комп'ютерів. Практично відразу мікропроцесори отримали широке застосування в різних системах управління від космічних апаратів до побутових приладів.

Упродовж наступних десятиліть, наслідуючи закон Мура, тривало збільшення швидкості та інтеграції мікропроцесорів. З'явилися надвеликі інтегральні схеми, що включають сотні тисяч і навіть мільйони елементів на один кристал. Це дозволило продовжити зменшення розмірів і вартості комп'ютерів і підвищити їхню продуктивність і надійність.

Практично одночасно з мікропроцесорами з'явилися мікрокомп'ютери, або персональні комп'ютери, відмітною особливістю яких стали невеликі розміри і низька вартість. Завдяки своїм характеристикам

персональні комп'ютери надали можливість практично будь-якій людині познайомитися з обчислювальною технікою. Комп'ютери перестали бути прерогативою великих компаній і державних установ, а перетворилися на товар масового споживання.

Одним з піонерів у виробництві персональних комп'ютерів була компанія Apple. Її засновники Стів Джобс і Стів Возняк зібрали першу модель персонального комп'ютера в 1976 році і назвали її Apple I. У 1977 році вони представили свій комп'ютер членам комп'ютерного клубу в Каліфорнії і наступного дня отримали замовлення на 50 подібних комп'ютерів. Вартість першого персонального комп'ютера складала всього 500 доларів. У тому ж 1977 році компанія Apple представила наступну модель персонального комп'ютера – Apple II. У новій моделі був витончений пластиковий корпус із вбудованою клавіатурою. Уперше комп'ютер набув рис побутового приладу. Продажі персональних комп'ютерів різко зросли. Apple II остаточно зруйнував уявлення про комп'ютер, як про "величезного залізного монстра", у нього був витончений дизайн і дружній інтерфейс взаємодії з користувачем.

Великі компанії не були зацікавлені у персональних комп'ютерах до 1979 року, коли з'явився перший процесор електронних таблиць – VisiCalc. Ідея VisiCalc була запропонована студентом Гарварду Даном Брісклін, якому довелося вирішувати складні фінансові завдання, що вимагають великої кількості обчислень. Із своїм другом Бобом Франкстоном вони написали VisiCalc для комп'ютера Apple II. Програма виявилася настільки зручною для фінансових обчислень, що багато компаній стали купувати Apple II з VisiCalc для своїх співробітників.

У 1981 році найбільша комп'ютерна компанія IBM представила свій перший персональний комп'ютер – IBM PC. Впродовж двох років було продано більше п'яти мільйонів цих комп'ютерів. В той же час компанія Microsoft починає випуск програмного забезпечення для IBM PC. З'являються клони IBM PC, але усі вони, так або інакше, відбивають стандарти, закладені IBM. Поява клонів IBM PC сприяла зростанню промислового виробництва персональних комп'ютерів.

У 1984 році компанія Apple представила комп'ютер "Макінтош". Операційна система "Макінтоша" включала графічний інтерфейс користувача, що дозволяв вводити команди, вибираючи їх за допомогою покажчика "миша". Самі команди були представлені у вигляді невеликих графічних зображень – значків. Простота використання у поєднанні з великим набором текстових і графічних програм зробила цей комп'ютер ідеальним для невеликих офісів, видавництв, шкіл і навіть дитячих садів. З появою "Макінтоша" персональний комп'ютер став ще доступнішим. Для роботи з ним більше не були потрібні ніяких спеціальних навичок, а тим більше знання програмування. У 1984 році компанією Apple був запущений на те-

лебаченні перший ролик, присвячений рекламі персонального комп'ютера. Комп'ютер дійсно перестав бути чимось особливим і перетворився на звичайний побутовий прилад.

Протягом усього 50 років комп'ютери перетворилися з незграбних дивовижних електронних монстрів в потужний, гнучкий, зручний і доступний інструмент. Комп'ютери стали символом прогресу в XX столітті. У міру того як людині знадобиться обробляти усе більшу кількість інформації, удосконалюватимуться і засоби її обробки – комп'ютери.

**П'яте покоління комп'ютерів (теперішний час).** ЕОМ п'ятого покоління базуються не стільки на прагненні витягнути додаткові переваги з надкомпактної упаковки надвеликих інтегральних схем (НВІС), скільки на реалізації широкої програми, яка передбачає досягнення принципових зрушень в усіх областях, пов'язаних з конструюванням, виробництвом, технічним обслуговуванням і використанням комп'ютерів. При цьому можна виділити:

- розробку експертних систем, які включають бази знань, що забезпечують рішення складних завдань і ухвалення рішень, а також передбачають введення і виведення інформації за допомогою голосу і графічних зображень;
- проблеми розробки мов надвисокого рівня, у тому числі функціональних і логічних, таких, що забезпечують досконалі способи програмування; проблеми децентралізованних обчислень за допомогою мереж ЕОМ, як великих, таких, що знаходяться на значній відстані один від одного, так і мініатюрних МІКРОЕОМ, розміщених на одній друкованій платі або навіть на одному кристалі напівпровідника;
- проблеми технології НВІС для ЕОМ загального застосування і спеціалізованих ЕОМ, які мають величезну обчислювальну потужність при помірній вартості.

Таким чином, засаднича ідея створення ЕОМ п'ятого покоління полягає в розробці збалансованої по своїй архітектурі системи комп'ютерів, щоб кожен користувач такої системи мав можливість використовувати її величезний обчислювальний потенціал.

Для ЕОМ наступних поколінь, напевно, елементна база ґрунтуватиметься на оптико-електронних елементах. У цих машинах носіями енергії служать не електрони, а фотони, що значно підвищує швидкість передачі сигналів, так що швидкодія ЕОМ може досягти сотень мільйонів операцій в секунду. Перетворення електричних сигналів в оптичні в машинах здійснюється лазерами і світловипромінюючими діодами, використовуються світлопроводи і фотоприймачі. Подальший розвиток отримує процес утворення обчислювальних систем (ВС), зрощення машин і обчислювальних центрів з системою зв'язку. Змінюється і уявлення про

систему зв'язку, яка робить не лише послуги передачі інформації, але і її зберігання і обробки.

Нині розроблені ЕОМ зі швидкодією до мільярда операцій в секунду. Ця швидкість підвищується завдяки паралельній роботі усіх пристроїв, у тому числі декількох процесорів. Стає можливим здійснення паралельного перетворення інформації типу тій, яка представляється у вигляді голограм за допомогою систем лазерних елементів.

**Шосте покоління ЕОМ.** Шосте покоління ЕОМ припускає електронні і оптоелектронні комп'ютери з масовим паралелізмом, нейронною структурою, з розподіленою мережею великого числа (десятки тисяч) мікропроцесорів, що моделюють архітектуру нейронних біологічних систем.

### 1.2.2 Загальні відомості про персональний комп'ютер

#### 1.2.2.1 Склад персонального комп'ютера

При розгляді складу персонального комп'ютера (ПК) прийнято виходити з того, які задачі передбачається вирішувати користувачу за допомогою комп'ютера.

Найбільш розповсюдженою задачею є створення різних документів, що відбивають діяльність організації (плани, договори, контракти, звіти, прайс-листи, прості бази даних тощо).

У зв'язку з цим можна виділити три основних види конфігурації комп'ютерного устаткування, що відрізняються кількістю вхідних у них пристроїв:

I. Стандартна конфігурація (базова).

II. Розширена конфігурація.

III. Спеціальна конфігурація.

I. Стандартна (базова) конфігурація є найбільш розповсюдженим офісним варіантом комп'ютерного устаткування, при якому до складу його входить тільки чотири пристрої (рис. 1.7):

1. Блок процесора.
2. Монітор або дисплей.
3. Клавіатура.
4. Маніпулятор типу "миша" (надалі миша).

II. Розширена конфігурація відрізняється від стандартної наявністю додаткових пристроїв, до яких можна віднести наступні:

1. Факс-модем – пристрій, що забезпечує зв'язок між віддаленими комп'ютерами через телефонних каналах зв'язку. При його наявності можливо:

– відправляти факсимільні повідомлення безпосередньо з робочих вікон текстових, табличних або графічних редакторів,

що значно заощаджує час і папір, тому що немає необхідності друкувати документ перед його відправленням, як це має місце при відправленні документа по телефону-факсу. При відсутності факсу-модему в приймаючої сторони документ буде прийнятий звичайним телефоном-факсом. Якщо ж у приймаючої сторони також мається факс-модем, то документ автоматично буде поміщений у журнал вхідних телефонограм, що знаходиться на жорсткому диску комп'ютера. Прийняте повідомлення (документ) може бути переглянуто користувачем у будь-який момент після закінчення його прийому і при необхідності виведено на друк для одержання твердої копії;

– встановлювати за допомогою телефонних лініях зв'язок між двома комп'ютерами, обладнаними факсами-модемами, для обміну програмами і документами. При такому виді зв'язку, так званому "віддаленому доступі", користувачі можуть бачити всю інформацію, що зберігається на жорсткому диску віддаленого комп'ютера, і при бажанні переписати потрібні програми або документи на свій жорсткий диск;
– підключатися до інформаційних мереж типу BBS, Relcom, Інтернет та ін. з розділів, які цікавлять користувача (наука, медицина, бізнес тощо).



Рисунок 1.7. Базова конфігурація комп'ютерної системи

2. Сканер – пристрій введення в комп'ютер зображення (тексту, рисунків тощо) шляхом зчитування з паперу;

3. CD ROM (compact disk read only memory) – пристрій для читання компакт-дисків. Компакт-диски (часто їх називають лазерними дисками) випускаються в трьох модифікаціях:

– з набором програмного продукту (Windows різних версій, різні додатки Windows типу Microsoft Office тощо);

– музикальні або аудіодиски;
– відео компакт-диски (відеофільми, різного роду навчальні програми та ін.);

4. Звукова карта, до виходу якої підключаються звукові колонки. Звукова карта забезпечує можливість відтворювати музичні твори, записані на аудіо та відео компакт-дисках;

5. Комунікаційне устаткування – спеціальне устаткування, використовуване для об'єднання комп'ютерів у мережу для спільної роботи.

Перераховане устаткування, що складає розширену конфігурацію, останнім часом стає найкращим як для будь-якого офісу, так і для домашнього робочого місця. У домашньому варіанті виключається комунікаційне устаткування.

III. Що стосується спеціальної конфігурації, то склад устаткування для неї визначається задачами, які мають бути вирішувати з його допомогою. До таких задач можна віднести:

– нелінійний відеомонтаж, що застосовується при створенні відеофільмів (кліпів), для чого до складу конфігурації додатково включаються відеокамера, спеціальний відеомагнітофон, спеціальний телевізор та інше відеоустаткування;
– медична спрямованість: діагностику захворювань за допомогою комп'ютерного томографа, проведення аналізу крові з використанням спеціального устаткування, керованого комп'ютером;
– видавничу діяльність, для здійснення якої до складу конфігурації повинна входити апаратура так званої настільної видавничої системи;
– керування різними технологічними процесами. У цьому випадку додатковим устаткуванням є вільно програмувальний контролер (СПК). СПК є спеціалізованим обчислювальним пристроєм, що підключається до комп'ютера для забезпечення можливості його програмування за допомогою останнього. СПК за закладеній у нього за допомогою комп'ютера програмі керує різними виконавчими автоматами, встановленими уздовж технологічної лінії (конвеєра). Як приклади таких технологічних ліній можна привести лінії з виробництва деталей меблів, з пошиття взуття, одягу, з розливу різних рідких харчових продуктів та ін.

Перелік подібних задач можна було б продовжити. Але простіше, мабуть, сказати, що в даний час немає практично жодної задачі, при рішенні якої не використовувався б комп'ютер разом з керованим ним технологічним або іншим устаткуванням.

### 1.2.2.2 Загальні відомості про пристрої, що входять до складу персонального комп'ютера

Мета даного навчального питання – познайомити користувача з тим кругом понять, що допоможуть йому правильно орієнтуватися в наступних ситуаціях:

- при купівлі комп'ютера;
- при бажанні встановити на комп'ютері програмний продукт;
- при бажанні поліпшити характеристики вже наявного комп'ютера з метою підвищення його продуктивності, мінімізуючи витрати на це;
- як, при можливості, відгородити себе від шкідливого впливу комп'ютера при частину значному обсязі робочого часу проведеному біля нього.

ПК має модульну структуру (рис. 1.8). Усі модулі пов'язані з системною магістраллю (шиною).




Рисунок 1.8. Загальна структура персонального комп'ютера з приєднаними периферійними пристроями

Найбільший інтерес для користувача в цьому сенсі має блок процесора і його складові.

**Склад блоку процесора і призначення деяких вхідних його елементів.** Системний блок (блок процесора) є центральним серед усіх при-

строїв, що входять до складу перерахованих вище конфігурацій. Саме від характеристик характеристик системного блоку та вхідних компонентів залежить швидкодія і продуктивність комп'ютера в цілому.

До основних характеристик, що представляють інтерес для користувача, можна віднести:

1. Швидкісні характеристики, які суб'єктивно можна оцінивати за:

– величиною тактової частоти процесора (У літературі часто для простоти сприйняття процесор порівнюють з мозком людини, називаючи його за аналогією мозком комп'ютера.Але таке порівняння є досить суб'єктивним. На користь такої аналогії говорить те, що багато операцій, які виконуються мозком людини при рішенні логічних і математичних задач, у чомусь подібні операціям, які виконуються процесором при рішенні аналогічних задач за заданою процесору програмою. Але не треба забувати, що людський мозок спирається не тільки на вивчені людиною формальні закони логіки і математики, але і широко використовує так звану інтуїтивно-підсвідому логіку, що, як правило, тим ефективніша, чим багатший інтелектуальний багаж, накопичений людиною. Процесор же керується тільки формальною логікою, у межах тих її обсягів, що людина визначила в програмі рішення задачі.), що вимірюється в мегагерцах (МГц) (1 Гц – одне коливання в секунду, 1 МГц – один мільйон коливань у секунду). Чим вища тактова частота, тим продуктивніший ПК. У сучасних процесорах (основні виробники Intel, AMD, Cyrix, IDT, Rise та інші) тактова частота сягає більше 2-х ГГц.

Тактова частота процесора в кратне число разів перевищує тактову частоту системної шини. Для забезпечення їх синхронної роботи встановлюється спеціальна *кеш-пам'ять* (cash – "тайник") першого і другого рівня.

MMX (MultiMedia eXtention) – різновид процесора, в якому передбачено додаткові команди для обробки звуку і графіки.

– швидкості обміну даними між процесором і іншими пристроями й елементами ПК, що оцінюється також за допомогою частотних характеристик шини обміну даними і шини адреси, тобто в Мгц. На сьогоднішній день у кращих ПК тактова частота роботи зазначених шин обмежується величиною 100 Мгц;
– швидкості запису і зчитування інформації з жорсткого диску – пристрій, що використовується в комп'ютері для тривалого збереження інформації. Жорсткий диск зручно уявляти собі канцелярською шафою, у якій в папках зберігаються різні внутрішні і зовнішні документи, що характеризують вид діяльності організації (наукової, комерційної, медичної тощо). На корінцях папок ми

завжди побачимо написи, що пояснюють характер документів, що зберігаються в них (статті, доповіді, реферати, договори, акти, рахунки тощо);

– швидкості зміни зображення на екрані монітора. Дана характеристика набуває значення коли при використанні комп'ютера для створення і редагування відео сюжетів, перегляду відеоінформації. Якщо швидкість зміни графічної інформації є досить низькою, то враження при її перегляді буде таким же, яким воно було в глядачів перших кінофільмів;

2. Обсяг оперативного запам'ятовуючого пристрою (ОЗП) або просто – пам'яті, обчислювальний у мегабайтах (Мб);

3. Припустимий обсяг збереження інформації на жорсткому диску. Зараз широковідомі диски з обсягом більш 2 Терабайтів (Тб). Для того щоб одержати уявлення про значення зазначених цифр, можна скористатися простим розрахунком. Для їхнього здійснення надамо наступну додаткову інформацію:

- ✧ будь-яка буква, цифра або розділовий знак, записані в цифровому виді на твердий диск, займають обсяг один байт;
- ✧ один байт – це двійкове число, що складається з восьми позицій, на кожній з яких може знаходитися або одиниця, або нуль (наприклад, 10011011). Залежно від того, на яких позиціях знаходяться одиниці і нулі, кожна так називана кодова комбінація одиниць і нулів буде відповідати букві, цифрі або розділовому знаку;
- ✧ кількість кодових комбінацій, які можна одержати, починаючи з тієї, для якої на кожній позиції будуть тільки нулі, до тієї, у якій усі нулі заміняться поступово одиницями, обчислюється по відомій із шкільного курсу математики формулі $N = 2^8 = 256$. Отже, кількість можливих кодових комбінацій перевищує суму кількості букв у алфавіт будь-якої мови, кількості розділових знаків і цифр;
- ✧ на кожному рядку стандартного аркушу формату А4, при друку стандартним шрифтом (книжковим, розмір шрифту = 12 пунктів [1]) можна розмістити 80 символів. Таких рядкі – 40. Отже, кількість символів на аркуші буде дорівнювати 80 х 40 = 3 200 символів (разом із пробілами між словами). Таким чином, текст, що міститься на одній сторінці стандартного листа папера, записаний на твердий диск, займе на ньому ~ 3,2 Кбайт;
- ✧ для того щоб визначити, скільки сторінок такого тексту може вміститися на диск із зазначеним вище обсягом, нагадаємо

[1] Пункт – одиниця виміру висоти шрифту, застосовуваний у поліграфії (12 пунктів відповідає висоті шрифту приблизно 2 мм).

співвідношення між байтами, кілобайтами, мегабайтами і гігабайтами:

1Гб=1024Мб,
1Мб=1024 Кб,
1Кб=1024 байта.

**Тобто.** 1Гб = 1024 х 1024 х1024 = 1.073.741.824 байта.

Для визначення кількості сторінок з текстом, що можна записати на диск обсягом 1 Гб, отримане значення розділимо на обсяг тексту на одній сторінці стандартного аркуша:

1.073.741.824:3 200 = 335544 сторінок.

Якщо ви заглянете на останню сторінку роману "Війна та мир" Л.М. Толстого і спробуєте з'ясувати, скільки романів з таким обсягом сторінок можна зберегти на твердому диску обсягом усього в 1 Гб, то одержите цифру, що приблизно дорівнює 300.

Після таких досліджень у багатьох може виникнути природне запитання, навіщо прагнути купувати диски обсягом більше 1 Гб, якщо я, за все життя, не напишу стільки, щоб це відповідало трьомстам томам.

У попередніх міркуваннях були присутні дані, що стосуються текстових документів створених користувачем, але не стосувалися питань про те, що забезпечує можливість користувачу створювати документи, за допомогою яких інструментів користувач записує документи на жорсткий диск, що допомагає йому переглядати списки раніше створених і збережених документів, чим він користується для наведення порядку у своїх документах тощо.

Виявляється, щоб мати можливість створювати документи різної складності і вирішувати інші задачі роботи з документацією, на тому ж жорсткому диску користувач повинен мати програмний продукт, що займає обсяг часом більш одного гігабайту. Крім того, вище наведені розрахунки торкалися тільки текстового документу.

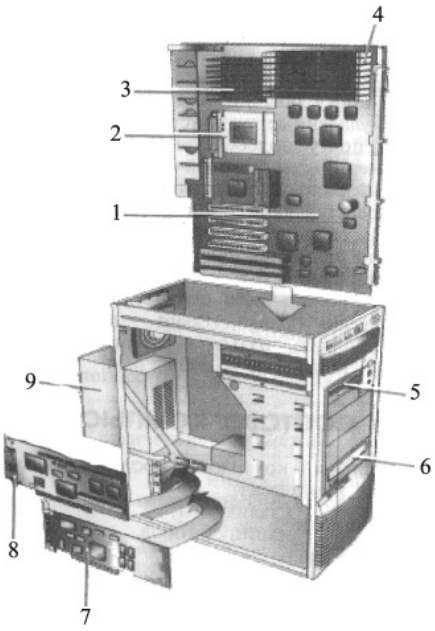
Весь зайвий обсяг зайняли малюнки і фотографії екранів дисплея, призначені полегшити читачеві сприйняття матеріалу і допомогти швидше набути практичних навичок у використанні комп'ютера.

Отже, повернемося до складу блоку процесора і з'ясуємо, з якими його елементами, які забезпечують наведені вище характеристики, користувач повинен більш детально ознайомитися.

До складу системного блоку входить (рис. 1.9):

- ✧ материнська плата (Motherboard) – основний елемент блоку процесора;
- ✧ блок живлення з вентилятором обдування, за працездатністю якого треба уважно стежити за характерним звуком. Якщо вентилятор не працює, комп'ютер можна вмикати не більше ніж на 10 – 15 хвилин;

- ✧ жорсткий диск (Hard disk, або HDD), магнітний пристрій для читання і запису великої місткості;
- ✧ дисковод для гнучких магнітних дискет (Floppy disk, або FDD). Гнучкі дискети призначені для запису на них архівів документів з метою забезпечення їхньої схоронності;
- ✧ пристрою для читання лазерних компакт-дисків (CD ROM – Compact Disk Read only memory – компакт-диск тільки для читання, CDD, DVDD);
- ✧ карти розширення : відеокарта, звукова карта, мережева карта;
- ✧ пристрої керування комп'ютером.




**Системний блок зі знятою кришкою:**

1 – системна плата;
2 – роз'їм додаткового другого процесора;
3 – центральний процесор з радіатором для виведення тепла;
4 – роз'їм оперативної пам'яті;
5 – накопичувач на гнучких магнітних дисках;
6 – накопичувач CD-ROM;
7 – мережева карта;
8 – графічний акселератор;
9 – блок живлення, що перетворює змінну напругу електромережі в постійну напругу різної полярності та величину, необхідну для живлення системної плати та внутрішніх пристроїв. Блок живлення містить вентилятор, що створює циркулюючі потоки повітря для охолодження системного блоку.

Рисунок 1.9. Системний блок

Усі комплектуючі системного блоку, мають бути розміщені в **корпус**. Існує декілька варіантів корпусів. Відсутність жорсткого стандарту дозволяє виробникам випускати різноманітну продукцію, яка відрізняється за дизайном, габаритами і способам зборки.

Корпус обирається виходячи з того, як він буде розташований – на робочому столі поряд з монітором, на підлозі або в спеціальному відсіку комп'ютерного столу. На даний момент найбільш поширені корпуси виду Tower (вежа) в різних модифікаціях. Випускаються корпусу і горизонталь-

ного розташування, так звані Desktop (робочий стіл) і низькопрофільний корпус типу Slim, що є різновидом попереднього.

Останніми роками став популярний термін моддинг (від англ. Modify – модифікувати), а точніше, його різновид case-modding – зміна зовнішнього вигляду корпусу із застосуванням креативних ідей і рішень. В основному це виражається в створенні прозорих стінок і підсвічуванні стандартних "веж", а також інших оригінальних рішеннях (рис 1.10).



Рисунок 1.10. Корпусу різної конфігурації (case-modding)

Разом з корпусом, зазвичай, поставляється блок живлення, що забезпечує електроживлення всіх внутрішніх компонентів системного блоку.

При цьому використовується стандартна за розмірами продукція спеціалізованих фірм, які виготовляють блоки живлення для комп'ютерів. Але ряд корпусів (особливо нетрадиційного дизайну) можу мати спеціально розроблений блок живлення, призначений тільки для даного корпусу.

Рекомендується купувати корпус із вже встановленим блоком живлення, оскільки в даному випадку фірма-виробник гарантує добре охолодження комплектуючих. Проте при модернізації ПК можуть зрости вимоги до його потужності, і як наслідок – його заміна. Для рядового користувача нині випускаються блоки живлення потужністю 350, 400, 430, 600, 850 Вт та ін.

На рис. 1.11 представлений типовий блок живлення стандарту ATX.

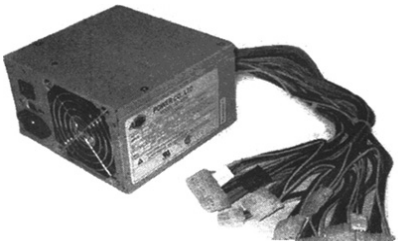


Рисунок 1.11. Блок живлення стандарту ATX

***Материнська плата.*** Головною складовою системного блоку комп'ютера є материнська або системна плата. Материнська плата повинна поєднувати такі важливі якості, як висока продуктивність, функціональність і надійність. Тому, важливо надавати перевагу перевіреним маркам (таким як Intel, ASUSTEK), та уникати дешевих системних плат сумнівного походження.

Чому ж ця плата називається системною? Справа в тому, що на ній сконцентровані майже всі пристрої, що забезпечують працездатність усієї системи (комп'ютера), а саме:

1. Процесор;
2. Оперативний запам'ятовуючий пристрій (ОЗП), що виконаний у виді окремих модулів встановлених у спеціально призначені розєми на системній платі;
3. Постійний запам'ятовуючий пристрій (ПЗП), у якому виробником плати записується програма BIOS (Basis Input Output System – базова система вхідної інформації);
4. Контролер дисководів (інтегрований у системну плату) для гнучких дискет (здатний керувати чотирма дисководами, але зазвичай встановлюється лише один);
5. Контролер жорстких дисків і дисків CD-ROM (інтегрований у системну плату), здатний керувати чотирма жорсткими дисками; (контролер дисководів (жорсткого диску) – пристрій керування процесами запису (читання) інформації на магнітних носіях (з магнітних носіїв) відповідних дисків);
6. Порти (port) – електронні плати, за допомогою яких до комп'ютера приєднуються інші пристрої (зовнішні чи внутрішні). Розрізняють:
   - паралельні (LPT1, LPT2, … – 25 гнізд) – приєднується принтер, сканер. LPT – Line Port Terminal.
   - послідовні (COM1, COM2… – 9 чи 25 штирів) – приєднується миша, модем. COM – комунікаційний.
   - USB (Universal Serial Bus – універсальна послідовна шина) – для під'єднання найновіших принтерів і сканерів, електронних фотоапаратів тощо.
   - IrDA (Infrared Data Communications) – інфрачервоні порти для дистанційного управління. Існують варіанти дистанційного підключення за допомогою радіосигналу.
7. Плата графічного адаптера (відеокарта), що може бути в одному з трьох виконань – AGP, PCI або ISA. Графічний адаптер перетворює цифрову інформацію в зображення, звичне для сприйняття людським оком (символи алфавіту, креслення, малюнки, фотографії і т.д.);

8. Плата звукової карти типу PCI або ISA;
9. Контролер сканера типу PCI або ISA.

Крім перерахованих елементів материнської плати, є й інші, але відомості про них становлять інтерес не для користувача, а для фахівців, що здійснюють ремонтом материнських плат.

***Пам'ять*** – функціональна частина комп'ютера, призначена для запису, зберігання і видачі інформації. Види пам'яті комп'ютера представлені на рисунку 1.12.

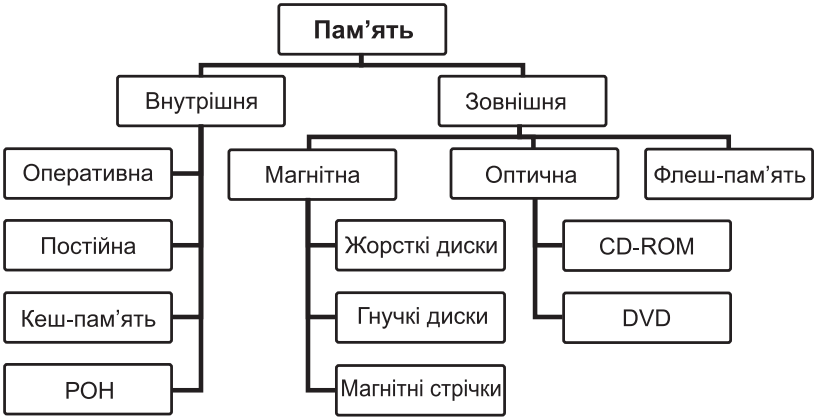



Рисунок 1.12. Види пам'яті комп'ютера

З огляду на те, що основна частина користувачів не має спеціальних знань в галузі радіоелектроніки й обчислювальної техніки, обсяг і характер знань повинні відповідати споживчим запитам користувача. Тому, говорячи про перераховані елементи тут і далі, якщо знадобиться знову звернутися до технічних характеристик яких-небудь інших пристроїв із наведених комп'ютерних конфігурацій, характеризувати їх будемо саме під цим кутом зору.

1. **Процесор**. Утилітарне уявлення про сутність процесора ви вже почерпнули. Виконано процесор у виді спеціалізованої мікросхеми, яку користувач може самостійно змінити на своїй материнській платі. Справа в тому, що материнські плати випускаються не під один вид процесор, а можуть підтримувати кілька їх різновидів, як різних виробників (Intel Pentium®, AMD, Cyrix, IBM), так і з різними швидкісними характеристиками. Які саме процесори підтримує конкретна материнська плата, вказується в її описі. Чим досконаліший процесор, тим вище, природно, його вартість. Причому істотна частина вартості всього блоку процесора приходиться саме на нього. Якщо користувач, купуючи комп'ютер, трохи обмежений у коштах, вигідніше придбати більш продуктивнішу материнську плату з

більш дешевим процесором, згодом замінивши на більш продуктивний, доплативши лише різницю в їхній вартості.

Таблиця 1.2

**Порівняльні характеристики поколінь процесорів**

| Тип МП | Покоління | Рік випуску | Разрядність шини даних, біт | Тактова частота, МГц |
|---|---|---|---|---|
| i8008 | 1 | 1970 | 4 | 1 |
| i8080 | 2 | 1974 | 8 | 1 |
| i8086, i8088 | 3 | 1978 | 16 | 6 |
| i80286 | 4 | 1984 | 16 | 8–20 |
| i80386 | 5 | 1986 | 32 | 25–40 |
| i80486 | 6 | 1989 | 32 | 25–100 |
| Pentium | 7 | 1993 | 64 | 60–200 |
| Pentium Pro | 7 | 1995 | 64 | 180–200 |
| Pentium II | 7 | 1997 | 64 | 200–400 |
| Pentium III | 7 | 2000 | 64 | 533–1260 |
| Pentium IV | 7 | 2001 | 64 | від 1500 і вище |

2. **ОЗП** *(англійська абревіатура RAM– Random Access Memory, що в перекладі означає – випадковий, або довільний, доступ до пам'яті)*. **Оперативна пам'ять** (RAM – random access memory, ОЗП) – пам'ять, призначена для зберігання оброблюваної інформації (даних) і програм, що управляють процесом обробки інформації. Конструктивно є набором мікросхем, розміщених на одній невеликій платі (модуль, планка). Модуль (модулі) оперативної пам'яті встановлюється у відповідний роз'єм материнської плати, дозволяючи таким чином зв'язуватися з іншими пристроями ПК (рис. 1.13). При визначенні необхідного обсягу пам'яті потрібно в першу чергу керуватися вимогами, пропонованими до неї самим програмним продуктом, яким ви бажаєте використовувати для своєї роботи (програмний продукт для офісу, нелінійний відеомонтаж, виробництво складних математичних розрахунків і т.д.).

Уявлення про те, скільки ОП повинно бути в типовому комп'ютері, безперервно змінюється. В середині 80-х років обсяг пам'яті розміром 1 Мбайт здавалося величезним, на початку 90-х років достатнім вважався об'єм 4 Мбайт, до середини 90-х років він збільшився до 8 Мбайт, а потім і до 16 Мбайт. Сьогодні типовим вважається розмір оперативної пам'яті 2-4 Гбайт.

Опанувавши навички оптимізації використання наявної пам'яті при роботі з конкретним програмним продуктом, ви завжди зможете домогтися такої ж швидкодії, як і за рахунок нарощування обсягу пам'яті без застосування методів оптимізації її використання. Тим самим ви зменшите витрати на досягнення поставленої мети з підвищення швидкодії при роботі з об-

раним програмним продуктом. При придбанні модулів пам'яті (рис. 1.13) необхідно також дотримуватися рекомендацій з їх типу (SIMM, DIMM тощо), що наводяться в описі до материнської плати. Крім того, необхідно у фахівців, мова про які йшла вище, проконсультуватися, чи відповідають швидкісні характеристики модуля пам'яті, що купується, швидкісним характеристикам шини наявної материнської плати.



Рисунок 1.13. Модулі оперативної пам'яті

Конструктивні модулі пам'яті мають два виконання – однорядні (SIMM-модулі) і дворядні (DIMM-модулі). На комп'ютерах з МП Pentium однорядні модулі можна застосовувати тільки парами (кількість роз'ємів для їх установки на материнській платі завжди парна), а DIMM-модулі можна встановлювати по одному. Багато моделей материнських плат мають роз'єми як того, так і іншого типу, але комбінувати їх на одній платі не можна.

Основними характеристиками модулів ОП являються об'єм пам'яті і час доступу. SIMM-модулі поставляються об'ємами 4, 8, 16, 32 Мбайт, а DIMM-модулі – 16, 32, 64, 128, 256 Мбайт і більше. Час доступу показує, скільки часу необхідно для звернення до елементів пам'яті – чим воно менше, тим краще. Час доступу вимірюється в мільярдних частках секунди (наносекундах – нс). Типовий час доступу до ОП для SIMM-модулей – 50-70 нс. Для сучасних DIMM-модулей – 7-10 нс.

3. **ПЗП з записаною в ньому програмою BIOS**. У комп'ютерах старих зразків (286, 386, 486) застосовувалися ПЗП, перезаписати в який програму можна було тільки за допомогою спеціального пристрою, так званого програматора. Це обумовлено тим, що запис програми в ПЗП можна було зробити тільки за умови підключення ПЗП до джерела живлення з підвищеною напругою (наприклад, 24 В). Таке джерело напруги мали програматори. На материнську плату комп'ютера подається кілька напруг живлення, найбільша – 12 В. Тому, ПЗП цього типу, встановлене на материнську плату, забезпечувало тільки читання програми з нього, а запис при таких напругах зробити було неможливо. У сучасних комп'ютерах почали застосовувати ПЗП, що давали змогу перепрограмувати в умовах материнської плати за допомогою програми PFLASH.EXE, яка додається на дискеті або компакт диску в комплекті з материнською платою. На цій

же дискеті записано кілька версій (варіантів) BIOS. Тобто, користувач сам може вибрати, з яким варіантом BIOS він буде працювати. Однак робити це не рекомендується. Надайте процедуру перезапису BIOS виконувати тільки фахівцям сервісних центрів з ремонту й обслуговуванню комп'ютерів. Застосування таких типів ПЗП призвело до того, що відтепер і програма BIOS стала доступною для зараження комп'ютерним вірусом.

У цьому плані важко зрозуміти розроблювачів материнських плат, що мають прекрасне уявлення про серйозний збиток який може бути нанесений комп'ютерними вірусами роботі користувача, про широке поширення могутніх вірусних програм, але, незважаючи на це, неначебто б свідомо підклали користувачам додаткову міну уповільненої дії у виді програми, що відтепер уражається, BIOS. Досить згадати, яких неприємностей завдав користувачам вірус "Чорнобиль", що активізується 26 числа кожного місяця і ліквідує не тільки доступ до жорсткого диску, але в багатьох машин знищує саму програму BIOS.

І ще одне нагадування з приводу програми BIOS. Уважно ознайомтеся з описом материнської плати вашого блоку процесора, а саме з розділом, що стосується BIOS, застосованої для вашої материнської плати[15] і постарайтеся усвідомити, з якою метою розробники програми внесли інформацію про неї в цей опис. Якщо після прочитання цього розділу ви самостійно ні до якого висновку прийти не змогли, зверніться за допомогою до фахівців сервісного центра. Вони вам підкажуть, які висновки ви повинні зробити для себе з почерпнутої інформації.

У цьому ж ПЗП зберігається ще одна службова програма – SETUP, або CMOS SETUP.

4. **Графічний адаптер** *(відеокарта)*. Вимоги до відеокарти визначаються переліком задач, що стоять перед користувачем. Якщо комп'ютер використовується лише для виготовлення текстових документів (договорів, контрактів тощо), вимоги до цієї плати можуть бути знижені. У цьому випадку ви орієнтуєтеся тільки на вимоги до неї з боку програмного продукту, а саме – версії Windows, що ви для себе оберете. Якщо ж комп'ютер буде використовуватися дизайнерами, працівниками рекламних агентств і інших користувачів, для яких основним видом документів є високоякісні графічні документи, вимоги до відеокарти різко зростають. Навіть комп'ютер у таких випадках називають графічною станцією, для якої потрібно підбирати не тільки високопрофесійну (а виходить, і дорогу) відеокарту, але і більш продуктивну материнську плату.

5. **Звукова карта**. Чим вище клас карти, тим краще якість відтвореного звуку, а за якість, як відомо, треба платити. Тому вибір типу карти визначається тільки відношенням користувача до якості звучання музичних добутків. Яка карта найкраще задовольнить вас, вам знов-таки підкаже тільки фахівець або ви визначитеся методом прослуховування того самого музичного твору при використанні різних звукових карт.

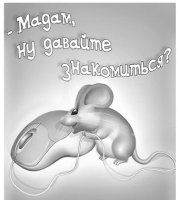
6. **Блок живлення**. Донедавна блоки живлення відрізнялися лише вихідною потужністю (150, 200, 250, 300 Вт). Вибір блоку живлення зводився лише до того, щоб його вихідна потужність відповідала споживаній потужності устаткування в обраній вами конфігурації. Ці блоки живлення мали стандартний набір вихідних напруг – +5, -5, +12, -12В і сигнал готовності блоку до роботи, що вироблявся через якийсь час після його включення (після того як у блоці живлення, у його стабілізаторах напруг, закінчаться перехідні процеси і на їхніх виходах установляться стабільні напруги з перерахованими вище значеннями. Цей сигнал дозволяв подавати зазначені напруги у відповідні ланцюги материнської плати).

Зараз у зв'язку із широким впровадженням пристроїв розширеного керування живленням комп'ютера, що забезпечує програмне відключення багатьох його вузлів і пристроїв (монітора, жорсткого диска й інших енергоємних пристроїв), якщо користувач протягом установленого ним часу не доторкається ні до клавіатури, ні до миші, що забезпечує різке зниження споживання електроенергії комп'ютером від електромережі, почали застосовуватися блоки живлення нового типу (блоки живлення ATX). У таких блоках живлення є додаткове джерело вихідної напруги +5 VSB (напруга живлення в режимі простою), розраховане на споживання струму 300 mA. Якщо материнська плата передбачена для роботи з таким блоком живлення, на ній установлюється спеціальний перемикач-перемичка, що дозволяє відключити це додаткове джерело або включити його, що дозволяє відключити або включити "пробуджуючу" функцію клавіатури. Якщо ви хочете використовувати клавіатуру для "пробудження" комп'ютера (це робиться натисканням клавіші пробілу), установите перемичку в положення "включене". Оскільки не у всіх комп'ютерів є відповідний блок живлення ATX, то за замовчуванням перемичка встановлена в положення "виключене". Якщо установити її в положення "включено" на комп'ютері, що не має потрібного блоку живлення ATX, комп'ютер працювати не буде. Розташування цієї перемички на материнській платі і способи її переключення приводяться в описі материнської плати.

7. ***Жорсткий диск (HDD).*** Якщо комп'ютер не використовується як графічна станція або для нелінійного відеомонтажу, жорсткий диск краще вибирати із шиною IDE, обсягом від 500 Гб, що значно дешевше своїх побратимів.

Усі пристрої, що підключаються до блоку процесора, відносяться до периферійних пристроїв (монітор, клавіатура, миша, принтер і ін.).

**День народження комп'ютерної миші**




9 грудня 1968 року була придумана і зроблена перша комп'ютерна миша. Перший маніпулятор, винайдений Дугласом Энгельбартом в Стенфордском дослідницькому інституті, був дерев'яним і складався з двох перпендикулярних коліс, виступаючих з корпусу пристрою. Назва "мишу" маніпулятор отримав із-за схожості сигнального дроту з хвостом однойменного гризуна.

1.2.2.3 Пристрої введення-виведення інформації

**Пристрої введення інформації.**

**Пристрої введення** – пристрої, що забезпечують введення в комп'ютер даних в різних формах: чисел, текстів, зображень, звуків.

Один із способів класифікації пристроїв введення наведений на рис. 1.14.

**Клавіатура і миша.**

Стандартним пристроєм для введення інформації в комп'ютер є клавіатура. Інформація, що зазвичай вводиться з клавіатури, в цілях контролю відображується на екрані монітора.

Як правило, використовується 101-103-клавишная клавіатура американського стандарту.

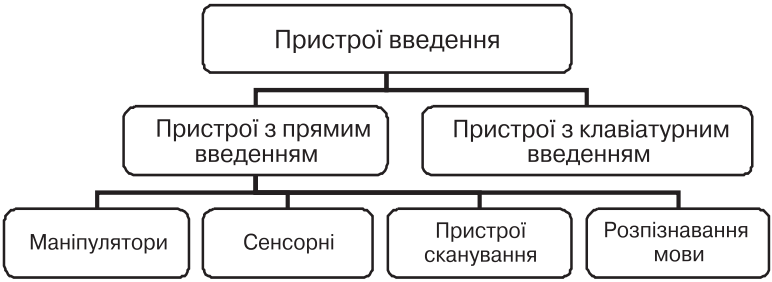



Рисунок 1.14. Класифікація пристроїв введення

Клавіатури розрізняються механізмом дії і розділяються на два типи – механічні і плівкові. Механічні клавіатури мають механічний перемикач під кожною клавішею, а плівкові – під клавішами прокладено дві великі плівки з електропровідними доріжками з фольги і натисненням клавіші замикається відповідна ділянка плівки.

Якщо розглядати клавіатури з точки зору надійності і ціни, то механічні надійніше і відповідно коштують дорожче.

Розрізняють клавіатури і по роз'єму з'єднання. Основні варіанти підключення до системного блоку є порти PS/2 і USB.

За формою більшість клавіатур однакові. Існують і так звані "ергономічні" варіанти (рис. 1.15).

Основне завдання ергономічної клавіатури: понизити стомлюваність м'язів рук при тривалій роботі за комп'ютером, а також звести до мінімуму ризик виникнення професійних захворювань.

Існують моделі, у яких відсутні сполучні кабелі. Принцип дії у таких клавіатур заснований на застосуванні радіосигналу чи інфрачервоного випромінювання.



Рисунок 1.15. Ергономічна клавіатура

Для власників кишенькових ПК і ноутбуків розроблені спеціальні гнучкі силіконові клавіатури. Матеріал легкий і міцний, завдяки чому її зручно транспортувати (можна просто згорнути в рулон).

Існує спеціальний компактний пристрій, який підключається до ноутбука або кишеньковому ПК через Bluetooth і проектує за допомогою лазера на рівну поверхню віртуальну клавіатуру. При натисненні на "клавішу" видається характерний сигнал.

Широке використання графічного інтерфейсу привело до появи маніпулятора **"мишу"** (рис. 1.16).



Рисунок 1.16. Перша мишка

За способом прочитування інформації їх можна класифікувати на:

- ✧ механічні;
- ✧ оптико-механічні;
- ✧ оптичні.

У механічних мишах робочим органом є металева куля, покрита гумою. У миші вона обертається при переміщенні її корпусу по горизонтальній поверхні. Обертання кулі передається двом пластмасовим валам, положення яких з великою точністю прочитується інфрачервоними фотопарами (парами "світловипромінювач – фотоприймач") і потім перетвориться в електричний сигнал, що управляє рухом покажчика миші на екрані.

Миша містить дві або три клавіші і датчики переміщення в двох взаємно перпендикулярних напрямах. Є миші з додатковою кнопкою, яка розташовується посередині основних кнопок. Вона призначена для прокрутки вгору або вниз зображення і тексту. Миша працює під управлінням

спеціальної програми – драйвера, наприклад mouse.com, і підключається або до послідовного порту комп'ютера, або до спеціального роз'єму PS/2, або до USB-порту.

Сучасні оптичні миші розроблені на базі мікросхеми, що містить фотосенсор і МП обробки зображення. Здешевлення і мініатюризація комп'ютерної техніки дозволили розмістити усе це в одній мікросхемі. Фотосенсор періодично сканує ділянку робочої поверхні під мишею. При зміні малюнка МП визначає, в яку сторону і на яку відстань змістилася миша. Сканована ділянка підсвічує світлодіодом (зазвичай – червоного кольору) під косим кутом.

Нині використовуються безпровідні миші. Вони використовують радио- та інфрачервоний сигнали. Перевага радіотехнології полягає в тому, що миша і приймач сигналу не зобов'язані знаходитися в зоні прямої видимості один одного.

Безпровідні миші трохи важче звичайних (рука втомлюється вже через декілька годин) і мають не найстійкіший зв'язок, що особливо погіршується при розряді акумуляторів.

**Сканери**

Сканером називається пристрій для введення в комп'ютер графічної інформації: фотографій, малюнків, слайдів, а також текстових документів. У ньому яскравість (чи колірний відтінок) кожної точки документу перетвориться в цифровий код, при цьому формується точковий графічний образ сторінки. Сканер виключає стомливу процедуру введення тексту за допомогою клавіатури і формування малюнка за допомогою миші. Отриману копію зображення можна редагувати: змінювати масштаб, додавати і видаляти деталі, змінювати колір і так далі. Електронну копію зображення можна тривалий час зберігати на магнітному або оптичному носію.

Відповідно до конструктивного виконання сканери бувають (рис 1.17):

- ✧ ручні;
- ✧ планшетні;
- ✧ барабанні;
- ✧ проекційні;
- ✧ та ін.

Сканери характеризуються:

- ✧ роздільною здатністю;
- ✧ за глибиною кольорів;
- ✧ максимальним форматом сканованого документу;
- ✧ швидкодією і способом підключення.

*Роздільна здатність* – це кількість точок, які сканер може розрізнити на відрізку одиничної довжини. Вимірюється в одиницях dpi. Символи dpi означають число точок на дюйм (dot per inch). Існує дві величини дозволу: в горизонтальному і вертикальному напрямах.





Планшетний сканер Ручний сканер Барабанний сканер

Рисунок 1.17. Види сканерів

*Глибина кольору* – це число розрядів, що відводяться для кодування кольору кожної точки. Вимірюється в бітах. Від цього залежить кількість відтінків, які можна закодувати двійковим числом відповідної розрядності.

Швидкість процесу сканування залежить від великої кількості чинників. Мають значення характеристики механізмів сканера і характеристики комп'ютера, до якого він підєднаний. Тому дана швидкість зазвичай вимірюється емпірично.

Отримана від сканера цифрова інформація може оброблюватись як графічний образ, або перетворюватися в текст.

Використання сканера поєднується з системами розпізнавання образів типу OCR (Optical Character Recognition). Система OCR розпізнає лічені сканером з документу мозаїчні портрети символів (букв, цифр, розділових знаків) і перетворить їх в байти відповідно до кодової таблиці. За рахунок системи OCR можна прочитувати машинописний і рукописний тексти. Але, в останньому випадку використовуються складні алгоритми розпізнавання образів, засновані на теорії штучного інтелекту.

**Інші пристрої введення.**

До пристрою введення відноситься і **TouchPad (тачпад)**, що є прямокутною панеллю, чутливою до натиснення пальців. Доторкнувшись пальцем до поверхні тачпада і переміщаючи його, користувач маневрує курсором так само, як і при використанні миші.

У іграх часто використовується **джойстик** – важіль, за допомогою якого можна направляти, наприклад літак, управо, вліво, вгору, вниз (рис.1.18).

При виборі предметів (наприклад в магазині) людина іноді показує на потрібний об'єкт пальцем. Саме таким чином вводиться інформація в ЕОМ за допомогою сенсорних екранів (СЕ). **Се́нсорний екран** – пристрій введення-виведення інформації, що є екраном який реагує на дотик до нього.

За принципом дії СЕ розділяються на ультразвукові, фотоелектричні, резистивні і місткості екрани. Головне завдання СЕ полягає у визначенні координати дотику пальця до екрану. Визначивши координату, далі можна за допомогою меню управляти роботою ЕОМ.



Рисунок 1.18. Джойстик

У ультразвукових СЕ на краях екрану розміщуються ультразвукові перетворювачі (датчики), які створюють на поверхні екрану акустичні хвилі. Ультразвукові коливання розходяться по склу монітора подібно до кіл на воді. Ультразвукові перетворювачі одночасно виконують функції передавача і приймача акустичних хвиль. Час проходження від передавача до приймача постійний, якщо акустична хвиля не натрапляє на який-небудь обурюючий об'єкт (палець). Точку дотику можна досить точно визначити методом ехолокації, шляхом виміру часу приходу відбитих хвиль. Аналогічно в аеропорту радіолокатор визначає відстань до літака.

**Цифрові (графічні) планшети – диджитайзери** (рис.1.19) забезпечують перенесення зображення з аркуша паперу, що накладається, в ЕОМ за допомогою переміщення по планшету спеціального покажчика. Диджитайзеры дозволяють створювати креслення відразу в електронному вигляді. Робота з графічним планшетом аналогічна малюванню олівцем. Особливо вони зручні для формування штрихових малюнків і креслень.




Рисунок 1.19. Графический планшет

Введення об'ємних зображень (будівель, автомобілів і так далі) в ЕОМ здійснюється за допомогою цифрових камер. **Цифрові камери** (відеокамери і фотоапарати) дозволяють отримувати відеозображення і фотознімки безпосередньо в цифровому коді. Цифрові відеокамери можуть бути постійно підключені до комп'ютера і забезпечувати запис відеозображення на жорсткий диск або його передачу комп'ютерними мережами.

**Цифрові фотоапарати** дозволяють отримувати високоякісні фотографії, для зберігання яких використовуються спеціальні модулі пам'яті або жорсткі диски дуже маленького розміру. Запис зображення на жорсткий диск комп'ютера може здійснюватися за допомогою підключення камери до USB порту комп'ютера.

При встановленні в комп'ютер спеціальної плати – **ТБ-тюнера** і підключення його до входу телевізійної антени можна переглядати телепередачі безпосередньо на комп'ютері.

В майбутньому роботою ЕОМ управлятимуть переважно голосом, за допомогою мікрофону. Для введення звукової інформації використовується мікрофон, який підключається до входу звукової карти.

**Пристрої виведення інформації.**

Пристрої виводу – пристрої перетворення комп'ютерного (машинного) представлення інформації у форму, зрозумілу людині. Класифікація пристроїв виводу представлена на рисунку 1.20.

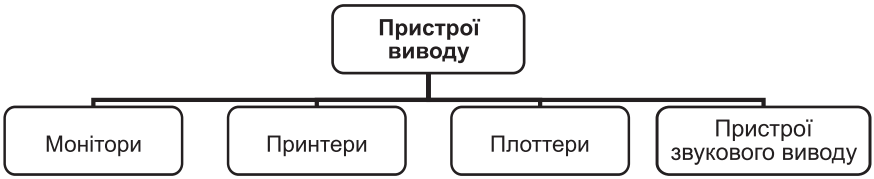



Рисунок 1.20. Класифікація пристроїв виводу

**Монітори**

Монітор (Monitor) – універсальний пристрій візуального відображення усіх видів інформації, що складається з дисплея (display) і пристроїв, призначених для виводу текстової, графічної і відеоінформації на дисплей.

За кольором дисплеї поділяються на: монохромні (чорно-білі) і кольорові (до декількох мільйонів кольорів).

Класифікація моніторів за видом виведеної інформації:

– алфавітно-цифрові:

дисплеї, що відображають тільки алфавітно-цифрову інформацію;

дисплеї, що відображають псевдографічні символи;

– інтелектуальні дисплеї:
  що мають редакторські можливості,
  що здійснюють попередню обробку даних;

графічні:
  векторні;
  растрові.

Класифікація моніторів за будовою:

- ЕПТ – на основі електронно-променевої трубки (рис. 1.21);
- РК – рідкокристалічний монітор (рис. 1.22);
- плазмовий – на основі плазмової панелі;
- проекційний – відеопроектор і екран, розміщені окремо або об'єднані в одному корпусі (як варіант – через дзеркало або систему дзеркал);
- OLED-монитор – на технології OLED – органічний світлоізлучаючий діод;
- віртуальний ретинальный монітор – технологія пристроїв виводу, що формує зображення безпосередньо на сітківці ока.



Рисунок 1.21. ЕПТ-монітор



Рисунок 1.22. TFT-монитор

**Відеомонітори на базі ЕПТ.**

До складу монітора входять:

- електронно-променева трубка;
- блок розгорток;
- відеопідсилювач;
- блок живлення і так далі.

Електронно-променева трубка (ЕПТ, CRT, Cathode Ray Tube, катодно-променева трубка) є запаяною вакуумною скляною колбою, дно (екран) якої покрите шаром люмінофора, а в горловині встановлена електронна гармата, що випускає потік електронів. За допомогою системи, яка формує і відхиляє потік електронів, він модулюється для відображення потрібного символу і спрямовується на потрібне місце екрану. Енергія, що виділяється електронами, які потрапляють на люмінофор, примушує його світитися.

Точки люмінофора, що світяться, формують зображення, яке сприймається візуально. У комп'ютерах застосовуються як монохромні, так і кольорові монітори.

**Відеомонітори на плоских панелях**

Останні забезпечують набагато нижчий рівень шкідливого випромінювання і сприяють меншій втомлюваності очей.

Відеомонітори на плоских панелях (ВМПП) дуже різноманітні. Зараз застосовуються:

- ✧ монітори на рідкокристалічних індикаторах (LCD – Liquid Crystal Display);
- ✧ плазмові монітори (PDP – Plasma Display Panels);
- ✧ електролюмінесцентні монітори (FED – Field Emission Display);
- ✧ світловипрямінювальні монітори (LEP – Light Emitting Polymer).

*Монітори на рідкокристалічних індикаторах (РКІ, LCD – Liquid Crystal Display)* – це цифрові плоскі монітори. Дані монітори використовують спеціальну прозору рідину, яка при певних напругах електростатичного поля кристалізується, при цьому змінюються її прозорість, коефіцієнти поляризації і заломлення світлових променів. Ці ефекти і використовуються для формування зображення.

*У плазмових моніторах* (PDP – Plasma Display Panels) формування зображення супроводжується випромінюванням світла газовими розрядами в пікселах панелі.

*Електролюмінесцентні монітори* (FED – Field Emission Display) як панель використовують дві тонкі скляні пластини з нанесеними на них прозорими дротами. Один з цих пластинів покритий шаром люмінофора. Пластини складаються так, що їх дроти перетинаються, утворюючи сітку. Між пересічними дротами утворюються піксели. На пару пересічних дротів подається напруга, що створює електричне поле, достатнє для збудження світіння люмінофора в пікселі.

*У світловипрямінюваних моніторах* (LEP – Light Emitting Polymer) використовується в якості панелі напівпровідникова полімерна пластина, елементи якої під дією електричного струму починають світитися. Конструкція панелі приблизно така ж, як панелі FED, але через напівпровідникові піксели пластини пропускається струм (а не створюється електричне поле).

Основні характеристики монітора:

- ✧ розмір екрану за діагоналлю (Screen Size) – від 9" дюймів (23 см) до 42" (106 см) і більше.
- ✧ роздільна здатність (Resolution) – максимальна кількість точок (пікселів) зображення. Наприклад, якщо роздільна здатність дисплея 1280*1024, то це значить, що на екран виводиться 1024

рядків по 1280 пікселів у кожному. Відстань між пікселями (dot pitch) називається зернистістю і обчислюється в долях міліметра.

- ✧ частота кадрової розгортки (Refresh Rate) (регенерації) – швидкість оновлення зображення. Чим вища частота, тим чіткіше і стійкіше зображення, і тим менше втомлюються очі користувача. При частоті регенерації 60 Гц миготіння екрану видно неозброєним оком. Нормальною вважається 85 Гц, а комфортною – 100 Гц і більше.

Керує роботою монітора спеціальна драйвер-програма – **відеоадаптер**. Типи відеоадаптерів: старі – EGA (Enhanced Graphics Adapter – удосконалений графічний адаптер), VGA (Video Graphics Adapter – відеографічний адаптер), нові – SVGA (Super VGA).

Для забезпечення відповідної роздільної здатності та інших параметрів виділяється спеціальна відеопам'ять. Так, наприклад, відеоадаптер з пам'яттю 2 Мб може формувати зображення з 65536 кольорів при роздільній здатності 1024*768 пікселів.

Для прискорення вводу складних зображень використовуються спеціальні прискорювачі – графічні акселератори. Для ігор з 3-х-вимірною графікою використовуються тривимірні 3D (3 dimension) акселератори.

З метою забезпечення режиму енергозбереження в сучасних моніторах передбачено різні режими роботи: робочий режим, очікування, тимчасове відключення і повне відключення.

У зв'язку з наявністю шкідливого випромінювання для безпечної роботи потрібно використовувати дисплеї з позначкою LR (Low Radiation – низьке випромінювання) або ті, які оснащені захисними екранами.

**Принтери**

Процес друку називається вивід на друк, а документ, що вийшов, – роздруковка або тверда копія.

**Принтер (printer)** – пристрій, призначений для виведення текстової і графічної інформації на твердий носій, в основному – на папір. Для зменшення завантаженості комп'ютера, під управлінням якого вони працюють, принтери мають власний вузькоспеціалізований МП і ОП (буфер), в яку поміщається повністю або частково інформація, що виводиться на друк.

Принтери класифікуються за п'ятьмп основними позиціями:

- ✧ принцип роботи друкуючого механізму;
- ✧ максимальний формат аркуша паперу;
- ✧ використання кольорового друку;
- ✧ за рекомендованим місячним навантаженням, яке взаємозв'язане із швидкістю друку.

За принципом дії принтери діляться на:

- ✧ матричні;

- ✧ струминні;
- ✧ лазерні;
- ✧ інші.

**Матричні принтери (dot matrix printer)** – є принтерами ударної дії, друкують за допомогою спеціальних голок (від 9 шт. до 48 шт.), що вибивають символи на папері через спеціальну фарбовану стрічку. Можуть друкувати одночасно декілька екземплярів (під копірку). Друкуюча голівка матричного принтера складається з вертикального стовпця маленьких стержнів, які під впливом магнітного поля "виштовхуються" з голівки і ударяють в папір через фарбувальну стрічку (рис. 1.23). Переміщаючись, друкуюча голівка залишає на папері рядок символів. Недоліками матричних принтерів є повільний друк, багато шуму при друку та якість самого друку. Переваги матричних принтерів визначаються здатністю працювати з будь-яким папером і низькою вартістю друку. Зараз матричні принтери використовуються тоді, коли важливий сам відтиск, приклад – роздрукування квитків на потяг.

У **струминних принтерах** (рис. 1.24) (***струменеві* jet printer**) – друкують за допомогою спеціальної фарби, яка під тиском викидається через мініатюрні сопла (від 16 до кількох сотень сопел), діаметри яких тонші за людський волос. Переміщаючись уздовж паперу, друкуюча голівка залишає рядок символів або смужку зображення. Передбачається кольоровий друк. Недорогі порівняно з іншими типами, але для друкування з високою роздільною здатністю вимагають папір із спеціальним полімерним покриттям. Недолік – засихання фарби в соплах при несистематичному друкуванні.

Відомі декілька принципів дії струминних друкуючих голівок.

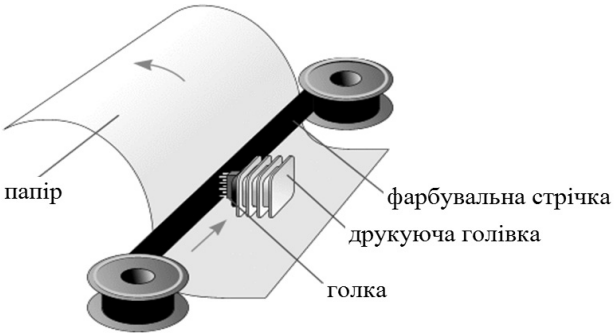



Рисунок 1.23. Принцип роботи матричного принтера

У одній з конструкцій на вхідному кінці кожного сопла розташований маленький резервуар з чорнилом. Позаду резервуару розташовується на-

грівач (тонкоплівковий резистор). Коли резистор нагрівається струмом, що проходить через нього, до температури 500°С, навколишнє його чорнило скіпає, утворюючи бульбашку пари. Ця бульбашка, що розширюється, виштовхує з сопла краплі чорнила діаметром 50-85 мкм зі швидкістю близько 700 км/год.

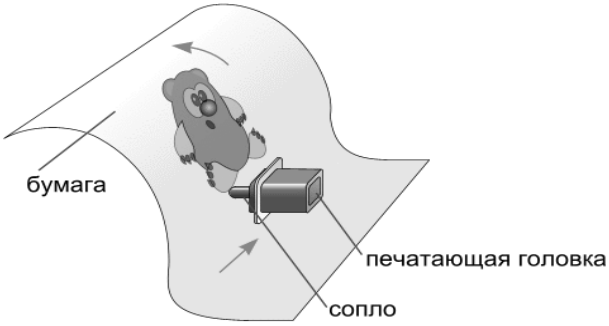



Рисунок 1.24. Принцип роботи струминного принтера

У іншій конструкції друкуючої голівки джерелом тиску служить мембрана, що приводиться в рух п'єзоелектричним способом.

У усіх конструкціях принтерів електромеханічні пристрої переміщають друкуючі голівки і папір так, щоб друк відбувався в потрібному місці.

Струминні принтери друкують досить швидко, чинять мало шуму. Якість друку визначається роздільною здатністю струминних принтерів, яка складає 600 dpi і вище. Проте, вони дуже вимогливі до паперу. На папері низької якості чорнило розпливається. Також отримане зображення чутливе до дії вологи.

У **лазерних принтерах** (**laser printer**) (рис. 1.25) друк відбувається за допомогою нанесення порошку (тонера) і закріплення його лазерним променем. Найбільш швидкодіючі, забезпечують найкращу якість друку, але їх вартість вища, ніж матричних чи струменевих. При друку чорно-білих і кольорових зображень використовується електрографічний принцип створення зображення. Процес друку включає створення невидимого рельєфу електростатичного потенціалу в шарі напівпровідника з наступною його візуалізацією. Візуалізація здійснюється за допомогою часток сухого порошку – тонера, що наноситься на папір. Тонер є шматочками заліза, покритими пластиком. Найбільш важливими частинами лазерного принтера є напівпровідниковий барабан, лазер і прецизійна оптико-механична система, що переміщає промінь.

Лазер генерує тонкий світловий промінь, який, відбиваючись від дзеркала, що обертається, формує електронне зображення на світлочутливому напівпровідниковому барабані.

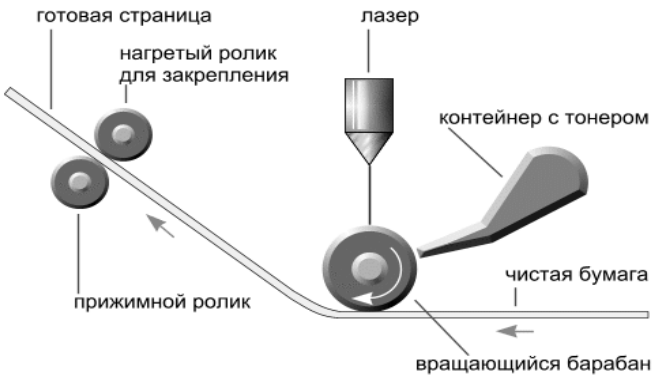



Рисунок 1.25. Принцип роботи лазерного принтера

Поверхні барабана заздалегідь повідомляється статичний заряд. Для отримання зображення на барабані лазер повинен вмикатися і вимикатися, що забезпене схемою управління. Дзеркало, що обертається, необхідне для розвороту променя лазера в рядок, що формується на поверхні барабана. Поворот барабана на новий рядок здійснює позиційний кроковий двигун. Процес розгортки зображення на барабані багато в чому нагадує побудову зображення на екрані монітора (створення растру).

Коли промінь лазера потрапляє на заздалегідь заряджений барабан, заряд "стікає" з освітленої поверхні. Освітлені і неосвітлені лазером ділянки барабана мають різний заряд. В результаті сканування усієї поверхні напівпровідникового барабана на ньому створюється приховане (електронне, не видиме для людини) зображення.

На наступному етапі роботи принтера відбувається прояв зображення, тобто перетворення прихованого електронного зображення на видиме зображення. Заряджені частки тонера притягуються тільки до тих місць барабана, які мають протилежний заряд по відношенню до заряду тонера.

Коли видиме зображення на барабані побудоване і він покритий тонером відповідно до оригіналу, аркуш паперу, що подається, заряджається таким чином, що тонер з барабана притягується до паперу. Прилипший порошок закріплюється на папері за рахунок нагріву часток тонера до температури плавлення. В результаті цих операцій формується водотривкий відбиток.

Лазерні принтери забезпечують практично безшумний друк, високу швидкість друку, який досягається посторінковим друком. Сторінка друкується цілком. Роздільна здатність лазерних принтерів досягає 1200 dpi і вище.

Окрім лазерних принтерів існують так звані LED-принтери (Light Emitting Diode), які отримали свою назву через те, що напівпровідниковий лазер в них замінений "гребінкою" (лінійкою) світлодіодів. Зображення одного рядка на напівпровідниковому барабані формується одночасно.

Кольорові лазерні принтери поки не ідеальні. Для отримання кольорового зображення з якістю, близькою до фотографічного, використовують термічні принтери, або, як їх ще називають, кольорові принтери високого класу. Нині поширення набули три технології кольорового термодруку: струминне перенесення розплавленого барвника (термопластичний друк); контактне перенесення розплавленого барвника (термовосковая друк); термоперенос барвника (друк сублімації).

Швидкість друку термічних принтерів внаслідок інерційності теплових ефектів невисока. Для принтерів-сублімацій від 0,1 до 0,8 сторінок в хвилину, а для термовосковых – 0,5-4 сторінки в хвилину.

Також отримали поширення багатофункціональні пристрої (БФП), в яких в одному приладі об'єднані принтер, сканер, копир і факс. Таке об'єднання раціональне технічно і зручне в роботі.

Нові модифікації лазерних, струминних і термосублімаціонних технологій друку дають гарні результати і відносяться до комбінованих.

Окремо слід виділити портативні принтери, які обслуговують ноутбуки. Очевидно, за швидкодією, якістю друку та економічними показниками вони поступаються стаціонарним принтерам.

**3D-принтер –** пристрій, що використовує метод пошарового створення фізичного об'єкту за цифровою 3D-моделлю.

3D-друк може здійснюватися різними способами і з використанням різних матеріалів, але в основі будь-якого з них лежить принцип пошарового створення (вирощування) твердого об'єкту.

Застосовуються дві технології формування шарів:

**Лазерна**

1. *Лазерна стереолітографія* – ультрафіолетовий лазер поступово, піксель за пікселем, засвічує рідкий фотополімер, або фотополімер засвічується ультрафіолетовою лампою через фотошаблон, що змінюється з новим шаром. При цьому рідкий полімер твердне і перетворюється на досить міцний пластик.
2. *Лазерний сплав* – melting – при цьому лазер сплавляє порошок з металу або пластика, шар за шаром, в контур майбутньої деталі.
3. *Ламінування* – деталь створюється з великої кількості шарів робочого матеріалу, які поступово накладаються один на одного і склеюються, при цьому лазер вирізує в кожному контурі перерізи майбутньої деталі.

**Струминна**

1. *Застигання матеріалу при охолодженні* – роздаюча голівка видавлює на охолоджувану платформу-основу краплі розігрітого термопластика. Краплі швидко застигають і злипаються один з одним, формуючи шари майбутнього об'єкту.
2. *Полімеризація фотополімерного пластика під дією ультрафіолетової лампи* – спосіб схожий на попередній, але пластик твердне під дією ультрафіолету.
3. *Склеювання або спікання порошкоподібного матеріалу* – схоже на лазерне спікання, тільки порошкова основа (часом на основі подрібненого паперу або целюлози) склеюється рідкою (іноді клеючою) речовиною, що поступає із струминної голівки. При цьому можна відтворити забарвлення деталі, використовуючи речовини різних кольорів. Існують зразки 3D-принтеров, що використовують голівки струминних принтерів.
4. Густі керамічні суміші теж застосовуються в якості самоотверждаемого матеріалу для 3D-друку великих архітектурних моделей.
5. *Біопринтери* – друк 3D-структури майбутнього об'єкту (органу для пересадки) робиться стовбуровими клітинами. Далі ділення, зростання і модифікації клітин забезпечує остаточне формування об'єкту.

**Застосування технології**

- ✧ Для швидкого прототипування, тобто швидкого виготовлення прототипів моделей і об'єктів для подальшого доведення. Вже на етапі проектування можна кардинальним чином змінити конструкцію вузла або об'єкту в цілому. У інженерії такий підхід здатний істотно понизити витрати у виробництві і засвоєнні нової продукції.
- ✧ Для швидкого виробництва – виготовлення готових деталей з матеріалів, що підтриуються 3D-принтерами. Це позитивне рішення для малосерійного виробництва.
- ✧ Виготовлення моделей і форм для ливарного виробництва.
- ✧ Конструкція з прозорого матеріалу дозволяє побачити роботу механізму "зсередини". Такий спосіб, зокрема, був використаний інженерами Porsche при вивченні струму масла в трансмісії автомобіля ще при розробці.
- ✧ Виробництво різних дрібниць в домашніх умовах.
- ✧ Виробництво складних, масивних, міцних і недорогих систем. Наприклад безпілотний літак Polecat[en] компанії Lockheed, більша частина деталей якого було виготовлено методом швидкісного тривимірного друку.

- ✧ Розробки університету Міссурі, що дозволяють наносити на спеціальний био-гель згустки клітин заданого типу. Розвиток цієї технології – вирощування повноцінних органів.
- ✧ У медицині, при протезуванні і виробництві імплантатів (фрагменти скелета, черепи, кісток, хрящові тканини). Ведуться експерименти з друку донорських органів.

*Додатки.* Після створення 3D-моделі використовуються САПР-системи, що підтримують управління 3D-друку. В більшості випадків для друку використовують формат файлу STL. Практично усі принтери мають свій власний софтвер для управління друком, причому частина – комерційні, частина з відкритим початковим кодом. Наприклад, 3D принтер Picaso Builder – програма Polygon, 3DTouch – Axon 2, MakerBot – MakerWare, Ultimaler – Cura.

*Самовідтворення.* Основна стаття: Проект RepRap.

Частково реплицирующийся (здатний відтворити самого себе) тривимірний принтер RepRap версія 2.0 (Мендель).

До недавнього часу вважалися науковою фантастикою 3D-принтеры, здатні відтворювати деталі власної конструкції, тобто відтворювати самі себе. Зараз розробка такої машини ведеться проектом RepRap, на даний момент принтер вже робить більше половини власних деталей. Проект є розробкою із загальнодоступними напрацюваннями і уся інформація про конструкцію поширюється за умовами ліцензії GNU General Public License.

Проект першого в історії недорогого тривимірного принтера, що самовідтворюється (тобто здатного відтворити принаймні частину самого себе), – RepRap активно реалізується в наші дні англійськими конструкторами університету Бата. "Найголовніша особливість RepRap, – як зазначає Адріан Боуэр, один із співробітників проекту RepRap, полягає в тому, що із самого початку він був задуманий як реплицирующаяся система: принтер, який сам себе роздруковує".

**Інші пристрої виводу**

**Плоттером (графічним пристроєм)** називається пристрій для виведення широкоформатної графічної інформації на папір (плакатів, креслень, електричних і електронних схем тощо). Принцип дії плоттерів такий же, як і у струминних принтерів. Принциповою відмінністю плоттера від принтера є здатність наносити безперервні лінії. Плоттери характеризуються максимальним форматом паперу і можливістю перенесення кольорів.

Виведення звукової інформації здійснюється за допомогою акустичних колонок і головних телефонів, які підключаються через **спеціальний адаптер** (**контроллер, звукову плату**).

**Модем –** пристрій для передачі цифрової інформації через телефонні або виділені канали зв'язку. Підключається до комп'ютера через

послідовний порт (зовнішній модем) або включається в роз'єм системної плати (внутрішній модем). За технічними характеристиками зовнішні і внутрішні модеми практично не відрізняються. Модеми обмінюються між собою правилами, набір яких називається протоколом. Найбільш важливими параметром модему являються максимальна швидкість передачі (пропускна спроможність) в бодах (1 бод = 1 біт/с), методи корекції помилок і стискування даних. Реальна швидкість передачі обмежується якістю підключеного каналу зв'язку і може бути істотно менша за максимальну.

**Мережева карта** – пристрій для високошвидкісного міжкомп'ютерного обміну цифровою інформацією на невеликих відстанях, включається в системну плату комп'ютера. Вона пов'язана з аналогічним пристроєм іншого комп'ютера високочастотною лінією. За відсутності в комп'ютері завантажувального диска мережева карта забезпечує завантаження ОС з іншого комп'ютера.

До периферійних пристроїв відносяться також FDD і пристрої для читання CD ROM, хоча вони розташовуються в блоці процесора. Таке їх розташування визначене виходячи з вимог ергономіки **(ергономіка** – наука, що займається розробкою апаратури, органи керування якої призначені для керування ними людиною-оператором, повинні бути сконструйовані з урахуванням фізіологічних особливостей людського організму так, щоб робота з такими органами керування мінімально стомлювала людини протягом робочого дня).

**Дисководи для гнучких дисків (Floppy Disk Drive, FDD)** – пристрої для роботи з дискетами (floppy disks).

*Дискета* – гнучкий пластиковий диск з нанесеним на обидві сторони магнітним покриттям, поміщений в досить твердий пластиковий конверт для оберігання від механічних ушкоджень (рис.1.26).

Інформація на диск наноситься уздовж концентричних кіл (рис. 1.27) – доріжок.

Кожна доріжка розбита на декілька секторів (зазвичай 18) – мінімально можливих адресних ділянок. Стандартна місткість сектора – 512 байт.




Рисунок 1.26. Дискета формата 3,5"

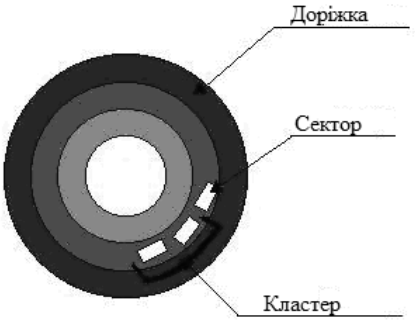



Рисунок 1.27. Логічна структура поверхні магнітного диску

Дискети за розмірами бувають 2-х типів: великі – 5,25 дюймів (вже майже не використовуються) і малі – 3,5 дюйми (1дюйм = 2,54 см). У побуті їх називають відповідно "п'ятидюймовими" і "тридюймовими". Кожна дискета повинна бути певним чином підготовлена до роботи (відформатована). Велика дискета може вмістити від 360 КБ до 1,2 МБ інформації, а мала – від 720 КБ до 1,44 МБ (є спеціальні малі дискети до 2,88).

Дискети використаються, як правило, лише для оперативного зберігання інформації. Навіть фірмові дискети не витримують більш як 20-и разовий перезапис.

Для захисту "п'ятидюймової" дискети від запису на неї даних потрібно заклеїти (але так, щоб можна було при потребі відклеїти) спеціальний проріз. Для такого ж захисту "тридюймової" – відкрити заслінку у віконечку внизу.

Структура дискети подібна до структури жорсткого диску.

При роботі з дискетами необхідно дотримуватись наступних правил:

- ✧ не торкатись робочої поверхні диску;
- ✧ не згинати дискету;
- ✧ не намагатися зняти металеву пластинку, яка захищає дискету від потрапляння на неї пилу чи сміття;
- ✧ зберігати дискети якнайдалі від магнітних полів, що можуть утворюватись поблизу електропристроїв, магнітів, а також нагрівальних пристроїв;
- ✧ перед користуванням дискетою, яка побувала в чужому комп'ютері, обов'язково перевірити її на наявність вірусів;
- ✧ не користуйтесь ненадійними дискетами, тобто такими, які були "трішки зіпсовані, а потім вилікувані".

Не варто також редагувати документ "з дискети", так як не завжди вдається зберегти нову версію документа на цій же дискеті. Це може бути

пов'язано з появою зіпсованих ділянок на дискеті або з великим обсягом редагованого файлу.

**Жорсткий магнітний диск (вінчестер, HDD – Hard Disk Drive) –** накопичувач, призначений для довготривалого зберігання усієї наявної в комп'ютері інформації. ОС, постійно використовувані програми завантажуються з жорсткого диска, на нім зберігаються більшість документів. Жорсткий диск є незмінним носієм.

Жорсткий диск є герметичною коробочкою (рис. 1.28), усередині якої заховано декілька дисків, покритих магнітним шаром. Над ними дуже швидко рухаються декілька голівок читання-запису.



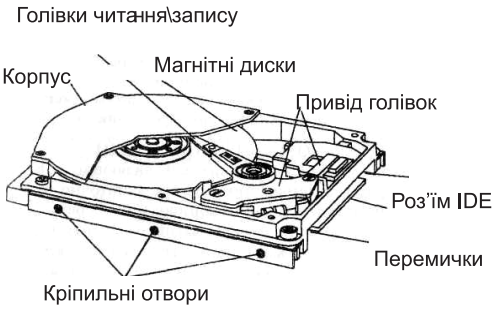



Рисунок 1.28. Жорсткий диск

Основні параметри жорсткого диска.

- Місткість – вінчестер має об'єм від 500 Гб до 2 Тб і більше.
- Швидкість читання даних. Середній показник нині – близько 8 Мбайт/с.
- Середній час доступу. Вимірюється в мілісекундах і характеризує час, необхідний диску для доступу до будь-якої обраної вами ділянки. Середній показник - 9 мс.
- Швидкість обертання диска. Показник, безпосередньо пов'язаний із швидкістю доступу і швидкістю читання даних (7200 об/хв). Швидкість обертання жорсткого диска в основному впливає на скорочення середнього часу доступу (пошуку). Підвищення загальної продуктивності особливе помітно при вибірці великого числа файлів.
- Розмір кеш-пам'яті – швидкій буферній пам'яті невеликого об'єму, в яку комп'ютер поміщає найбільш часто використовувані дані. У вінчестера є своя кеш-пам'ять розміром 32 Мбайт і більше.
- Фірма-виробник. Опанувати сучасні технології можуть тільки найбільші виробники, що пов'язано з великих великими витратами фінансових та інтелектуальних фондів. Нині жорсткі диски ви-

готовляють сім компаній: Fujitsu, IBM - Hitachi, Maxtor, Samsung, Seagate, Toshiba і Western Digital. При цьому кожна модель одного виробника має свої, тільки їй властиві особливості.

**Пристрої для читання CD ROM** удосконалюються з кожним роком. Це удосконалювання йде по шляху збільшення швидкості читання і надійності розпізнавання інформації, що читається. Цікаво відзначити, що в процесі збільшення швидкості читання інформації розроблювачі зіштовхнулися з проблемою зниження вірогідності інформації, що зчитується, з компакт-дисків. В той же час, в малошвидкісних дисководамх ця ж інформація зчитується без помилок. Як вихід із такої ситуації, розроблювачам довелося в сучасні дисководи додати пристрій автоматичного підбору швидкості читання для забезпечення вірогідності зчитуваної інформації. Дисководи з наявністю такого пристрою стали відрізнятися маркіруванням від своїх побратимів, що не мають такого пристрою. Наприклад, якщо на лицьовій панелі дисковода нанесена мітка, що вказує на швидкісні можливості дисковода без слова *max,* такий дисковод має фіксовану швидкість читання, що відповідає зазначеній цифрі. Якщо ж після цифри мається слово *max,* такий дисковод здатний працювати з зазначеною швидкістю читання, але і здатний знижувати цю швидкість, якщо йому запропонують прочитати старий компакт-диск (*32x max).* При виборі дисковода бажано купувати дисководи, що мають пристрій автоматичного підбору швидкості читання інформації, щоб виключити проблеми читання старих компакт-дисків. Крім того, бажано з'ясувати, хто з виробників дисководів користується на ринку авторитетом.

***Дисководи для лазерних компакт-дисків*** – пристрої, які здійснюють зчитування (запис) інформації з компакт-дисків за допомогою лазерної оптики.

Компакт-диски виготовляють з прозорого пластику діаметром 120 мм або 80 мм і товщиною 1,2 мм. На пластикову поверхню напилюють відбиваючий шар алюмінію або золота. Для запису інформації на поверхні диску видавлюють відповідні комбінації впадин – "піти" (від англ. pit – ямка). Від центру до краю на компакт-диск наноситься єдина доріжка даних у вигляді спіралі.

На компакт-диск діаметром 120 мм можна записати до 700 Мб інформації, а на компакт-диск 80 мм – до 210 Мб. Зчитування інформації відбувається за допомогою лазерного променя і відповідної оптичної системи.

Вставляють компакт у висувний лоток системного блоку дзеркальною поверхнею вниз.

Одна з найважливіших характеристик дисководу для компакт-дисків – швидкість передачі даних (Data Transfer Rate), яка вимірюється в Кб/с. Вона характеризує максимальну швидкість, з якою дані передаються з компакт-

диску в оперативну пам'ять комп'ютера. Як правило, для позначення даної швидкості використовується число, кратне швидкості звичайного програвача аудіокомпакт-дисків – 150 Кб/с. Отже, якщо дисковод позначається 52X, то це значить, що максимальна швидкість передавання з нього дорівнює 52*150 Кб/с = 7800 Кб/с.

За допомогою дисководів для компакт-дисків можна прослуховувати і музикальні компакт-диски.

Є багато різновидів форматів компакт-дисків. Наприклад:

CD-ROM (Compact Disk Read-Only Memory) – формат, при якому записана на компакт-диску інформація використовується лише для зчитування.

CD-DA (Digital Audio) – формат для запису звичайного цифрового музичного компакт-диску (час звучання 74 хв.).

CD-ROM-XA (eXtended Architecture) – розширений формат запису даних, що об'єднують звук і зображення для одночасного (синхронного) їх зчитування.

Photo-CD – формат для фото компакт-диску.

CD-I (Interactive) – збереження відео зображення із звуком (час тривання 90 хв.).

При наявності спеціальних дисководів та програмного забезпечення на комп'ютері можна здійснювати запис інформації на компакт-диски.

CD-R – інформація записується один раз, а зчитуватись може багато разів.

CD-RW (ReWrite) – можливе багаторазове перезаписування інформації.

У випадку наявності обох дисководів, CD-R і CD-RW, можливе безпосереднє копіювання інформації з одного компакт-диску на інший.

DVD (Digital Versatile Disk – цифровий універсальний диск) – це двосторонній диск, на якому за рахунок більшої щільності запису і меншої довжини хвилі лазера можна помістити набагато більше інформації.

Існує 4-и формати DVD: односторонній одношаровий (4,7 ГБ), односторонній двошаровий (8,5 ГБ), двосторонній одношаровий (9,4 ГБ) і двосторонній двошаровий (17 ГБ).

Формула 40/20/48 для CD-RW означає: швидкість запису CD – 40X; швидкість запису CD-RW – 20X; швидкість читання CD – 48X.

Формула 32/10/40/16 для комбо-дисковода DVD-ROM/CD-RW означає те ж саме, що і в попередньому випадку, лише остання цифра характеризує швидкість читання DVD-ROM.

На дисководі для DVD-дисків можна записувати і звичайні CD-диски.

Механічно компакт-диск є досить надійним, але побоюється впливу сонячного проміння.

Flash-пам'ять. Пристрої, засновані на кристалах електрично перепрограмованій пам'яті, що не мають рухливих частин, називаються флеш-

пам'ять. Фізичний принцип організації осередків флэш-памяти можна вважати однаковим для усіх пристроїв, що випускаються, як би вони не називалися. Розрізняються такі пристрої за інтерфейсом і вживаному контроллеру, що обумовлює різницю в місткості, швидкості передачі даних і енергоспоживанні.

Недоліки:

- ✧ час зберігання заряду дуже великий і вимірюється роками, але все таки обмежено. Закони термодинаміки і дифузії свідчать, що концентрація електронів в різних областях рано чи пізно вирівняється. *А це означає, що без перезапису, без оновлення осередків хоч би раз в декілька років Flash "обнуляється" так само, як оперативна пам'ять, після виключення комп'ютера*.
- ✧ обмежена кількість циклів запису-перезапису: від ста тисяч до декількох мільйонів. З часом неминуче відбувається деградація самого матеріалу і p-n-переходов. Наприклад, карти Kingston Compact Flash розраховані на 300 000 циклів перезапису. Transcend Compact Flash – на 1 000 000, а flash – диск Transcend 32 Gb USB – всього на 100 000.

Існують дві архітектури flash-памяти. Вони відрізняються способом звернення до осередків і, відповідно, організацією внутрішніх провідників.

Пам'ять NOR (логічне ИЛИ-НЕ) дозволяє звертатися до осередків по одній. До кожного осередку підходить окремий провідник (двовимірна організація). Адресний простір NOR -памяти дозволяє працювати з окремими байтами або словами (кожне слово містить 2 байти). Така архітектура накладає серйозні обмеження на максимальний об'єм пам'яті на одиниці площі кристала. Пам'ять NOR сьогодні використовується лише в мікросхемах BIOS і інших ПЗП малої місткості, наприклад в стільникових телефонах.

У пам'яті архітектури NAND (логічне І-НЕ) кожен осередок виявляється на перетині "лінії біт" і "лінії слів" (організація може бути і тривимірною). Осередки групуються в невеликі блоки за аналогією до кластерів жорсткого диска. І прочитування, і запис здійснюються лише цілими блоками або рядками. Усі сучасні знімні носії побудовані на пам'яті NAND.

У будь-якому випадку при купівлі комп'ютера необхідно:

- ✧ вимагати у фірми, що торгує комп'ютерною технікою, гігієнічний сертифікат і сертифікат відповідності міжнародному стандарту на монітор, блок процесора, принтер, сканер та інші пристрої, що породжують електромагнітне випромінювання і статистичний заряд у процесі роботи;
- ✧ пам'ятати, що, купуючи дешевий комп'ютер, ми заощаджуємо на власному здоров'ї.

Усе, що входить до складу комп'ютера, незалежно від обраної користувачем конфігурації, прийнято називати **апаратним забезпеченням** (Hard Ware).

Саме по собі апаратне забезпечення ще не є комп'ютером як таким. Комп'ютер, якщо так можна сказати, відбудеться тоді і тільки тоді, коли Hard Ware буде підкріплено **програмним забезпеченням** (Soft Ware).

## 1.3 Основні поняття економічної інформатики

### 1.3.1 Поняття економічної інформації, її особливості

Термін *інформація* походить від латинського informatio, що означає роз'яснення, інформування, виклад. З позиції матеріалістичної філософії інформація є віддзеркалення реального світу за допомогою відомостей (повідомлень). **Повідомлення** – це форма представлення інформації у вигляді мови, тексту, зображення, цифрових даних, графіків, таблиць, тощо. У широкому сенсі інформація – це загальнонаукове поняття, що включає обмін відомостями між людьми, обмін сигналами між живою і неживою природою, людьми і пристроями.

**Інформація** – відомості про об'єкти і явища довкілля, їх параметри, властивості і стан, які зменшують наявну про них міру невизначеності, неповноту знань**.**

Інформатика розглядає інформацію як концептуально пов'язані між собою відомості, дані, поняття, змінюючі наші уявлення про явище або об'єкт навколишнього світу. Разом з інформацією в інформатиці часто вживається поняття **дані**.

**Дані** можуть розглядатися як ознаки або записані спостереження, які з якихось причин не використовуються, а тільки зберігаються. У такому випадку, якщо з'являється можливість використовувати ці дані для зменшення невизначеності про що-небудь, дані перетворюються на інформацію. Тому можна стверджувати, що інформацією є використовувані дані.

***Приклад 1.1.*** Напишіть на листі десять номерів телефонів у вигляді послідовності десяти чисел і покажіть їх вашому другу. Він сприйме ці цифри як дані, оскільки вони не надають йому ніяких відомостей.

Потім проти кожного номера вкажіть назву фірми і вид діяльності. Для вашого друга незрозумілі цифри набудуть визначеності і перетворяться з даних в інформацію, яку він надалі міг би використовувати.

Одним з найважливіших різновидів інформації є інформація економічна. Її *відмінна риса* – зв'язок з процесами управління колективами людей, організацією. Економічна інформація супроводжує процеси виробництва, розподілу, обміну і споживання матеріальних благ і послуг. Значна частина її пов'язана з громадським виробництвом і може бути названа *виробничою* інформацією.

**Економічна інформація –** сукупність відомостей, що відбивають соціально-економічні процеси і службовців для управління цими процесами і колективами людей у виробничій і невиробничій сфері.

При роботі з інформацією завжди є її джерело і споживач (одержувач). Шляхи і процеси, що забезпечують передачу повідомлень від джерела інформації до її споживача, називаються *інформаційними комунікаціями*.

Для споживача інформації дуже важливою характеристикою є її адекватність.

**Адекватність інформації –** це певний рівень відповідності створюваного за допомогою отриманої інформації образу реальному об'єкту, процесу, явищу і тому подібне.

У реальному житті навряд чи можлива ситуація, коли ви зможете розраховувати на повну адекватність інформації. Завжди присутня деяка міра невизначеності. Від міри адекватності інформації реальному стану об'єкту або процесу залежить правильність ухвалення людиною рішень.

***Приклад 1.2.*** Ви успішно закінчили школу і хочете продовжити освіту за економічному напряму. Поговоривши з друзями, ви дізнаєтеся, що подібну підготовку можна отримати в різних ВНЗ. В результаті таких бесід ви отримуєте дуже суперечні відомості, які не дозволяють вам прийняти рішення на користь того або іншого варіанту, тобто отримана інформація неадекватна реальному стану справ. Для того, щоб отримати достовірніші відомості, ви купуєте довідник для вступників до вузів, з якого отримуєте вичерпну інформацію. В цьому випадку можна говорити, що інформація, отримана вами з довідника, адекватно відбиває напрями навчання у ВНЗ і допомагає вам визначитися в остаточному виборі.

**Форми адекватності інформації**

Адекватність інформації може виражатися в трьох формах: семантичній, синтаксичній, прагматичній.

**Синтаксична адекватність.** Вона відображує формально-структурні характеристики інформації і не зачіпає її смислового змісту. На синтаксичному рівні враховуються тип носія і спосіб представлення інформації, швидкість, передачі і обробки, розміри кодів представлення інформації, надійність і точність перетворення цих кодів і тому подібне. Інформацію, що розглядається тільки з синтаксичних позицій, зазвичай називають даними, оскільки при цьому не має значення смислова сторона. Ця форма сприяє сприйняттю зовнішніх структурних характеристик, тобто синтаксичної сторони інформації.

**Семантична (смислова) адекватність.** Ця форма визначає міру відповідності образу об'єкту і самого об'єкту. Семантичний аспект припускає облік смислового змісту інформації. На цьому рівні аналізуються ті відомості, які відбиває інформація, розглядаються смислові зв'язки. У

інформатиці встановлюються смислові зв'язки між кодами представлення інформації. Ця форма служить для формування понять і представлень, виявлення сенсу, змісту інформації і її узагальнення.

**Прагматична (споживча) адекватність.** Вона відбиває відношення інформації і її споживача, відповідність інформації меті управління, яка на її основі реалізується. Проявляються прагматичні властивості інформації тільки за наявності єдності інформації (об'єкту), користувача і мети управління. Прагматичний аспект розгляду пов'язаний з цінністю, корисністю використання інформації при виробленні споживачем рішення для досягнення своєї мети. З цієї точки зору аналізуються споживчі властивості інформації. Ця форма адекватності безпосередньо пов'язана з практичним використанням інформації, з відповідністю її цільової функції діяльності системи.

**Міри інформації.**

**Класифікація мір інформації.**

Для виміру інформації вводяться два параметри: кількість інформації $I$ і об'єм даних $V_д$.

Ці параметри мають різні вирази і інтерпретацію залежно від даної форми адекватності. Кожній формі адекватності відповідає своя міра кількості інформації і об'єму даних (рис. 1.29).

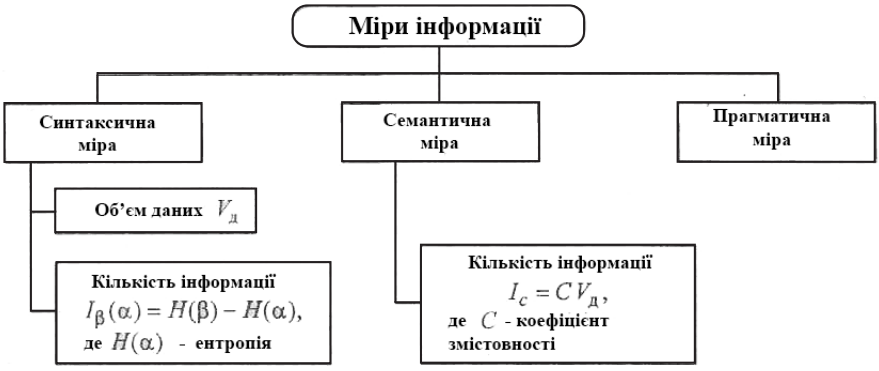



Рисунок 1.29. Міри інформації

**Синтаксична міра інформації**

Цей захід кількості інформації оперує зі знеособленою інформацією, що не виражає смислового відношення до об'єкту.

*Об'єм даних* $V_д$ в повідомленні вимірюється кількістю символів (розрядів) в цьому повідомленні. У різних системах числення один розряд має різну вагу і відповідно міняється одиниця виміру даних*:*

- ✧ у двійковій системі числення одиниця виміру – біт (*bit – binary digit* – двійковий розряд);
- ✧ у десятковій системі числення одиниця виміру – дит (десятковий розряд).

***Приклад 1.3.*** Повідомлення в двійковій системі у вигляді восьмирозрядного двійкового коду 10111011 має об'єм даних $V_д = 8$ біт.

Повідомлення в десятковій системі у вигляді шестирозрядного числа 275903 має об'єм даних $V_д = 6$ дит.

*Кількість інформації I* на синтаксичному рівні неможливо визначити без розгляду поняття невизначеності стану системи (ентропії системи). Дійсно, отримання інформації про яку-небудь систему завжди пов'язане із зміною міри непоінформованості одержувача про стан цієї системи. Розглянемо це поняття.

Нехай до отримання інформації споживач має деякі попередні (апріорні) відомості про систему α. Мірою його непоінформованості про систему є функція H(α), яка в той же час служить і мірою невизначеності стану системи.

Після отримання деякого повідомлення β одержувач придбав деяку додаткову інформацію $I_β(α)$, що зменшила його апріорну непоінформованість так, що апостеріорна (після отримання повідомлення β) невизначеність стану системи стала $H_β(α)$.

Тоді кількість інформації $I_β(α)$ про систему, отриманої в повідомленні β, визначиться як

$$I_β(α) = H(α) - H_β(α), \tag{1.1}$$

тобто кількість інформації вимірюється зміною (зменшенням) невизначеності стану системи.

Якщо кінцева невизначеність $H_β(α)$ звернеться в нуль, то початкове неповне знання заміниться повним знанням і кількість інформації $I_β(α)=H(α)$. Іншими словами, ентропія системи *H(a)* може розглядатися як міра відсутньої інформації.

Ентропія системи $H(α)$, що має N можливих станів, згідно з формулою Шеннона, дорівнює:

$$H(\alpha) = -\sum_{i=1}^{N} P_i \log P_i, \tag{1.2}$$

де $P_i$ – імовірність того, що система знаходиться в і-му стані.

Для випадку, коли всі стани системи рівноймовірні, тобто їх ймовірності рівні $P_i = \frac{1}{N}$, її ентропія визначається співвідношенням

$$H(\alpha) = -\sum_{i=1}^{N} \frac{1}{N} \log \frac{1}{N}. \tag{1.3}$$

Часто інформація кодується числовими кодами в тій чи іншій системі числення, особливо це актуально при поданні інформації в комп'ютері. Природно, що одне і та ж кількість розрядів в різних системах числення може передати різне число станів відображуваного об'єкта, що можна представити у вигляді співвідношення

$$N = m^n, \tag{1.4}$$

де $N$ – число всіляких відображуваних станів;

$m$ – основа системи числення (різноманітність символів, застосовуваних в алфавіті);

$n$ – число розрядів (символів) у повідомленні.

***Приклад 1.4***. По каналу зв'язку передається n-розрядне повідомлення, яке використовує m різних символів. Так як кількість всіляких кодових комбінацій буде $N=m^n$ то при рівній імовірності появи будь-який з них кількість інформації, придбаної абонентом в результаті отримання повідомлення, буде $I= \log N = n \log m$ – формула Хартлі.

Якщо в якості підстановки логарифма взяти $m$, то $I = n$. У даному випадку кількість інформації (за умови повного апріорного незнання абонентом змісту повідомлення) буде дорівнює обсягу даних $I= V_Д$, отриманих по каналу зв'язку. Для не рівноімовірних станів системи завжди $I< V_Д = n$.

Найбільш часто використовуються двійкові і десяткові логарифми. Одиницями виміру в цих випадках будуть відповідно біт та діт.

*Коефіцієнт (ступінь) інформативності* (лаконічність) повідомлення визначається відношенням кількості інформації до обсягу даних, тобто

$$Y = \frac{I}{V_Д}, \text{ причому } 0<Y<1. \tag{1.5}$$

Із збільшенням $Y$ зменшуються обсяги роботи з перетворення інформації (даних) в системі. Тому прагнуть до підвищення інформативності, для чого розробляються спеціальні методи оптимального кодування інформації.

**Семантична міра інформації.**

Для вимірювання смислового змісту інформації, тобто її кількості на семантичному рівні, найбільше визнання отримала тезаурусна міра, яка пов'язує семантичні властивості інформації зі здатністю користувача приймати повідомлення, що надійшло. Для цього використовується поняття *тезаурус користувача*.

**Тезаурус –** це сукупність відомостей, якими володіє користувач або система.

Залежно від співвідношень між смисловим змістом інформації $S$ і тезаурусом користувача $S_p$ змінюється кількість семантичної інформації

$I_C$, сприйманої користувачем і що включається їм надалі в свій тезаурус. Характер такої залежності показано на рис. 1.30. Розглянемо два граничних випадки, коли кількість семантичної інформації $I_C$ рівна 0:

- ✧ при $Sp \approx 0$ користувач не сприймає, не розуміє інформацію, що надходить;
- ✧ при $Sp \to \infty$ користувач все знає, і інформація, що надходить, йому не потрібна.

Максимальна кількість семантичної інформації $I_C$ споживач набуває при узгодженні її смислового змісту S зі своїм тезаурусом $S_p$ ($S_p = S_{p\,opt}$), коли надходить інформація зрозуміла користувачеві і несе йому раніше не відомі (відсутні в його тезаурусі) відомості.

Отже, кількість семантичної інформації в повідомленні, кількість нових знань, одержуваних користувачем, є величиною відносною. Одне і те ж повідомлення може мати смисловий зміст для компетентного і бути беззмістовним (семантичний шум) для некомпетентного.

При оцінці семантичного (змістовного) аспекту інформації необхідно прагнути до узгодження величин $S$ і $S_p$.

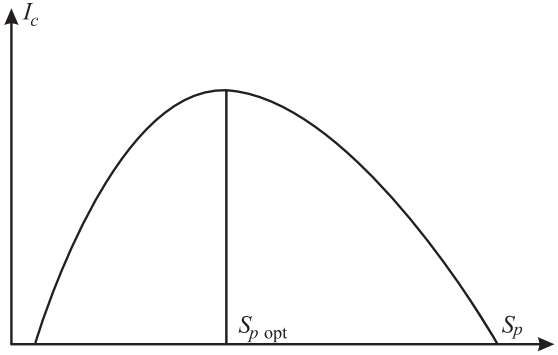



Рисунок 1.30. Залежність кількості семантичної інформації, яка сприймається споживачем, від його тезауруса $I_c = f(S_p)$

Відносною мірою кількості семантичної інформації може служити коефіцієнт змістовності С, який визначається як відношення кількості семантичної інформації до її об'єму:

$$C = \frac{I_C}{V_Д}. \qquad (1.6)$$

**Прагматична міра інформації.**

Ця міра визначає корисність інформації (цінність) для досягнення користувачем поставленої мети. Ця міра також величина відносна, обумовлена особливостями використання цієї інформації в тій чи іншій системі.

Цінність інформації доцільно вимірювати у тих же самих одиницях (або близьких до них), в яких вимірюється цільова функція.

***Приклад 1.5.*** В економічній системі прагматичну властивість (цінність) інформації можна визначити приростом економічного ефекту функціонування, досягнутим завдяки використанню цієї інформації для управління системою:

$$I_{n\beta}(\gamma) = П(\gamma / \beta) - П(\gamma),$$

де $I_{n\beta}(\gamma)$ – цінність інформаційного повідомлення β для системи управління $\gamma$,

П($\gamma$) – апріорний очікуваний економічний ефект функціонування системи управління $\gamma$,

П($\gamma$/β) – очікуваний ефект функціонування системи $\gamma$ за умови, що для управління буде використана інформація, що міститься в повідомленні β.

Для зіставлення введених заходів інформації представимо в табл. 1.3.

Таблиця 1.3

**Одиниці виміру інформації та приклади**

| Міра інформації | Одиниці виміру | Приклади (для комп'ютерної області) |
|---|---|---|
| Синтаксична: шенінівский підхід комп'ютерний підхід | Ступінь зменшення невизначеності Одиниці представлення інформації | Ймовірність події Біт, байт, Кбайт і т.д. |
| Семантична | Тезаурус<br><br>Економічні показники | Пакет прикладних програм, персональний комп'ютер, комп'ютерні мережі і т.д. Рентабельність, продуктивність, коефіцієнт амортизації і т.д. |
| Прагматична | Цінність використання | Ємність пам'яті, продуктивність комп'ютера, швидкість передачі даних і т.д. Ціннісний вираз Час обробки інформації та прийняття рішень |

**Якість інформації.**

Можливість і ефективність використання інформації обумовлюються такими основними її споживчими *показниками якості*, як репрезентативність, змістовність, достатність, доступність, актуальність, своєчасність, точність, достовірність, стійкість.

*Репрезентативність* інформації пов'язана з правильністю її добору і формування в цілях адекватного відображення властивостей об'єкта.

Найважливіше значення тут мають:

- ❑ правильність концепції, на базі якої сформульовано вихідне поняття;
- ❑ обгрунтованість відбору істотних ознак і зв'язків відображуваного явища.

Порушення репрезентативності інформації призводить нерідко до суттєвих її похибок.

*Змістовність* інформації відображає семантичну ємність, рівну відношенню кількості семантичної інформації в повідомленні до обсягу оброблюваних даних, тобто

$$C = \frac{I_C}{V_Д}. \tag{1.6}$$

Зі збільшенням змістовності інформації зростає семантична пропускна здатність інформаційної системи, так як для отримання одних і тих же відомостей потрібно перетворити менший обсяг даних.

*Достатність (повнота)* інформації означає, що вона містить мінімальний, але достатній для прийняття правильного рішення склад (набір показників). Поняття повноти інформації пов'язане з її смисловим змістом (семантикою) і прагматикою. Як неповна, тобто недостатня для прийняття правильного рішення, так і надлишкова інформація знижує ефективність прийнятих користувачем рішень.

*Доступність* інформації сприйняттю користувача забезпечується виконанням відповідних процедур її отримання та перетворення. Наприклад, в інформаційній системі інформація перетворюється в доступну та зручну для сприйняття користувача форму. Це досягається, зокрема, і шляхом узгодження її семантичної форми з тезаурусом користувача.

*Актуальність* інформації визначається ступенем збереження цінності інформації для управління в момент її використання і залежить від динаміки зміни її характеристик і від інтервалу часу, що пройшов з моменту виникнення даної інформації.

*Своєчасність* інформації означає її вступ не пізніше заздалегідь призначеного моменту часу, узгодженого з часом вирішення поставленого завдання.

*Точність* інформації визначається ступенем близькості одержуваної інформації до реального стану об'єкта, процесу, явища і т.д. Для інформації, яка відображається цифровим кодом, відомі чотири класифікаційних поняття точності:

- ✧ формальна точність, вимірювана значенням одиниці молодшого розряду числа;
- ✧ реальна точність, обумовлена значенням одиниці останнього розряду числа, вірність якого гарантується;

- ✧ максимальна точність, яку можна отримати в конкретних умовах функціонування системи;
- ✧ необхідна точність, обумовлена функціональним призначенням показника.

*Достовірність* інформації визначається її властивістю відображати реально існуючі об'єкти з необхідною точністю. Вимірюється достовірність інформації довірчою ймовірністю необхідної точності, тобто ймовірністю того, що відображуване інформацією значення параметра відрізняється від істинного значення цього параметра в межах необхідної точності.

*Стійкість* інформації відображає її здатність реагувати на зміни вихідних даних без порушення необхідної точності. Стійкість інформації, як і репрезентативність, зумовлена обраною методикою її відбору і формування.

На закінчення слід зазначити, що такі параметри якості інформації, як репрезентативність, змістовність, достатність, доступність, стійкість, цілком визначаються на методичному рівні розробки інформаційних систем. Параметри актуальності, своєчасності, точності та достовірності обумовлюються в більшій мірі також на методичному рівні, однак на їх величину істотно впливає і характер функціонування системи, в першу чергу її надійність. При цьому параметри актуальності і точності жорстко зв'язані відповідно з параметрами своєчасності та достовірності.

### 1.3.2 Класифікація й кодування економічної інформації

1.3.2.1 Система класифікації

**Загальні відомості**

Важливим поняттям при роботі з інформацією є *класифікація* об'єктів.

**Класифікація –** система розподілу об'єктів (предметів, явищ, процесів, понять) по класах відповідно за певною ознакою.

Під *об'єктом* розуміється будь-який предмет, процес, явище матеріальної або нематеріальної властивості. Система класифікації дозволяє згрупувати об'єкти і виділити певні класи, які будуть характеризуватися рядом загальних властивостей. Класифікація об'єктів – це процедура угруповання на якісному рівні, направлена на виділення однорідних властивостей. Стосовно інформації як об'єкту класифікації виділені класи називають *інформаційними об'єктами*.

***Приклад 1.6.*** Всю інформацію про університет можна класифікувати за численними інформаційнимими об'єктами, які будуть характеризуватися загальними властивостями:

- – інформація про студентів – у вигляді інформаційного об'єкту "Студент";

- інформація про викладачів – у вигляді інформаційного об'єкту "Викладач";
- інформація про факультети – у вигляді інформаційного об'єкту "Факультет" і т.п.

Властивості інформаційного об'єкту визначаються інформаційними параметрами, званими *реквізитами*. Реквізити представляються або числовими даними, наприклад вага, вартість, рік, або ознаками, наприклад колір, марка машини, прізвище.

**Реквізит –** логічно неподільний інформаційний елемент, що описує певну властивість об'єкта, процесу, явища і т.д.

***Приклад 1.7.*** Інформація про кожного студента у відділі кадрів університету систематизована і представлена за допомогою однакових реквізитів:

- прізвище, ім'я, по батькові;
- стать;
- рік народження;
- адреса проживання;
- факультет, де проходить навчання студент, і т.д.

Всі перераховані реквізити характеризують властивості інформаційного об'єкту "Студент".

Окрім виявлення загальних властивостей інформаційного об'єкта класифікація потрібна для розробки правил (алгоритмів) і процедур обробки інформації, представленою сукупністю реквізитів.

***Приклад 1.8.*** Алгоритм обробки інформаційних об'єктів бібліотечного фонду дозволяє отримати інформацію про всіх книгах з певної тематики, про авторів, абонентів і т.д.

Алгоритм обробки інформаційних об'єктів фірми дозволяє отримати інформацію про обсяги продажів, про прибуток, замовників, види виготовленої продукції і т. д.

Алгоритми обробки в тому і іншому випадку переслідують різні цілі, обробляють різну інформацію, реалізуються різними способами.

При будь-якій класифікації бажано дотримуватися наступних вимог:

- повнота охоплення об'єктів в розглянутій області;
- однозначність реквізитів;
- можливість включення нових об'єктів.

У будь-якій країні розроблені і застосовуються державні, галузеві, регіональні класифікатори. Наприклад, класифіковані: галузі промисловості, обладнання, професії, одиниці виміру, статті витрат і т.д.

**Класифікатор –** систематизоване зведення найменувань і кодів класифікаційних угруповань.

При класифікації широко використовуються поняття *класифікаційна ознака* і *значення класифікаційної ознаки*, які дозволяють встановити подібність або відмінність об'єктів. Можливий підхід до класифікації з об'єднанням цих двох понять в єдине, назване як ознака класифікації.

***Приклад 1.9.*** В якості ознаки класифікації обирається вік, який складається з трьох значень: до 20 років, від 20 до 30 років, понад 30 років.

Можна в якості ознак класифікації використовувати: вік до 20 років, вік від 20 до 30 років, вік понад 30 років.

Розроблено три методи класифікації об'єктів: ієрархічний, фасентний, дескрипторний. Ці методи розрізняються різною стратегією застосування класифікаційних ознак. Розглянемо основні ідеї цих методів для створення систем класифікації.

**Ієрархічна система класифікації**

*Ієрархічна система класифікації* (рис. 1.31) будується таким чином:

- вихідна безліч елементів складає 0-й рівень і ділиться залежно від обраної класифікаційної ознаки на класи (угрупування), які утворюють 1-й рівень;
- кожен клас 1-го рівня у відповідності зі своїм, притаманною йому класифікаційною ознакою, ділиться на підкласи, які утворюють 2-й рівень;
- кожен клас 2-го рівня аналогічно ділиться на групи, які утворюють 3-й рівень, і т.д.

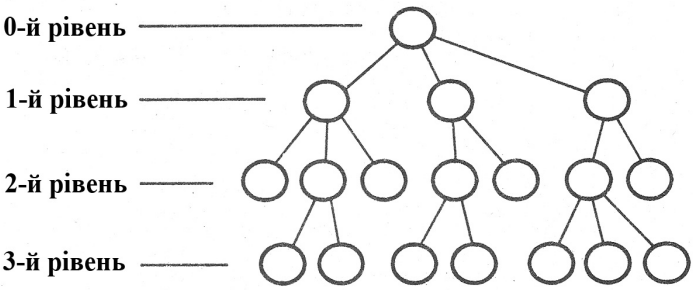



Рисунок 1.31. Ієрархічна система класифікації

Враховуючи достатньо жорстку процедуру побудови структури класифікації, необхідно перед початком роботи визначити її мету, тобто якими властивостями повинні володіти об'єкти, що об'єднуються в класи. Ці властивості приймаються надалі за ознаки класифікації.

***Запам'ятайте!*** *В ієрархічній системі класифікації через жорстку структуру особливу увагу слід приділити вибору класифікаційних ознак.*

В ієрархічній системі класифікації кожен об'єкт на будь-якому рівні повинен бути віднесений до одного класу, який характеризується конкретним значенням обраної класифікаційної ознаки. Для подальшого угруповання в кожному новому класі необхідно задати свої класифікаційні ознаки і їх значення. Таким чином, вибір класифікаційних ознак залежатиме від семантичного змісту того класу, для якого необхідне угруповання на наступному рівні ієрархії.

Кількість рівнів класифікації, відповідних числу ознак, обраних у якості підстави поділу, характеризує *глибину класифікації.*

Переваги ієрархічної системи класифікації:

- простота побудови;
- використання незалежних класифікаційних ознак у різних гілка ієрархічної структури.

Недоліки ієрархічної системи класифікації:

- жорстка структура, яка призводить до складності внесення змін, оскільки доводиться перерозподіляти всі класифікаційні угруповання;
- неможливість групувати об'єкти за заздалегідь не передбаченим сполученням ознак.

***Приклад 1.10.*** Поставлено завдання – створити ієрархічну систему класифікації для інформаційного об'єкту "Факультет", яка, дозволить класифікувати інформацію про всіх студентів за такими класифікаційними ознаками: факультет, на якому він навчається, віковий склад студентів, стать студента, для жінок – наявність дітей.

Система класифікації представлена на рис.1.32 і матиме такі рівні:

0-й-рівень. Інформаційний об'єкт "Факультет";

1-й рівень. Вибирається класифікаційна ознака – назва факультету, що дозволяє виділити декілька класів з різними назвами факультетів, в яких зберігається інформація про всіх студентів.

2-й рівень. Вибирається класифікаційна ознака – вік, який має три градації: до 20 років, від 20 до 30 років, понад 30 років. За кожному факультету виділяються три вікові підкласи студентів.

3-й рівень. Вибирається класифікаційна ознака – стать. Кожен підклас 2-го рівня розбивається на дві групи. Таким чином, інформація про студентів кожного факультету в кожному віковому підкласі розділяється на дві групи – чоловіків і 1 жінок.

4-й рівень. Вибирається класифікаційна ознака – наявність дітей у жінок: є, немає.

Створена ієрархічна система класифікації має глибину класифікації, рівну чотирьом.

**Фасетна система класифікації**

*Фасетна система* класифікації на відміну від ієрархічної дозволяє вибирати ознаки класифікації незалежно як один від одного, так і від семантичного змісту класифікуючого об'єкта. Ознаки класифікації називаються *фасетами (facet – рамка)*. Кожен фасет ($Ф_i$) містить сукупність однорідних значень даної класифікаційної ознаки. Причому, значення в фасетах можуть розташовуватися в довільному порядку, хоча краще їх упорядковувати.

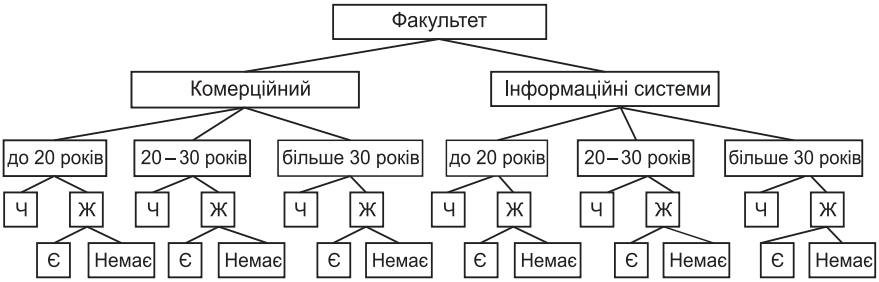



Рисунок 1.32. Приклад ієрархічної системи класифікації для інформаційного об'єкту "Факультет"

***Приклад 1.11.*** Фасетний колір містить значення: червоний, білий, зелений, чорний, жовтий.

Фасетна спеціальність містить назви спеціальностей.

Фасетна освіта містить значення: середня, середня спеціальна, вища.

Схема побудови фасетної системи класифікації у вигляді таблиці відображена на рис. 1.33. Назви стовпців відповідають виділеним класифікаційним ознакам (фасетам), позначеним $Ф_1, Ф_2, ..., Ф_i, ..., Ф_n$ – Наприклад, колір, розмір одягу, вага і т.д. Зроблена нумерація рядків таблиці. У кожній клітці таблиці зберігається конкретне значення фасета. Наприклад, фасетний колір, позначений $Ф_2$, містить значення: червоний, білий, зелений, чорний, жовтий.

Процедура класифікації полягає в привласненні кожному об'єкту відповідних значень з фасетів. При цьому можуть бути використані не всі фасети. Для кожного об'єкта задається конкретне групування фасетною структурною формулою, в якій відбивається їх порядок проходження:

$$К_s = (Ф_1, Ф_2, \ldots, Ф_i, \ldots, Ф_n), \qquad (1.7)$$

де $Ф_1$ – і-й фасет;

n – кількість фасетів.

При побудові фасетної системи класифікації необхідно, щоб значення, використовувані в різних фасетах, не повторювалися. Фасетну систему легко можна модифікувати, вносячи зміни в конкретні значення будь-якої фасети.

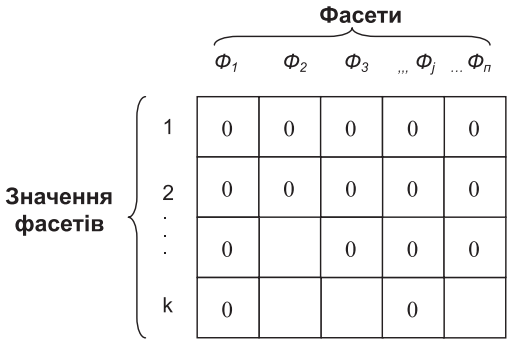



Рисунок 1.33. Фасетна система класифікації

Переваги фасетної системи класифікації:

- можливість створення великої ємності класифікації, тобто використання великої кількості ознак класифікації і їх значень для створення угруповань;
- можливість простої модифікації всієї системи класифікації без зміни структури існуючих угруповань.

Недоліком фасетної системи класифікації є складність її побудови, оскільки необхідно враховувати все різноманіття класифікаційних ознак.

***Приклад 1.12.*** Зверніться до змісту прикладу 10, де показано побудова ієрархічної системи класифікації. Для зіставлення розробимо фасетну систему класифікації.

Згрупуємо і представимо у вигляді таблиці (рис. 1.34) всі класифікаційні ознаки за фасетам:

- фасет *назва факультету* з п'ятьма назвами факультетів;
- фасет *вік* з трьома віковими групами;
- фасет *стать* з двома градаціями;
- фасет *діти* з двома градаціями.

| Назва факультету | Вік | Стать | Діти |
|---|---|---|---|
| Радіотехнічний | До 20 років | Ч | Є |
| Машинобудівний | 20–30 років | Ж | Немає |
| Комерційний | Більше30 років | | |
| Інформаційні системи | | | |
| Математичний | | | |

Рисунок 1.34. Приклад фасетної системи класифікації
для інформаційного об'єкту "Факультет"

Структурну формулу будь-якого класу можна представити у вигляді: Ks – (Факультет, Вік, Стать, Діти)

Привласнюючи конкретні значення кожному фасет, отримаємо наступні класи:

- $K_1$ = (Радіотехнічний факультет, вік до 20 років, чоловік, є діти);
- $K_2$ = (Комерційний факультет, вік від 20 до 30 років, чоловік, дітей немає);
- $K_3$ = (Математичний факультет, вік до 20 років, жінка, дітей немає) і т.д.

**Дескрипторна система класифікації**

Для організації пошуку інформації, для ведення тезаурусів (словників) ефективно використовується дескрипторна (описова) система класифікації, мова якої наближається до природної мови опису інформаційних об'єктів. Особливо широко вона використовується в бібліотечній системі пошуку.

Суть дескрипторного методу класифікації полягає в наступному:

- відбирається сукупність ключових слів або словосполучень, що описують певну предметну область або сукупність однорідних об'єктів. Причому серед ключових слів можуть знаходитися синоніми;
- вибрані ключові слова та словосполучення піддаються нормалізації, тобто із сукупності синонімів вибирається один або декілька найбільш вживаних;
- створюється словник дескрипторів, тобто словник ключових слів і словосполучень, відібраних в результаті процедури нормалізації.

***Приклад 1.13***. В якості об'єкта класифікації розглядається успішність студентів. Ключовими словами можуть бути вибрані: оцінка, іспит, залік, викладач, студент, семестр, назва предмету. Серед них немає синонімів, і тому зазначені ключові слова можна використовувати як словник дескрипторів. В якості предметної області вибирається учбова діяльність у вищому навчальному закладі. Ключовими словами можуть бути вибрані: студент, учень, учень, викладач, вчитель, педагог, лектор, асистент, доцент, професор, колега, факультет, підрозділ університету, аудиторія, кімната, лекція, практичне заняття, заняття і т.д. Серед зазначених ключових слів зустрічаються синоніми, наприклад: студент, учень, учень; викладач, вчитель, педагог; факультет, підрозділ університету і т.д. Після нормалізації словник дескрипторів складатиметься з наступних слів: студент, викладач, лектор, асистент, доцент, професор, факультет, аудиторія, лекція, практичне заняття і т.д.

Між дескрипторами встановлюються зв'язки, які дозволяють розширити область пошуку інформації. Зв'язки можуть бути трьох видів:

- *синонімічні*, що вказують деяку сукупність ключових слів як синоніми;
- *родо-видові*, що відображають включення деякого класу об'єктів в більш представницький клас;
- *асоціативні*, що з'єднують дескриптори, що мають загальні властивості.

***Приклад 1.14***. Синонімічний зв'язок: студент – учень – учень. Родовидова зв'язок: університет – факультет – кафедра. Асоціативний зв'язок: студент – іспит – професор – аудиторія.

1.3.2.2 Система кодування

**Загальні поняття**

Система, кодування застосовується для заміни назви об'єкта на умовне позначення (код) в цілях забезпечення зручної і більш ефективної обробки інформації.

**Система кодування** – сукупність правил кодового позначення об'єктів.

Код будується на базі алфавіту, що складається з букв, цифр та інших символів. Код характеризується:

- *довжиною* – число позицій в коді;
- *структурою* – порядок розташування в коді символів, використовуваних для позначення класифікаційної ознаки.

Процедура присвоєння об'єкту кодового позначення називається *кодуванням.* Можна виділити дві групи методів, що використовуються в системі кодування (рис.1.35), які утворюють:

- *класифікаційну систему* кодування, орієнтовану на проведення попередньої класифікації об'єктів або на основі ієрархічної системи, або на основі фасетної системи;
- *регістраційну систему* кодування, яка не вимагає попередньої класифікації об'єктів.

Розглянемо представлену на рис. 1.35 систему кодування.

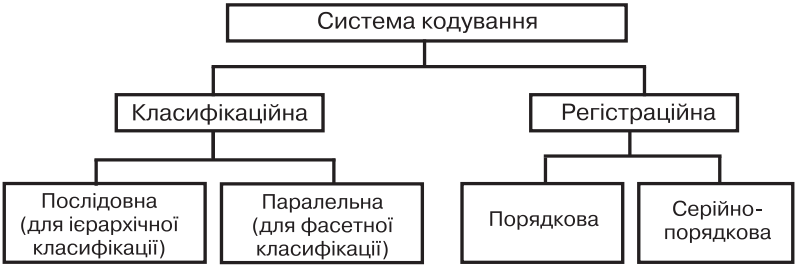



Рисунок 1.35. Система кодування, що використовує різні методи

**Класифікаційне кодування**

Класифікаційне кодування застосовується після проведення класифікації об'єктів. Розрізняють послідовне і паралельне кодування.

*Послідовне* кодування використовується для ієрархічної класифікаційної структури. Суть методу полягає в наступному: спочатку записується код старшої угрупування 1-го рівня, потім код угрупування 2-го рівня, потім код групировки 3-го рівня і т.д. У результаті виходить кодова комбінація, кожен розряд якої містить інформацію про специфіку виділеної групи на кожному рівні ієрархічної структури. Послідовна система кодування володіє тими ж перевагами та недоліками, що і ієрархічна система класифікації.

***Приклад 1.15.*** Проведемо кодування інформації, класифікованої за допомогою ієрархічної схеми (див. рис. 1.30). Кількість кодових угруповань буде визначатися глибиною класифікації і дорівнює 4. Перш ніж почати кодування, необхідно визначитися з алфавітом, тобто які будуть використовуватися символи. Для більшої наочності виберемо десяткову систему числення – 10 арабських цифр. Аналіз схеми на рис. 1.30 показує, що довжина коду визначається 4 десятковими розрядами, а кодування угруповання на кожному рівні можна робити шляхом послідовної нумерації зліва направо. У загальному вигляді код можна "записати як XXXX, де X – значення десяткового розряду. Розглянемо структуру коду, починаючи зі старшого розряду:

1-й (старший) розряд виділений для класифікаційної ознаки "назва факультету" і має наступні значення: 1 – комерційний; 2 – інформаційні системи; 3-для наступної назви факультету і т.д.;

2-й розряд виділений для класифікаційної ознаки "вік" і має наступні значення: 1 – до 20 років; 2 – від 20 до 30 років; 3 – понад 30 років;

3-й розряд виділений для класифікаційної ознаки "стать" і має наступні значення: 1 – чоловіки; 2 – жінки;

4-й розряд виділений для класифікаційної ознаки "наявність дітей у жінок" і має наступні значення: 1 – є діти; 2 – немає дітей, 0 – для чоловіків, оскільки подібної інформації не вимагається.

Прийнята система кодування дозволяє легко розшифрувати будь-який код угруповання, наприклад: 1310 – студенти комерційного факультету, понад 30 років, чоловіки; 2221 – студенти факультету інформаційних систем, від 20 до 30 років, жінки, які мають дітей.

*Паралельне* кодування використовується для фасетної системи класифікації. Суть методу полягає в наступному: всі фасети кодуються незалежно один від одного; для значень кожного фасета виділяється певна кількість розрядів коду. Паралельна система кодування володіє тими ж перевагами та недоліками, що і фасетної системи класифікації.

***Приклад 1.16***. Проведемо кодування інформації, класифікованої за допомогою фасетної схеми (див. рис. 1.33). Кількість кодових угруповань визначається кількістю фасетів і дорівнює 4. Виберемо десяткову систему числення як алфавіт кодування, що дозволить для значень фасетів виділити один розряд і мати довжину коду, рівну 4. На відміну від послідовного кодування для ієрархічної системи класифікації в даному методі не має значення порядок кодування фасетів. У загальному вигляді код можна записати як XXXX, де X – значення десяткового розряду. Розглянемо структуру коду, починаючи зі старшого розряду:

1-й (старший) розряд виділений для фасета "стать" і має наступні значення: 1 – чоловіки; 2 – жінки;

2-й розряд виділений для фасета "наявність дітей у жінок" і має наступні значення: 1 – є діти; 2 – немає дітей, 0 – для чоловіків, оскільки подібної інформації нетреба;

3-й розряд виділений для фасета "вік" і має наступні значення: 1 – до 20 років; 2 – від 20 до 30 років; 3 – понад 30 років;

4-й розряд виділений для фасета "назва факультету" і має наступні значення: 1 –радіотехнічний, 2 – машинобудівний, 3 – комерційний; 4 – інформаційні системи; 5 – математичний і т.д.

Прийнята система кодування дозволяє легко розшифрувати будь-який код угруповання, наприклад:

2135 – жінки у віці понад 30 років, що мають дітей і є студентами математичного факультету;

1021 – чоловіки віку від 20 до 30 років, які є студентами радіотехнічного факультету.

**Реєстраційне кодування**

Реєстраційне кодування використовується для однозначної ідентифікації об'єктів і не вимагає попередньої класифікації об'єктів. Розрізняють порядкову і серійно-порядкову систему.

*Порядкова* система кодування припускає послідовну нумерацію об'єктів числами натурального ряду. Цей порядок може бути випадковим або визначатися після попереднього упорядкування об'єктів, наприклад за алфавітом. Цей метод застосовується в тому випадку, коли кількість об'єктів невелика, наприклад кодування назв факультетів університету, кодування студентів у навчальній групі.

*Серійно-порядкова* система кодування передбачає попереднє виділення груп об'єктів, які складають серію, а потім у кожній серії проводиться порядкова нумерація об'єктів. Кожна серія також буде мати порядкову нумерацію. За своєю суттю серійно-порядкова система є змішаною: классифицирующей й ідентифікує. Застосовується тоді, коли кількість груп невелика.

***Приклад 1.17***. Всі студенти одного факультету розбиваються на навчальні групи (у даній термінології – серії), для яких використовується порядкова нумерація. Усередині кожної групи проводиться упорядкування прізвищ студентів за алфавітом і кожному студенту привласнюється номер.

### 1.3.3 Класифікація економічної інформації за різними ознаками

Будь-яка класифікація завжди відносна. Один і той самий об'єкт може бути класифікований за різними ознаками або критеріями. Часто зустрічаються ситуації, коли в залежності від умов зовнішнього середовища об'єкт може бути віднесений до різних класифікаційних угруповань. Ці міркування особливо актуальні при класифікації видів інформації без урахування її предметної орієнтації, так як вона часто може бути використана в різних умовах, різними споживачами, для різних цілей.

На рис. 1.36 наведена одна зі схем класифікації циркулюючої в організації (фірмі) інформації. В основу класифікації покладено п'ять найбільш загальних ознак: місце виникнення, стадія обробки, спосіб відображення, стабільність, функція управління.

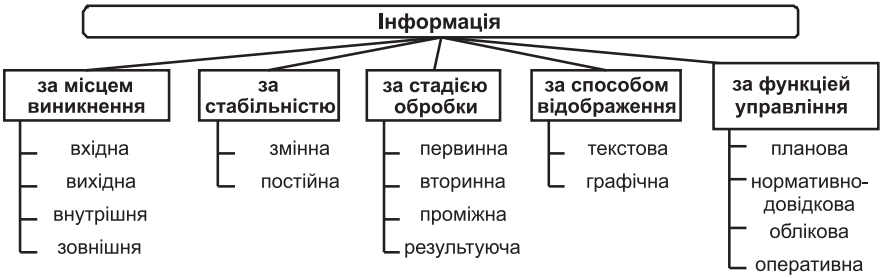



Рисунок 1.36. Класифікація інформації, що циркулює в організації

**Місце виникнення.** За цією ознакою інформацію можна розділити на вхідну вихідну, внутрішню, зовнішню**.**

*Вхідна інформація* – це інформація, що надходить у фірму або її підрозділу.

*Вихідна інформація* – це інформація, що надходить з фірми в іншу фірму, організацію (підрозділ).

Одна і та ж інформація може бути вхідною для однієї фірми, а для іншої, тієї, що є її виробником – вихідною. За відношенням до об'єкта управління (фірма або її підрозділ: цех, відділ, лабораторія) інформація може бути визначена як внутрішня, так і зовнішня.

*Внутрішня інформація* виникає всередині об'єкта, зовнішня інформація – за межами об'єкта.

***Приклад 1.18***. Зміст указу уряду про зміну рівня стягнутих податків для фірми є, з одного боку, зовнішньою інформацією, з іншого боку – вхідною. Відомості фірми в податкову інспекцію про розмір відрахувань до держбюджету є, з одного боку, вихідною інформацією, з іншого боку – зовнішньої, по відношенню до податкової інспекції.

**Стадія обробки.** Відповідно до стадії обробки інформація може бути первинною, вторинною, проміжної, результатної.

*Первинна інформація* – це інформація, яка виникає безпосередньо в процесі діяльності об'єкта і реєструється на початковій стадії.

*Вторинна інформація* – це інформація, яка виходить в результаті обробки первинної інформації і може бути проміжної і результативної.

*Проміжна інформація* використовується в якості вихідних даних для подальших розрахунків.

*Результативна інформація* отримується в процесі обробки первинної і проміжної інформації і використовується для прийняття управлінських рішень.

***Приклад 1.19***. В художньому цеху, де проводиться розпис чашок, в кінці кожної зміни реєструється загальна кількість виробленої продукції і кількість розписаних чашок кожним працівником. Це первинна інформація. В кінці кожного місяця майстер підводить підсумки первинної інформації. Це буде, з однієї сторони, вторинною проміжною інформація, а з іншого боку – результативною.

Підсумкові дані надходять до бухгалтерії, де проводиться розрахунок заробітної плати кожного працівника в залежності від його виробітку. Отримані розрахункові дані - результативна інформація.

**Спосіб відображення.** За способом відображення інформація підрозділяється на текстову та графічну.

*Текстова інформація* – це сукупність алфавітних, цифрових і спеціальних символів, за допомогою яких подається інформація на фізичному носії (папір, зображення на екрані дисплея).

*Графічна інформація* – це різного роду графіки, діаграми, схеми, малюнки і т.д.

**Стабільність.** За стабільності інформація може бути змінною (поточної) і постійної (умовно-постійної).

Змінна інформація відображає фактичні кількісні та якісні характеристики виробничо-господарської діяльності фірми. Вона може змінюватися для кожного випадку як за призначенням, так і за кількістю. Наприклад, кількість виробленої продукції за зміну, щотижневі витрати на доставку сировини, кількість справних верстатів і т.п.

*Постійна (умовно-постійна) інформація* – це незмінна і використовувана багато разів протягом тривалого періоду часу інформація. Постійна інформація може бути довідкової, нормативної, планової:

- постійна довідкова інформація включає опис постійних властивостей об'єкта у вигляді стійких тривалий час ознак. Наприклад, табельний номер службовця, професія працівника, номер цеху і т.п.;
- постійна нормативна інформація містить місцеві, галузеві і загальнодержавні нормативи. Наприклад, розмір податку на прибуток, стандарт на якість продуктів певного виду, розмір мінімальної оплати праці, тарифна сітка оплати державним службовцям;
- постійна планова інформація містить багаторазово використовувані в фірмі планові показники. Наприклад, план випуску телевізорів, план підготовки фахівців певної кваліфікації.

**Функція управління**. За функціями управління зазвичай класифікують *економічну інформацію*. При цьому виділяють такі групи: планову, нормативно-довідкову, облікову та оперативну (поточну).

*Планова інформація* – інформація про параметри об'єкта управління на майбутній період. На цю інформацію йде орієнтація всієї діяльності фірми.

***Приклад 1.20.*** Планової інформацією фірми можуть бути такі показники, як план випуску продукції, планований прибуток від реалізації, очікуваний попит на продукцію і т.д.

*Нормативно-довідкова інформація* містить різні нормативні та довідкові дані. Її оновлення відбувається досить рідко.

***Приклад 1.21***. Нормативно-довідковою інформацією на підприємстві є:

- час, призначений для виготовлення типовий деталі (норми трудомісткості);
- середньоденна оплата робочого за розрядом;
- оклад службовця;
- адресу постачальника або покупця і т.д.

*Облікова інформація* – це інформація, яка характеризує діяльність фірми за певний минулий період часу. На підставі цієї інформації можуть бути проведені наступні дії: скоригована планова інформація чи аналізу господарської діяльності фірми, прийняті рішення з підвищення ефективності управління роботи фірми, тощо. На практиці в якості облікової інформації може виступати інформація бухгалтерського обліку, статистична інформація та інформація оперативного обліку.

***Приклад 1.22***. Обліковою інформацією є: кількість проданої продукції за певний період часу; середньодобове завантаження або простий верстатів і т.п.

*Оперативна (поточна) інформація* – це інформація, яка використовується в оперативному управлінні і характеризує виробничі процеси в

поточний (даний) період часу. До оперативної інформації пред'являються вимоги зі швидкості надходження і обробки, а також за ступенем, її достовірності. Від того, наскільки швидко і якісно проводиться її обробка, багато в чому залежить успіх фірми на ринку.

***Приклад 1.23***. Оперативною інформацією є:
- кількість виготовлених деталей за годину, зміну, день;
- кількість проданої продукції за день або певний час;
- обсяг сировини від постачальника на початок робочого дня і т.д.

## 1.4 Арифметичні основи цифрових ЕОМ

### 1.4.1 Загальні відомості про системи числення

Практична діяльність людини нерозривно пов'язана з числами, за допомогою яких відображаються найрізноманітніші кількісні співвідношення. Будь-яке число утворюється за допомогою набору символів-цифр. Сукупність символів, за допомогою яких записується система числення, називається ***алфавітом*** системи числення. Кількість символів, що становлять алфавіт, називається його ***розмірністю***. Зображення цифр та запис чисел підпорядковані конкретним правилам що утворюють систему числення. Таким чином, **під системою числення** розуміють сукупність способів найменування, побудови та зображення чисел за допомогою обмеженого набору символів (цифрі), що мають визначене кількісне значення.

Визначення системи числення підкреслює, що її характерними ознаками є:

конкретний обмежений набір символів для зображення чисел;

певний спосіб зображення чисел за допомогою цього набору символів;

визначений спосіб читання чи найменування чисел.

В залежності від способу зображення чисел за допомогою цифр системи числення розподіляються на два класи:

клас непозиційних систем числення;

клас позиційних систем числення.

**Особливістю непозиційних систем числення є те, що вага цифри запишається постійною і не залежить від її місця в запису числа.** Найпоширенішим прикладом непозиційної системи є відома всім алфавітна система запису чисел (римська система числення). Запис чисел тут виконується за допомогою цифр I (один), V (п'ять), X (десять), L (п'ятдесят), C (сто), D (п'ятсот), M (тисяча). Кількісний зміст числа, записаного у римській системі числення утворюється відніманням чи додаванням цифр в залежності від місця їх розташування у запису. Наприклад, число XI означає одинадцять, а IX – дев'ять.

До класу непозиційних систем також відносяться:
- система числення в остатніх класах (СОК);
- одинична;
- давньоєгипетська – для позначення ключових чисел використовували спеціальні значки-ієрогліфи (рис. 1.37).

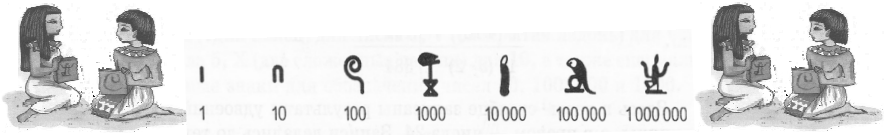



Рисунок 1.37. Давньоєгипетська система числення

Незручності при виконанні арифметичних операцій над числами, записаними у непозиційній системи числення, обмежили їх використання на практиці.

**Система числення є позиційною, якщо цифри у зображенні числа мають різне кількісне значення в залежності від їх позиції у запису числа.** Кожна позиція чи розряд числа має свою вагу, яка визначається застосованою системою числення. В свою чергу, вага позиції визначає кількість цифр, що використовуються для запису чисел. Набір цифр, за допомогою якого утворюються числа у позиційній системі числення, називається основою системи. Основа системи числення визначає її назву. Так, загальновідома десяткова система числення названа так тому, що її основу складають десять цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Якщо основу будуть складати п'ять цифр (наприклад: 0, 1, 2, 3, 4), систему числення назвемо п'ятірковою.

В залежності від основи системи числення визначається вага кожної позиції у запису числа. Це легко зрозуміти, якщо розглянути запис десяткового числа:

$$71625{,}349 = 7\cdot10^4 + 1\cdot10^3 + 6\cdot10^2 + 2\cdot10^1 + 5\cdot10^0 + 3\cdot10^{-1} + 4\cdot10^{-2} + 9\cdot10^{-3}, \quad (1.8)$$

тобто вага позиції визначається цілими степенями основи системи. Кожна позиція у даному прикладі відрізняється від сусідньої рівно у 10 разів.

Позиції цифр у записі числа називаються розрядами числа.

Сказане можна узагальнити і на інші позиційні системи. Позначимо основу системи літерою R. Якщо R<10, для запису чисел в R-ій системі числення можна використовувати звичні цифри 0, 1, 2, …, зберігаючи їх кількісний зміст. При цьому буде збережений і зовнішній вигляд чисел, але їх кількісний зміст буде змінюватись в залежності від обраної основи. Візмемо набір цифр 243,4. Щоб зрозуміти, яке число у він утворює, треба

вказати значення R. Це дозволить записати число у розгорнутій формі і визначити його кількісний зміст. Наприклад, R = 5, тобто використовуються цифри 0, 1, 2, 3, 4.

Тоді,

$$243{,}4_{(5)} = 2\cdot 5^2 + 4\cdot 5^1 + 3\cdot 5^0 + 4\cdot 5^{-1} = 73{,}8_{(10)} \tag{1.9}$$

(індекс у дужках позначає основу системи).

Якщо R = 8, те ж саме число змінить свій зміст:

$$243{,}4_{(8)} = 2\cdot 8^2 + 4\cdot 8^1 + 3\cdot 8^0 + 4\cdot 8^{-1} = 163{,}5_{(10)} \tag{1.10}$$

Оскільки зміст числа змінюється в залежності від основи R, вже не можна називати його так, як воно виглядає. Наприклад, прочитати число 243,4 як "двісті сорок три та чотири десятих" можна лише у десятковій системі числення, яка є загальновідомою. В інших системах числення число читається як набір цифр: "два, чотири, три, кома, чотири".

Основа R системи числення може бути будь-якою. Але якщо R>10, треба ввести позначення цифр системи, що перевищують звичну цифру 9. Всього для позиційної системи з основою R треба ввести R цифр. Підкреслимо, що кожна цифра будь-якої системи числення може займати лише одну позицію (розряд) числа.

Підсумовуючи викладене, запишемо розгорнуту форму числа A у позиційній, системі числення з основою R:

$$A_{(R)} = \sum_{k=-m}^{n} a_k R^k, \tag{1.11}$$

де буквою *a* позначено будь яку цифру в набору (0, 1, 2, (R-1)) що утворює основу системи. Підкреслимо, що як цифри будь-якої системи числення використовуються звичні цифри десяткової системи : a=0, a=1 і т. д.

Якщо основу складають більш як 10 цифр, цифри десяткової системи доповнюються спеціальними символами.

Формулу (4) можна записати в іншому вигляді, відділяючи цілу та дрібну частину числа:

$$A_{(R)} = \sum_{k=0}^{n-1} a_k R^k + \sum_{k=-1}^{-m} a_k R^k. \tag{1.12}$$

На практиці використовують скорочену форму запису числа, опускаючи запис степенів R. Ціла та дрібни частини числа відокремлюються, як і в десятковій системі числення, комою:

$$A_{(2)} = a_{n-1}a_{n-2}...a_0 a_{-1}a_{-2}...a_{-m}, \tag{1.13}$$

де як коефіцієнти $a_{n-1}a_{n-2}...a_0 a_{-1}a_{-2}...a_{-m}$ можуть стояти будь-які з R цифр, використовуваних у системі числення: для двійкової системи

(R=2) це – 0, 1; для вісімкової (R=8) – 0, 1, 2, 3, 4, 5, 6, 7; для шістнадцяткової (R=16) – 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F (A відповідає 10, B – 11, C – 12, D – 13, E – 14, F – 15).

***Приклад 1.24.***

Десяткове число $175 = 1 \cdot 10^{0^{2}} + 7 \cdot 10^{1} + 5 \cdot 10$

Суми $100 + 70 + 5 = 175$

У шістнадцяткової системі число 175 буде мати вигляд:

$10 \cdot 16^{1} + 15 \cdot 16^{0} = AF_{16}$

Суми $160 + 15 = 175$

У вісімкової системі йому відповідає число 257: $2 \cdot 8^{2} + 5 \cdot 8^{1} + 7 \cdot 8^{0} = 257_{8}$

Суми $128 + 40 + 7 = 175$

а в двійкової формі – число 10101111:

$1 \cdot 2^{7} + 0 \cdot 2^{6} + 1 \cdot 2^{5} + 0 \cdot 2^{4} + 1 \cdot 2^{3} + 1 \cdot 2^{2} + 1 \cdot 2^{1} + 1 \cdot 2^{0} = 10101111_{2}$

суми $128 + 0 + 32 + 0 + 8 + 4 + 2 + 1 = 175$

Для перекладу двійкового числа у вісімкове або шістнадцяткове варто розбити двійкове число праворуч ліворуч на тріади або тетради відповідно і кожну з них перевести в число

$$\begin{array}{cccccc} 10 & 101 & 111 & 1010 & & 1111 \\ 2 & 5 & 7 & 10 = A & & 15 = F \end{array} \qquad (1.14)$$

### 1.4.2 Системи кодування інформації в ЕОМ

Для автоматизації роботи з даними, що відносяться до різних типів, дуже важливо уніфікувати їхню форму представлення – для цього звичайно використовується прийом *кодування,* тобто вираження даних одного типу через дані іншого типу. Природні людські *мови* – це не що інше, як системи кодування понять для вираження думок за допомогою мови. До мов близько примикають *абетки* (системи кодування компонентів мови за допомогою графічних символів). Історія знає цікаві, хоча і безуспішні спроби створення "універсальних" мов і абеток. Очевидно, безуспішність спроб їхнього впровадження зв'язана з тим, що національні і соціальні утворення природним чином розуміють, що зміна системи кодування суспільних даних неодмінно приводить до зміни суспільних методів (тобто норм права і моралі), а це може бути пов'язане із соціальними потрясіннями.

Та ж проблема універсального засобу кодування досить успішно реалізується в окремих галузях техніки, науки і культури. Як приклади можна привести систему запису математичних виражень, телеграфну абетку, морську флажковую абетку, систему Брайля для сліпих і багато чого іншого (рис. 1.38).

Своя система існує й в обчислювальній техніці – вона називається *двійковим кодуванням* і заснована на представленні даних послідовністю

усього двох знаків: 0 та 1. Ці знаки називаються *двійковими цифрами,* по-англійському – *binary digit,* або, скорочено, *bit (біт)*. **Біт** – найменша міра інформації і найменша величина, за допомогою якої інформацію можна перевести в дані.

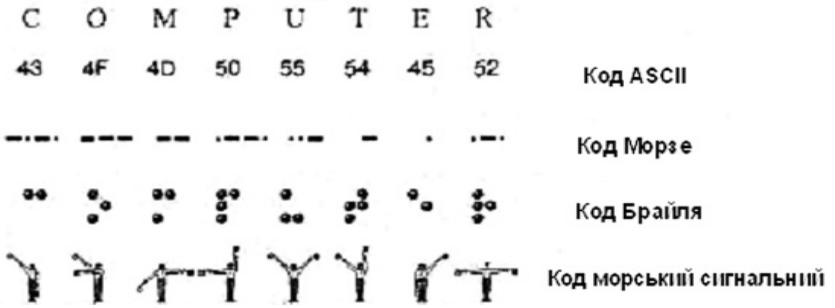



Рисунок 1.38. Приклади різних систем кодування

Одним бітом можуть бути виражені два поняття: 0 або 1 *(так* чи *ні, чорна* або *біле, істина* або *неправда* і т.п.). Якщо кількість бітов збільшити до двох, то вже можна виразити чотири різних поняття:

$$00 \quad 01 \quad 10 \quad 11 \tag{1.15}$$

Трьома бітами можна закодувати вісьмох різних значень:

$$000 \; 001 \; 010 \; 011 \; 100 \; 101 \; 110 \; 111 \tag{1.16}$$

Таким чином, збільшуючи на одиницю кількість розрядів у системі двійкового кодування, ми збільшуємо в два рази кількість значень, що може бути виражене в даній системі.

Для кодування цілих чисел від 0 до 255 досить мати 8 розрядів двійкового коду (8 біт).

$$\begin{aligned} &0000\;0000 = 0 \\ &0000\;0001 = 1 \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ &1111\;1110 = 254 \\ &1111\;1111 = 255 \end{aligned} \tag{1.17}$$

Шістнадцять біт дозволяють закодувати цілі числа від 0 до 65535, а 24 біта – уже більш 16,5 мільйонів різних значень.

Для кодування дійсних чисел використовують 80-розрядне кодування.

### 1.4.2.1 Кодування символьних (текстових) даних

Спочатку в ЕОМ використовувався лише латинській алфавіт (26 букв). Для позначення кожного символу вистачило би п'яти розрядів (бітів). Але в тексті є знаки пунктуації, десяткові цифри т.ін. Тому в перших англо-

мовних комп'ютерах байт – машинний склад – включав шість бітів. Потім сім – не тільки для того, щоб відрізнити заглавні літери від малих, але і для збільшення числа кодів керування принтерами, сигнальними лампочками и іншим обладнанням.

Якщо кожному символу алфавіту зіставити визначене ціле число (наприклад порядковий номер), то за допомогою двійкового коду можна кодувати і текстову інформацію. Восьми двійкових розрядів досить для кодування 256 різних символів. Цього вистачить, щоб виразити різними комбінаціями всі символи англійського та російського алфавіту, як рядкові, так і прописні, а також розділові знаки, символи основних арифметичних дій і деякі загальноприйняті спеціальні символи, наприклад символ "§".

Технічно це виглядає дуже просто, однак, завжди існували досить вагомі організаційні складності. В перші роки розвитку обчислювальної техніки вони були зв'язані з відсутністю необхідних стандартів, а в даний час викликані, навпаки, вагомою кількістю одночасно діючих і суперечливих стандартів. Для того щоб увесь світ однаково кодував текстові дані, потрібні єдині таблиці кодування, а це поки неможливо через протиріччя між символами національних алфавітів, та протиріч корпоративного характеру.

Для англійської мови, що захопили де-факто нішу міжнародного засобу спілкування, протиріччя вже зняті. Інститут стандартизації США *(ANSI – American National Standard Institute)* ввів у дію систему кодування *ASCII (American Standard Code for Information Interchange – стандартний код інформаційного обміну США).* У системі *ASCII* закріплені дві таблиці кодування: *базова* і *розширена* (рис. 1.39). Базова таблиця закріплює значення кодів від 0 до 127, а розширена відноситься до символів з номерами від 128 до 255.

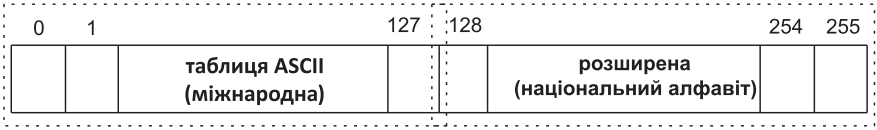



Рисунок 1.39. Система кодування *ASCII*

Перші 32 коди базової таблиці (табл. 1.4а), починаючи з нульового, віддані виробникам апаратних засобів (у першу чергу виробникам комп'ютерів і друкувальних пристроїв).

У цій області розміщаються так називані *керуючі коди,* яким не відповідають ніякі символи мови, і, відповідно, ці коди не виводяться ні на екран, ні на пристрої друку, але за допомогою них можна керувати тим, як виробляється висновок інших даних (0-31 керуючі символи: 7 – дзвоник, 10 – новий рядок, 13 – повернення каретки, 27 – Esc).

Таблиця 1.4а

**Базова таблиця кодування ASCII (коди символів (0-31), що управляють)**

| Код | Обозначение | Клавиша | Значение | Отображаемый символ |
|---|---|---|---|---|
| 0 | nul | ^@ | Нуль | |
| 1 | Soh | ^A | Начало заголовка | ☺ |
| 2 | Stx | ^B | Начало текста | ● |
| 3 | Etx | ^C | Конец текста | ♥ |
| 4 | Eot | ^D | Конец передачи | ♦ |
| 5 | Enq | ^E | Запрос | ♣ |
| 6 | Ack | ^F | Подтверждение | ♠ |
| 7 | Bel | ^G | Сигнал (звонок) | • |
| 8 | Bs | ^H | Забой (шаг назад) | ◘ |
| 9 | Ht | ^I | Горизонтальная табуляция | ○ |
| 10 | Lf | ^J | Перевод строки | ■ |
| 11 | Vt | ^K | Вертикальная табуляция | ♂ |
| 12 | Ff | ^L | Новая страница | ♀ |
| 13 | Cr | ^M | Возврат каретки | ♪ |
| 14 | So | ^N | Выключить сдвиг | ♫ |
| 15 | Si | ^O | Включить сдвиг | ☼ |
| 16 | Dle | ^P | Ключ связи данных | ► |
| 17 | dc1 | ^Q | Управление устройством 1 | ◄ |
| 18 | dc2 | ^R | Управление устройством 2 | ↕ |
| 19 | dc3 | ^S | Управление устройством 3 | ‼ |
| 20 | dc4 | ^T | Управление устройством 4 | ¶ |
| 21 | nak | ^U | Отрицательное подтверждение | § |
| 22 | syn | ^V | Синхронизация | ▬ |
| 23 | etb | ^W | Конец передаваемого блока | ↨ |
| 24 | can | ^X | Отказ | ↑ |
| 25 | em | ^Y | Конец среды | ↓ |
| 26 | sub | ^Z | Замена | → |
| 27 | esc | ^[ | Ключ | ← |
| 28 | fs | ^\ | Разделитель файлов | ∟ |
| 29 | gs | ^] | Разделитель группы | ↔ |
| 30 | rs | ^^ | Разделитель записей | ▲ |
| 31 | us | ^_ | Разделитель модулей | ▼ |

Починаючи з коду 32 по код 127 (табл. 1.4б) розміщені коди символів англійського алфавіту (65-90 великі латинські літери A-Z, 97-122 рядкові латинські літери a-z), розділових знаків (знаки пунктуації: . , : ; ! ?), цифр (48-57 цифри від 0,…,9), арифметичних дій ( +, -, *, /, (), {}, []) і деяких допоміжних символів. Базова таблиця кодування *ASCII* приведена в таблицях 1.4а, 1.4б.

Таблиця 1.4**б**

**Базова таблиця кодування ASCII (символи з кодами (32-127))**

| Код | Символ | Код | Символ | Код | Символ | Код | Символ |
|---|---|---|---|---|---|---|---|
| 32 | пробел | 56 | 8 | 80 | P | 104 | H |
| 33 | ! | 57 | 9 | 81 | Q | 105 | I |
| 34 | “ | 58 | : | 82 | R | 106 | J |
| 35 | # | 59 | ; | 83 | S | 107 | K |
| 36 | $ | 60 | < | 84 | T | 108 | L |
| 37 | % | 61 | = | 85 | U | 109 | m |
| 38 | & | 62 | > | 86 | V | 110 | n |
| 39 | ' | 63 | ? | 87 | W | 111 | o |
| 40 | ( | 64 | @ | 88 | X | 112 | p |
| 41 | ) | 65 | A | 89 | Y | 113 | q |
| 42 | * | 66 | B | 90 | Z | 114 | r |
| 43 | + | 67 | C | 91 | [ | 115 | s |
| 44 | , | 68 | D | 92 | \ | 116 | t |
| 45 | - | 69 | E | 93 | ] | 117 | u |
| 46 |  | 70 | F | 94 | ^ | 118 | v |
| 47 | / | 71 | G | 95 | _ | 119 | w |
| 48 | 0 | 72 | H | 96 | ` | 120 | x |
| 49 | 1 | 73 | I | 97 | A | 121 | y |
| 50 | 2 | 74 | J | 98 | b | 122 | z |
| 51 | 3 | 75 | K | 99 | c | 123 | { |
| 52 | 4 | 76 | L | 100 | d | 124 | \| |
| 53 | 5 | 77 | M | 101 | e | 125 | } |
| 54 | 6 | 78 | N | 102 | f | 126 | ~ |
| 55 | 7 | 79 | O | 103 | g | 127 | del |

Аналогічні системи кодування текстових даних були розроблені й в інших країнах. Так, наприклад, у СРСР у цій області діяла система кодування КОИ-7 *(код обміну інформацією, семизначний).* Однак підтримка

виробників устаткування і програм вивела американський код *ASCII* на рівень міжнародного стандарту, і національним системам кодування довелося "відступити" у другу, розширену частину системи кодування, що визначають значення кодів з 128 по 255 (для киритиці: **CP-866** – для системи *MS DOS*, **CP-1251** – для системи *Windows*, **КОИ8-Р** – для системи *UNIX*).

Відсутність єдиного стандарту в цій області призвело до множинності одночасно діючих кодувань. Тільки в Росії можна вказати три діючі стандарти кодування і ще два застарілих.

Так, наприклад, кодування символів російської мови, відома як кодування *Windows-1251,* була введена "ззовні" – компанією Microsoft, але, з огляду на широке поширення операційних систем і інших продуктів цієї компанії в Росії, вона закріпилася і знайшла широке розповсюдження (таблиця 1.5). Це кодування використовується на більшості локальних комп'ютерів, що працюють на платформі Windows. Де-факто вона стала стандартною в російському секторі World Wide Web.

Таблиця 1.5

**Кодировка Windows-1251 (синоним CP1251)**

| | .0 | .1 | .2 | .3 | .4 | .5 | .6 | .7 | .8 | .9 | .A | .B | .C | .D | .E | .F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8. | Ђ 402 | Ѓ 403 | ‚ 201A | ѓ 453 | „ 201E | … 2026 | † 2020 | ‡ 2021 | € 20AC | ‰ 2030 | Љ 409 | ‹ 2039 | Њ 40A | Ќ 40C | Ћ 40B | Џ 40F |
| 9. | ђ 452 | ‘ 2018 | ’ 2019 | “ 201C | ” 201D | • 2022 | – 2013 | — 2014 | | ™ 2122 | љ 459 | › 203A | њ 45A | ќ 45C | ћ 45B | џ 45F |
| A. | 0A0 | Ў 40E | ў 45E | J 408 | ¤ A4 | Ґ 490 | ¦ A6 | § A7 | Ё 401 | © A9 | Є 404 | « AB | ¬ AC | AD | ® AE | Ї 407 |
| B. | ° B0 | ± B1 | I 406 | і 456 | ґ 491 | µ B5 | ¶ B6 | · B7 | ё 451 | № 2116 | є 454 | » BB | j 458 | S 405 | s 455 | ї 457 |
| C. | А 410 | Б 411 | В 412 | Г 413 | Д 414 | Е 415 | Ж 416 | З 417 | И 418 | Й 419 | К 41A | Л 41B | М 41C | Н 41D | О 41E | П 41F |
| D. | Р 420 | С 421 | Т 422 | У 423 | Ф 424 | Х 425 | Ц 426 | Ч 427 | Ш 428 | Щ 429 | Ъ 42A | Ы 42B | Ь 42C | Э 42D | Ю 42E | Я 42F |
| E. | а 430 | б 431 | в 432 | г 433 | д 434 | е 435 | ж 436 | з 437 | и 438 | й 439 | к 43A | л 43B | м 43C | н 43D | о 43E | п 43F |
| F. | р 440 | с 441 | т 442 | у 443 | ф 444 | х 445 | ц 446 | ч 447 | ш 448 | щ 449 | ъ 44A | ы 44B | ь 44C | э 44D | ю 44E | я 44F |

Інше розповсюджене кодування має ім'я КОИ-8 *(код обміну інформацією, восьмизначный)* – її походження відноситься до часів дії Ради Економічної Взаємодопомоги держав Східної Європи (таблиця 1.6). На базі цього кодування нині діють кодування КОИ8-Р (російська) і КОИ8-У (українська). Сьогодні кодування КОИ8-р має широке поширення в

комп'ютерних мережах на території Росії й у деяких службах російського сектора Інтернету. Зокрема, у Росії вона де-факто є стандартною в повідомленнях електронної пошти і телеконференцій.

Таблиця 1.6

**Кодування КОИ-8**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ─ 128 | │ 129 | ┌ 130 | ┐ 131 | └ 132 | ┘ 133 | ├ 134 | ┤ 135 | ┬ 136 | ┴ 137 | ┼ 138 | ▀ 139 | ▄ 140 | ▉ 141 | ▌ 142 | ▐ 143 |
| ░ 144 | ▒ 145 | ▓ 146 | ⌠ 147 | ■ 148 | ∙ 149 | √ 150 | ≈ 151 | ≤ 152 | ≥ 153 | nbsp 154 | ⌡ 155 | ° 156 | ² 157 | · 158 | ÷ 159 |
| ═ 160 | ║ 161 | ╒ 162 | ё 163 | є 164 | ╔ 165 | і 166 | ї 167 | ╗ 168 | ╘ 169 | ╙ 170 | ╚ 171 | ╛ 172 | ╜ 173 | ╝ 174 | ╞ 175 |
| ╟ 176 | ╠ 177 | ╡ 178 | Ё 179 | Є 180 | ╣ 181 | І 182 | Ї 183 | ╦ 184 | ╧ 185 | ╨ 186 | ╩ 187 | ╤ 188 | ╥ 189 | ╬ 190 | © 191 |
| ю 192 | а 193 | б 194 | ц 195 | д 196 | е 197 | ф 198 | г 199 | х 200 | и 201 | й 202 | к 203 | л 204 | м 205 | н 206 | о 207 |
| п 208 | я 209 | р 210 | с 211 | т 212 | у 213 | ж 214 | в 215 | ь 216 | ы 217 | з 218 | ш 219 | э 220 | щ 221 | ч 222 | Ъ 223 |
| Ю 224 | А 225 | Б 226 | Ц 227 | Д 228 | Е 229 | Ф 230 | Г 231 | Х 232 | И 233 | Й 234 | К 235 | Л 236 | М 237 | Н 238 | О 239 |
| П 240 | Я 241 | Р 242 | С 243 | Т 244 | У 245 | Ж 246 | В 247 | Ь 248 | Ы 249 | З 250 | Ш 251 | Э 252 | Щ 253 | Ч 254 | Ъ 255 |

ISO *(International Standard Organization – Міжнародний інститут стандартизації)* є міжнародним стандартом, у якому передбачене кодування символів російського алфавіту. На практиці дане кодування використовується рідко (таблиця 1.7).

На комп'ютерах, що працюють в операційних системах *MS-DOS,* можуть діяти ще два кодування (кодування *ДСТ* і кодування *Дст-альтернативна)*. Перше з них вважалося застарілим навіть у перші роки появи персональної обчислювальної техніки, але друга використовується і донині (таблиця 1.8).

У зв'язку з достатком систем кодування текстових даних, які діють у Росії, виникає задача міжсистемного перетворення даних – це одна з розповсюджених задач інформатики.

**Універсальна система кодування текстових даних**

Якщо проаналізувати організаційні труднощі, зв'язані зі створенням єдиної системи кодування текстових даних, то можна прийти до висновку, що вони викликані обмеженим набором кодів (256). У той же час очевидно, що якщо, наприклад, кодувати символи не восьмірозрядними двійковими числами, а числами з великою кількістю розрядів, те і діапазон можливих значень кодів стане набагато більше. А якщо об'єднати всі символи в одну таблицю?

Таблиця 1.7

**Кодування ISO**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| sp 32 | ! 33 | " 34 | # 35 | $ 36 | % 37 | & 38 | ' 39 | ( 40 | ) 41 | * 42 | + 43 | , 44 | – 45 | . 46 | / 47 |
| 0 48 | 1 49 | 2 50 | 3 51 | 4 52 | 5 53 | 6 54 | 7 55 | 8 56 | 9 57 | : 58 | ; 59 | < 60 | = 61 | > 62 | ? 63 |
| @ 64 | A 65 | B 66 | C 67 | D 68 | E 69 | F 70 | G 71 | H 72 | I 73 | J 74 | K 75 | L 76 | M 77 | N 78 | O 79 |
| P 80 | Q 81 | R 82 | S 83 | T 84 | U 85 | V 86 | W 87 | X 88 | Y 89 | Z 90 | [ 91 | \ 92 | ] 93 | ^ 94 | _ 95 |
| ` 96 | a 97 | b 98 | c 99 | d 100 | e 101 | f 102 | g 103 | h 104 | i 105 | j 106 | k 107 | l 108 | m 109 | n 110 | o 111 |
| p 112 | q 113 | r 114 | s 115 | t 116 | u 117 | v 118 | w 119 | x 120 | y 121 | z 122 | { 123 | \| 124 | } 125 | ~ 126 | |
| 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 | 137 | 138 | 139 | 140 | 141 | 142 | 143 |
| 144 | 145 | 146 | 147 | 148 | 149 | 150 | 151 | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 |
| nbsp 160 | Ё 161 | Ђ 162 | Ѓ 163 | Є 164 | S 165 | I 166 | Ї 167 | J 168 | Љ 169 | Њ 170 | Ћ 171 | Ќ 172 | shy 173 | Ў 174 | Џ 175 |
| А 176 | Б 177 | В 178 | Г 179 | Д 180 | Е 181 | Ж 182 | З 183 | И 184 | Й 185 | К 186 | Л 187 | М 188 | Н 189 | О 190 | П 191 |
| Р 192 | С 193 | Т 194 | У 195 | Ф 196 | Х 197 | Ц 198 | Ч 199 | Ш 200 | Щ 201 | Ъ 202 | Ы 203 | Ь 204 | Э 205 | Ю 206 | Я 207 |
| а 208 | б 209 | в 210 | г 211 | д 212 | е 213 | ж 214 | з 215 | и 216 | й 217 | к 218 | л 219 | м 220 | н 221 | о 222 | п 223 |
| р 224 | с 225 | т 226 | у 227 | ф 228 | х 229 | ц 230 | ч 231 | ш 232 | щ 233 | ъ 234 | ы 235 | ь 236 | э 237 | ю 238 | я 239 |
| № 240 | ё 241 | ђ 242 | ѓ 243 | є 244 | s 245 | i 246 | ї 247 | j 248 | љ 249 | њ 250 | ћ 251 | ќ 252 | § 253 | ў 254 | џ 255 |

Таблица 1.8

**Дст-альтернативное кодирование (синоним CP866)**

| | .0 | .1 | .2 | .3 | .4 | .5 | .6 | .7 | .8 | .9 | .A | .B | .C | .D | .E | .F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8. | А 410 | Б 411 | В 412 | Г 413 | Д 414 | Е 415 | Ж 416 | З 417 | И 418 | Й 419 | К 41A | Л 41B | М 41C | Н 41D | О 41E | П 41F |
| 9. | Р 420 | С 421 | Т 422 | У 423 | Ф 424 | Х 425 | Ц 426 | Ч 427 | Ш 428 | Щ 429 | Ъ 42A | Ы 42B | Ь 42C | Э 42D | Ю 42E | Я 42F |
| A. | а 430 | б 431 | в 432 | г 433 | д 434 | е 435 | ж 436 | з 437 | и 438 | й 439 | к 43A | л 43B | м 43C | н 43D | о 43E | п 43F |
| B. | ▒ 2591 | ▓ 2592 | ▓ 2593 | ┃ 2502 | ┤ 2524 | ╡ 2561 | ╢ 2562 | ╖ 2556 | ╕ 2555 | ╣ 2563 | ║ 2551 | ╗ 2557 | ╝ 255D | ╜ 255C | ╛ 255B | ┐ 2510 |
| C. | └ 2514 | ┴ 2534 | ┬ 252C | ├ 251C | ─ 2500 | ┼ 253C | ╞ 255E | ╟ 255F | ╚ 255A | ╔ 2554 | ╩ 2569 | ╦ 2566 | ╠ 2560 | ═ 2550 | ╬ 256C | ╧ 2567 |
| D. | ╨ 2568 | ╤ 2564 | ╥ 2565 | ╙ 2559 | ╘ 2558 | ╒ 2552 | ╓ 2553 | ╫ 256B | ╪ 256A | ┘ 2518 | ┌ 250C | ▉ 2588 | ▄ 2584 | ▌ 258C | ▐ 2590 | ▀ 2580 |
| E. | р 440 | с 441 | т 442 | у 443 | ф 444 | х 445 | ц 446 | ч 447 | ш 448 | щ 449 | ъ 44A | ы 44B | ь 44C | э 44D | ю 44E | я 44F |
| F. | Ё 401 | ё 451 | Є 404 | є 454 | Ї 407 | ї 457 | Ў 40E | ў 45E | ° B0 | ∙ 2219 | · B7 | √ 221A | № 2116 | ¤ A4 | ■ 25A0 | A0 |




В кінці минулого століття з'явився новий міжнародний стандарт Unicode, в якому один символ представляється двобайтовим двійковим кодом. Застосування цього стандарту – продовження розробки універсального міжнародного стандарту, що дозволяє розв'язати проблему сумісності національних кодувань символів.

Така система, заснована на 16-розрядному кодуванні символів, одержала назву *універсальної – UNICODE.* Шістнадцять розрядів дозволяють забезпечити унікальні коди для 65536 різних символів – цього полючи досить для розміщення в одній таблиці символів більшості мов планети (110 182 символа (2012); кожному символу присвоєн код кириллица: А – 041016, Б – 041116, а – 043016, б – 043116, коди 0..10FFFF16, всего 1 114 112, загальновживані символи 0..65535 = 216-1 (0..FFFF16)).

Таблиця 1.9

**Кодова таблиця 0400 стандарту Unicode**

| | 01 Ё | 02 Ђ | 03 Ѓ | 04 Є | 05 Ѕ | 06 І | 07 Ї | 08 Ј | 09 Љ | 0A Њ | 0B Ћ | 0C Ќ | | 0E Ў | 0F Џ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 А | 11 Б | 12 В | 13 Г | 14 Д | 15 Е | 16 Ж | 17 З | 18 И | 19 Й | 1A К | 1B Л | 1C М | 1D Н | 1E О | 1F П |
| 20 Р | 21 С | 22 Т | 23 У | 24 Ф | 25 Х | 26 Ц | 27 Ч | 28 Ш | 29 Щ | 2A Ъ | 2B Ы | 2C Ь | 2D Э | 2E Ю | 2F Я |
| 30 а | 31 б | 32 в | 33 г | 34 д | 35 е | 36 ж | 37 з | 38 и | 39 й | 3A к | 3B л | 3C м | 3D н | 3E о | 3F п |
| 40 р | 41 с | 42 т | 43 у | 44 ф | 45 х | 46 ц | 47 ч | 48 ш | 49 щ | 4A ъ | 4B ы | 4C ь | 4D э | 4E ю | 4F я |
| | 51 ё | 52 ђ | 53 ѓ | 54 є | 55 ѕ | 56 і | 57 ї | 58 ј | 59 љ | 5A њ | 5B ћ | 5C ќ | | 5E ў | 5F џ |
| 60 Ѡ | 61 ѡ | 62 Ѣ | 63 ѣ | 64 Ѥ | 65 ѥ | 66 Ѧ | 67 ѧ | 68 Ѩ | 69 ѩ | 6A Ѫ | 6B ѫ | 6C Ѭ | 6D ѭ | 6E Ѯ | 6F ѯ |
| 70 Ѱ | 71 ѱ | 72 Ѳ | 73 ѳ | 74 Ѵ | 75 ѵ | 76 Ѷ | 77 ѷ | 78 Ѹ | 79 ѹ | 7A Ѻ | 7B ѻ | 7C Ѽ | 7D ѽ | 7E Ѿ | 7F ѿ |
| 80 Ҁ | 81 ҁ | 82 ҂ | | | | | | | | | | | | | |
| 90 Ґ | 91 ґ | 92 Ғ | 93 ғ | 94 Ҕ | 95 ҕ | 96 Җ | 97 җ | 98 Ҙ | 99 ҙ | 9A Қ | 9B қ | 9C Ҝ | 9D ҝ | 9E Ҟ | 9F ҟ |
| A0 Ҡ | A1 ҡ | A2 Ң | A3 ң | A4 Ҥ | A5 ҥ | A6 Ҧ | A7 ҧ | A8 Ҩ | A9 ҩ | AA Ҫ | AB ҫ | AC Ҭ | AD ҭ | AE Ү | AF ү |
| B0 Ұ | B1 ұ | B2 Ҳ | B3 ҳ | B4 Ҵ | B5 ҵ | B6 Ҷ | B7 ҷ | B8 Ҹ | B9 ҹ | BA Һ | BB һ | BC Ҽ | BD ҽ | BE Ҿ | BF ҿ |
| C0 Ӏ | C1 Ӂ | C2 ӂ | C3 Ӄ | C4 ӄ | | | C7 Ӈ | C8 ӈ | | | CB Ӌ | CC ӌ | | | |
| D0 Ӑ | D1 ӑ | D2 Ӓ | D3 ӓ | D4 Ӕ | D5 ӕ | D6 Ӗ | D7 ӗ | D8 Ә | D9 ә | DA Ӛ | DB ӛ | DC Ӝ | DD ӝ | DE Ӟ | DF ӟ |
| E0 Ӡ | E1 ӡ | E2 Ӣ | E3 ӣ | E4 Ӥ | E5 ӥ | E6 Ӧ | E7 ӧ | E8 Ө | E9 ө | EA Ӫ | EB ӫ | | | EE Ӯ | EF ӯ |
| F0 Ӱ | F1 ӱ | F2 Ӳ | F3 ӳ | F4 Ӵ | F5 ӵ | | | F8 Ӹ | F9 ӹ | | | | | | |

Незважаючи на тривіальну очевидність такого підходу, простий механічний перехід на дану систему, довгий час стримувався через

недостатні ресурсних засобів обчислювальної техніки (у системі кодування *UNICODE* усі текстові документи автоматично стають удвічі довше). В другій половині 90-х років технічні засоби досягли необхідного рівня забезпеченості ресурсами, і сьогодні ми спостерігаємо поступовий перехід документів і програмних засобів на універсальну систему кодування. Для індивідуальних користувачів це ще більше додало турбот за узгодженням документів, виконаних у різних системах кодування, із програмними засобами, але це треба сприймати як труднощі перехідного періоду.

До переваг можна віднести одночасне використовування символів різних мов (Інтернет). До недоліків: розмір файлу збільшується.

1.4.2.2 Кодування графічних даних

Графічна інформація, представлена у вигляді малюнків, фотографій, слайдів, рухливих зображень (анімація, відео), схем, креслень, може створюватися і редагуватися за допомогою комп'ютера, при цьому вона відповідним чином кодується. Нині існує досить велика кількість прикладних програм для обробки графічної інформації, але усі вони реалізують три види комп'ютерної графіки:

- ✧ растрову;
- ✧ векторну;
- ✧ фрактальну.

Якщо розглянути за допомогою збільшувального скла чорно-біле графічне зображення, надруковане в газеті або книзі, то можна побачити, що воно складається з дрібних крапок, що утворять характерний візерунок, який має назву *растр* (рис. 1.37), або пильніше розглянути графічне зображення на екрані монітора комп'ютера, то можна побачити велику кількість різноколірних точок (пікселів – від англійської "pixel", утвореного від "picture element" – елемент зображення), які, будучи зібрані разом, і утворюють це графічне зображення. **Піксель** – це найменший елемент рисунка, для якого можна незалежно встановити колір (**роздільна здатність**: число пікселів на дюйм, *pixels per inch (ppi*): е**кран 96** ppi, **друк 300-600** ppi, **типографія 1200** ppi).

Скільки пам'яті потрібно для зберігання кольору 1 пікселя? R: 256=28 варіантів, потрібно 8 біт = 1 байт. R G B: всього 3 байта.

З цього можна зробити висновок: графічне зображення в комп'ютері певним чином кодується і повинно бути представлено у вигляді графічного файлу.

Файл є основною структурною одиницею організації і зберігання даних в комп'ютері і в даному випадку повинен містити інформацію про те, як представити цей набір точок на екрані монітора.

Файли, створені на основі *векторної графіки*, містять інформацію у вигляді математичних залежностей (математичних функцій, які описують

лінійні залежності) і відповідних даних про те, як побудувати зображення об'єкта за допомогою відрізків ліній (векторів) при виведенні його на екран монітора комп'ютера.

Файли, створені на основі *растрової графіки*, припускають зберігання даних про кожну окрему точку зображення. Для відображення растрової графіки не потрібно складні математичні розрахунки, досить лише отримати дані про кожну точку зображення (її координати і колір) і відобразити їх на екрані монітора комп'ютера (рис. 1.40).

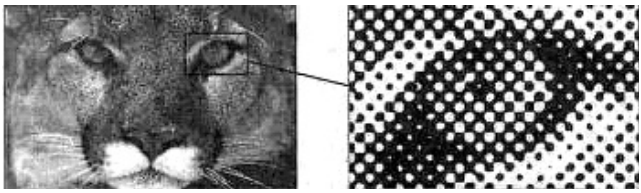


Рисунок 1.40. Растр – це метод кодування графічної інформації, здавна прийнятий у поліграфії

Оскільки лінійні координати й індивідуальні властивості кожної крапки (яскравість) можна виразити за допомогою цілих чисел, то можна сказати, що растрове кодування дозволяє використовувати двійковий код для представлення графічних даних. Загальноприйнятим на сьогоднішній день вважається представлення чорно-білих ілюстрацій у виді комбінації крапок з 256 градаціями сірого кольору, і, таким чином, для кодування яскравості будь-якої крапки звичайно досить восьмирозрядного двійкового числа.

В процесі кодування зображення робиться його просторова дискретизація, тобто зображення розбивається на окремі точки і кожній точці задається код кольору (жовтий, червоний, синій і так далі).

Для кодування кольорових графічних зображень застосовується *принцип декомпозиції* довільного кольору на основні складові. У якості таких складових використовують три основних кольори: червоний *(Red, R),* зелений *(Green, G)* і синій *(Blue, У)*. На практиці вважається (хоча теоретично це не зовсім так), що будь-який колір точки, який сприймається людським оком, можна отримати шляхом адитивного (пропорційного) складання (змішення) трьох основних кольорів – червоного, зеленого і синього. Така система кодування називається колірною системою RGB по перших буквах назв основних кольорів.

Файли графічних зображень, в яких застосовується колірна система RGB, представляють кожну точку зображення у вигляді колірного триплета – трьох числових величин R, G і B, що відповідають інтенсивностям червоного, зеленого і синього кольорів.

Процес кодування графічного зображення здійснюється за допомогою різних технічних засобів (сканера, цифрового фотоапарата, цифрової відеокамери і так далі); в результаті отримують растрове зображення. При відтворенні кольорових графічних зображень на екрані кольорового монітора комп'ютера колір кожної точки (пікселя) такого зображення виходить шляхом змішення трьох основних кольорів R, G і B.

Скільки пам'яті потрібно для зберігання кольору 1 пікселя? R: 256=28 варіантів, потрібно 8 біт = 1 байт. R G B: всього 3 байта.

Якщо для кодування яскравості кожної з основних складових використовувати 256 значень (вісім двійкових розрядів), як це прийнято для напівтонових чорно-білих зображень, то на кодування кольору однієї крапки треба витратити 24 розряду. При цьому система кодування забезпечує однозначне визначення 16,5 млн різних кольорів, що насправді близько до чутливості людського ока.

Скільки різних кольрів можна кодувати? 256·256·256 = 16 777 216 (*True Color*).

Режим представлення кольорової графіки з використанням 24 двійкових розрядів називається *повнокольоровим (True Color)*.

Кожному з основних кольорів можна поставити у відповідність додатковий колір, тобто колір, що доповнює основний колір до білого. Неважко помітити, що для кожного з основних кольорів додатковим буде колір, утворений сумою пари інших основних барв. Відповідно, додатковими кольорами є:

- ✧ голубий *(Cyan, З),*
- ✧ пурпурний *(Magenta, M)*,
- ✧ жовтий (*Yellow, Y*).

Принцип декомпозиції довільного кольору на складові компоненти можна застосовувати не тільки для основних кольорів, але і для додаткових, тобто будь-який колір можна представити у виді суми блакитної, пурпурної та жовтої складовий. Такий метод кодування кольору прийнятий у поліграфії, але в поліграфії використовується ще і четверта фарба – чорна *(Black, K)*. Тому дана система кодування позначається чотирма буквами *CMYK* (чорний колір позначається буквою К, щоб не плутати з *В в* моделі *RGB*), і для представлення кольорової графіки в цій системі треба мати 32 двійкових розрядів. Такий режим теж називається *повнокольоровим (True Color)*.

Якщо зменшити кількість двійкових розрядів, використовуваних для кодування кольору кожної крапки, то можна скоротити обсяг даних, але при цьому діапазон кольорів, які кодуються, помітно скорочується. Кодування кольорової графіки 16-розрядними двійковими числами називається режимом *High Color*.

При кодуванні інформації про колір за допомогою восьми біт даних можна передати тільки 256 колірних відтінків. Такий метод кодування кольору називається *індексним.*

Зміст назви в тім, що, оскільки 256 значень зовсім недостатньо, щоб передати весь діапазон кольорів, доступні для людського ока, код кожної крапки растра виражає не колір сам по собі, а тільки його номер *(індекс)* у якійсь довідковій таблиці, що має назву *палітра* (рис.1.41). Зрозуміло, ця палітра повинна прикладатися до графічних даних – без неї неможливо скористатися методами відтворення інформації на екрані або папері (тобто, скористатися, звичайно, можна, але через неповноту даних отримана інформація не буде адекватною: листя на деревах можуть виявитися червоними, а небо – зеленим).

**Растровое кодування з палітрою**

Крок 1. Вибрати кількість кольрів: 2, 4, … 256.

Крок 2. Вибрати 256 кольорів з палітри:

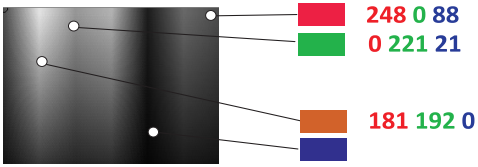
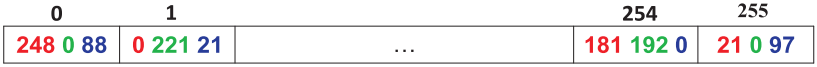



Рисунок 1.41. Палітра

Крок 3. Скласти палітру (кожному кольору – номер 0..255), палітра зберігається на початку файла.




Крок 4. Код пікселя = номеру його кольору в палітрі

2 45 65 14 … 12 23

Файл з палітрою:




Таблиця 1.9

**Форматы файлов (растровые рисунки)**

| Формат | True Color | Палитра | Прозрачность |
|---|---|---|---|
| BMP | + | + | - |
| JPG | + | - | - |
| GIF | - | + | + |
| PNG | + | + | + |

До переваг можна віднести: кращий спосіб для зберігання фотографій і зображень без чітких границь; спецефекти (тіні, ореоли тощо).

Недолики: є втрата інформації; при зміні розмірів рисунка він спотворюється; розмір файлу не залежить від складності рисунка.

**Векторні рисунки**

Векторні рисунки будуються з геометричних фігур:

- ✧ відрізки, ламані, прямокутники;
- ✧ кола, еліпси, дуги;
- ✧ згладжені лінії.

Для кожної фігури в пам'яті зберігаються:

- ✧ розміри і координати на рисунку;
- ✧ колір і стиль границі;
- ✧ колір і стиль заливки (для замкнених фігур).

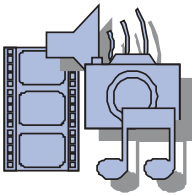
Формати файлів:

- ✧ **WMF** (Windows Metafile);
- ✧ **CDR** (CorelDraw);
- ✧ **AI** (Adobe Illustrator);
- ✧ **SVG** (Inkscape) для Web.

Переваги: кращий спосіб для зберігання креслень, схем, карт; при кодуванні немає втрат інформації; при зміні розміру немає спотворень; менший розмір файла, залежить від складності рисунка.

До недоліків можна віднести: неефективно використовувати для фотографій та розмитих зображень.

### 1.4.2.3 Кодування звукової та відео інформації



Прийоми і методи роботи зі звуковою інформацією прийшли в обчислювальну техніку найбільше пізно. До того ж, на відміну від числових, текстових і графічних даних, у звукозаписів не було настільки ж тривалої і перевіреної історії кодування. У підсумку методи кодування звукової інформації двійковим кодом далекі від стандартизації. Безліч окремих компаній розробили свої корпоративні стандарти.

Існують дві форми подання інформації:

- ✧ **неперервна (аналогова)** – характеризує процес, який не має перерв і теоретично може змінюватися в будь-який момент часу и на любую величину (наприклад, мова людини, музичні твори).
- ✧ **переривиста (цифрова, дискретна)** – цифровий сигнал може змінюватися лише в певні моменти часу і приймає лише задані значення.

Для перетворення аналогового сигналу в цифровий сигнал потрібно провести дискретизацію неперервного сигналу в часі, квантування по рівню, а потім кодування відібраних значень.

**Дискретизація** – заміна неперервного (аналогового) сигналу послідовністю відокремлених по часу відліків цього сигналу (рис. 1.42).

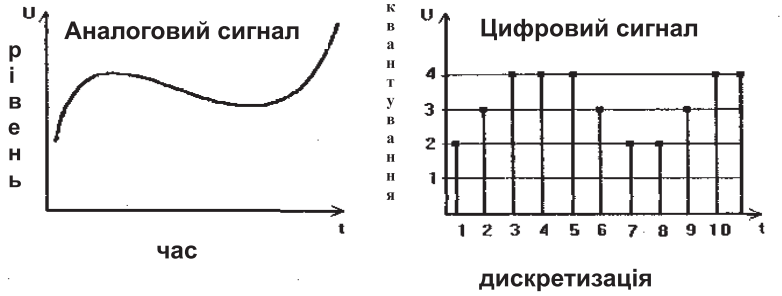



Рисунок 1.42. Процес дискретизації

Після такого перетворення неперервний сигнал подають послідовністю чисел. Показаний на рисунку 1.42 неперервний сигнал замінюється числами 2-3-4-4-4-3-2-2-3-4-4 (так відбувається квантування по рівням). Потім десяткові числа кодуються послідовністю одиниць і нулів.

Чим більше рівнів і вище частота вибірок сигналів, тем вище якість кодування.

Табліця 1.10

Якість звука

| Параметр | Глибина кодування звука | Частота дискретизації |
|---|---|---|
| Радіотрансляція | 8 бит | до 8 кГц |
| Середня якість | 8 або16 біт | від 8 до 48 кГц |
| Звук CD | 16 біт | 48 кГц |

**Кодування відеосигналів.**

При кодуванні відеосигналів застосовуються операції, які забезпечують дуже високу міру стискування.

Цифрове відео є послідовністю цифрових кадрів, тобто фотографій. У випадках, коли відеофільм має порівняно невеликий за сьогоднішніми мірками розмір $512 \times 512$ пікселів, для зберігання 1 секунди фільму потрібно $786432 \times 24 = 18874368$ байт.

Для кодування кожного кадру відеофільму застосовується стискування у форматі JPEG, що навіть при порівняно невеликій втраті інформації може дати помітне (до 10 разів) зменшення розміру. Окрім стискування

кожного кадру, для зменшення розміру відеофільму, що зберігається, використовують деякі додаткові прийоми: проріджування, віднімання, віднімання по блоках, компенсація руху.

В результаті поєднання усіх цих операцій при кодуванні відеосигналу можна отримати дуже високу міру стискування. Усі ці методи кодування реалізовані в сучасних форматах кодування цифрового відео, таких як ***AVI*** (*Audio Video Interleave* – звук і відео, які чергуються), ***MPEG*** (*Motion Picture Expert Group*), ***WMV*** (*Windows Media Video,* формат фірми *Microsoft*), ***MP4*** (***MPEG-4***, стиснуте відео та звук), ***MOV*** (*Quick Time Movie*, формат фірмы *Apple*), ***WebM*** (відкритий формат, підтримується браузерами).

Цифрове відео може бути отримане трьома способами: оцифруванням кіно, що має, знятого на плівку; зйомкою відео на професійну або любительську відеокамеру; створенням відеофайлів в спеціалізованих програмах.

Таким чином, кодування – це, власне кажучи, процес перетворення інформації, вираженої буквами одного алфавіту, і представлення її засобами іншого.

### 1.4.3 Традиційні системи числення

Позиційну систему числення називають ***традиційною***, якщо її базис утворюють члени геометричної прогресії, а значення цифр є цілі ненегативні числа. Традиційні системи числення з основою *R* інакше називають ***R – ичными***. До традиційних систем числення відносяться: змішані, двійкова, трійкова, вісімкова, шістнадцятирична.

**Двійкова система числення**

Відповідно до її назви основу двійкової системи числення складають дві цифри: 0 (нуль) та 1 (одиниця), тобто $R_{(10)} = 2$ чи $R_{(2)} = 10$.

У зв'язку з цим система має значні переваги порівняно з іншими системами числення:

- вона повністю відповідає принципу двійкового кодування інформації в ЕОМ;
- вона має найбільш прості правила виконання арифметичних операцій;
- технічна реалізація системи потребує елементи, що мають лише два усталених стани; один з них відображує нуль, другий-одиницю.

Наведемо приклад числа у двійковій системі числення:

$$A_{(2)} = 101011{,}1011$$

Відповідно до формули (13) визначимо структуру числа, подавши її у вигляді таблиці (табл. 1.11).

Таблиця 1.11

| Двійкове число | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| Номер розряду | 5 | 4 | 3 | 2 | 1 | 0 | -1 | -1 | -3 | -4 |
| Вага розряду | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ |

Щоб визначити кількісний зміст числа А, запишемо його у розгорнутій формі, як це випливає з формули (11). При цьому для зручності показники степенів та основу R запишемо у десятковій системи числення:

$$A = 1\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 + 1\cdot 2^{-1} + 0\cdot 2^{-2} + 1\cdot 2^{-3} + 1\cdot 2^{-4} = \quad (1.18)$$
$$= 32 + 0 + 8 + 0 + 2 + 1 + 0{,}5 + 0 + 0{,}125 + 0{,}0625 = 43{,}6875$$

З розглянутого приклада випливає методичний висновок: щоб зрозуміти кількісний зміст числа, записаного у будь-якій системі числення, його переводять у десяткову систему, оскільки остання є самою звичною для людини.

**Вісімкова система числення**

Як основу системи використовують перші вісім цифр десяткової системи: 0, 1, 2, …, 7, тобто $R_{(10)} = 8$, $R_{(8)} = 10$.

Тому зовнішнє число у вісімковій системі числення виглядає як десяткове число. Але його кількісний зміст буде іншим, оскільки вага позиції визначається степенем вісімки, а не десятки.

Наприклад. $A_{(8)} = 275{,}146$

Структура цього числа розкрита у табл. 1.12.

Таблиця 1.12

| Вісімкове число | 2 | 7 | 5, | 1 | 4 | 6 |
|---|---|---|---|---|---|---|
| Номер розряду | 2 | 1 | 0 | -1 | -2 | -3 |
| Вага розряду | $8^2$ | $8^1$ | $8^0$ | $8^{-1}$ | $8^{-2}$ | $8^{-3}$ |

Розгорнута форма числа має вигляд:

$$A_{(10)} = 2\cdot 8^2 + 7\cdot 8^1 + 5\cdot 8^0 + 1\cdot 8^{-1} + 4\cdot 8^{-2} + 6\cdot 8^{-3} \quad (1.19)$$

давши указані дії в десятковій системі, одержимо кількісне значення цього вісімкового числа:

$$A_{(10)} = 2\cdot 64 + 7\cdot 8 + 5 + \frac{1}{8} + \frac{4}{64} + \frac{6}{512} = 189\frac{102}{512}. \quad (1.20)$$

**Шістнадцяткова система числення**

Основу системи складають шістнадцять цифр, що мають кількісне значення відповідно від нуля до п’ятнадцяти. Як перші десять цифр у системі

використовують звичайні десяткові цифри 0, 1, 2, …, 9. Щоб відобразити цифри "десять", "одинадцять", …, "п'ятнадцять", вводять спеціальні символи. Найпоширенішим варіантом є використання перших великих літер латинського алфавіту: A = 10, B = 11, C = 12, D = 13, E = 14, F = 15. Тому число у шістнацятковій системи числення має трохи незвичайний вигляд: K=5A,3E; L=D8,0C2 і т. д.

Оскільки для неї $R=16_{(10)} = 10_{(16)}$, покажемо структуру числа у вигляді табл. 1.13: R = 2C5,E7.

Таблиця 1.13

| Шістнадцяткове число | 2 | C | 5, | E | 7 |
|---|---|---|---|---|---|
| Номер розряду | 2 | 1 | 0 | -1 | -2 |
| Вага розряду | $16^2$ | $16^1$ | $16^0$ | $16^{-1}$ | $16^{-2}$ |

Тому розгорнута форма числа має вигляд:

$$R = 2 \cdot 16^2 + C \cdot 16^1 + 5 \cdot 16^0 + E \cdot \frac{1}{16^1} + 7 \cdot \frac{1}{16^2} \qquad (1.21)$$

(тут для зручності основа R та показники степенів записані у десятковій системі числення).

Ця форма дає змогу визначити кількісний зміст вибраного числа:

$$R = 2 \cdot 256 + 12 \cdot 16 + 5 \cdot 1 + \frac{14}{16} + \frac{7}{256} = 709\frac{231}{256_{(10)}} \qquad (1.22)$$

Підсумовуючи розглянуте, зробимо декілька висновків.

1. Оскільки зовнішній вигляд чисел, записаних у різних системах числення, може бути однаковим, необхідно обов'язково позначити систему числення в запису числа (наприклад, індексом у дужках).
2. Число у будь-якій системі числення (окрім десяткової) читається як перелік цифр, що його складають, починаючи з старшого розряду: 274 – "вісімкове число два, сім, чотири"; 5A3,8Д – "шістнадцяткове число п'ять, А, три кома, вісім Д".
3. Чим більшою є основа системи R, тим менш розрядів потрібно для запису цілих чисел, однакових за своїм кількісним змістом. Наприклад, число $M_{(2)}= 1101110$, $M_{(8)}= 156$, $M_{(110)}= 110$, $M_{(16)}= 6E$, має однакове кількісне значення.
4. У будь-якій позиційній системі числення зберігається знак числа, тобто є числа додатні та від'ємні.

Наведемо таблицю відповідності чисел у розглянутих системах числення (табл. 1.14).

Таблиця 1.14

**Відповідності чисел у системах числення**

| Система числення | | | |
|---|---|---|---|
| двійкова | вісімкова | десяткова | шістнадцяткова |
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 10 | 2 | 2 | 2 |
| 11 | 3 | 3 | 3 |
| 100 | 4 | 4 | 4 |
| 101 | 5 | 5 | 5 |
| 110 | 6 | 6 | 6 |
| 111 | 7 | 7 | 7 |
| 1000 | 10 | 8 | 8 |
| 1001 | 11 | 9 | 9 |
| 1010 | 12 | 10 | A |
| 1011 | 13 | 11 | B |
| 1100 | 14 | 12 | C |
| 1101 | 15 | 13 | Д |
| 1110 | 16 | 14 | E |
| 1111 | 17 | 15 | F |
| 10000 | 20 | 16 | 10 |
| 10001 | 21 | 17 | 11 |
| 10010 | 22 | 18 | 12 |
| 10011 | 23 | 19 | 13 |
| 10100 | 24 | 20 | 14 |

**Двійково-десяткова система числення**

Ця система не є класичною позиційною системою у тому розумінні, як це викладено вище. Вона залишається десятковою системою числення, але кожний десятковий символ (цифра) записується у двійковій системі числення. Як випливає з табл. 4, для запису цифр від 0 до 9 треба відвести чотири розряди (тетраду). Тому, запис числа у двійково-десятковій системі має вигляд:

$$A_{(10)} = 973{,}25$$
$$A_{(2\text{-}10)} = 1001.0111.0011{,}0010.0101. \qquad (1.23)$$

Двійково-десяткову систему числення називають також двійково-вокодованою системою із кодом 8421, де цифри 8, 4, 2, 1 визначають вагу відповідних розрядів у запису цифр десяткового числа.

Розглянуте двійкове кодування цифр десяткового числа за допомогою двійкової системи числення не е єдиним. Можна запропонувати безліч варіантів подання цифр десяткового числа сукупністю одиниць та нулів. При цьому двійкові коди будуть мати довжину не менш як чотири розряди

(тетради), оскільки необхідно мати десять (по кількості цифр десяткової системи) рівних комбінацій 1 та 0. Прикладами таких двійкових кодів є коди з "надлишком 3", "надлишком 5" та інші.

**Поняття про системи числення в остатніх класах**

Системи числення в остатніх класах (СОК) є непозиційними системами числення і використовуються в ЕОМ для підвищення їх продуктивності.

В цих системах використовуються в ЕОМ для підвищення їх продуктивності.

В цих системах використовують не одну, а декілька основ. Як основу СОК застосовують взаємно прості числа, наприклад, 2, 3, 5, 7, 11 і т.д.

Ціле додатнє число подається у вигляді послідовності остач здобутих в результаті ділення його на сукупність обраних остач СОК.

Припустимо, що основами СОК є цілі додатні числа $r_1, r_2 \ldots r_n$

Тоді будь-яке число А в СОК можна подати у вигляді:

$$A = (L_1, L_2, \ldots, L_n), \tag{1.24}$$

Де $L_i$ – остачі від ділення числа А на основи СОК $r_i$:

$$L_i = A - \left[\frac{A}{r_i}\right] r_i, i \cdot 1,2,\ldots,n \tag{1.25}$$

Тут у квадратних дужках відзначено цілу частину від ділення А на r.

**Наприклад**, задане число А=125 треба подати у СОК, що має основи $r_1=2$, $r_2=3$, $r_3=5$, $r_4=7$

Визначимо остачі $L_i$ (i = 1,2,3,4)

$$L_1 = 125 - \left[\frac{125}{2}\right] 2 = 125 - 124 = 1;$$

$$L_2 = 125 - \left[\frac{125}{3}\right] 3 = 125 - 123 = 2;$$

$$L_3 = 125 - \left[\frac{125}{5}\right] 5 = 125 - 125 = 0;$$

$$L_4 = 125 - \left[\frac{125}{7}\right] 7 = 125 - 119 = 6;$$

Тому задане число А=125 у цій СОК має вигляд:

$$A=(1,2,0,6)$$

Подання числа в остачах буде єдиним, якщо добуток всіх основ СОК перевищує задане число. Для розглянутого приклада діапазон чисел

визначається добутком Д = 2 · 3 · 5 · 7 = 210. Добуток Д=210 визначає, що запис числа в СОК з вибраною основою буде єдиним в діапазоні від 0 до 209. В діапазоні 210-419, 420-629 і т.д. запис чисел буде ідентичним, тобто запис А=(1,2,0,6) відповідає числам 355, 545 і т.д.

Таким чином, в СОК нема однозначної відповідності між числами та його зображенням. Тому основи r треба обирати таким чином, щоб їх добуток Д перекривав весь діапазон чисел, що обробляються ЕОМ.

Існують і інші системи числення, але вони не використовуються в обчислювальній техніці. Наприклад.

**Змішана система**

*Змішана система числення* є узагальненням системи числення з основою *b* і її часто відносять до позиційних систем числення. Основою змішаної системи є послідовність чисел, що зростає, $\{b_k\}_{k=0}^{\infty}$ кожне число x представляється як **лінійна комбінація**:

$$x = \sum_{k=0}^{n} a_k b_k \tag{1.26}$$

де на коефіцієнти накладаються деякі обмеження.

Якщо $b_k = b_k$ для деякого b, то змішана система співпадає з 6-основною системою числення.

Найвідомішим прикладом змішаної системи числення є представлення часу у вигляді кількості діб, годин, хвилин і секунд. При цьому величина *d днів h годин m хвилин s секунд* відповідає значенню

$$d \cdot 24 \cdot 60 \cdot 60 + h \cdot 60 \cdot 60 + m \cdot 60 + s \quad \text{секунд.}$$

### 1.4.4 Нетрадиційні системи числення

Якщо в позиційній системі числення базис не є геометричною прогресією або значення цифр можуть набувати негативних значень, таку систему числення називають ***нетрадиційною***. Їх можна розділити на урівноважені (урівноважена трійкова) і інші (факторіальна, фибоначчиева та ін.) системи числення.

**Система числення Фібоначчі**

Представлення засновується на числах Фібоначчі:

$$x = \sum_{k=0}^{n} f_k F_k \ , \tag{1.27}$$

де $F_k$ – числа Фібоначчі $f_k \in \{0,1\}$, при цьому у записі $f_n, f_{n-1}, \ldots, f_0$ не зустрічаються дві одиниці підряд.

**Факторіальна система числення**

Представлення використовує факторіал натуральних чисел:

$$x = \sum_{k=1}^{n} d_k k! , \tag{1.28}$$

де $0 \le d_k \le k$ .

**Біномальна система числення**

Представлення використовує біномальні коефіцієнти:

$$x = \sum_{k=1}^{n} \binom{c_k}{k} , \tag{1.29}$$

де $0 \le c_1 < c_2 < \ldots < c_n$ .

**Система числення майя**

Майя використовували двадцяткову систему числення за одним винятком: у другому розряді було не 20, а 18 ступенів, тобто після числа (17)(19) відразу йшло число (1)(0)(0). Це було зроблено для полегшення розрахунків календарного циклу, оскільки (1)(0)(0) дорівнювало 360, що приблизно дорівнює кількості днів у сонячному році.

### 1.4.5 Способи переведення чисел з однієї системи числення в іншу

Арифметичні операції в усіх позиційних системах числення виконуються за одними і тими ж правилами. Для проведення арифметичних операцій над числами, представленими в різних системах числення, необхідно заздалегідь перетворити їх в одну систему числення і врахувати те, що перенесення в наступний розряд при операції складання і позика із старшого розряду при операції віднімання визначається величиною підстави системи числення.

Основна перевага двійкового кодування інформації полягає в його простоті, тому що дозволяє реалізувати надзвичайно простими технічними засобами усі види інформаційних процесів (збір, збереження, обробка і передача інформації) в обчислювальних пристроях і системах зв'язку.

Арифметичні операції в двійковій системі числення засновані на таблицях складання, віднімання і множення однорозрядних двійкових чисел і легко можуть здійснюватися за допомогою автоматичних пристроїв.

Саме такими пристроями і є усі без винятку ЕОМ.

При складанні двох одиниць відбувається переповнювання розряду і робиться перенесення одиниці в старший розряд, при відніманні 0 – 1 робиться позика із старшого розряду, в таблиці "Віднімання" ця позика позначена 1 з рисою над цифрою.

| Складання | Віднімання | Множення |
|---|---|---|
| 0 + 0 = 0 | 0 - 0 = 0 | 0 · 0 = 0 |
| 0 + 1 = 1 | 0 - 1 = 1 | 0 · 1 = 0 |
| 1 + 0 = 1 | 1 - 0 = 1 | 1 · 0 = 0 |
| 1 + 1 = 10 | 1 - 1 = 0 | 1 · 1 = 1 |

Щоб переконатися у тому, що правила виконання арифметичних операцій залишаються без будь-яких змін, розглянемо послідовно чотири дії: додавання, віднімання, множення та ділення чисел у двійковій системі числення.

**Операція додавання**

Додавання виконується слідуючим чином: два двійкових числа записуються стовпцем, додавання починається з молодшого розряду. Так, як і у десятковій системі, при утворенні одиниці переносу вона додається до результату у наступному розряді.

***Приклад***. Скласти два двійкових числа $A_{(a)}$=11010,101 та $B_{(2)}$=1011,110

Запишемо числа стовпцем. Додавання почнемо з молодших розрядів, користуючись таблицею додавання. Одиниці переноса, що утворюються, покажемо зверху, позначаючи їх стрілками:

$$\begin{array}{r} 11010{,}101 \\ +1011{,}110 \\ \hline 100110{,}011 \end{array}$$

Утворений результат можна перевірити переведенням всіх чисел у десяткову систему:

$$A_{(10)}=26{,}625,\ B_{(10)}=11{,}75,\ A_{(10)}+B_{(10)}=28{,}375$$

**Операція віднімання**

Як і додавання, операція віднімання починається з наймолодшого розряду. Якщо будь-який розряд зменшуваного дорівнює 0, а однойменний розряд від'ємного дорівнює 1, позначають одиницю сусіднього старшого розряду. При цьому в молодшому розряді утворюється дві одиниці (кожний розряд двійкового числа відрізняється від сусіднього у 2 рази).

***Приклад 1.25***. Виконати операцію віднімання над числами $A_{(2)}$=11011,101 та $B_{(2)}$=1101,011

$$\begin{array}{r} 11011{,}101 \\ -1101{,}011 \\ \hline 1110{,}010 \end{array}$$

Вірність виконаної операції знову можна перевірити, якщо всі числа перевести у десяткову систему.

$$A_{(10)}=27{,}625,\ B_{(10)}=13{,}375,\ A_{(10)} — B_{(10)}=14{,}25$$

**Операція множення**

Множення виконується шляхом утворення проміжних добутків, що потім підсумовуються. Згідно з таблицею множення проміжні добутки – це множене, якщо у розряді множника стоїть 1, або нуль, якщо у розряді множника стоїть 0. Це означає, що множення у двійковій системі виконується значно простіше, ніж у десятковій системі, тому що виконується лише зсув та підсумковування множеного. Позиція коми визначається так як і при множенні десяткових чисел.

***Приклад 1.26.*** Виконати операцію множення над двійковими числами A=1011,1 та B=101,01.

```
   1011,1
 x 101,01
 ---------
   10111
  00000
 10111
00000
10111
-----------
111100,011
```

**Операція ділення**

Як і у десятковій системі числення, операція ділення двійкових чисел у загальному випадку є нескінченною. Тому її зупиняють, коли досягнута потрібна точність результата. Зовнішнє ділення не відрізняється (як і множення) від аналогічної операції у десятковій системі.

***Приклад 1.27***. Розділити число $A_{(2)}=10111$ на $B_{(2)}=101$

```
11011 |101
-     |---
101   101,011
---
  111
  101
  ---
 1000
  101
 ----
  110 і. т.д.
```

Розглянуті приклади дозволяють зробити висновок про те, що арифметичні операції у двійковій системі виконуються значно простіше ніж у десятковій системі. Головним чином це стосується множення та ділення, під час виконання яких виконується зсув чисел та їх додавання (віднімання).

На закінчення розглянемо особливості виконання арифметичної операції додавання у двійководесятковій системі числення. Вона потребує урахування властивостей цієї системи, а саме:

усередині однієї тетради (одного десяткового розряду) додавання виконується як число десять у двійковій системі, тобто $10_{(10)}=1010_{(2)}$.

У залежності від десяткових цифр додатків у даному розряді можливі три випадки:

1. Результат додавання менший ніж $10_{(2)}$ цей результат є остаточним у даному розряді.

2. Результат лежить у межах від $10_{(10)}$ ($1010_{(2)}$) до $15_{(10)}$($1111_{(2)}$).

Він перевищує $10_{(10)}$, тому потрібно виробити одиницю переносу у сусідній старший розряд, а у даному розряді утворити остаточний результат відніманням $10_{(10)}$-$1010_{(2)}$

3. Результат лежить у межах від $16_{(10)}$($10000_{(2)}$) до $19_{(10)}$($10011_{(2)}$).

У цьому випадку при додаванні у двійковій системі утворюється одиниця переносу у старший п'ятий розряд. Але ця одиниця "важить" $15_{(10)}$ а не 10, як потрібно при роботі у десятковій системі. Тому треба виконати перенос у старший розряд, а до остачі у даному розряді додати число $6_{(10)}=0110_{(2)}$

***Приклад 1.28***. Виконати додавання чисел $A_{(10)}=985$ та $B_{(10)}=963$, поданих у двійково-десятковій системі:

$$A_{(2-10)} = \underbrace{1001}_{9}\ \underbrace{1000}_{8}\ \underbrace{0101}_{5}$$

$$B_{(2-10)} = \underbrace{1001}_{9}\ \underbrace{0110}_{6}\ \underbrace{0011}_{3}$$

Додавання виконуємо стовпчиком, починаючи з наймолодшого розряду:

$$\begin{array}{r} 1001\ \ 1000\ \ 0101 \\ +\ 1001\ \ 0110\ \ 0011 \\ \hline \underbrace{10010}_{18}\ \underbrace{1110}_{14}\ \underbrace{1000}_{8} \end{array}$$

У другому розряді утворилося число більше десяти. Тому треба від результату відняти $10_{(10)}$($1010_{(2)}$) і зробити перенос у старший розряд, а в даному розряді залишити число 1110-1010-0100 ($4_{(10)}$). З урахуванням одиниці переносу у третьому розряді утворюється число 10010+0001-10011, що має одиницю у п'ятому розряді. Ця одиниця переноситься у четвертий розряд, а до остачі у третьому розряді треба додати число $0110_{(2)}$-$6_{(10)}$:

0011 – 0011+0110- 1001 ($9_{(10)}$). Тому остаточний результат складання має вигляд:

$$A_{(2-10)} + B_{(2-10)} = \underbrace{0001}_{1}\ \underbrace{1001}_{9}\ \underbrace{0100}_{4}\ \underbrace{1000}_{8}$$

Подібні правила виконання арифметичних операцій існують для двійково-десяткової системи числення, якщо кодування десяткових цифр у розрядах виконується іншим способом, наприклад, кодом з "надлишком 3", чи кодом з "надлишком 6".

В ЕОМ використовуються всі системи числення, розглянуті вище. Вся вихідна інформація, що задається людиною, подається у звичайній для неї десятковій системі. В записах програми вживають вісімкову та шістнадцяткову системи числення. Обробка інформації ЕОМ виковується, як правило, у двійковій системі. Тому виникає потреба у переведенні чисел з однієї системи числення в іншу. Способи переведення залежать від заданого числа (ціле чи дрібне) та заданих систем числення. Тому поряд з універсальними способами переведення, існують такі, застосовуються лише для конкретній систем числення.

**Переведення цілого числа**

Універсальний спосіб переведення цілих чисел із системи з основою R в систему з основою $R_n$ стає зрозумілим якщо звернутися до розгорнутої форми запису числа(1.11). При цьому запишемо цифри та основу R символами вихідної системи:

$$A_{(R)} = a_{n-1}R_H^{n-1} + a_{n-2}R_H^{n-2} + \ldots + a_2R_H^2 + a_1R_H^1 + a_0 \qquad (1.30)$$

З (1.30) випливає, що цифра нульового розряду $a_0$ утворюється як остача від ділення заданого числа А на основу нової системи числення $R_н$. Результат ділення буде мати вигляд:

$$\frac{A_{(R)}}{R_H} = a_{n-1}R_H^{n-2} + a_{n-2}R_H^{n-3} + \ldots + a_2R_H + a_1. \qquad (1.31)$$

Якщо цей результат розділити знову на $R_н$, здобудемо остачу $a_1$ що є цифрою першого розряду.

Продовжуючи ділення, одержимо послідовно всі цифри числа А в системі числення з основою $R_н$, зображені символами вихідної системи. Останнє має значення тоді, коли $R > R_н$, тому що заключною операцією є переведення здобутих цифр в цифри нової системи.

***Приклад 1.29***. Перевести. десяткове число $A_{(10)} = 29$ у двійкову систему числення.

29:2=14 остача 1 ↑ напрям
14:2=7 остача 0 | запису
7:2=3 остача 1 | числа
3:2=1 остача 1 |
1:2=0 остача 1 |

Звідси $A_{(2)} = 11101$ .

***Приклад 1.30***. Перевести десяткове число $A_{(10)} = 75$ у вісімкову систему числення.

75:8=9 остача 3 ↑
9:8=1 остача 1 |
1:8=0 остача 1 |

Тому $A_{(8)} = 113$ .

***Приклад 1.31***. Перевести десяткове число $A_{(10)} = 167$ у шістнадцяткову систему числення.

167:16=10 остача 7 ↑
10:16=0 остача 10=A |

Тому $A_{(16)} = A7$ .

Цей приклад показує, що після одержання результату у цифрах десяткової системи (оскільки основа 16 більше 10) треба перевести цифри у шістнадцяткову систему.

***Приклад 1.32***. Перевести число $A_{(2)} = 1101011$ у десяткову систему числення. Оскільки ділення виконується у вихідній системі, основу $R_{(10)} = 10$ треба записати у двійковій системі: $R_{(2)}1010$

1101011:1010=1010 остача 111=7 ↑
1010:1010=1 остача 0=0 |
1:1010=0 остача 1=1 |

Переводячи остачі в десяткову систему числення, одержимо результат: $A_{(10)} = 107$ .

**Переведення дрібного числа**

Підкреслимо одразу що оскільки мова йде про дрібне число, ціла частина залишається нульовою у будь-якій системі числення. Щоб здобути спосіб утворення цифр дрібної частини, запишемо число у розгорнутій формі, використовуючи цифри вихідної (заданої) системи числення.

$$A_{(R)} = a_{-1}R_H^{-1} + a_{-2}R_H^{-2} + \ldots + a_{-m+1}R_H^{-m+1} + \ldots + a_{-m}R_H^{-m} \qquad (1.32)$$

З цього запису випливає, що цифра $a_{-1}$ першого після коми розряду утворюється як ціла частина числа; якщо помножити його на основу $R_н$ системи, у яку воно переводиться:

$$A_{(R)}R_H = a_{-1} + a_{-2}R_H^{-1} + \ldots + a_{-m+1}R^{-m+2} + \ldots + a_{-m}R^{-m+1}. \qquad (1.33)$$

Виключивши з цього числа цілу частину $a_{-1}$, знову виконаємо множення на $R_н$. При цьому одержимо цілу частину $a_{-2}$, що є цифровою другого розряду після коми. Продовжуючи множення дрібної частини, послідовно одержимо цифри. інших розрядів. Слід підкреслити, що цей процес у загальному випадку є нескінченним. Тому його припиняють, коли досягнута потрібна точність переведення.

***Приклад 1.33***. Перевести десяткове число $A_{(10)} = 0{,}361$ у двійкову систему числення:

Ціла частина ← 0,361

$a_{-1}$ ← $\frac{2}{0{,}722}$

$a_{-2}$ ← $\frac{2}{1{,}444}$

$a_{-3}$ ← $\frac{2}{0{,}888}$

$a_{-4}$ ← $\frac{2}{1{,}776}$

$a_{-5}$ ← $\frac{2}{1{,}552}$ і т.д.

Звідси $A_{(2)} = 0{,}01011\ldots$

***Приклад 1.34***. Перевести десяткове число $A_{(10)} = 0{,}427$ у вісімкову систему числення:

Ціла частина ← 0,427

$a_{-1}$ ← $\frac{8}{3{,}416}$

$a_{-2}$ ← $\frac{8}{3{,}328}$

$a_{-3}$ ← $\frac{8}{2{,}624}$

$a_{-4}$ ← $\frac{8}{4{,}992}$

$a_{-5}$ ← $\frac{8}{7{,}936}$

Тому $A_{(8)} = 0{,}33247$ .

### 1.4.6 Одиниці вимірювання даних

Існує багато різних систем і одиниць виміру даних. Кожна наукова дисципліна і кожна область людської діяльності може використовувати свої, найбільш зручні або традиційно устояні одиниці. В інформатиці для виміру даних використовують той факт, що різні типи даних мають універсальне двоичное представлення, і тому уводять свої одиниці даних, засновані на ньому.

Найменшою одиницею виміру є байт. Оскільки одним байтом, як правило, кодується один символ текстової інформації, то для текстових документів розмір у байтах відповідає лексичному обсягу в символах (поки виключення представляє розглянута вище універсальне кодування *UNICODE).*

Більш велика одиниця виміру – кілобайт (Кбайт). Умовно можна вважати, що 1 Кбайт приблизно дорівнює 1000 байт. Умовність зв'язана з тим, що для обчислювальної техніки, що працює з двоичными числами, більш зручне представлення чисел у виді ступеня двійки, і тому насправді 1 Кбайт дорівнює $2^{10}$ байт (1024 байт). Однак усюди, де це не принципово, з погрішністю до 3% "забувають" про "зайві" байти.

У кілобайтах вимірюють порівняно невеликі обсяги даних. Умовно можна вважати, що одна сторінка неформатированного машинописного тексту складає близько 2 Кбайт.

Більш великі одиниці виміру даних утворяться додаванням префіксів *мега-, гига- тера-;* у більш великих одиницях поки немає практичної потреби.

1 Кілобайт (Кбайт) = $2^{10}$ байт = 1024 байт,
1 Мегабайт (Мбайт) = $2^{10}$ Кбайт = 1024 Кбайт = $2^{20}$ байт,
1 Гігабайт (Гбайт) = $2^{10}$ Мбайт = 1024 Мбайт = $2^{30}$ байт,
1 Терабайт (Тбайт) = $2^{10}$ Гбайт = 1024 Гбайт = $2^{40}$ байт,
1 Петабайт (Пбайт) = $2^{10}$ Тбайт = 1024 Тбайт = $2^{50}$ байт,
1 Ексабайт (Ебайт) = $2^{10}$ Пбайт = 1024 Пбайт = $2^{60}$ байт,
1 Зетабайт (Збайт) = $2^{10}$ Ебайт = 1024 Ебайт =$2^{70}$ байт,
1 Йотабайт (Йбайт) = $2^{10}$ Збайт = 1024 Збайт = $2^{80}$ байт.

Особливо оборотний увага на те, що при переході до більш великих одиниць погрішність, зв'язана з округленням, накопичується і стає неприпустимої, тому на старших одиницях виміру округлення виробляється рідше.

Одиниці виміру кількості інформації, в назві яких є приставки "кіло", "мега" і так далі, з точки зору теорії вимірів не є коректними, оскільки ці приставки використовуються в метричній системі заходів, в якій в якості множників кратних одиниць використовується коефіцієнт $10^n$, де n = 3, 6, 9 і так далі.

Для усунення цієї некоректності міжнародной організацією International Electrotechnical Commission, яка займається створенням стандартів для галузі електронних технологій, затвердила ряд нових приставок для одиниць виміру кількості інформації: кібі (kibi), мебі (mebi), гібі (gibi), тебі (tebi), петі (peti), ексбі (exbi). Проте доки використовуються старі позначення одиниць виміру кількості інформації, і потрібно час, щоб нові назви почали широко застосовуватися (табл.1.15).

Таблиця 1.15

**Нові приставки для одиниць виміру кількості інформації**

| Кількість байтів | | | | |
|---|---|---|---|---|
| Префікси СІ | | | Бінарні префікси | |
| Назва (скорочення) | Префікс СІ | Альтернативне використання | Назва (скорочення) | Значення |
| кілобайт (КБ) | $10^3$ | $2^{10}$ | кібібайт (КіБ) | $2^{10}$ |
| мегабайт (МБ) | $10^6$ | $2^{20}$ | мебібайт (МіБ) | $2^{20}$ |
| гігабайт (ГБ) | $10^9$ | $2^{30}$ | гібібайт (ГіБ) | $2^{30}$ |
| терабайт (ТБ) | $10^{12}$ | $2^{40}$ | тебібайт (ТіБ) | $2^{40}$ |
| петабайт (ПБ) | $10^{15}$ | $2^{50}$ | петібайт (ПіБ) | $2^{50}$ |
| ексабайт (ЕБ) | $10^{18}$ | $2^{60}$ | ексбібайт (ЕіБ) | $2^{60}$ |
| зетабайт (ЗБ) | $10^{21}$ | $2^{70}$ | зебібайт (ЗіБ) | $2^{70}$ |
| йотабайт (ЙБ) | $10^{24}$ | $2^{80}$ | йобібайт (ЙіБ) | $2^{80}$ |

### 1.4.7 Форми подання числових даних в комп'ютер

Посилаючись на принципи побудови ЕОМ, знову підкреслимо, що безпосередня обробка інформації машиною виконується над даними, що подані двійковими кодами. З цього випливає, що основною системою кодування чисел є двійкова система числення. Тому надалі будемо розглядати лише двійкові числа.

Для розміщення чисел в ЕОМ відводиться певне число розрядів, що завжди обмежене. Це число визначається під час розробки машини і далі залишається незмінним. Кількість розрядів, призначених для запису чисел в ЕОМ, утворює розрядну сітку машини. Зрозуміло, що довжина розрядної сітки визначає діапазон чисел, з яким оперує машина, і тому є однією з основних її характеристик.

Зараз використовується дві форми подання чисел в ЕОМ:

Природна форма та нормальна форма.

Розглянемо особливості запису чисел в розрядній сітці машини.

1.4.7.1 Природна форма подання чисел

Числові дані обробляються в комп'ютері в двійковій системі числення. Числа зберігаються в пам'яті комп'ютера в двійковому коді, тобто у вигляді послідовності нулів і одиниць, і можуть бути представлені у форматі з фіксованою або плаваючою комою.

Цілі числа зберігаються в пам'яті у форматі з фіксованою комою. При такому форматі представлення чисел для зберігання цілих ненегативних чисел відводиться регістр пам'яті, якій складається з восьми елементів пам'яті (8 біт). Кожному розряду елементу пам'яті відповідає завжди один і той же розряд числа, а кома знаходиться справа після молодшого розряду і поза розрядною сіткою. Наприклад, число 110011012 зберігатиметься в регістрі пам'яті таким чином:

| 8р | 7р | 6р | 5р | 4р | 3р | 2р | 1р (розряд) |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

Це зображення числа у вигляді послідовності цифр, де комою відокремлюється його ціла та дрібна частина: $A_{(2)} = 1011{,}11$; $B_{(2)} = -10011{,}11$. Ця форма в найпоширенішою у математиці і тому знайома кожному. У розрядній сітці машини кома ніяк не позначається, але її місце чітко визначене. Тому природна форма числа має іншу назву: форма з фіксованою комою.

Для отримання більше цілісного уявлення про цей період доцільно познайомитися зі зміною поколінь електронно-обчислювальних машин (ЕОМ) і зіставити ці відомості з етапами в області обробки і передачі інформації.

Максимальне значення цілого ненегативного числа, яке може зберігатися в регістрі у форматі з фіксованою комою, можна визначити з формули: $2^n - 1$, де $n$ – число розрядів числа. Максимальне число при цьому дорівнюватиме $2^8 - 1 = 255_{10} = 1111111$ Ці мінімальне $0_{10} = 00000\ 000_2$. Таким чином, діапазон зміни цілих ненегативних чисел знаходитиметься в межах від 0 до $255_{10}$.

На відміну від десяткової системи в двійковій системі числення при комп'ютерному представленні двійкового числа відсутні символи, що означають знак числа : позитивний (+) або негативний (-), тому для представлення цілих чисел зі знаком в двійковій системі використовуються два формати представлення числа : формат значення числа зі знаком і формат додаткового коду. У першому випадку для зберігання цілих чисел зі знаком відводиться два регістри пам'яті (16 біт), причому старший розряд (крайній ліворуч) використовується під знак числа: якщо число позитивне, то в знаковий розряд записується 0, якщо число негативне, то – 1. Наприклад, число $536_{10} = 0000001000011000_2$ буде представлено в регістрах пам'яті в

наступному виді:

| 16р | 15р | 14р | 13р | 12р | 11р | 10р | 9р | 8р | 7р | 6р | 5р | 4р | 3р | 2р | 1р |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

а негативне число $-536_{10} = 1000001000011000_2$ у виді:

| 16р | 15р | 14р | 13р | 12р | 11р | 10р | 9р | 8р | 7р | 6р | 5р | 4р | 3р | 2р | 1р |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

Реально у сучасних ЕОМ місце коми у запису числа визначається:

- ✧ перед старшим розрядом числа. Це означає, що машина оперує лише з числами, меншими одиниці, тобто з дрібними числами;
- ✧ після наймолодшого розряду числа. Це означає, що машина оперує лише з цілими числами.

Вибір місця розташування коми у розрядній сітці визначає людина при використанні машини для розв'язання конкретної задачі.

Довжина розрядної сітки впливає на можливий діапазон чисел, що може обробляти машина. Припустимо, що для розміщення числової частини числа відводиться *m* розрядів (один розряд залишається для знака числа) (рис. 1.43)

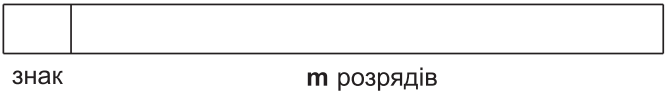



Рисунок 1.43. Розрядна сітка цілого числа

Максимальним числом А, що може бути записане у m розрядах є таке, що має одиницю у кожному розряді. Якщо машина працює з цілими числами, це число має вигляд:

$$A_{\max(2)} = \underbrace{111...11}_{m \text{ розрядів}} \tag{1.34}$$

Це число у звичній десятковій системі має вигляд:

$$A_{\max(10)} = 2^m - 1. \tag{1.35}$$

Наприклад, якщо m=15, $A_{\max(10)} = 2^{15} - 1 = 32767$ .

Це означає, що машина може оперувати з числами, що лежать у діапазоні .Тому під час підготовки задачі для розв'язання треба задавати вихідні дані таким чином, щоб при виконанні обчислень будь-який результат не вийшов за цей діапазон. Якщо це не передбачити, може статися переповнення розрядної сітки, коли результат по довжині перевищить m розрядів. Цифри розрядів, більших *m*-го, будуть загублені, і результат буде зіпсований.

Якщо машина працює з дрібними числами. Максимальне число, що може бути записане у m розрядах, має вигляд:

$$A_{\max(2)} = \underbrace{0{,}111\ldots11}_{m\ \text{розрядів}} \tag{1.36}$$

Переводячи число у десяткову систему одержимо:

$$A_{\max(10)} = 1 - 2^{-m} \ . \tag{1.37}$$

Наприклад, для m = 15 $A_{\max(10)} = 1 - 2^{-15} = 0{,}9997$ .

Якщо будь-який результат у ході обчислень перевищить $A_{\max}$, відбувається переповнення розрядної сітки, що приводить до знівечення розв'язання задачі.

Розрядна сітка обмежує також подання малих чисел. Найменшим числом, що може бути записане у розрядній сітці, є

$$A_{\min(10)} = \underbrace{0{,}000\ldots001}_{m\ \text{розрядів}} \text{, тобто } A_{\min(10)} = 2^{-m} \tag{1.38}$$

Тому, якщо в ході обчислень утворюється число $A<2^{-m}$, воно втрачається машиною, оскільки у всіх m розрядах будуть стоять нулі. Цей випадок називається машиним нулем і також веде до спотворення результату обчислень.

Таким чином, машина, що працює з дрібними числами, поданими у формі з фіксованою комою, вірно обробляє числа, що лежать у діапазоні

$$2^{-m} \le |A| < 1 - 2^{-m} \ . \tag{1.39}$$

Обмеженість розрядної сітки призводить до похибки у поданні чисел. Абсолютна похибка, що виникає внаслідок округлення числа, не перевищує половини наймолодшого розряду, тобто

$$\Delta A_{\max} = 2^{-m-1} = 2^{-(m+1)} \tag{1.40}$$

Ця похибка є постійною для всього діапазону чисел. В той же час відносна похибки суттєво залежить від конкретних чисел, що беруть участь у обчисленнях. Тому вона лежить у межах:

$$\left.\begin{aligned} S_{A_{\min}} &\frac{\Delta A_{\max} \cdot 2^{m-1}}{A_{\max} \cdot 1 - 2^{m}} \approx 2^{-m-1} \\ S_{A_{\min}} &= \frac{\Delta A_{\max} \cdot 2^{-m-1}}{A_{\min} \cdot 2^{-m}} = 2^{-1} \end{aligned}\right\} \tag{1.41}$$

З цього випливає, що точність результату буде тим більшою, чим більше числа, що обробляються машиною. Для малих чисел похибка обчислень може стати неприпустимою. Цей випадок треба мати на увазі під

час підготовки задачі для розв'язання на ЕОМ. Щоб забезпечити вірний результат та виключити появу переповнення розрядної сітки чи машинного нуля, виконується масштабування вихідних даних.

Комп'ютерне представлення чисел у форматі з фіксованою комою має свої переваги і недоліки. До *переваг* відносяться простота представлення чисел і алгоритмів реалізації арифметичних операцій, до *недоліків* – кінцевий діапазон представлення чисел, який може бути недостатнім для вирішення багатьох завдань практичного характеру (математичних, економічних, фізичних і так далі).

1.4.7.2 Нормальна форма подання чисел

Дійсні числа (кінцеві і нескінченні десяткові дроби) обробляються і зберігаються в комп'ютері у форматі з плаваючою комою. При такому форматі представлення числа положення коми в записі може змінюватися. Будь-яке дійсне число А у форматі з таваючою комою може бути представлено у виді:

$$A = \pm MR^{\pm P} \ , \qquad (1.42)$$

де M – мантиса числа,
R –основа системи числення,
P – порядок числа.

Вираження (1.42) для десяткової системи числення прикмет вид: $A_{10} = \pm A\ 10^{\pm p}$, для двійкової – $A_2 = \pm M\ 2^{\pm p}$, для вісімкової – $A_8 = \pm M\ 8^{\pm p}$, для шістнадцятиричної – $A_{16} = \pm M\ 16^{\pm p}$ і так далі

Така форма представлення числа також називається *нормальною.* Зі зміною порядку кома в числі зміщується, тобто як би плаває вліво або управо. Тому нормальну форму представлення чисел називають формою з плаваючою комою. Десяткове число 15,5, наприклад, у форматі з плаваючою комою може бути представлено у виді: $0{,}155\cdot10^{2}$; $1{,}55\cdot10^{1}$; $15{,}5\cdot10^{0}$; $155{,}0\cdot10^{-1}$; $1550{,}0\cdot10^{-2}$ і так далі. Ця форма запису десяткового числа 15,5 з плаваючою комою не використовується при написанні комп'ютерних програм і введенні їх в комп'ютер (пристрої введення комп'ютерів сприймають тільки лінійний запис даних). Виходячи з цього вираження (1.43) для представлення десяткових чисел і введення їх в комп'ютер перетворюють до виду

$$A = \pm ME \pm P, \qquad (1.43)$$

де P – порядок числа,

замість основи системи числення 10 пишуть букву E, замість коми – точку, і знак множення не ставиться. Таким чином, число 15,5 у форматі з плаваючої коми і лінійного запису (комп'ютерне представлення) буде записано у виді: 0.155E2; 1.55E1; 15.5E0; 155.0E-1; 1550.0E-2 і так далі.

Незалежно від системи числення будь-яке число у формі з плаваючою комою може бути представлене нескінченною безліччю чисел. Така форма запису називається *ненормалізованою*. Для однозначного представлення чисел з плаваючою комою використовують нормалізовану форму запису числа, при якому мантиса числа повинна відповідати умові

$$1/h <= |A| < 1 , \qquad (1.44)$$

де |A| – абсолютне значення мантиси числа.

Умова (1.44) означає, що мантиса має бути правильним дробом і мати після коми цифру, відмінну від нуля, або, іншими словами, якщо після коми в мантисі стоїть не нуль, то число називається нормалізованим. Так, число 15,5 в нормалізованому виді (нормалізована мантиса) у формі з плаваючою комою виглядатиме таким чином: $0{,}155 \cdot 10^2$, тобто нормалізована мантиса буде A = 0,155 і порядок P = 2, або в комп'ютерному представленні числа 0.155E2.

Числа у формі з плаваючою комою мають фіксований формат і займають в пам'яті комп'ютера чотири (32 біт) або вісім байт (64 біт). Якщо число займає в пам'яті комп'ютера 32 розряди, то це число звичайної точності, якщо 64 розряди, то це число *подвійної точності*. При записі числа з плаваючою комою виділяються розряди для зберігання знаку мантиси, знаку порядку, мантиси і порядку (рис.1.44). Кількість розрядів, яка відводиться під порядок числа, визначає діапазон зміни чисел, а кількість розрядів, відведених для зберігання мантиси, – точність, з якою задається число.




Рисунок 1.44. Умовний вигляд розрядної сітки у нормальній формі

Ця форма широко використовується в математиці, оскільки дозволяє зробити запис числа більш стислим. Прикладами десяткових чисел у нормальній формі є: $A_{(10)} = 3 \cdot 10^8$; $B_{(10)} = -8{,}73 \cdot 10^{-4}$; $C_{(10)} = 0{,}35 \cdot 10^5$ тощо. З цих прикладів випливає, що запис числа у нормальній формі є неоднозначним:

$$A_{(10)} =, 56{,}83 \cdot 10^4 = 0{,}5683 \cdot 10^6 = 568{,}3 \cdot 10^3 \qquad (1.45)$$

тобто положення коми у мантисі змінюється в разі змінення величини порядку. Тому така форма має іншу назву: форма з плаваючою комою.

Форма з плаваючою комою, має здатність обробляти числа, що лежать у діапазоні

$$2^{-2^K} \le |A| \le \left(1 - 2^{-m}\right) 2^{2^K - 1} . \qquad (1.46)$$

Порівнюючи (1.46) з (1.41) приходимо до висновку, що в разі однакової довжини розрядної сітки діапазон чисел у машинах з плаваючою комою значно ширший ніж діапазон чисел у машинах з фіксованою комою. Щоб переконатися у цьому, розглянемо приклад. Нехай загальна довжина розрядної сітки є 32 розряди. У машині з фіксованою комою один розряд відводиться на знак числа. Тому для неї m-31. У машині з плаваючою комою один розряд відводиться на знак натиси і один розряд – знак порядку.

Припустимо, що на числову частину порядку відводиться 6 розрядів. Тому для неї к=6, m=24.

Звертаючись до формули (1.41) для машини з фіксованою комою одержимо:

$$2^{-31} \le |A| \le 1 - 2^{-31}$$

Для машини з плаваючою комою, як випливає з формули (1.46)

$$2^{64} \le |A| \le (1 - 2^{-31})3^{+63}$$

Довжина діапазону подання чисел у машинах з плаваючою комою є суттєвою її перевагою у порівнянні з машиною з фіксованою комою. Другою перевагою машини з плаваючою комою є точність подання чисел. Для них відносна похибка $\delta_A$ визначається так:

$$\left.\begin{aligned} \delta_{A_{\min}} &= \frac{\Delta A_{\max}}{A_{\max}} \approx 2^{-m-1} \\ \delta_{A_{\max}} &= \frac{\Delta A_{\min}}{A_{\min}} = 2^{-m} \end{aligned}\right\} \qquad (1.47)$$

З цього випливає, що точність подання чисел у формі з плаваючою комою практично не залежить від модулей чисел. Це можна пояснити тим, що у машині відбувається нормалізація чисел, тобто мантиса чисел завжди відповідає умові (1.42).

При виконанні арифметичних операцій (складання і віднімання) над числами, представленими у форматі з плаваючою комою, реалізується наступний порядок дій (алгоритм):

- ✧ робиться вирівнювання порядків чисел, над якими здійснюються арифметичні операції (порядок меншого по модулю числа збільшується до величини порядку більшого по модулю числа, мантиса при цьому зменшується в таку ж кількість разів);
- ✧ виконуються арифметичні операції над мантисами чисел;
- ✧ робиться нормалізація отриманого результату.

Пояснимо сказане вище на прикладах.

***Приклад 1.35***.

Зробимо складання двох чисел $0{,}5 \cdot 10^2$ і $0{,}8 \cdot 10^3$ у форматі з плаваючою комою.

Рішення.

Проведемо вирівнювання порядків і складання мантис $0{,}05 \cdot 10^3 + 0{,}8 \cdot 10^3 = 0{,}85 \cdot 10^3$. Отримана мантиса 0,85 є нормалізованою, оскільки задовольняє умові (1.39).

***Приклад 1.36***.

Зробимо складання двох чисел $0{,}1 \cdot 2^2$ і $0{,}1 \cdot 2^3$ у форматі з плаваючою комою.

Рішення.

Проведемо вирівнювання порядків і складання мантис: $0{,}01 \cdot 2^3 + 0{,}1 \cdot 2^3 = 0{,}11 \cdot 2^3$. Отримана мантиса 0,11 є нормалізованою, оскільки задовольняє умові (1.39).

**9 вересня – День тестувальника**

9 вересня 1945 р. учені Гарвардського університету, що тестували обчислювальну машину Mark II Aiken Relay Calculator, знайшли метелика, застряглого між контактами електромеханічного реле. Виконана робота вимагала опису, і слово було знайдене – "debugging" (дослівно: позбавлення від комахи). Так і нині називається процес виявлення і усунення багов – причин неправильної роботи комп'ютера.

## 1.5 Сучасні інформаційні системи та технології

### 1.5.1 Поняття інформаційної технології

Технологія при перекладі з грецького (techne) означає мистецтво, майстерність, уміння, а це не що інше, як процеси. Під процесом варто розуміти визначену сукупність дій, спрямованих на досягнення поставленої мети. Процес повинний визначатися обраної людиною стратегією і реалізуватися за допомогою сукупності різних засобів і методів.

Під технологією матеріального виробництва розуміють процес, який використовує сукупність засобів і методів обробки, виготовлення, зміни стану, властивостей, форми сировини або матеріалу. Технологія змінює якість або первісний стан матеріальних ресурсів з метою одержання матеріального продукту (рис. 1.45).

Інформація є одним з найцінніших ресурсів суспільства поряд з такими традиційними матеріальними видами ресурсів, як нафта, газ, корисні копалини, і іншими, а процес її переробки за аналогією з процесами переробки

матеріальних ресурсів можна сприймати як технологію. Тоді справедливо наступне визначення:

**інформаційна технологія (ІТ)** – процес, який використовує сукупність засобів і методів збору, обробки і передачі даних (первинної інформації) для одержання інформації нової якості про стан об'єкта, процесу або явища.




Рисунок 1.45. Інформаційна технологія як аналог технології переробки матеріальних ресурсів

*Мета технології матеріального виробництва* – випуск продукції, яка задовольняє потреби людини або системи.

*Мета інформаційної технології* – виробництво інформації для її аналізу людиною і прийняття на його основі рішення щодо виконання певної дії.

Відомо, що, застосовуючи різні технології до одного й того ж матеріального ресурсу, можна одержати різні вироби, продукти. Те ж саме буде справедливим і для технології переробки інформації.

Для порівняння в табл. 1.16 приведені основні компоненти технологій для виробництва матеріальних і інформаційних продуктів.

Таблиця 1.16

**Співставлення основних компонентів технологій**

| № з/п | Компоненти технологій для виробництва продуктів | |
|---|---|---|
| | матеріальних | інформаційних |
| 1 | Підготовка сировини і матеріалів | Збір даних або первинної інформації |
| 2 | Виробництво матеріального продукту | Обробка даних і отримання результатної інформації |
| 3 | Збут зроблених продуктів споживачам | Передача результатної інформації користувачеві для прийняття на її основі рішень |

Інформаційна технологія – досить загальне поняття і як інструмент може використовуватися різними користувачами, як непрофесіоналами в комп'ютерній галузі, так і розробниками нових ІТ.

Функціональна частина економічної інформаційної системи (ЕІС) завжди пов'язана з предметною областю і поняттям інформаційних технологій. Взагалі кажучи, технологія як деякий процес присутній у будь-якій предметній галузі. Так, наприклад, технологія видачі кредиту банком

може мати свої особливості залежно від виду кредиту, виду застави та ін. У ході виконання цих технологічних процесів співробітник банка обробляє відповідну інформацію.

Вирішення економічних і управлінських задач тісно пов'язане з виконанням ряду операцій із збору інформації, необхідної для вирішення цих задач, переробці її за деякими алгоритмами і видачі її особі, яка приймає рішення (ОПР), у зручній формі. Очевидно, що технологія прийняття рішень завжди мала інформаційну основу, хоча обробка даних і здійснювалася вручну. Однак із упровадженням засобів обчислювальної техніки в процес керування з'явився спеціальний термін *інформаційна технологія.*

Щоб термінологічно виділити традиційну технологію вирішення економічних і управлінських задач, вводиться термін *предметна технологія,* яка є послідовністю технологічних етапів з модифікації первинної інформації в результатну. Наприклад, технологія бухгалтерського обліку передбачає надходження первинної документації, що трансформується у форму бухгалтерської проводки. Остання, змінюючи стан аналітичного обліку, приводить до зміни рахунків синтетичного обліку і далі – балансу.

### 1.5.2 Класифікація інформаційних технологій

ІТ відрізняються ***по типу оброблюваної інформації*** (рис. 1.46), але можуть поєднуватися в інтегровані технології.

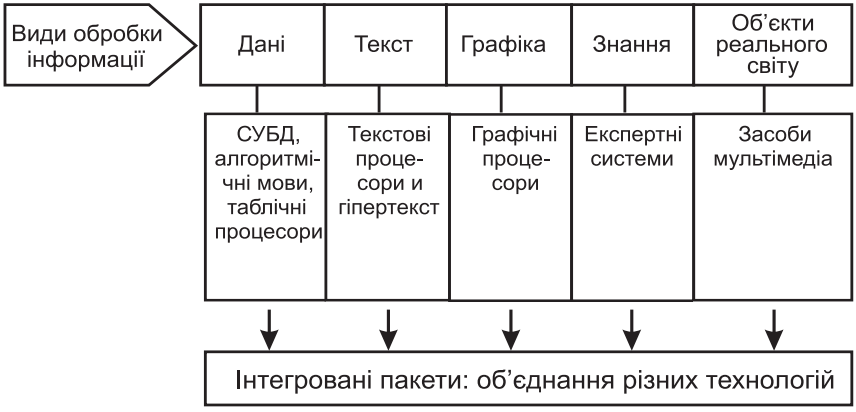



Рисунок 1.46. Класифікація ІТ залежно від типу оброблюваної інформації

Наведена схема певною мірою є умовною, оскільки більшість цих ІТ дозволяє підтримувати й інші види інформації. Так, у текстових про-

цесорах передбачена можливість виконання примітивних розрахунків, табличні процесори можуть обробляти не тільки цифрову, але і текстову інформацію, а також мають вбудований апарат генерації графіки. Однак кожна з цих технологій все-таки акцентована на обробці інформації певного виду.

Очевидно, що модифікація елементів, які складають поняття ІТ, дає можливість утворення величезної кількості їх у різних комп'ютерних середовищах.

Сьогодні можна говорити про забезпечуючі ІТ (ЗІТ) і функціональні ІТ (ФІТ).

*Забезпечуючі ІТ* – технології обробки інформації, які можуть використовуватися як інструментарій у різних предметних областях для вирішення різних задач. Інформаційні технології цього типу, можуть бути класифіковані щодо класів задач, на які вони орієнтовані. Забезпечуючі технології базуються на різних платформах, що обумовлено розмаїттям видів комп'ютерів і програмних середовищ, тому при їхньому об'єднанні на основі предметної технології виникає проблема системної інтеграції. Вона полягає в необхідності приведення різних ІТ до єдиного стандартного інтерфейсу та обміну даними.

*Функціональна ІТ* є такою модифікацією забезпечуючої ІТ, при якій реалізується яка-небудь із предметних технологій. Наприклад, робота співробітника кредитного відділу банку з використанням ЕОМ обов'язково передбачає застосування сукупності банківських технологій оцінки кредитоспроможності позичальника, формування кредитного договору і строкових зобов'язань, розрахунку графіка платежів і інших технологій, реалізованих у якій-небудь інформаційній технології: СУБД, текстовому процесорі та ін. Трансформація забезпечуючої інформаційної технології в чистому виді у функціональну (модифікація деякого загальновживаного інструментарію в спеціальний) може бути зроблена як фахівцем-проектувальником, так і самим користувачем. Це залежить від того, наскільки складна така трансформація, тобто від того, наскільки вона доступна самому користувачу-економісту. Ці можливості все більше розширюються, оскільки забезпечуючі технології рік від року стають "дружнішими". Таким чином, в арсеналі співробітника кредитного відділу можуть бути як технології, з якими він постійно працює (текстові і табличні процесори), так і спеціальні функціональні технології (СУБД, експертні системи, які реалізують предметні технології).

Предметна та інформаційна технології впливають одна на одну. Так, наприклад, наявність пластикових карток як носія фінансової інформації принципово змінює предметну технологію, надаючи такі можливості, які без цього носія просто були відсутні. З іншого боку, предметні технології, наповняючи специфічним змістом ІТ, акцентують їх на певні функції. Такі

технології можуть носити типовий або унікальний характер, що залежить від ступеня уніфікації технології виконання цих функцій.

Як приклад можна навести банківську технологію роботи з картотекою № 3, яка містить документи, що надійшли на обробку і не виконані через закриття особового рахунка за мотивами фінансового контролю. У цьому випадку спочатку закривається рахунок. Потім, якщо застосовується інформаційна технологія, цей запис позначається номером картотеки, для того щоб усі інші документи, що зменшують залишок на рахунку, попадали б у цю картотеку. У структурі операційно-облікового відділу банку перша і друга функції можуть виконуватися як одним виконавцем, так і різними операціоністами. Крім того, процеси виконання цих функцій можуть бути рознесені в часі. Таким чином, позначка в особовому рахунку, зроблена при його тимчасовому закритті одним операціоністом, використовується іншим операціоністом у процесі обробки документів, що надходять до сплати. У той же час ця позначка може бути зроблена тим операціоністом, що є відповідальним виконавцем з даного рахунку (відкриває, закриває рахунок, забезпечує операції з рахунком, нарахування відсотків та ін.).

Класифікація ІТ ***за типом користувальницького інтерфейсу*** (рис. 1.47) дозволяє говорити про системний і прикладний інтерфейси. І якщо останній пов'язаний з реалізацією деяких функціональних ІТ, то системний інтерфейс – це набір прийомів взаємодії з комп'ютером, що реалізується операційною системою або її надбудовою. Сучасні операційні системи підтримують командний, WIMP- і SILK-інтерфейси. В даний час поставлена проблема створення суспільного інтерфейсу (social interface).

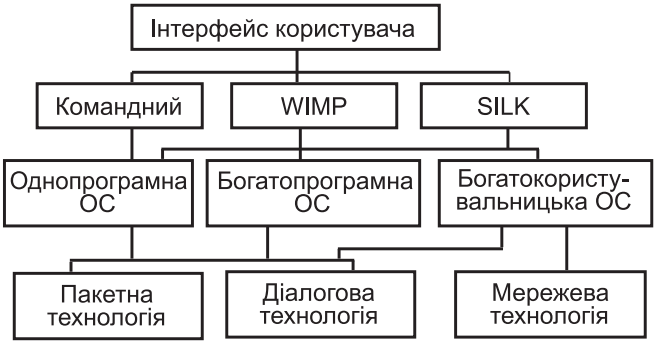



Рисунок 1.47. Класифікація ІТ
за типом користувальницького інтерфейсу

***Командний інтерфейс*** – найпростіший. Він забезпечує видачу на екран системного запрошення для введення команди. Наприклад, в операційній системі MS-DOS запрошення виглядає як З:\>, а в операційній системі UNIX – це звичайно знак долара.

***WIMP-інтерфейс*** розшифровується як Windows (вікно) Image (образ) Menu (меню) Pointer (покажчик). На екрані висвітлюється вікно, що містить образи програм і меню дій. Для вибору одного з них використовується покажчик.

***SILK-інтерфейс*** розшифровується – Spich (мова) Image (образ) Language (мова) Knowledge (знання). При використанні SILK-інтерфейсу на екрані за мовною командою здійснюється переміщення від одних пошукових образів до інших по семантичних зв'язках.

Суспільний інтерфейс буде містити в собі кращі рішення WIMP- і SILK-інтерфейсів. Передбачається, що при використанні суспільного інтерфейсу не потрібно буде розбиратися в меню. Екранні образи однозначно вкажуть подальший шлях. Переміщення від одних пошукових образів до інших буде проходити по семантичних зв'язках.

Операційні системи (ОС) поділяють на однопрограмні, багатопрограмні і багатокористувальницькі.

До однопрограмних операційних систем відносяться, наприклад, MS-DOS. Багатопрограмні операційні системи, наприклад UNIX (XENIX), Windows, починаючи з версії 3.1, DOS 7.0, OS/2 та ін., дозволяють одночасно виконувати кілька додатків. Розрізняються вони алгоритмом поділу часу. Якщо однопрограмні системи працюють або в пакетному режимі, або в діалоговому, то багатопрограмні можуть об'єднувати зазначені режими. Таким чином, ці системи забезпечують і пакетну, і діалогову технології.

Багатокористувальницькі системи реалізуються мережними операційними системами. Вони забезпечують віддалені мережні технології, а також пакетні і діалогові технології для спілкування на робочому місці. Усі три типи інформаційних технологій мають широке застосування в економічних інформаційних системах.

Більшість забезпечуючих і функціональних ІТ можуть бути використані управлінським працівником без додаткових посередників (програмістів). При цьому користувач може впливати на послідовність застосування тих або інших технологій. Таким чином, з погляду участі або неучасті користувача в процесі виконання функціональних ІТ, усі вони можуть бути розділені на пакетні і діалогові.

Економічні задачі, розв'язувані в пакетному режимі, характеризуються наступними властивостями:

- ✧ алгоритм рішення задачі формалізований, процес її рішення не вимагає втручання людини;
- ✧ мається великий обсяг вхідних і вихідних даних, значна частина яких зберігається на магнітних носіях;
- ✧ розрахунок виконується для більшості записів вхідних файлів;
- ✧ великий час рішення задачі обумовлений великими обсягами даних;

✧ задачі вирішуються з заданою періодичністю (регламентність).

Діалоговий режим є не альтернативою пакетного, а його розвитком. Якщо застосування пакетного режиму дозволяє зменшити втручання користувача в процес вирішення задачі, то діалоговий режим передбачає відсутність твердої закріпленої послідовності операцій обробки даних (якщо вона не обумовлена предметною технологією).

Особливе місце займають мережні технології, які забезпечують взаємодію багатьох користувачів.

Інформаційні технології розрізняються ***за ступенем їхньої взаємодії між собою*** (рис. 1.48). Вони можуть бути реалізовані різними технічними засобами (дискетна і мережна взаємодія), а також з використанням різних концепцій обробки і збереження даних (розподілена інформаційна база і розподілена обробка даних).

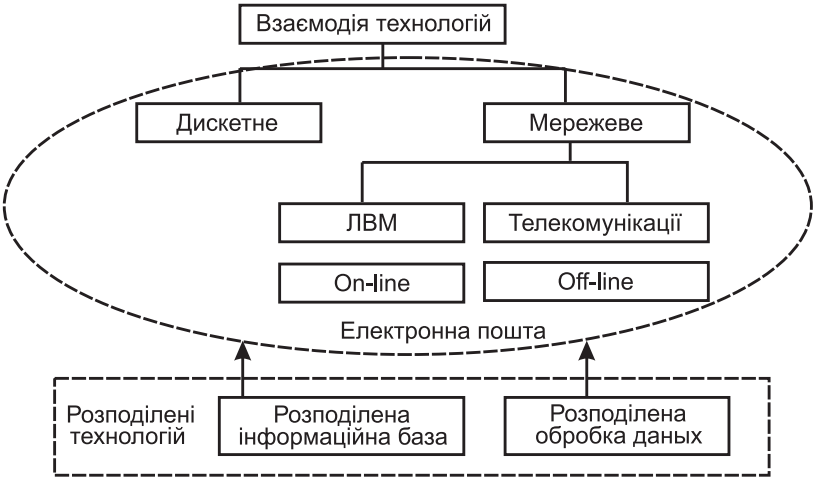



Рисунок 1.48. Класифікація ІТ за ступенем їхньої взаємодії

### 1.5.3 Інформаційні процеси і системи

Інформаційна технологія тісно пов'язана з інформаційними системами, що є для неї основним середовищем. На перший погляд може здатися, що визначення інформаційної технології і системи дуже схожі між собою. Однак це не так.

У загальному випадку роль інформації може обмежуватися емоційною дією на людину, проте найчастіше вона використовується для вироблення керуючих дій, в автоматичних (чисто технічних) і автоматизованих (людино-машинних) системах. У подібних системах можна виділити

окремі етапи (фази) обігу інформації, кожний з яких характеризується певними діями.

Послідовність дій, які виконуються з інформацією, називають **інформаційним процесом.**

Системи, які реалізовують інформаційні процеси, називають **інформаційними системами (ІС).**

Основні етапи (фази) звернення інформації в ІС:

- ✧ збір (сприйняття) інформації;
- ✧ підготовка (перетворення) інформації;
- ✧ передача інформації;
- ✧ обробка (перетворення) інформації;
- ✧ берігання інформації;
- ✧ відображення (відтворення) інформації.

Оскільки матеріальним носієм інформації є сигнал, то реально це будуть етапи обігу і перетворення сигналів (рис. 1.49).

На *етапі сприйняття* інформації здійснюється цілеспрямоване витягання і аналіз інформації про який-небудь об'єкт (процес), внаслідок чого формується образ об'єкту, проводяться його упізнання і оцінка. Головне завдання на цьому етапі – виокремити корисну інформацію від тієї, яка заважає (шумів), що у ряді випадків пов'язано із значними труднощами. Простим видом сприйняття є розрізнення двох протилежних станів: наявності ("так") і відсутності ("ні"), складнішим – вимірювання.

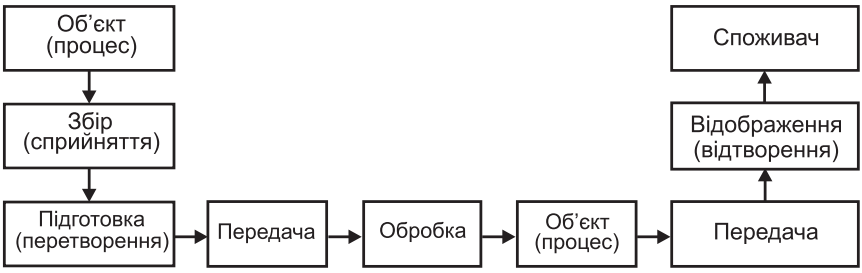



Рисунок 1.49. Етапи обігу інформації в ІС

На *етапі підготовки* інформації здійснюється її первинне перетворення. На цьому етапі проводяться такі операції, як нормалізація, аналого-цифрове перетворення, шифрування. Іноді етап підготовки розглядається як допоміжний на етапі сприйняття. В результаті сприйняття і підготовки виходить сигнал у формі, зручній для передачі, зберігання або обробки.

На *етапі передачі* інформація пересилається з одного місця в інше (від відправника до одержувача-адресата). Передача здійснюється по каналах різної фізичної природи, найпоширенішими з яких є електричні,

електромагнітні і оптичні. Витягання сигналу на виході каналу, схильного до дії шумів, носить характер вторинного сприйняття.

На *етапі обробки* інформації виявляються її загальні і істотні взаємозалежності, важливі для системи. Перетворення інформації на етапі обробки (як і на інших етапах) здійснюється або засобами інформаційної техніки, або людиною.

У загальному випадку під *обробкою* інформації розуміють будь-яке її перетворення, що проводиться за законами логіки, математики, а також за неформальними правилами, заснованими на "здоровому глузді", інтуїції, узагальненому досвіді, поглядах, які склалися, і нормах поведінки. Результатом обробки є теж інформація, яка представлена в інших формах (наприклад, впорядкована за якимись ознаками), або така, яка містить відповіді на поставлені питання (наприклад, рішення деякої задачі). Якщо процес обробки формалізується, він може виконуватися технічними засобами. Кардинальні зрушення в цій галузі сталися завдяки створенню ЕОМ – універсального перетворювача інформації, у зв'язку з чим з'явилися поняття *дані* і *обробка даних*.

**Дані –** факти, відомості, представлені у формалізованому вигляді (закодовані), занесені на ті або інші носії і які допускають обробку за допомогою спеціальних технічних засобів (в першу чергу ЕОМ).

**Обробка даних** передбачає проведення операцій над даними, в першу чергу арифметичних і логічних, для створення нових даних, які об'єктивно потрібні (наприклад, при підготовці відповідальних рішень).

На *етапі зберігання* інформацію записують у запам'ятовуючий для наступного використання. Для зберігання інформації використовують в основному напівпровідникові, магнітні, оптичні носії. Вирішення завдань витягання збереженої інформації (пошуку інформації) пов'язане з розробкою класифікаційних ознак і схем розміщення збереженої інформації, систематизацією, правилами доступу до неї, порядком її поповнення і оновлення, тобто усім тим, що визначає можливість цілеспрямованого пошуку і оперативного витягання збереженої інформації.

*Етап відображення* інформації повинен передувати етапам за участю людини. Мета цього етапу – надати людині потрібну йому інформацію за допомогою пристроїв, здатних впливати на його органи відчуття.

### 1.5.4 Класифікація ІС

***Інформаційна система*** може розглядатися як середовище, яке забезпечує цілеспрямовану діяльність організації. Тобто вона є сукупністю таких компонентів як інформація, процедури, персонал, апаратне і програмне забезпечення, об'єднаних регульованими взаєминами для формування організації як єдиного цілого і забезпечення її цілеспрямованої діяльності (рис. 1.50).

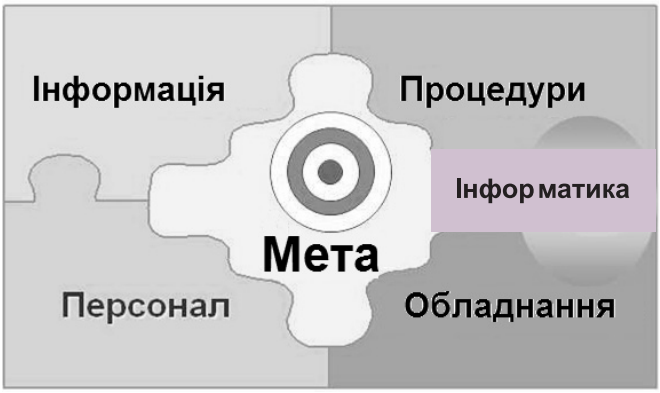



Рисунок 1.50. Основні компоненти інформаційної системи.

Наслідком прийняття такого визначення є висновок, що ефективність інформаційної системи може бути оцінена тільки в термінах її внеску в досягнення організацією її цілей.

**Інформаційні системи** можуть значно розрізнятися за типами об'єктів, характером і обсягом вирішуваних завдань та низкою інших ознак.

Загальноприйнятої класифікації ІС досі не існує, тому їх можна класифікувати за різними ознаками, що викликало існування декількох різних класифікацій ІС (рис.1.51).

Зокрема, інформаційні системи – підрозділяються:

*за масштабами застосування* – на настільні, офісні, корпоративні;

*за ознакою структурованості завдань* – на структуровані (що формалізуються), такі, що не структуруються (що не формалізуються) і такі, що частково структуруються. Частково структуровані діляться на: ІС репортингу та ІС розробки альтернативних рішень (модельні, експертні). Експертні у свою чергу діляться:

на централізовані, децентралізовані і колективного використання з інтеграцією за рівнями управління, за рівнями планування тощо.

*за функціональною ознакою* – на виробничі, маркетингові (аналізу ринку, рекламні, постачальницькі тощо), фінансові (бухгалтерські, статистичні, і тому подібне), кадрові;

*за кваліфікацією персоналу і рівнями управління* – на стратегічні (топ-менеджери), функціональні (менеджери середньої ланки) і оперативні (фахівці);

*за характером обробки інформації*: на системи обробки даних, системи управління, системи підтримки ухвалення рішень;

*за оперативністю обробки даних* – на пакетну обробку і оперативну;

*за мірою автоматизації* – на ручні, автоматичні, автоматизовані;

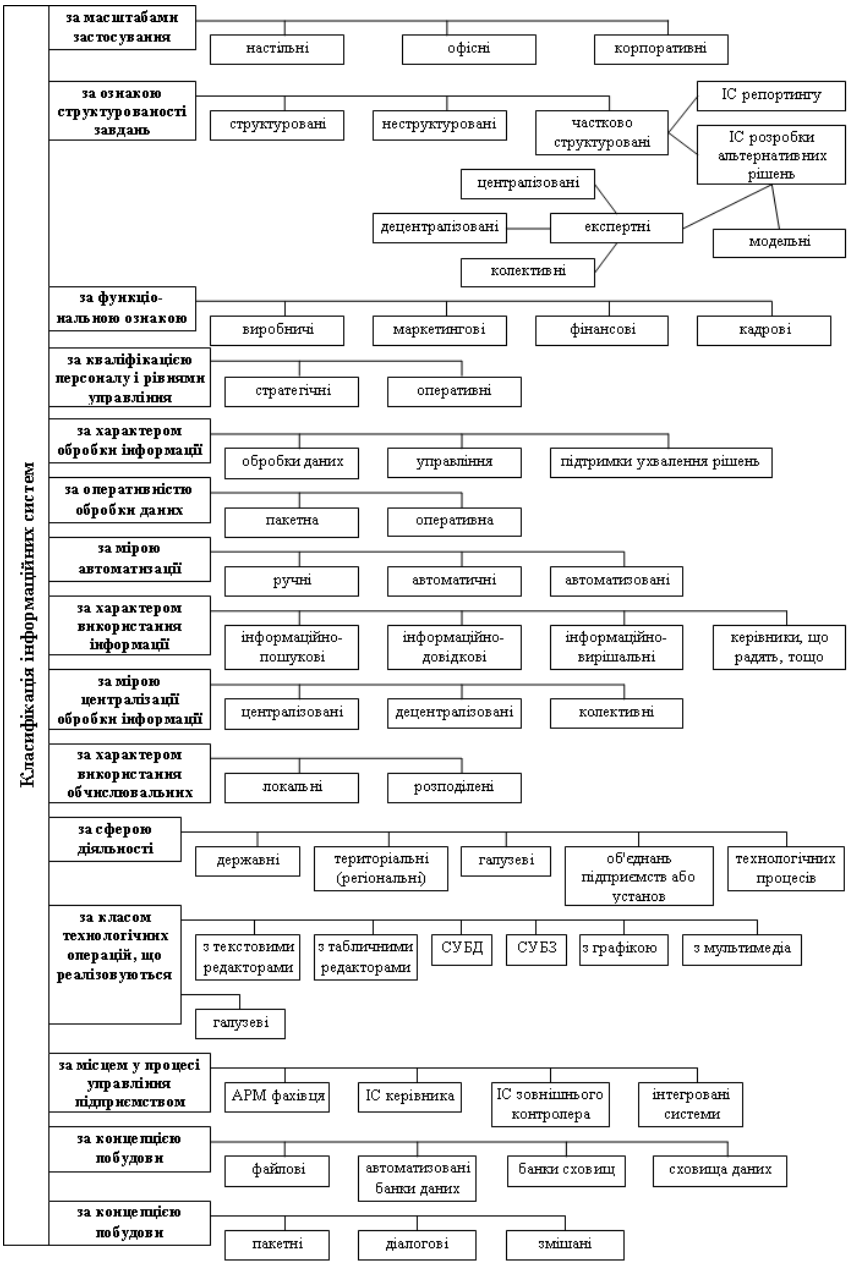



Рисунок 1.51. Класифікація інформаційних систем

**Автоматизовані ІС** – системи, в яких автоматизація може бути неповною (тобто потрібне постійне втручання персоналу). **Автоматичні ІС** – системи, в яких автоматизація є повною, тобто втручання персоналу не потрібно або вимагається тільки епізодично. Поняття "Автоматизована інформаційна система", "комп'ютерна інформаційна система" і просто "інформаційна система" є синонімами.

Більшість автоматизованих ІС є локальними системами і функціонують на рівні підприємств і установ. Нині відбувається інтенсивний процес інтеграції таких систем в корпоративні системи і далі – в регіональні і глобальні системи. Системи вищого рівня стають територіально розосередженими, ієрархічними як за функціональним принципом, так і за їхньою технічною реалізацією. Забезпечення взаємодії територіально розосереджених систем вимагає протяжних високошвидкісних і надійних каналів зв'язку, а збільшення обсягу оброблюваної інформації – ЕОМ високої продуктивності. Це приводить до необхідності колективного використання дорогих засобів автоматизації (ЕОМ і ліній зв'язку) і оброблюваної інформації (баз даних). Технічний розвиток як самих ЕОМ, так і засобів зв'язку дозволив розв'язати цю проблему шляхом переходу до створення *розподілених інформаційно-обчислювальних мереж колективного користування*.

Централізація різних видів інформації в одній мережі дає можливість використовувати її для вирішення широкого спектру завдань, пов'язаних з адміністративним управлінням, плануванням, науковими дослідженнями, конструкторськими розробками, технологією виробництва, постачанням, обліком і звітністю.

Якщо інформація, яка поставляється, витягається з якого-небудь об'єкту (процесу), а вихідна застосовується для цілеспрямованої зміни стану того ж об'єкту (процесу), причому абонентом, який використовує інформацію для вибору основних управляючих дій (ухвалення рішення), є людина, то таку автоматизовану ІС називають **автоматизованою системою управління (АСУ)**.

АСУ знайшли широке застосування в усіх сферах сучасного суспільства в першу чергу як системи управління технологічними процесами і колективами людей. АСУ технологічними процесами служать для автоматизації різних функцій на виробництві. Вони широко використовуються при організації потокових ліній, виготовленні мікросхем, для підтримки технологічного циклу в машинобудуванні і тому подібне. ІС організаційного управління призначені для автоматизації функцій управлінського персоналу, наприклад, ІС управління банками, готелями, торговими фірмами і тому подібне.

*за характером використання інформації* – на інформаційно-пошукові, інформаційно-довідкові, інформаційно-вирішальні, керівники, що радять і тому подібне;

*за мірою централізації обробки інформації* – на централізовані, децентралізовані, інформаційні системи колективного використання;

*за характером використання обчислювальних ресурсів* – на локальні і розподілені;

*за сферою діяльності* – на державні, територіальні (регіональні), галузеві, об'єднань, підприємств або установ, технологічних процесів;

*за класом технологічних операцій, що реалізовуються,* – на системи з текстовими редакторами, системи з табличними редакторами, СУБД, СУБЗ, системи з графікою, мультимедіа, гіпертекстом;

*за місцем у процесі управління підприємством* – на АРМ фахівця, ІС керівника, ІС зовнішнього контролера, інтегровані системи, що об'єднують у собі частину або усе з цих функцій;

*за концепцією побудови* – файлові, автоматизовані банки даних, банки знань, сховища даних (СД);

*за режимом роботи* – на пакетні, діалогові і змішані.

### 1.5.4.1 Класифікація ІС за функціональною ознакою

Функціональна ознака визначає призначення системи, а також її основні цілі, завдання і функції. Структура ІС може бути представлена як сукупність її функціональних підсистем, тому функціональна ознака може бути використана при класифікації ІС.

Тип ІС залежить від того, чиї інтереси вона обслуговує і на якому рівні управління. На рисунку 1.52 показаний варіант класифікації ІС за функціональною ознакою з урахуванням рівнів управління і рівнів кваліфікації персоналу.

З рисунка видно, що чим вище за значущістю рівень управління, тим менший обсяг робіт, які виконуються фахівцем і менеджером за допомогою ІС. Проте при цьому зростають складність і інтелектуальні можливості ІС, її роль у прийнятті менеджером рішень. Будь-який рівень управління потребує інформації з усіх функціональних систем, але в різних обсягах і з різною мірою узагальнення.

Основу піраміди складають ІС, за допомогою яких співробітники-виконавці займаються операційною обробкою даних, а менеджери нижчої ланки – оперативним управлінням. Нагорі піраміди на рівні стратегічного управління ІС змінюють свою роль і стають стратегічними, такими, що підтримують діяльність менеджерів вищої ланки з ухвалення рішень в умовах поганої структурованості поставлених завдань.

У господарській практиці виробничих і комерційних об'єктів **типовими видами діяльності**, які визначають функціональну ознаку класифікації ІС, є:

*Виробнича*

Пов'язана з безпосереднім випуском продукції і спрямована на створення і впровадження у виробництво науково-технічних нововведень;

*Маркетингова*

Включає:

аналіз ринку виробників і споживачів продукції, що випускається, аналіз продажів;

організацію рекламної кампанії з просування продукції;

раціональну організацію матеріально-технічного постачання;

*Фінансова*

Пов'язана з організацією контролю і аналізу фінансових ресурсів фірми на основі бухгалтерської, статистичної, оперативної інформації;

*Кадрова*

Спрямована на підбір і розставляння необхідних фірмі фахівців, а також ведення службової документації.

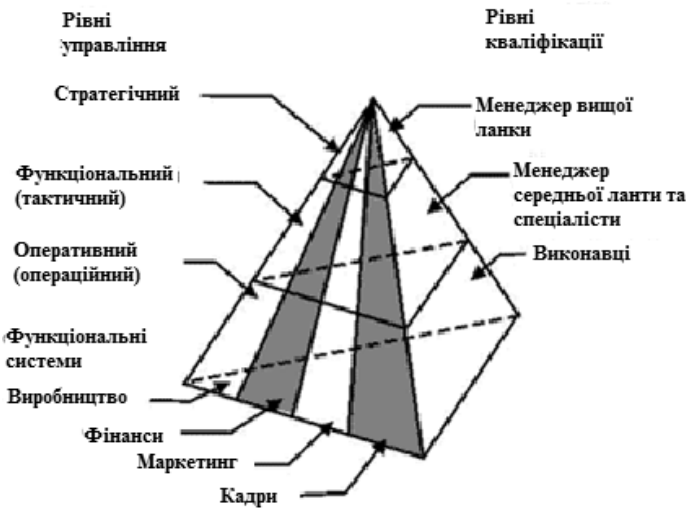



Рисунок 1.52. Типи ІС залежно від функціональної ознаки з урахуванням рівнів управління і кваліфікації персоналу

Зазначені напрями діяльності визначили типовий набір ІС:

**виробничі системи; системи маркетингу; фінансові і облікові системи; системи кадрів (людських ресурсів);** інші типи, що виконують допоміжні функції залежно від специфіки діяльності фірми.

У великих фірмах основна ІС функціонального призначення може складатися з декількох підсистем для виконання підфункцій. Наприклад

Підсистеми виробничої ІС – інформаційної системи

- конструкторської підготовки виробництва;
- технологічної підготовки виробництва;
- управління матеріально-технічним постачанням;

– управління виробничим процесом;
– комп'ютерного інжинірингу тощо.

Для кращого розуміння функціонального призначення ІС у таблиці 1.17 наведені кожний з розглянутого вище виду та вирішувані в них типові завдання.

Таблиця 1.17

**Функції інформаційних систем**

| Система маркетингу | Виробничі системи | Фінансові і облікові системи | Система кадрів (людських ресурсів) | Інші системи (наприклад ІС керівництво) |
|---|---|---|---|---|
| Дослідження ринку і прогнозування продажів | Планування об'ємів робіт і розробка календарних планів | Управління портфелем замовлень | Аналіз і прогнозування потреби в трудових ресурсах | Контроль за діяльністю фірми |
| Управління продажами | Оперативний контроль і управління виробництвом | Управління кредитною політикою | Ведення архівів записів про персонал | Виявлення оперативних проблем |
| Рекомендації по виробництву нової продукції | Аналіз роботи устаткування | Розробка фінансового плану | Аналіз і планування підготовки кадрів | Аналіз управлінських і стратегічних ситуації |
| Аналіз і встановлення ціни | Участь у формуванні замовлень постачальникам | Фінансовий аналіз і прогнозування | | Забезпечення процесу вироблення стратегічних рішень |
| Облік замовлень | Управління запасами | Контроль бюджету | | |
| | | Бухгалтерський облік і розрахунок зарплати | | |

Відповідно до характеру обробки інформації в ІС на різних рівнях управління економічною системою (оперативному, тактичному і стратегічному) виділяються декілька типів ІС.

**Системи обробки даних – СОД (EDP – Electronic Data Processing, СОД)** призначені для обліку і оперативного регулювання господарських операцій, підготовки стандартних документів для зовнішнього середовища (рахунків, накладних, платіжних доручень, розрахунку заробітної плати, статистичній звітності і тому подібне). Такі системи разом з функціями введення, вибірки, корекції інформації виконують математичні розрахунки без застосування методів оптимізації. Горизонт оперативного управління господарськими процесами складає від одного до декілька днів і реалізує

реєстрацію і обробку подій (оформлення і моніторинг виконання замовлень, прихід і витрата матеріальних цінностей на складі, ведення табеля обліку робочого часу і так далі). Ці завдання мають ітеративний, регулярний характер, виконуються безпосередніми виконавцями господарських процесів (робітниками, комірниками, адміністраторами і т.д.) і пов'язані з оформленням і пересилкою документів відповідно до чітких певних алгоритмів. Результати виконання господарських операцій через екранні форми вводяться у базу даних.

1.5.4.2 Класифікація ІС за характером обробки інформації

***Інформаційні системи (ІС) управління – ІСУ (MIS – Management Information System, ІСУ)*** орієнтовані на тактичний рівень управління: середньострокове планування, аналіз і організацію робіт впродовж декількох тижнів (місяців), наприклад аналіз і планування постачань, збуту, складання виробничих програм. Для цього класу завдань характерні регламентована (періодична повторюваність) формування результатних документів і чіткий певний алгоритм рішення завдань, наприклад зведення замовлень для формування виробничої програми і визначення потреби в комплектуючих деталях і матеріалах на основі специфікації виробів. Рішення подібних завдань призначене для керівників різних служб підприємств (відділів матеріально-технічного постачання і збуту, цехів і так далі). Завдання вирішуються на основі накопиченої бази оперативних даних.

***Системи підтримки прийняття рішень – (DSS – Decision Support System, СППР)*** використовуються в основному на верхньому рівні управління (керівництво фірм, підприємств, організацій), що має стратегічне довгострокове значення впродовж року або декількох років. До таких завдань відносяться формування стратегічних цілей, планування залучення ресурсів, джерел фінансування, вибір місця розміщення підприємств тощо. Рідше за завдання класу СППР вирішуються на тактичному рівні, наприклад при виборі постачальників або укладенні контрактів з клієнтами. Завдання СППР мають, як правило, нерегулярний характер. Для завдань СППР властиві недостатність наявної інформації, її суперечність і нечіткість, переважання якісних оцінок цілей і обмежень, слабка формалізованість алгоритмів рішення. В якості інструментів узагальнення найчастіше використовуються засоби складання аналітичних звітів довільної форми, методи статистичного аналізу, експертних оцінок і систем, математичного та імітаційного моделювання. При цьому використовуються бази узагальненої інформації, інформаційні сховища, бази знань про правила і моделі прийняття рішень.

Ідеальною вважається ІС, яка включає усі три типи перерахованих ІС.

Інформаційна технологія є процесом, який складається з чітко регламентованих правил виконання операцій, дій, етапів різного ступеня

складності над даними, що зберігаються в комп'ютерах. Основна мета інформаційної технології – у результаті цілеспрямованих дій з переробки первинної інформації одержати необхідну для користувача інформацію.

Інформаційна система є середовищем, яке складається з елементів, якими є комп'ютери, комп'ютерні мережі, програмні продукти, бази даних, люди, різного роду технічні і програмні засоби зв'язку тощо. Основна мета інформаційної системи – організація збереження і передачі інформації. Інформаційна система є людино-комп'ютерною системою обробки інформації.

Реалізація функцій інформаційної системи неможлива без знання застосовуваної інформаційної технології. *Інформаційна технологія може існувати і поза сферою інформаційної системи*, наприклад, інформаційна технологія роботи в середовищі текстового процесора Word, що не є інформаційною системою; інформаційна технологія мультимедіа, де за допомогою телекомунікаційного зв'язку здійснюються передача й обробка на комп'ютері зображення і звуку.

Таким чином, інформаційна технологія є більш ємним поняттям, що відбиває сучасне представлення про процеси перетворення інформації в інформаційному суспільстві. У вмілому сполученні двох інформаційних технологій – управлінської і комп'ютерної – запорука успішної роботи інформаційної системи.

Узагальнюючи усе вищенаведене, можна дати трохи вужчі, ніж введені раніше, визначення інформаційної системи та інформаційної технології, які реалізовані засобами комп'ютерної техніки.

**Інформаційна технологія** – сукупність чітко визначених цілеспрямованих дій персоналу з переробки інформації на комп'ютері.

**Інформаційна система** – людино-комп'ютерна система для підтримки прийняття рішень і виробництва інформаційних продуктів, яка використовує комп'ютерну інформаційну технологію.

**Інформаційна система** – організаційно впорядкована сукупність документів, інформаційних технологій, у тому числі з використанням засобів обчислювальної техніки і зв'язку, які реалізовують інформаційні процеси.

У загальному випадку під *інформаційними ресурсами* розуміють увесь наявний в ІС обсяг інформації, відчуженої від її творців і призначеної для громадського використання.

### 1.5.5 Нова інформаційна технологія

Інформаційна технологія є найважливішою складовою процесу використання інформаційних ресурсів суспільства. Дотепер вона пройшла кілька еволюційних етапів, зміна яких визначалася головним чином науково-технічним прогресом, появою нових технічних засобів переробки

інформації. У сучасному суспільстві основним технічним засобом переробки інформації служить персональний комп'ютер, що істотно вплинув як на концепцію побудови і використання технологічних процесів, так і на якість результатної інформації. Впровадження персонального комп'ютера в інформаційну сферу і застосування телекомунікаційних засобів зв'язку визначили новий етап розвитку інформаційної технології і, як наслідок, зміну її назви за рахунок приєднання одного із синонімів: "нова", "комп'ютерна" або "сучасна".

Прикметник "нова" підкреслює новаторський, а не еволюційний характер цієї технології. Її впровадження є новаторським актом у тому сенсі, що вона істотно змінює зміст різних видів діяльності в організаціях. У поняття нової інформаційної технології включені також комунікаційні технології, що забезпечують передачу інформації різними засобами, а саме – телефон, телеграф, телекомунікації, факс та ін. У таблиці 1.18 наведені основні характерні риси нової інформаційної технології.

Таблиця 1.18

**Основні характеристики нової інформаційної технології**

| Методологія | Основні ознаки | Результат |
|---|---|---|
| Принципово нові засоби опрацювання інформації | Інтеграція в технологію управління | Нова технологія комунікацій |
| Цілісні технологічні системи | Інтеграція функцій фахівців і менеджерів | Нова технологія опрацювання інформації |
| Цілеспрямоване створення, передача, зберігання і відображення інформації | Врахування закономірностей соціального середовища | Нова технологія ухвалення управлінських рішень |

**Нова інформаційна технологія (НІТ)** – інформаційна технологія з "дружнім" інтерфейсом роботи користувача, яка використовує персональні комп'ютери і телекомунікаційні засоби.

Прикметник "комп'ютерна" підкреслює, що основним технічним засобом її реалізації є комп'ютер.

Існує три основних принципи нової (комп'ютерної) інформаційної технології:

- інтерактивний (діалоговий) режим роботи з комп'ютером;
- інтегрованість (стикування, взаємозв'язок) з іншими програмними продуктами;
- гнучкість процесу зміни як даних, так і постановок задач.

Більш точним варто вважати все-таки термін *нова,* а не *комп'ютерна, інформаційна технологія,* оскільки він відбиває в її структурі не тільки технології, засновані на використанні комп'ютерів, але і технології,

засновані на інших технічних засобах, особливо на засобах, які забезпечують телекомунікацію.

Слід зазначити, що останнім часом термін НІТ поступово починає втрачати слово "нова", а під інформаційною технологією починають розуміти той зміст, що вкладається в НІТ.

Нова інформаційна технологія – це технологія, заснована на наступних принципах:

- ✧ повсюдному застосуванні ЕОМ і оргтехніки;
- ✧ активній участі користувачів (непрофесіоналів в галузі обчислювальної техніки і програмуванні ) в інформаційному процесі;
- ✧ високому рівні "дружнього" користувальницького інтерфейсу;
- ✧ широкому використанні пакетів прикладних програм (ППП) загального і проблемного призначення;
- ✧ можливості для користувача доступу до баз даних і програм, у тому числі і віддаленим, завдяки локальним і глобальним комп'ютерним мережам;
- ✧ аналізі ситуацій при виробленні і прийнятті управлінських рішень за допомогою автоматизованих робочих місць фахівців;
- ✧ застосуванні систем штучного інтелекту;
- ✧ впровадженні експертних систем;
- ✧ використанні телекомунікацій;
- ✧ створенні геоінформаційних систем та інших технологій.

Для позначення якісно нового етапу в розвитку комп'ютерних технологій пропонується використовувати термін "ноокомпьютинг" як ключового ноологизма (unicast –адресация, протокол нового покоління IPv6)

Застосування IPv6 (по мережі / 64 на абонента; використовується тільки unicast – адресация) забезпечить можливість використання більше 300 млн. IP – адресов на кожного жителя землі.

Спостерігається стрімкий ріст відкритих API – як в сегменті інтернет-сервісів (Google Sites API, Google Wave API, Мій Світ@Mail. Ru API або YouTube API і ін.) і соціальних мереж (ніби Facebook API, ВКонтакте API), так і в сегменті хмарних обчислень (ніби OpenCloud API) і ряду інших напрямів.

Фактично, в умовах ноокомпьютинга і ноосети необхідно вести мову вже і про ноопрограмування, єдиною платформою для якого є увесь глобальний мережевий простір.

Ще одно знакова подія сталася 5 жовтня 2011 року, коли фірма Google оголосила, що досягла важливої віхи в історії розвитку Google Earth (Планета Земля в російському варіанті): програмне забезпечення, в якості клієнта для ПК, мобільного застосування або плагина для браузерів.

Що формується в результаті "ноопарадигма" припускає, що разом з поняттям ноосфера центральне місце і ключову роль в процесах, що

відбуваються, покликаний грати ноокомпьютинг, становлення якого практично почалося і може в основному завершитися приклад до 2018 року. До цього часу суперкомп'ютерними системами буде здоланий рубіж в один экзафлопс, а рівень їх можливостей стане приблизно порівнянний з людським розумом.

### 1.5.6 Складові частини інформаційних технологій

Використовувані у виробничій сфері стосовно до технології такі технологічні поняття, як норма, норматив, технологічний процес, технологічна операція й інші, можуть застосовуватися і в інформаційній технології. Розробка їх у будь-якій технології, у тому числі і в інформаційній, починається з визначення мети. Потім випливає структурування всіх передбачуваних дій, що приводять до наміченої мети, і вибирається необхідний програмний інструментарій.

Технологічний процес переробки інформації може бути представлений у вигляді багаторівневої ієрархічної структури (рис. 1.53).

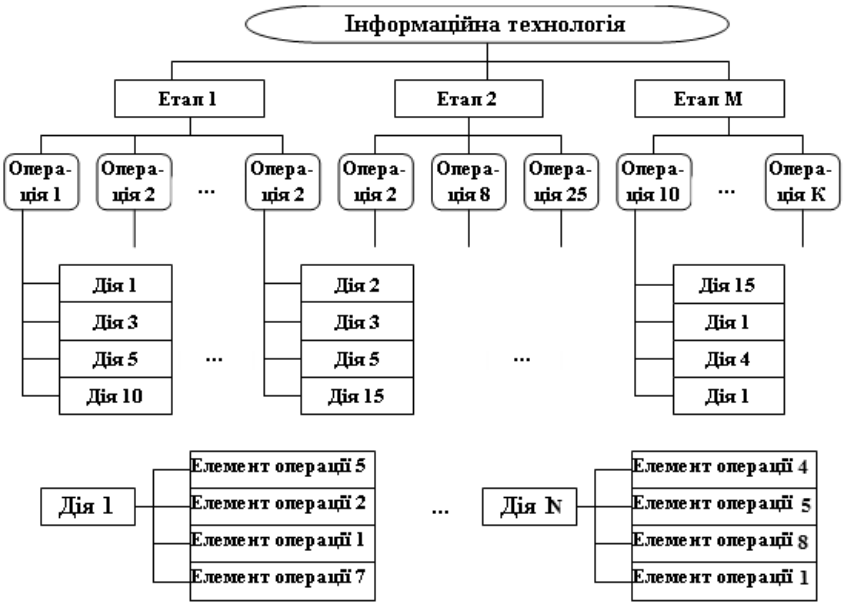



Рис. 1.53. Представлення інформаційної технології у вигляді ієрархічної структури, що складається з етапів, дій, операцій

**1-й рівень** – *етапи*, де реалізуються порівняно тривалі технологічні процеси, які складаються з операцій і дій наступних рівнів. Наприклад,

технологія створення шаблону форми документа в середовищі текстового процесора Word складається з наступних етапів:

етап 1 – створення постійної частини форми у виді текстів і таблиць;

етап 2 – створення постійної частини форми у виді кадру, куди потім розміщується малюнок;

етап 3 – створення змінної частини форми;

етап 4 – захист і збереження форми.

**2-й рівень** – *операції,* у результаті виконання яких буде створений конкретний об'єкт в обраній на 1-му рівні програмному середовищі. Так, наприклад, етап 2 технології створення постійної частини форми документа у виді кадру в середовищі текстового процесора Word складається з наступних операцій:

операція 1 – створення кадру;

операція 2 – настроювання кадру;

операція 3 – впровадження в кадр малюнка.

**3-й рівень** – *дії* – сукупність стандартних для кожного програмного середовища прийомів роботи, що приводять до виконання поставленої у відповідній операції мети. Кожна дія змінює зміст екрана. Приміром, операція 3 – впровадження в кадр малюнка в середовищі текстового процесора Word, складається з наступних дій:

дія 1 – установка курсору в кадрі;

дія 2 – виконання команди **ВСТАВКА, Рисунок;**

дія 3 – установка значень параметрів у діалоговому вікні.

**4-й рівень** – *елементарні операції* з керування мишею і клавіатурою. Елементарними операціями можуть бути: введення команди, натискання правої кнопки миші, вибір пункту меню тощо.

Необхідно розуміти, що опанування інформаційною технологією і подальше її використання повинні звестися до того, що спочатку варто добре освоїти набір елементарних операцій, число яких обмежене. З цього обмеженого числа елементарних операцій у різних комбінаціях складається дія, а з дій, також у різних комбінаціях, складаються операції, що визначають той або інший технологічний етап. Сукупність технологічних етапів утворить технологічний процес (технологію).

Технологічний процес необов'язково повинний складатися з усіх рівнів, представлених на рис. 1.53. Він може починатися з будь-якого рівня і не включати, наприклад, етапи або операції, а складатися тільки з дій. Для реалізації етапів технологічного процесу можуть використовуватися різні програмні середовища.

Інформаційна технологія, як і будь-яка інша, повинна відповідати наступним вимогам:

– забезпечувати високий ступінь розчленовування всього процесу обробки інформації на етапи (фази), операції, дії;

– включати весь набір елементів, необхідних для досягнення поставленої мети;
– мати регулярний характер. Етапи, дії, операції технологічного процесу можуть бути стандартизовані й уніфіковані, що дозволить ефективніше здійснювати цілеспрямоване керування інформаційними процесами.

### 1.5.6.1 Інструментарій інформаційної технології і його види. Переваги, що забезпечує комп'ютерна технологія

Реалізація технологічного процесу матеріального виробництва здійснюється за допомогою різних технічних засобів, до яких відносяться: устаткування, верстати, інструменти, конвеєрні лінії тощо.

За аналогією і в інформаційній технології повинні існувати відповідні технічні засоби. Такими технічними засобами виробництва інформації є апаратне, програмне і математичне забезпечення цього процесу. З їхньою допомогою здійснюють переробку первинної інформації в інформацію нової якості. Окремо з цих засобів виділяються програмні продукти, так звані інструментарії інформаційної технології.

**Інструментарій інформаційної технології –** один або кілька взаємозалежних програмних продуктів для певного типу комп'ютера, технологія роботи з якими дозволяє досягти поставленої користувачем мети.

Як інструментарій можна використовувати такі розповсюджені види програмних продуктів для персонального комп'ютера: текстовий процесор (редактор), настільні видавничі системи, електронні таблиці, системи управління базами даних, електронні записні книжки, електронні календарі, інформаційні системи функціонального призначення (фінансові, бухгалтерські, для маркетингу й ін.), експертні системи і т.д.

При розгляді переваг, що забезпечує комп'ютерна технологія, можна виділити три етапи.

*1-й етап* (з початку 1960-х рр.) характеризується досить ефективною обробкою інформації при виконанні рутинних операцій з орієнтацією на централізоване колективне використання ресурсів обчислювальних центрів. Основним критерієм оцінки ефективності створюваних інформаційних систем була різниця між витраченими на розробку і зекономленими в результаті впровадження засобами. Основною проблемою на цьому етапі була психологічна – погана взаємодія користувачів, для яких створювалися інформаційні системи, і розроблювачів через розходження їхніх поглядів і розуміння розв'язуваних проблем. Як наслідок цієї проблеми, створювалися системи, які користувачі погано сприймали і, незважаючи на їхній досить великі можливості, не використовували повною мірою.

*2-й етап* (із середини 1970-х рр.) пов'язаний з появою персональних комп'ютерів. Змінився підхід до створення інформаційних сис-

тем – орієнтація змістилася у бік індивідуального користувача для підтримки прийнятих ним рішень. Користувач зацікавлений у проведеній розробці, налагоджується контакт із розробником, виникає взаєморозуміння обох груп фахівців. На цьому етапі використовується як централізована обробка даних, характерна для першого етапу, так і децентралізована, що базується на вирішенні локальних задач і роботі з локальними базами даних на робочому місці користувача.

*3-й етап* (з початку 1990-х рр.) пов'язаний з поняттям аналізу стратегічних переваг у бізнесі і заснований на досягненнях телекомунікаційної технології розподіленої обробки інформації. Інформаційні системи мають на меті не просте збільшення ефективності обробки даних і допомогу керівнику. Відповідні інформаційні технології повинні допомогти організації вистояти в конкурентній боротьбі й одержати перевагу.

### 1.5.6.2 Приклади інформаційних технологій в економічній діяльності

Найефективніше інформатизації піддаються наступні види діяльності: бухгалтерський облік, довідкове й інформаційне забезпечення економічної діяльності, організація праці керівника, документообіг, економічна і фінансова діяльність, навчання.

Найбільше число пакетів прикладних програм ППП створено для бухгалтерського обліку. Серед них можна відзначити "1С:Бухгалтерія", "Турбо-Бухгалтер", "Инфо-Бухгалтер", "Вітрило", "ABACUS", "Бэмби+", "Бухкомплекс", "Бэст", "Лука".

Довідкове й інформаційне забезпечення економічної діяльності представлений наступними ППП: "ГАРАНТ" (податки, бухоблік, аудит, підприємництво, банківська справа, валютне регулювання, митний контроль), "КОНСУЛЬТАНТ+" (податки, бухоблік, аудит, підприємництво, банківська справа, валютне регулювання, митний контроль).

Економічна і фінансова діяльність підтримується наступними ППП:

- ✧ "Економічний аналіз і прогноз діяльності фірми, організації" (фірма "ИНЕК"), що реалізує функції: економічний аналіз діяльності фірми, підприємства; бізнес-план; техніко-економічне обґрунтування повернення кредитів; аналіз і добір варіантів діяльності; прогноз балансу, потоків коштів і готової продукції;
- ✧ "Фінансовий аналіз підприємства" (фірма "Инфософт"), що реалізує функції: загальна оцінка фінансового стану; аналіз фінансової стійкості; аналіз ліквідності балансу; аналіз фінансових коефіцієнтів (ліквідність, маневреність, покриття, співвідношення позикових і власних засобів); аналіз коефіцієнтів ділової активності; розрахунок і аналіз коефіцієнтів оборотності; оцінка рентабельності виробництва. В галузі ство-

рення фінансово-кредитних систем працюють фірми "Диа-софт", "Інверсія", R-Style, Программбанк, "Асофт" і ін.

В умовах конкуренції виграють ті підприємства, чиї стратегії в бізнесі поєднуються зі стратегіями в галузі інформаційних технологій. Тому реальною альтернативою варіанту вибору єдиного пакета є підбор деякого набору пакетів різних постачальників, що задовольняють щонайкраще тієї або іншої функції АІС (підхід mix-and-match). Такий підхід пом'якшує деякі проблеми, що виникають при впровадженні і прив'язці програмних засобів, а АІТ буде краще відповідати функціям конкретної індивідуальності предметної області.

Останнім часом усе більше число банків, організацій, підприємств воліють купувати готові пакети і технології, а якщо необхідно, додавати до них своє програмне забезпечення, тому що розробка власних АІС і АІТ пов'язана з високими витратами і ризиком. Ця тенденція привела до того, що постачальники систем змінили раніше існуючий спосіб виходу на ринок. Як правило, розробляється і пропонується тепер базова система, яка адаптується відповідно до побажань індивідуальних клієнтів. При цьому користувачам надаються консультації, які допомагають мінімізувати терміни впровадження систем і технологій, найефективніше їх використовувати, підвищити кваліфікацію персоналу.

Наприклад, банківська АІС Atlas фірми Internet спроектована для будь-яких можливих конфігурацій системи. Банки можуть, використовуючи свій власний персонал, настроїти конфігурацію системи відповідно до своїх вимога. Для цього в системі Atlas мається повний набір засобів розробки – навчання, консультації і підтримка.

Аналогічна ситуація склалася при розробці АІС в інших галузях економіки. Так, наприклад, розробка АІС для страхової діяльності під силу тільки спеціалізованим організаціям, що узагальнюють практичний досвід роботи страховиків, тісно взаємодіючим з аудиторськими організаціями і які мають штат висококваліфікованих постановників задач і програмістів.

В галузі автоматизації проектування АІС і АІТ за останнє десятиліття сформувався новий напрямок – CASE (Computer-Aided Software/System Engineering). Лавиноподібне розширення галузей застосування ПЕОМ, зростання складності інформаційних систем, підвищення вимог до них привели до необхідності індустріалізації технологій їхнього створення. Важливий напрямок у розвитку технологій склали розробки інтегрованих інструментальних засобів, що базуються на концепціях життєвого циклу і керування якістю АІС і АІТ, що представляють собою комплексні технології, орієнтовані на створення складних автоматизованих управлінських систем і підтримку їхнього повного життєвого циклу або ряду його основних етапів. Подальший розвиток робіт у цьому напрямку привів до створення низки концептуально цілісних, оснащених

високорівневими засобами проектування і реалізації варіантів, доведених по якості і легкості тиражування до рівня програмних продуктів технологічних систем, що одержали назва CASE-систем або CASE-технологій.

В даний час не існує загальноприйнятого визначення CASE. Зміст цього поняття звичайно визначається переліком задач, розв'язуваних за допомогою CASE, а також сукупністю застосовуваних методів і засобів. CASE-технологія є сукупністю методів аналізу, проектування, розробки і супроводу АІС, підтриманою комплексом взаємозалежних засобів автоматизації. CASE – це інструментарій для системних аналітиків, розроблювачів і програмістів, що дозволяє автоматизувати процес проектування і розробки АС, що міцно ввійшов у практику створення і супроводу АІС і АІТ. При цьому CASE-системи використовуються не тільки як комплексні технологічні конвейєри для виробництва АІС і АІТ, але і як могутній інструмент рішення дослідницьких і проектних задач, таких як структурний аналіз предметної області, специфікація проектів засобами мов програмування четвертого покоління, випуск проектної документації, тестування реалізацій проектів, планування і контроль розробок, моделювання ділових пропозицій з метою вирішення задач оперативного і стратегічного планування і керування ресурсами тощо.

Основна мета CASE-технології полягає в тому, щоб виокремити проектування АІС і АІТ від її кодування і наступних етапів розробки, а також максимально автоматизувати процеси розробки і функціонування систем.

При використанні CASE-технологій змінюється технологія ведення робіт на всіх етапах життєвого циклу автоматизованих систем і технологій, при цьому найбільші зміни стосуються етапів аналізу і проектування. У більшості сучасних CASE-систем застосовуються методології структурного аналізу і проектування, засновані на наочних діаграмних техніках, при цьому для опису моделі проектованої АІС використовуються графи, діаграми, таблиці і схеми. Такі методології забезпечують строгий і наочний опис проектованої системи, що починається з її загального огляду і потім деталізується, здобуваючи ієрархічну структуру з усе більшим числом рівнів.

CASE-технології успішно застосовуються для побудови практично всіх типів АІС, однак стійке положення вони займають в галузі забезпечення розробки ділових і комерційних АІС. Широке застосування CASE-технологій обумовлене масовістю цієї прикладної галузі, у якій CASE застосовується не тільки для розробки АІС, але і для створення моделей систем, які допомагають комерційним структурам вирішувати задачі стратегічного планування, керування фінансами, визначення політики фірм, навчання персоналу та ін. Цей напрямок одержав свою власну назву – бізнес-аналіз. Наприклад, для найбільш швидкої й ефективної розробки

високоякісної банківської системи фінансисти усі частіше звертаються до технології CASE. Постачальники цієї технології зважають на становище фінансистів і швидко розширюють ринок засобів. Найшвидшому впровадженню технології CASE сприяє також ускладнення банківських систем.

Поява CASE – не революція в автоматизації проектування АІС, а результат природного еволюційного розвитку всієї галузі засобів, які раніше малі назву інструментальних або технологічних. Однією із ключових ознак є підтримка методологій структурного системного аналізу і проектування.

Із самого початку метою розвитку CASE-технологій було подолання обмежень при використанні структурних методологій проектування 1960-1970-х рр. (складності розуміння, великій трудомісткості і вартості використання, труднощі внесення змін у проектні специфікації тощо) за рахунок їхньої автоматизації та інтеграції підтримуючих засобів. Таким чином, CASE-технології не можуть вважатися самостійними методологіями, вони тільки розвивають структурні методології і роблять більш ефективним їхнє застосування за рахунок автоматизації.

Крім автоматизації структурних методологій і, як наслідок, можливості застосування сучасних методів системної і програмної інженерії, CASE-технології мають такі основні достоїнства:

- ✧ поліпшують якість створюваних АІС (АІТ) за рахунок засобів автоматичного контролю (насамперед, контролю проекту);
- ✧ дозволяють за короткий час створювати прототип майбутньої АІС (АІТ), що надає можливість на ранніх етапах оцінити очікуваний результат;
- ✧ прискорюють процес проектування і розробки системи;
- ✧ звільняють розроблювача від рутинної роботи, дозволяючи йому цілком зосередитися на творчій частині розробки;
- ✧ підтримують розвиток і супровід розробки АІС (АІТ);
- ✧ підтримують технології повторного використання компонентів розробки.

Більшість CASE-засобів заснована на науковому підході, який одержав назву "методологія/метод/нотація/засіб". Методологія формулює провідні вказівки для оцінки і вибору проекту розроблювальної АІС, кроки роботи і їхню послідовність, а також правила застосування і призначення методів.

На сьогодні CASE-технологія оформилася в самостійний наукомісткий напрямок, який спричинив за собою утворення могутньої CASE-індустрії, яка поєднує сотні фірм і компаній різної орієнтації. Серед них виділяються:

- ✧ компанії-розроблювачі засобів аналізу і проектування АІС і АІТ із широкою мережею дистриб'юторських і дилерських фірм; фірми-розроблювачі спеціальних засобів з орієнтацією на вузькі предметні області або на окремі етапи життєвого циклу АІС;

- ✧ навчальні фірми, які організовують семінари і курси підготовки фахівців;
- ✧ консалтингові фірми, які здійснюють практичну допомогу при використанні CASE-пакетів для розробки конкретних АІС;
- ✧ фірми, які спеціалізуються на випуску періодичних журналів і бюлетенів з CASE-технологій.

Практично жоден серйозний закордонний проект АІС і АІТ не здійснюється в даний час без використання CASE-засобів.

## 1.6 Поняття та призначення програмного забезпечення

### 1.6.1 Поняття та призначення програмного забезпечення, класифікація

Призначення будь-якої системи обробки даних полягає в тому, щоб перетворювати дані в корисну інформацію. Для досягнення цієї мети існує обчислювальна система – взаємозалежна сукупність апаратних засобів обчислювальної техніки і програмного забезпечення, яка призначена для обробки інформації. Дуже важливо, щоб усі компоненти цієї системи використовувалися ефективно. Створення операційних систем (ОС) мотивувалося саме цією метою.

Призначення й функції сучасних ОС, безсумнівно обумовлюються поточними потребами людини в комп'ютерній техніці, однак більша частина цих функцій властива ОС із самої їхньої появи.

Персональний комп'ютер (ПК) не може працювати без програмного забезпечення. Для того щоб він "ожив", необхідний комплекс різних програм. Для ПК розроблено безліч програм.

Сукупність програм, яка призначена для вирішення задач на ПК, називається ***програмним забезпеченням***. Склад програмного забезпечення ПК називають ***програмною конфігурацією***.

Залежно від призначення все програмне забезпечення для ПК можна розділити на три групи: системне забезпечення, системи програмування і прикладне забезпечення (рис. 1.54).

Системне забезпечення складається з операційних систем і засобів контролю і діагностики.

**Операційна система (ОС)** – це сукупність програм, які забезпечують керування процесом обробки інформації і взаємодію між апаратними засобами і користувачем. ОС складає основу програмного забезпечення ПК.

До складу ОС включаються також системні програми: програми-оболонки, драйвери та утиліти.

Програми-оболонки – програми, які забезпечують більш зручний і наочний спосіб спілкування користувача з ПК, ніж штатні засоби ОС. Для

MS DOS використовується звичайно оболонка Norton Commander, для Windows 3.1 – Norton Desktop і для Windows 95 – Norton Navigator.

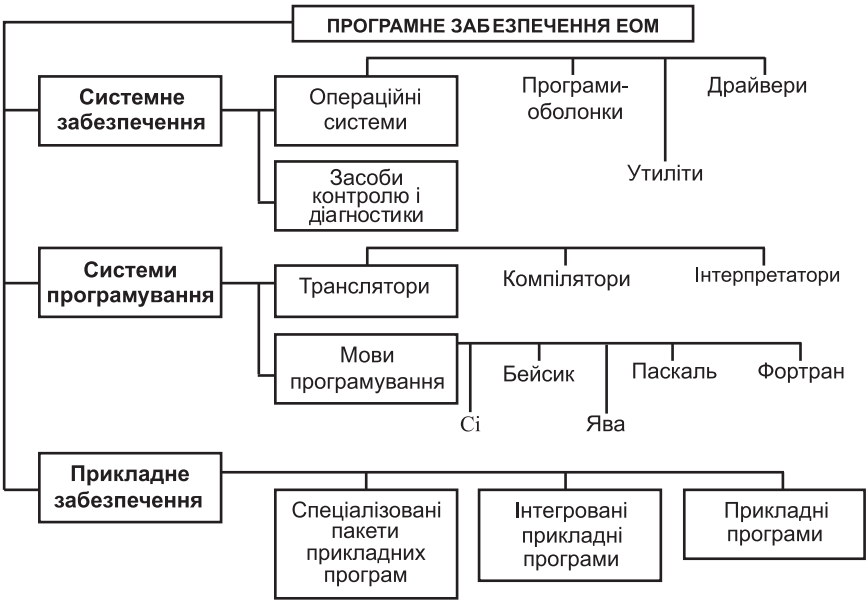



Рисунок 1.54. Структура програмного забезпечення ПК

Драйвери – програми, які забезпечують взаємодію ОС із зовнішніми пристроями (принтерами, накопичувачами на магнітних дисках, лазерними дисками CD-ROM, клавіатурою, маніпулятором – "миша" та ін.

Утиліти – допоміжні програми, що забезпечують розширення можливостей ОС (русифікатори, антивірусні, резервування, архіватори, оптимізації дисків, захисту доступу, динамічного стискання та ін.).

Система контролю і діагностики призначена для перевірки пристроїв ПК, вона автоматично виявляє несправності, їхнє місце і сповіщає про це користувачу.

**Системи програмування** (інструментальне програмне забезпечення) містять у собі мови програмування і транслятори, що дозволяють розробляти як системне, так і прикладне програмне забезпечення.

Мови програмування – це штучні мови, створені для цілком визначених цілей – опису алгоритмів обробки даних. У даний час у світі нараховується кілька сотень мов програмування, однак тільки десяток з них одержали широке поширення. Це ФОРТРАН, КОБОЛ, ПАСКАЛЬ, БЕЙСИК, ЛИСП, СІ, СІ++, ТУРБО-ПАСКАЛЬ, ТУРБО-СІ, АДА. ПРОЛОГ, ЯВА та ін. За допомогою мов програмування розробляють програ-

ми для ПК. ***Програма*** – сукупність команд, керуючих діями комп'ютера, записаних відповідно до прийнятого синтаксису.

Звичайно програма пишеться на символічній, близькій до англійської, мові і є в такому вигляді вихідним модулем. Однак для того, щоб цю мову зрозумів комп'ютер, необхідно перевести програму, написану символічною мовою, у сукупність машинних команд або в об'єктний модуль. Роль перекладача виконують транслятори.

**Транслятор** – перекладач з мови програмування на мову ЕОМ, тобто в команди, що складаються з кодів ЕОМ. Розрізняють основні види трансляторів: інтерпретатори і компілятори.

**Інтерпретатор** – транслятор, який забезпечує покомандний переклад у машинні коди й одночасне виконання кожної команди програми. Перевагою інтерпретатора є можливість організації роботи в режимі діалогу користувач – ПК. Недолік полягає в низькій швидкості виконання програми.

**Компілятор** – транслятор, який переводить усю програму в машинні команди без її виконання. У результаті роботи компілятора створюється окремий модуль, ще не готовий до виконання. Для виконання програми необхідно створити завантажувальний (виконуваний) модуль, який може включати кілька об'єктних модулів, необхідних для виконання програми.

***Прикладне програмне забезпечення*** – сукупність програм різного призначення для автоматизації обробки різного виду інформації і виконання розрахунків у науці і техніки.

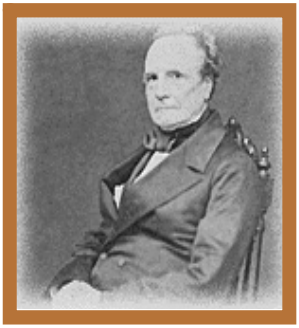
***Прикладні програми*** – програми, що розробляються, як правило, для конкретних користувачів або самими користувачами для поточних потреб. Використання прикладних програм іншими користувачами обмежено.

***Спеціалізовані пакети прикладних програм*** – програми для визначеного класу задач, конкретної предметної області, які призначені для забезпечення потреб великого числа користувачів. Пакети мають "дружній" інтерфейс і можуть експлуатуватися користувачами без участі програмістів. До подібних пакетів можна віднести: редактори (ЛЕКСИКОН, WORD), електронні таблиці (SuperCalc), бази даних (dBase), пакет проектування механізмів (AutoCad), пакет математичних обчислень (MatCad), довідкова законодавча система (ГАРАНТ), пакет допоміжних програм (NortonUtilite) та ін.

***Інтегровані прикладні системи*** – комплексні пакети прикладних програм, що володіють універсальними можливостями з обробки інформації і включають у себе: текстові редактори, текстові процесори, системи керуваннями базами даних, електронні таблиці і засоби ділової графіки. Характерними представниками подібних систем є: QuartoPro, Works, Російський офіс, Microsoft Office. Подібні системи підвищують продуктивність праці і полегшують роботу бізнесменам, юристам, що служать офісів і інших працівників сфери керування.

## 1.6.2 Загальні основи операційних систем

### 1.6.2.1 Призначення і функції ОС



Ідея комп'ютера була запропонована англійським математиком Чарльзом Беббіджем (**Charles Babbage**) у середині дев'ятнадцятого століття, але внаслідок недостатнього розвитку технологій того часу вона не могла бути реалізована.

Справжнє народження цифрових обчислювальних машин відбулося в середині 40-х рр. XX ст. Саме тоді були створені перші лампові обчислювальні пристрої. Вони були скоріше предметом науково-дослідної роботи, непридатним для вирішення яких-небудь практичних завдань із інших прикладних областей. Їхнім проектуванням, програмуванням і експлуатацією займалася та сама група людей. Програмування здійснювалося винятково машинною мовою, не було ніякого системного програмного забезпечення, крім бібліотек математичних і службових підпрограм, які програміст міг використовувати для того, щоб не писати щораз коди, що обчислюють значення якої-небудь математичної функції або керуючі стандартним пристроєм введення-виведення. ОС тоді ще не з'явилися, а всі завдання організації обчислювального процесу вирішувалися вручну кожним програмістом з пульта керування – примітивного пристрою введення-виведення, який складався із кнопок, перемикачів та індикаторів.У середині 50-х років розпочався наступний період у розвитку обчислювальної техніки, пов'язаний з новою технічною базою – напівпровідниковими елементами. З їхньою появою зменшилися розміри комп'ютерів, виросла швидкодія процесорів, збільшилися обсяги оперативної й зовнішньої пам'яті. Комп'ютери стали надійнішими, тепер вони могли безупинно працювати настільки довго, що на них можна було покласти виконання дійсно важливих практичних завдань. Поряд з удосконалюванням апаратури помітний прогрес спостерігався також в області автоматизації програмування й організації обчислювальних робіт. У ці роки з'явилися алгоритмічні мови, і в такий спосіб до бібліотек математичних і службових програм додався новий тип системного програмного забезпечення – **транслятори**. Але як би швидко й надійно не працювали оператори, які здійснювали керування обчислювальним процесом, вони вже

не могли змагатися в продуктивності з роботою пристроїв комп'ютера. Більшу частину часу комп'ютер простоював чекаючи, поки оператор запустить чергове завдання. А оскільки процесор являв собою досить дорогий пристрій, то низька ефективність його використання означала низьку ефективність використання комп'ютера в цілому, при цьому витрати на комп'ютерну обробку даних зростали. Таким чином, максимальне завантаження апаратних засобів стало однією з головних вимог використання комп'ютерів, а мірою ефективності їхнього використання стала пропускна здатність. Для вирішення цієї проблеми були розроблені перші **системи пакетної обробки**, які автоматизували всю послідовність дій оператора з організації обчислювального процесу. Ранні системи пакетної обробки стали прообразом сучасних ОС, вони стали першими системними програмами не для обробки даних користувача, а для керування обчислювальним процесом. У ході реалізації систем пакетної обробки була розроблена формалізована мова керування завданнями, за допомогою якої програміст повідомляв систему й оператора, які дії і в якій послідовності він хоче виконати на обчислювальній машині. Типовий набір директив звичайно включав ознаку початку окремої роботи, виклик транслятора, виклик завантажника, ознаки початку й кінця вихідних даних. Оператор становив пакет завдань, які надалі без його участі послідовно запускалися на виконання керуючою програмою – **монітором**. Крім того, монітор був здатний самостійно обробляти аварійні ситуації, які найчастіше зустрічаються при роботі користувальницьких програм, такі як відсутність вихідних даних, переповнення регістрів, ділення на нуль, звертання до неіснуючої області пам'яті тощо. Зазвичай пакет був набором перфокарт, але для прискорення роботи він міг переноситися на зручніший носій, наприклад на магнітну стрічку або магнітний диск. Сама програма-монітор у перших реалізаціях також зберігалася на перфокартах або перфострічці, а в пізніших – на магнітній стрічці й магнітних дисках. Ранні системи пакетної обробки значно скоротили витрати часу на допоміжні дії з організації обчислювального процесу, а виходить, був зроблений ще один крок для підвищення ефективності використання комп'ютерів, однак при цьому програмісти-користувачі втратили безпосередній доступ до комп'ютера.

1961 р. – у продаж надійшла перша інтегральна схема (ІС) на пластині кремнію.



У 1965-1975 рр. відбувся перехід від окремих напівпровідникових елементів до інтегральних мікросхем, що відкрило шлях до появи наступного типу комп'ютерів. Більші функціональні можливості інтегральних схем уможливили здійснення на практиці складних комп'ютерних архітектур,

таких, наприклад, як IBM/360. У цей період були реалізовані практично всі основні механізми, властиві сучасним ОС: мультипрограмування, мультипроцесорність, підтримка багатотермінального багатокористувальницького режиму, віртуальна пам'ять, файлові системи, розмежування доступу й мережна робота. У ці роки величезного значення набуває системне програмування, яке з напрямку прикладної математики перетворюється в галузь індустрії, що робить практичний вплив на діяльність мільйонів людей.

Революційною подією даного етапу з'явилася промислова реалізація мультипрограмування – способу організації обчислювального процесу, при якому в пам'яті комп'ютера знаходилося одночасно декілька програм, які поперемінно виконуються. Таке вдосконалення значно поліпшило ефективність обчислювальної системи – комп'ютер використовувався тепер постійно. До цього часу відбулася серйозна зміна в розподілі функцій між апаратними й програмними засобами комп'ютера. ОС ставали невід'ємними елементами комп'ютерів, відіграючи роль "продовження" апаратури. У перших обчислювальних машинах програміст, прямо взаємодіючи з апаратурою, міг здійснити завантаження програмних кодів, а потім вручну запустити програму на виконання. У комп'ютерах 60-х років більшу частину дій з організації обчислювального процесу взяла на себе операційна система. У більшості ж сучасних комп'ютерів взагалі не передбачено навіть теоретичної можливості виконання якої-небудь обчислювальної роботи без участі ОС. Після включення живлення автоматично проводиться пошук, завантаження й запуск ОС, а у випадку її відсутності комп'ютер просто припиняє свою роботу. Реалізація мультипрограмування зажадала внесення дуже важливих змін в апаратуру комп'ютера, спрямованих безпосередньо на підтримку нового способу організації обчислювального процесу. При поділі ресурсів комп'ютера між програмами виконувалися завдання із забезпечення швидкого перемикання процесора з однієї програми на іншу, а також надійного захисту кодів і даних однієї програми від ненавмисного або навмисного псування іншою програмою. У процесорах з'явилися привілейований і користувальницький режими роботи, спеціальні регістри для швидкого перемикання з однієї програми на іншу, засоби захисту областей пам'яті, а також розвинена система переривань. Апаратна підтримка ОС стала з тих пор невід'ємною властивістю практично будь-яких комп'ютерних систем, включаючи персональні комп'ютери.

Ще однією важливою тенденцією цього періоду є створення сімейств програмно-сумісних машин і ОС для них. Прикладами таких сімейств, побудованих на інтегральних мікросхемах, є серії машин IBM/360 (рис.1.55) і IBM/370, їхні радянські аналоги – машини серії ЄС, PDP-11 і комп'ютери вітчизняного виробництва СМ-3 (рис. 1.56), СМ-4 і СМ-1420.

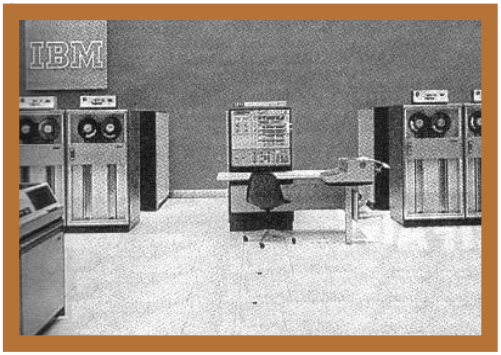



Рисунок 1.55.
1965 р. – розпочато випуск сімейства машин третього покоління IBM/360 (США)



Рисунок 1.56.
1970-і р. – розпочато випуск сімейства малих ЕОМ міжнародної системи (СМ ЕОМ). На фотографії ЕОМ СМ-3

Незабаром ідея програмно-сумісних машин стала загальновизнаною. Програмна сумісність вимагала й сумісності ОС, що припускає можливість роботи на великих і малих обчислювальних системах з різноманітною периферією в комерційних і науково-дослідних галузях. ОС, побудовані з наміром задовольнити всім цим украй суперечливим вимогам, виявилися надзвичайно складними. Вони складалися з мільйонів асемблерних рядків, написаних тисячами програмістів, і містили сотні тисяч помилок, що викликали нескінченний потік виправлень. ОС цього покоління були дуже дорогими – наприклад, розробка OS/360, обсяг коду якої становив усього (за сучасними мірками) 8 Мбайт, коштувала компанії IBM 80 мільйонів доларів. Варто помітити, що, незважаючи на безліч проблем і зауважень, OS/360 і інші, подібні їй, ОС задовольняли більшості вимог споживачів. За це десятиліття був зроблений величезний крок уперед і закладений міцний фундамент для створення сучасних ОС.

На початку 70-х років з'явилися перші мережні ОС, які на відміну від багатотермінальних ОС дозволяли не тільки розосередити користувачів, але й організувати розподілене зберігання й обробку даних між декількома комп'ютерами, зв'язаними електричними мере-

жами. Програмні модулі, що реалізують мережні функції, з'являлися в ОС поступово, у міру розвитку мережних технологій, апаратної бази комп'ютерів і виникнення нових завдань, що вимагають мережної обробки даних. І хоча теоретичні роботи із створення концепцій мережної міжкомп'ютерної взаємодії велися майже із самої появи обчислювальних машин, значущі практичні результати з об'єднання комп'ютерів у мережу були отримані наприкінці 60-х, коли за допомогою глобальних зв'язків і техніки комутації пакетів удалося реалізувати взаємодію машин класу мейнфреймів (мейнфрейм – універсальна ЕОМ великої потужності, зазвичай використовувана одночасно декількома користувачами, які працюють на терміналах, підключених до неї і суперкомп'ютерів. Ці дорогі комп'ютери часто зберігали унікальні дані й програми, доступ до яких необхідно було забезпечити широкому колу користувачів, що перебували в різних містах на значному видаленні від обчислювальних центрів. У 1969 році Міністерство оборони США ініціювало роботи з об'єднання суперкомп'ютерів оборонних і науково-дослідних центрів у єдину мережу ARPANET, у яку ввійшли комп'ютери різних типів, що працювали під керуванням різних ОС із доданими модулями, що реалізують комунікаційні протоколи, загальні для всіх комп'ютерів у мережі. Ця мережа з'явилася відправною точкою для створення найвідомішої тепер глобальної мережі – Інтернет (Internet).

До середини 70-х років, поряд мейнфреймами, широкого поширення набули такі міні-комп'ютери, як PDP-11, NOVA, HP. Міні-комп'ютери першими використовували переваги великих інтегральних схем, що дозволили реалізувати досить потужні функціональні можливості при порівняно невисокій вартості. Архітектура міні-комп'ютерів була значно спрощена порівняно з мейнфреймами, що, безсумнівно, знайшло своє відбиття і в їхніх ОС. Багато функцій багатотермінальних багатокористувальницьких ОС мейнфреймів були обмежено, з огляду на обмеженість ресурсів міні-комп'ютерів. ОС міні-комп'ютерів часто стали робити спеціалізованими, наприклад тільки для керування в реальному часі (ОС RT-11 для міні-комп'ютерів PDP-11) або тільки для підтримки режиму поділу часу (RSX-11M для тих же комп'ютерів). Ці ОС не завжди були багатокористувальницькими, що в багатьох випадках виправдувалося досить низькою вартістю міні-комп'ютерів порівняно з мейнфреймами.

Важливою подією в історії міні-комп'ютерів і взагалі в історії ОС з'явилося створення ОС UNIX. Спочатку вона призначалася для підтримки режиму поділу часу в міні-комп'ютері PDP-11. Із середини 70-х років почалося масове використання ОС UNIX. До цього часу програмний код для UNIX був на 90% написаний мовою високого рівня Сі, а широке поширення ефективних Сі-Компіляторів згодом зробило UNIX унікальною у своєму роді ОС, яка володіє можливістю порівняно легкого перенесення

на різні типи комп'ютерів. Оскільки ця ОС поставлялася разом з вихідними текстами, вона стала першою відкритою ОС, яку могли вдосконалювати або змінювати за своїм розсудом звичайні користувачі-програмісти. Хоча UNIX була спочатку розроблена для міні-комп'ютерів, гнучкість, елегантність, потужні функціональні можливості й відкритість дозволили їй зайняти гідні позиції у всіх класах комп'ютерів: суперкомп'ютерах, мейнфреймах, міні-комп'ютерах, серверах і робочих станціях, персональних комп'ютерах.

У 80-х роках був розроблений протокол TCP/IP, визнаний у 1983 р. Міністерством оборони США військовим стандартом і реалізований для ОС BSD UNIX. Все це десятиліття відзначалося постійною появою нових, усе більш довершених версій ОС UNIX. Серед них були й фірмові версії UNIX: SunOS, HP-UX, Irix, AIX, і багато інших до яких виробники комп'ютерів адаптували код ядра й системних утиліт для своєї апаратури. Розмаїття версій породило проблему їхньої сумісності, періодично розв'язувану різними організаціями. У результаті цієї роботи з'явилися стандарти POSIX і XPG, що визначають інтерфейси ОС для додатків, а спеціальний підрозділ компанії AT&T випустив кілька версій UNIX System III і UNIX System V, призначених для консолідації розроблювачів на рівні коду ядра.

Початок 80-х років також ознаменувався важливою подією в історії ОС – з'явилися персональні комп'ютери. Вони стали широко використовуватися неспеціалістами, що зажадало розробки так званого дружнього програмного забезпечення. Надання цих дружніх функцій стало, природно, прямим обов'язком ОС. Персональні комп'ютери внаслідок низької вартості одержали широке поширення й послужили причиною бурхливого розвитку локальних обчислювальних мереж. У результаті підтримка мережних функцій стала для ОС персональних комп'ютерів необхідною умовою. Однак, незважаючи на це, і дружній інтерфейс, і мережні функції з'явилися в персональних комп'ютерів далеко не відразу. Перша версія найбільш популярної ОС раннього етапу розвитку персональних комп'ютерів – MS-DOS компанії Microsoft – була позбавлена цих функціональних можливостей. Вона розроблялася як однопрограмна однокористувальницька ОС із **інтерфейсом** командного рядка, здатна стартувати з дискети. Основними завданнями для неї були керування файлами, розташованими на гнучких і жорстких дисках, а також почерговий запуск програм. Однак розроблювачі перших персональних комп'ютерів вважали, що при індивідуальному користуванні нема рації в мультипрограмуванні, тому не були передбачені механізми підтримки мультипрограмних систем, і процесор Intel 8088 не підтримував привілейованого режиму. Мережні функції також реалізовувалися в основному мережними оболонками, що працювали поверх ОС.

У 1987 році в результаті спільних зусиль Microsoft і IBM з'явилася на світ перша багатозадачна ОС для персональних комп'ютерів OS/2 із процесором Intel 80286, яка повною мірою використовує можливості захищеного режиму. Ця система була добре продумана й підтримувала витісняючу багатозадачність, віртуальну пам'ять, графічний користувальницький інтерфейс (правда, не з першої версії), а також віртуальну машину для виконання DOS-додатків. У персональних комп'ютерах застосовувалися не тільки спеціально розроблені ОС, подібні MS-DOS, NetWare і OS/2, але до них адаптувалися й уже існуючі ОС. Поява процесорів Intel 80286 і 80386 з підтримкою мультипрограмування дозволило перенести на платформу персональних комп'ютерів ОС UNIX. Найвідомішою системою цього типу стала версія UNIX компанії Santa Cruz Operation (SCO UNIX).В 90-і роки практично всі ОС, які займають помітне місце на ринку, стали мережними. Мережні функції сьогодні є невід'ємною частиною ОС і вбудовуються в її ядро. ОС отримали засоби для роботи з усіма основними технологіями локальних (Ethernet, Fast Ethernet, Gigabit Ethernet, Token Ring, FDDI, ATM) і глобальних (X.25, frame relay, ISDN, ATM) мереж, а також засобу для створення складених мереж (IP, IPX, AppleTalk, RIP, OSPF, NLSP). В ОС використовується мультиплексування декількох стеків протоколів, за рахунок якого комп'ютери можуть підтримувати одночасну мережну роботу з різними серверами й клієнтами. З'явилися спеціалізовані ОС, призначені винятково для виконання комунікаційних завдань. Наприклад, мережна ОС IOS компанії Cisco Systems, що працює в маршрутизаторах, організує в мультипрограмному режимі виконання набору програм, кожна з яких реалізує один з комунікаційних протоколів.

У другій половині 90-х років всі компанії-виробники ОС різко підсилили підтримку своїми системами засобів роботи з Інтернетом. Крім самого стеку протоколів TCP/IP, у комплект поставки почали включати утиліти, які реалізують такі популярні сервіси, як telnet, ftp, DNS і Web. Вплив Інтернету виявився і в тому, що комп'ютер перетворився із чисто обчислювального пристрою в засіб комунікації з розвиненими обчислювальними можливостями.

На сучасному етапі стратегічним напрямком еволюції ОС є створення багатофункціональної масштабованої довідкової служби. Від успіхів у цьому напрямку багато в чому залежить подальший розвиток Інтернету. Така служба необхідна для перетворення Інтернету в передбачувану й керовану систему, наприклад, для забезпечення необхідної якості обслуговування трафіку користувачів, підтримки великих розподілених додатків, побудови ефективної поштової системи тощо.Немаловажними для сучасних ОС постали й питання безпеки. Це пов'язано із зростаючою цінністю інформації, оброблюваної комп'ютерами, а також з підвищеним рівнем загроз, що існують при передачі даних по мережах. Багато ОС мають сьогодні

розвинені засоби захисту, засновані на шифруванні даних, аутентифікації та авторизації.Сучасним ОС властива багатоплатформенність, тобто здатність працювати на зовсім різних типах комп'ютерів. Багато ОС мають спеціальні версії для підтримки кластерних архітектур, які забезпечують високу продуктивність і відмовостійкість.

В останні роки одержала подальший розвиток тенденція збільшення зручності взаємодії людини з комп'ютером. Ефективність роботи користувача стає основним фактором, що визначає результативність обчислювальної системи в цілому. Людина повинна робити якнайменше зусиль на налаштування параметрів обчислювального процесу. Наприклад, у системах пакетної обробки для мейнфреймів кожний користувач за допомогою мови керування завданнями визначав велику кількість параметрів, що ставляться до організації обчислювальних процесів у комп'ютері. Так, для системи OS/360 таких параметрів налічувалося більше 40, серед яких були наступні: пріоритет завдання, вимоги до основної пам'яті, граничний час виконання завдання, перелік використовуваних пристроїв уведення-виводу й режими їхньої роботи.Сучасна ОС бере на себе виконання завдання вибору параметрів операційного середовища, використовуючи для цієї мети різні адаптивні алгоритми. Наприклад, тайм-аути в комунікаційних протоколах часто визначаються залежно від умов роботи мережі. Розподіл оперативної пам'яті між процесами відбувається автоматично за допомогою механізмів віртуальної пам'яті залежно від активності цих процесів і інформації про частоту використання тієї чи іншої сторінки. Миттєві пріоритети процесів визначаються динамічно залежно від передісторії, яка включає, наприклад, час знаходження процесу в черзі, відсоток використання виділеного кванта часу, інтенсивність введення-виведення тощо. Навіть у процесі встановлення більшість ОС пропонують режим вибору параметрів за замовчуванням, що гарантує нехай і не оптимальну, але завжди прийнятну якість роботи системи.

Включення в сучасні ОС розвинених графічних інтерфейсів, що використовують поряд із графікою звук і відеозображення, забезпечує підвищення зручності інтерактивної роботи з комп'ютером. Особливого значення це набуває при використанні комп'ютера як терміналу Інтернету. Користувальницький інтерфейс ОС стає усе більше інтелектуальним, направляючи дії людини в типових ситуаціях і приймаючи за нього рутинні рішення. Рівень зручностей у використанні ресурсів, які сьогодні надають користувачам, адміністраторам і розроблювачам додатків ОС ізольованих комп'ютерів, для мережних ОС є тільки привабливою перспективою. Поки користувачі й адміністратори мережі витрачають значну кількість часу на спроби з'ясувати, де перебуває той або інший ресурс, розроблювачі мережних додатків додають безліч зусиль для визначення місця розташування даних і програмних модулів у мережі. ОС майбутнього повинні забезпечити

високий рівень прозорості мережних ресурсів, взявши на себе завдання організації розподілених обчислень, перетворивши в такий спосіб мережа у віртуальний комп'ютер.

Розглянувши коротку історію розвитку операційних систем, ми можемо більш докладно зупинитися на обговоренні виконуваних ними функцій.

**Операційна система** (ОС) комп'ютера є комплексом програм, які організують обчислювальний процес в обчислювальній системі.

Сьогодні існує велика кількість різних ОС, які розрізняються за типами, способами реалізації, специфікою розв'язуваних завдань, використовуваними апаратними засобами і функціональними можливостями. Тому при вивченні ОС при всьому їхньому різноманітті важливо виділити ту частину, яка властива їм усім як класу продуктів. Це насамперед функції, багато з яких увійшли до складу ОС із самої їхньої появи, інші виникли відносно недавно й продовжують оновлюватися внаслідок постійного вдосконалювання апаратної платформи і змін у концепції взаємодії людини і комп'ютера.

ОС виконує дві групи функцій:

- ✧ надання користувачеві замість реальної апаратури комп'ютера якоїсь розширеної машини, з якою зручніше працювати і яку легше програмувати;
- ✧ підвищення ефективності використання комп'ютера шляхом раціонального керування його ресурсами відповідно до деякого критерію.

Для того щоб успішно вирішувати свої завдання, сучасний користувач або прикладний програміст може обійтися без досконального знання апаратного устрою комп'ютера, принципів функціонування його електронних блоків і електромеханічних пристроїв. Наприклад, при роботі з диском користувачеві досить представляти його у вигляді деякого набору файлів з певними іменами. Файл, готовий до виконання ОС, називається ***виконуючим файлом***. Такі технічні деталі, як використовувана при запису інформації на диск частотна модуляція або положення магнітної головки читання/запису, не повинні хвилювати користувача. В ідеальному випадку його взагалі не повинні стосуватися принципи й особливості роботи пристроїв, які становлять комп'ютер.

Саме ОС приховує від нього більшу частину особливостей функціонування апаратури, беручи на себе рутинні операції, пов'язані з керуванням апаратними пристроями, і надаючи можливість простої і зручної роботе.

ОС не тільки надає користувачам і програмістам зручний інтерфейс до апаратних засобів комп'ютера, але і є механізмом, що розподіляє ресурси комп'ютера. До **ресурсів** обчислювальної системи відносять такі її засоби, які можуть бути виділені процесу обробки даних.

Ресурси обчислювальної системи можна розбити на **первинні** – апаратні ресурси й **вторинні** – логічні, програмні й інформаційні ресурси.

До числа первинних ресурсів сучасних обчислювальних систем відносяться процесори, основна пам'ять, диски та ін., за якими стоять реальні апаратні засоби. Вони є найбільш значущими для обчислювального процесу.

Вторинні ресурси пов'язані з технічними пристроями побічно, тому що є логічними, віртуальними. Однак їхнє введення – це необхідна абстракція, зручна не тільки для творців ОС, але й для користувачів.

Керування ресурсами обчислювальної системи з метою їх найефективнішого використання становить важливу частину функцій будь-якої ОС. Критерії ефективності, відповідно до яких ОС організує керування ресурсами, можуть розрізнятися. Наприклад, в одних системах важлива пропускна здатність обчислювальної системи, а в інших – час її реакції. Часто ОС повинні одночасно задовольняти декільком суперечливим критеріям, що створює для розроблювачів серйозні проблеми.

Керування ресурсами включає вирішення наступних загальних, що не залежать від типу ресурсу завдань:

- ***планування ресурсу*** – визначення, якому процесу, коли й у якій кількості (якщо ресурс може бути розбитий на частини) варто виділити даний ресурс;
- задоволення запитів на ресурси;
- відстеження стану й ***облік використання ресурсу*** – підтримка оперативної інформації про зайнятість ресурсу й розподіленої його частки;
- вирішення конфліктів між процесами, які використовують один ресурс.

Для рішення цих загальних завдань керування ресурсами різні ОС використовують алгоритми, особливості яких в остаточному підсумку й визначають зовнішній вигляд ОС у цілому, включаючи характеристики їхньої продуктивності, галузь застосування й користувальницький інтерфейс. Розробка й реалізація алгоритмів керування ресурсами є дуже важливим етапом при проектуванні системи.

Організація ефективного спільного використання ресурсів декількома процесами – це досить складне завдання, обумовлене, в основному, випадковим характером виникнення запитів на споживання ресурсів. Аналіз і визначення оптимальних параметрів обслуговування запитів є предметом спеціальної області прикладної математики – теорії масового обслуговування. У рамках цієї теорії розроблена безліч методів, що дозволяють планувати розподіл критично важливих ресурсів з обліком усіляких стохастичних і детермінованих подій.

Функції ОС звичайно групуються або відповідно до типів локальних ресурсів, якими управляє ОС, або відповідно до специфічних завдань, застосовними до всіх ресурсів обчислювальної системи. Такі групи називаються **підсистемами**. Найважливіші – це підсистеми керування процесами, пам'яттю, файлами й зовнішніми пристроями, а підсистемами, загальними для всіх ресурсів, є підсистеми користувальницького інтерфейсу, захисту даних і адміністрування. Розглянемо покладені на них основні завдання.

Підсистема керування процесами генерує системні інформаційні структури, які містять дані про потреби в ресурсах обчислювальної системи, а також про фактично виділені ресурси для кожного завдання. **Процес** (завдання) – являє собою базове поняття сучасних ОС і часто коротко визначається як програма в стадії виконання. **Програма** – це статичний об'єкт, який представляє собою файл із кодами й даними. Процес є динамічним об'єктом, який виникає в операційній системі після того, як користувач або ОС вирішує запустити програму на виконання. У багатьох сучасних ОС для позначення мінімальної одиниці роботи ОС використовують термін "нитка, або "потік", при цьому змінюється суть терміну "процес". Щоб процес виконувався, ОС повинна призначити йому область оперативної пам'яті, у якій будуть розміщені коди і дані процесу, а також надати необхідну кількість процесорного часу й доступ до необхідних ресурсів інших типів. В інформаційні структури процесу часто включається додаткова інформація, яка характеризує історію перебування процесу в системі, його поточний стан, ступінь привілейованості процесу.

У мультипрограмній ОС одночасно можуть існувати кілька процесів, частина з яких, названа **користувальницькими процесами**, породжується з ініціативи користувачів. Інші процеси ініціюються самою ОС для виконання своїх функцій і називаються **системними**.

Оскільки процеси досить часто одночасно претендують на ті самі ресурси обчислювальної системи, на ОС лягає завдання підтримки черг заявок на такі ресурси.

Не менш важливим завданням ОС при керуванні процесами є захист виділених даному процесу ресурсів від втручання інших процесів. Найретельнішого захисту вимагають області оперативної пам'яті, які зберігають коди й дані процесу. Однак захист ресурсів процесу зовсім не має на увазі, що ОС забороняє їхнє спільне використання декількома процесами одночасно.

ОС бере на себе також функції синхронізації процесів, які дозволяють процесу припиняти своє виконання до настання якої-небудь події в системі, яким може бути, наприклад, завершення операції введення-виведення, здійснюваної ОС за його запитом.

У сучасних ОС не існує однозначної відповідності між процесами й програмами. Той самий процес у ході свого виконання може перемінити

програмний файл. Для реалізації складних програмних комплексів часто буває необхідно організувати їхню роботу у вигляді декількох процесів, що виконуються паралельно, які час від часу взаємодіють один з одним і обмінюються деякими даними. Оскільки ОС захищає ресурси процесів від взаємного втручання й не дозволяє одному процесу зчитувати й записувати дані в область пам'яті іншого процесу, то для можливості їхньої оперативної взаємодії ОС повинна мати у своєму складі особливі засоби, які називають **засобами міжпроцесної взаємодії**.

Таким чином, підсистема керування процесами планує виконання процесів, тобто розподіляє процесорний час між декількома одночасно існуючими в системі процесами, займається створенням і знищенням процесів, забезпечує процеси необхідними їм системними ресурсами, підтримує синхронізацію процесів, а також забезпечує взаємодію між процесами.

На **підсистему керування пам'яттю** покладені дуже важливі функції, оскільки процес може виконуватися тільки в тому випадку, якщо його коди й дані перебувають в оперативній пам'яті комп'ютера. Керування пам'яттю містить у собі розподіл наявної в обчислювальній системі фізичної пам'яті між всіма існуючими в цей момент у системі процесами, завантаження кодів і даних процесів у відведені їм області пам'яті, настроювання адресно-залежних частин кодів процесу на фізичні адреси виділеної області, а також захист областей пам'яті кожного процесу. Існує досить велика розмаїтість алгоритмів розподілу пам'яті, які можуть відрізнятися кількістю виділюваних процесу областей пам'яті, ступенем волі меж цих областей та інших параметрів.

Одним з найпопулярніших способів керування пам'яттю в сучасних ОС є так звана **віртуальна пам'ять**. Наявність в ОС механізму віртуальної пам'яті дозволяє програмістові писати програму так, начебто в його розпорядженні є однорідна оперативна пам'ять великого обсягу, що часто істотно перевищує обсяг наявної фізичної пам'яті. У дійсності всі дані, використовувані програмою, зберігаються на диску і при необхідності частинами відображаються у фізичній пам'яті. При переміщенні кодів і даних між оперативною пам'яттю й диском підсистема віртуальної пам'яті здійснює трансляцію віртуальних адрес, отриманих у результаті компіляції й компонування програми, у фізичні адреси чарунок оперативної пам'яті. Дуже важливо, що всі операції з переміщення кодів і даних між оперативною пам'яттю й дисками, а також трансляція адрес виконуються ОС зовсім прозоро для програміста.

**Захист пам'яті** – це вибірна здатність охороняти виконуване завдання від операцій запису або читання пам'яті, призначеному іншому завданню. Правильно написані програми не роблять некоректних спроб звернутися до пам'яті, використовуваної іншими програмами. Однак внаслідок то-

го, що в реальних програмах можуть бути помилки, такі спроби можуть виникати. Засоби захисту пам'яті, реалізовані в ОС, повинні припиняти несанкціонований доступ процесів до чужих областей пам'яті.

Таким чином, функціями ОС з керування пам'яттю є: відстеження вільної й зайнятої пам'яті, виділення пам'яті процесам і звільнення пам'яті після їхнього завершення, захист пам'яті, а також настроювання адрес програм на конкретну область фізичної пам'яті.

Здатність ОС до "екранування" складностей реальної апаратури добре виражено в одній з основних підсистем ОС – **файловій системі**. ОС віртуалізує окремий набір даних, що зберігаються на зовнішньому накопичувачі, у вигляді файлу – простої неструктурованої послідовності байтів, яка має символьне ім'я. Для зручності роботи з даними файли групуються в каталоги, які, у свою чергу, утворять групи – каталоги вищого рівня. Користувач, застосовуючи засоби ОС, може виконувати над файлами і каталогами такі дії, як пошук за іменем, видалення, виведення вмісту на зовнішній пристрій, зміну й збереження вмісту. Для того щоб велика кількість наборів даних, розміщених випадковим чином на циліндрах і поверхнях дисків різних типів, представити у вигляді ієрархічній структурі файлів і каталогів, ОС повинна вирішити безліч завдань. Файлова система ОС виконує перетворення символьних імен файлів, з якими працює користувач або прикладний програміст, у фізичні адреси даних на диску, організує спільний доступ до файлів, захищає їх від несанкціонованого доступу. При виконанні своїх функцій файлова система тісно взаємодіє з підсистемою керування зовнішніми пристроями, що по запитах файлової системи здійснює передачу даних між дисками й оперативною пам'яттю.

**Підсистема керування зовнішніми пристроями**, названа також підсистемою введення-виведення, є інтерфейсом до всіх пристроїв, підключеним до комп'ютера. Множина цих пристроїв дуже велика. Номенклатура накопичувачів, що випускаються, на твердих, гнучких і оптичних дисках, принтерів, сканерів, моніторів, плотерів, модемів, мережних адаптерів і спеціалізованих пристроїв введення-виведення (наприклад, аналого-цифрових перетворювачів, пристроїв зчитування шрих-коду тощо) нараховує тисячі моделей. Ці моделі відрізняються набором і послідовністю команд, використовуваних для обміну інформацією із процесором і пам'яттю комп'ютера, швидкістю роботи, кодуванням переданих даних, можливістю спільного використання, підтримуваними функціональними можливостями й безліччю інших деталей. Програма, яка управляє конкретною моделлю зовнішнього пристрою й враховує всі його особливості, називається **драйвером** цього **пристрою** (від англ. drive – управляти). Драйвер може управляти єдиною моделлю пристрою або ж групою пристроїв певного типу. Для кінцевого користувача важливе значення має наявність в ОС якомога більшої кількості різноманітних драйверів

(для широкого спектру пристроїв), оскільки це гарантує можливість підключення до комп'ютера великої кількості зовнішніх пристроїв різних виробників. Від наявності підходящих драйверів багато в чому залежить успіх ОС на ринку. Так, відсутність багатьох необхідних драйверів зовнішніх пристроїв було однією із причин низької популярності ОС OS/2 і ранніх версій Windows NT. Тому, перш ніж випускати на ринок нову ОС, розроблювачі намагаються включити в її склад драйвери для підтримки самого різного наявного в продажі обладнання. Створенням драйверів займаються як розроблювачі конкретної ОС, так і фахівці компаній, що випускають зовнішні пристрої. ОС повинна підтримувати чітко певний інтерфейс між драйверами та іншими частинами ОС, щоб розроблювачі з компаній-виробників пристроїв введення-виведення могли поставляти разом із своїми пристроями драйвери для даної ОС. Прикладні програмісти також можуть користуватися інтерфейсом драйверів при розробці своїх програм, однак це дуже незручно, оскільки такий інтерфейс звичайно є низькорівневими операціями, наділеними великою кількістю деталей.

Підтримка високорівневого уніфікованого інтерфейсу прикладного програмування до різнорідних пристроїв введення-виведення є одним з найважливіших завдань ОС. Із часу появи ОС UNIX такий уніфікований інтерфейс у більшості ОС будується на основі концепції файлового доступу. Її зміст полягає в тому, що обмін даними з будь-яким зовнішнім пристроєм виглядає як обмін з файлом, що має ім'я й представляє собою неструктуровану послідовність байтів. Як файл може виступати як реальний файл на диску, так і алфавітно-цифровий термінал, друкувальний пристрій або мережний адаптер. Таким чином, повною мірою реалізується властивість ОС підмінювати реальну апаратуру зручними для користувача й програміста абстракціями.

Безпека даних обчислювальної системи забезпечується засобами відмовостійкості ОС, спрямованими на захист від збоїв і відмов апаратури і помилок програмного забезпечення, а також засобами захисту від несанкціонованого доступу. В останньому випадку ОС захищає дані від помилкового або навмисного поводження користувачів системи.

ОС повинна надавати зручний інтерфейс не тільки для прикладних програм, але й для людини, яка працює з обчислювальною системою. Ця людина може бути як звичайним користувачем, так і адміністратором ОС, чи програмістом. У ранніх ОС пакетного режиму функції користувальницького інтерфейсу були зведені до мінімуму й не вимагали наявності терміналу. Команди набивалися на перфокарти, а результати обробки виводилися на друкувальний пристрій. Сучасні ОС підтримують розвинені функції користувальницького інтерфейсу для інтерактивної роботи за терміналами двох типів: алфавітно-цифровим і графічним. При роботі за алфавітно-цифровим терміналом користувач має у своєму розпорядженні

систему команд, потужність якої відображає функціональні можливості даної ОС. Елементарна пропозиція машинної мови, яка містить припис ЕОМ виконати деяку машинну операцію або дію, називається **командою ОС**.Звичайно командна мова ОС дозволяє запускати й зупиняти додатки, виконувати різні операції з файлами й каталогами, одержувати інформацію про стан ОС, адмініструвати систему. Команди можуть не тільки вводитися в інтерактивному режимі, але й зчитуватися з так званого **командного файлу**, що містить деяку послідовність команд. Програмний модуль ОС, відповідальний за читання окремих команд або послідовності команд із командного файлу, називається **командним інтерпретатором**. Інтерфейс у вигляді **командного рядка**, незважаючи на його гадану незручність, дотепер широко використовується, особливо в спеціалізованих ОС. Введення команди може бути спрощеним, якщо ОС підтримує **графічний користувальницький інтерфейс** (GUI) – інтерфейс користувача, заснований на засобах машинної графіки. У цьому випадку користувач для виконання потрібної необхідної йому дії за допомогою миші або іншого пристрою вказівки вибирає на екрані потрібний йому пункт меню.

Таким чином, призначення ОС полягає в наданні користувачеві деякої розширеної машини, яку легше програмувати і з якої зручніше працювати, ніж безпосередньо з апаратурою, що становить реальний комп'ютер, а також у керуванні ресурсами системи з метою їх найефективнішого використання.

### 1.6.2.2 Класифікація ОС

Розглянувши призначення ОС і виконувані ними функції, ми можемо класифікувати все різноманіття існуючих ОС, взявши за основу найзагальніші класифікаційні принципи:

а) *За кількістю одночасно існуючих програмних процесів* ОС діляться на ***однопрограмні*** й ***мультипрограмні***. У мультипрограмних ОС, на відміну від однопрограмних, обчислювальний процес організується таким чином, що в пам'яті комп'ютера перебуває одночасно кілька програм, які поперемінно виконуються на одному процесорі. Така організація дозволяє значно підвищити ефективність обчислювальної системи. Іноді говорять також про ***багатозадачаний режим***, однак, ***багатозадачний*** і мультипрограмний режими відрізняються. ***Мультипрограмний*** режим забезпечує паралельне виконання декількох додатків і при цьому програмісти, які їх створюють, не повинні піклуватися про механізми організації їхньої паралельної роботи. Ці функції бере на себе сама ОС, саме вона розподіляє між додатками, що виконуються, ресурси обчислювальної системи, здійснює необхідну

синхронізацію й взаємодію. ***Багатозадачаний*** режим, навпроти, припускає, що турбота про паралельне виконання й взаємодію додатків лягає саме на прикладних програмістів.

b) *За числом користувачів, які здійснюють доступ до обчислювальної системи*, розрізняють ***однокористувальницькі*** й ***багатокористувальницькі*** ОС. Багатокористувальницькі системи надають можливість одночасного доступу до обчислювальної системи декільком користувачам. При цьому кожний з них працює за власним терміналом, однак всі обчислення здійснюються на одному комп'ютері. Це приводить до більше ефективного використання обчислювальної техніки й зменшенню вартості обробки даних.

c) *За призначенням* ОС діляться на ***універсальні*** й ***спеціалізовані***. Спеціалізовані ОС, як правило, працюють із фіксованим набором програм. Застосування таких систем обумовлено неможливістю використання універсальної ОС з міркувань ефективності, а також внаслідок специфіки розв'язуваних завдань.

d) *За способом завантаження* можна виділити ОС, які ***завантажують*** і ***системи, що постійно перебувають у пам'яті*** обчислювальної системи. Останні, як правило, використовують для керування роботою спеціалізованих пристроїв.

e) *За особливістю галузі використання* ОС підрозділяють на ***системи пакетної обробки***, ***системи поділу часу*** і ***системи реального часу***. Системи пакетної обробки призначаються, в основному, для вирішення завдань обчислювального характеру, які не потребують швидкого одержання результатів. Критерієм ефективності таких систем є максимальна пропускна здатність, тобто вирішення максимального числа завдань в одиницю часу. Взаємодія користувача з обчислювальною системою зводиться до того, що він приносить завдання, віддає його операторові, а через якийсь час, після виконання пакета завдань, одержує результат.

Системи поділу часу організують обчислювальний процес таким чином, що кожному завданню виділяється квант процесорного часу, внаслідок чого жодне завдання не займає процесор надовго, і це дає можливість користувачеві вести діалог із своєю програмою. Критерієм ефективності систем поділу часу є зручність і ефективність роботи користувачів.

Системи реального часу застосовуються для керування різними технічними об'єктами або технологічними процесами. Критерієм ефективності для цих систем є здатність ОС витримувати заздалегідь задані інтервали часу між запуском програми й одержанням результату. Цей час називається **часом реакції**, а відповідна властивість системи – **реактивністю**.

Наведена класифікація носить досить загальний характер. Оскільки багато систем можуть сполучати в собі властивості систем різних типів, іноді буває важко віднести конкретну систему до того або іншого класу.

### 1.6.2.3 Операційні системи Windows

Компанія Microsoft почала роботу над операційною системою з графічним призначеним для користувача інтерфейсом у 1981 р. Перша версія Windows, випущена в листопаді 1985 року, мала набагато менші можливості, ніж Mac OS, і аж до випуску в 1990-му Windows 3.0 ця операційна система практично ніким серйозно не сприймалася.

**Windows 3.0 (1990 р.)**. – популярність нової версії Windows пояснювалася декількома причинами (графічний інтерфейс, багатозадачність, робота із зовнішніми пристроями різних виробників).

Windows NT (1992 р.) – ця версія Windows істотно підвищила надійність роботи системи і ефективність підтримки мережевої роботи.

**Windows 95 (1995 р.)** – значно змінився інтерфейс, виросла швидкість роботи програм. Однією з нових можливостей Windows 95 була можливість автоматичного налаштування додаткового устаткування комп'ютера для роботи без конфліктів один з одним. Іншою важливою особливістю системи стала можливість роботи з Інтернетом без використання додаткових програм.

**Windows 98 (1998 р.) –** відрізняється від Windows 95 тим, що в ній операційна система об'єднана з браузером Internet Explorer за допомогою інтерфейсу, виконаного у вигляді Web-браузера й оснащеного кнопками "Назад" і "Вперед" для переходу на попередню і наступну Web-сторінки. Поліпшена сумісність з новими апаратними засобами комп'ютера, вона однаково зручна як для використання на настільних, так і на портативних комп'ютерах.

**Windows 2000 (2000 р.)** – система Windows 2000 розроблена на основі Windows NT і успадкувала від неї високу надійність і захищеність інформації від стороннього втручання.

**Windows Me (2000 р.)** – операційна система Windows Me (Millennium Edition – редакція тисячоліття) стала спадкоємицею Windows 98, але придбала багато нових можливостей. Передусім, це поліпшена робота із засобами мультимедіа, можливість записувати не лише аудіо, але і відеоінформацію, потужні засоби відновлення інформації після збоїв і багато що інше.

**Windows XP (2002 р.)** – операційна система Microsoft Windows XP, створена на основі технології NT і є прямою спадкоємицею системи Windows 2000. В той же час, усі кращі нововведення, включені в Windows Me, можна виявити і в Windows XP.

***Система поставляється в декількох варіантах, орієнтованих на різні особливості застосування.***

Версія Microsoft Windows XP **Home Edition** призначена для індивідуальних користувачів, які найчастіше працюють на домашньому комп'ютері. У цій версії наголос зроблено на роботу з рисунками, аудіо і відео. Версія Microsoft Windows XP **Professional** призначена для професіоналів. Цю версію найчастіше використовують в організаціях для роботи в мережах.

**Windows Vista (2006 р.)** Версії ОС Vista мають звичайне умовне ділення на два класи, які усім знайомі по роботі з Windows XP. Перша категорія – "домашня", вона включатиме такі версії як: Starter, Home і Media Center Editions, друга категорія це "Pro", з такими версіями як Professional, Professional x64, і версія для Tablet PC. При цьому термінологія в Windows Vista трохи змінилася, категорії називаються Home і Business.

Microsoft випустила декілька "домашніх" версій: Windows Starter, Windows Vista Home Basic (і спеціальну європейську версію Home Basic N), Windows Vista Home Premium, і якнайповнішу Windows Vista Ultimate (раніше відома як "Uber").

Категорія Business буде представлена наступними версіями: Windows Vista Business (раніше відома як Professional Standard Edition; а так само вийде європейська версія Business N), і нарешті, Windows Vista Enterprise (раніше Professional Premium Edition). Кожна версія орієнтована для певного круга користувачів.

***Можливості операційної системи Windows Vista.***

- ✧ у версії Windows Vista сильно змінився зовнішній вигляд (інтерфейс) системи (технологія Aero Glass);
- ✧ ОС стала істотно стабільнішою і захищенішою порівняно з попередніми версіями Windows;
- ✧ гаджети – це невеликі графічні застосування, які встановлюються на робочий стіл. З їх допомогою, наприклад, можна отримати деяку інформацію з Інтернету і красиво відобразити її на робочому столі;
- ✧ перероблений механізм створення резервних копій системи за допомогою "Відновлення системи";
- ✧ vista пропонує вбудовану backup-програму. З її допомогою користувач може вручну або автоматично (за розкладом) робити копії і відновлення певних файлів, тек або цілих розділів. Зберегти резервні копії можна на CD-ROM, DVD-ROM, жорстких дисках і навіть на інших комп'ютерах, підключених до локальної мережі;
- ✧ у Vista вбудовано новий поштовий клієнт – Windows Mail. Ця програма настільки добре відпрацьована, що у більшості користувачів не виникне потреби в установці додаткового поштового клієнта;

- ✧ Photo Gallery – це вбудований у Vista додаток для створення колекцій цифрових фотографій з можливістю їх впорядкування і швидкого пошуку;
- ✧ оновлено Movie Maker у складі Vista. За допомогою цієї програми можна легко створювати DVD.
- ✧ новий Media Player 11 працює як в Windows Vista, так і Windows XP. Хоча, за попередніми відомостями, версія плеєра для Vista буде функціональнішою.

**Windows 7 (2009 р.)** заснована на Windows Vista і відрізняється вищою безпекою, надійністю, розширеними мережевими можливостями і новими можливостями взаємодії з користувачем. Одночасно з цим система Windows 7 дозволяє підвищити продуктивність праці користувачів, де б вони не знаходилися, а також надає засоби для ефективного управління ПК.

***Можливості операційної системи Windows 7.***

- ✧ щодо **продуктивності**, за словами розробників Windows 7, внесені істотні удосконалення, які прискорюють запуск комп'ютера, завершення роботи і вихід з режиму, що чекає, порівняно з попередніми версіями Windows. Пошук і індексація тепер повинні виконуватися швидше, а від зовнішніх USB-пристроїв флеш-пам'яті можна отримати більше переваг щодо підвищення продуктивності комп'ютера;
- ✧ удосконалення щодо **надійності** Windows 7 включають ряд нових засобів усунення недоліків, загального підвищення надійності і відмовостійкості, що дозволяє усунути найбільш поширені проблеми, пов'язані з управлінням пам'яттю; засіб відновлення процесів, що дозволяє скоротити час простою, пов'язаного з діагностикою; засіб діагностики зависань мережі, призначений для виявлення і діагностики найбільш поширених ситуацій зависання мережі і відновлення після таких зависань без повідомлення користувача;
- ✧ **сумісність додатків** – один з найважливіших моментів. Основна мета, поставлена перед розробниками, полягає в тому, щоб програмне забезпечення для Windows Vista працювало під Windows 7 так само або ефективніше. Для досягнення цього Windows 7 тестується з найширше використовуваними призначеними для користувача та корпоративними застосуваннями;
- ✧ **сумісність пристроїв.** Як і у випадку з програмним забезпеченням, при розробці Windows 7 була поставлена мета працездатності пристроїв, працюючих під Windows Vista, в системі Windows 7. Окрім цього, було значно розширено список пристроїв і периферійного устаткування, протестованого на сумісність;

- ✧ **безпека.** Основні функції безпеки Windows 7 побудовані на базі системи безпеки Windows Vista; у тому числі усі процедури і технології безпечної розробки ПЗ (Security Development Lifecycle); при цьому в ключові технології безпеки (обробка подій, контроль облікових записів, пристрої зберігання з підвищеною безпекою і перевірка достовірності) внесені істотні удосконалення. Фундаментальні функції безпеки, такі як захист ядра від змін, посилення захисту служб, запобігання виконанню даних, випадкова зміна структури адресного простору і обов'язкові рівні цілісності, як і раніше забезпечують захист від шкідливих програм і атак;
- ✧ **збільшення часу роботи батареї.** У системі управління живленням Windows 7 відбулися значні зміни, спрямовані на продовження часу роботи мобільних ПК. Так, засіб Powercfg включає нову функцію switch/energy, за допомогою якої можна діагностувати найбільш поширені проблеми витрати заряду батареї, а також протестувати працездатність батареї і визначити необхідність її заміни.

**Windows 8 и Windows RT (2012 р.)** побудовані на надійному фундаменті платформи Windows 7, що піддалася усебічному поліпшенню, і спроектовані для роботи з новим устаткуванням і новими пристроями. Радикально оновлено дизайн, підтримка сенсорного управління і миттєве відновлення роботи знаменують нову еру розвитку комп'ютерів.

Висока швидкість роботи, миттєвий запуск, триваліший час автономної роботи і чітке реагування на дотики дозволяють відчути новий рівень продуктивності комп'ютерів.

Windows відрізняється швидкістю, гнучкістю і привабливим дизайном. Ця операційна система дозволяє зручно представити усю важливу інформацію – сайти, контакти, додатки і багато що інше – прямо на початковому екрані. Windows 8 і Windows RT надають платформу для грандіозних інновацій у сфері обладнання, створюючи передумови для появи нового покоління планшетів, ноутбуків і моноблоків, що використовують переваги сенсорного управління, миші і клавіатури. І в кожному випадку операційна система забезпечує безпрецедентні можливості для швидкої і надійної роботи.

Усе, що треба, знаходиться на початковому екрані (рис. 1.57).

Тепер для входу на комп'ютер не треба запам'ятовувати пароль, що складається з букв і цифр. Віднині можна застосовувати графічний пароль.

Internet Explorer 10 – це абсолютно новий браузер, у якому реалізовані усі переваги Windows 8 і Windows RT (рис.1.58).

**Поліпшена безпека.** Windows 8 і Windows RT допомагають захистити комп'ютер у будь-яких ситуаціях.




Рисунок 1.57. Початковий екран Windows 8

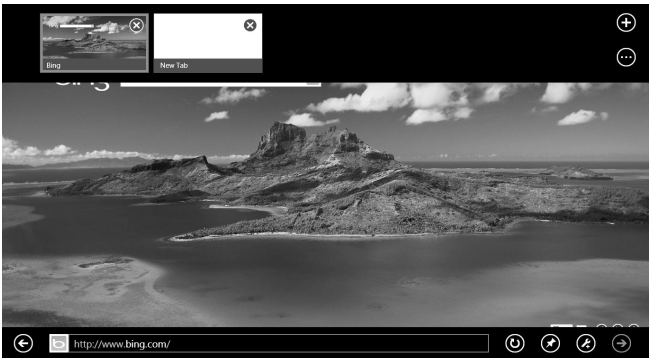



Рисунок 1.58. Internet Explorer 10 у Windows 8

**Windows 8 і Windows RT** змінюють сам стиль роботи на комп'ютері. Привабливий дизайн, швидкість і гнучкість поєднуються в них з продуктивністю Windows.

Оновлений **диспетчер завдань** дозволяє швидко побачити, які додатки і служби використовують ресурси комп'ютера (рис.1.59).

Оновлений провідник (рис. 1.60). Нова стрічка в провіднику значно спрощує виконання найпоширеніших завдань, таких як копіювання і вставка файлів, відображення розширень імен файлів і пошук файлів за датою, типом або за іншими параметрами.

**Відновлення і повернення комп'ютера в початковий стан**. Підтримується декілька можливостей відновлення комп'ютера. Можна навіть відновити комп'ютер, зберігши при цьому усі документи, облікові записи, особисті параметри і додатки Магазину Windows. Проте операційна система Windows при цьому повертається в початковий стан.

**Підтримка декількох моніторів**. Можна читати звіт на одному моніторі і працювати над презентацією на іншому або шукати додаток

в Магазині Windows і переглядати новітні огляди в Інтернеті. Гнучка підтримка декількох моніторів дозволяє добитися високої продуктивності.

**Windows 8 Професійна** містить усі функції Windows 8, а також дозволяє легко підключатися до корпоративних мереж, діставати доступ до файлів в дорозі і багато що інше.

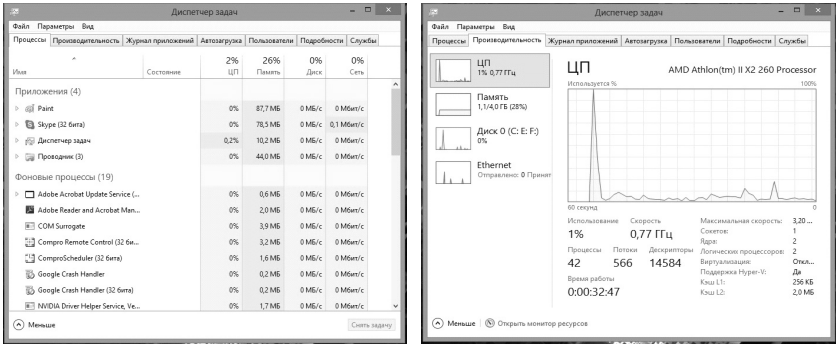



Рисунок 1.59. Диспетчер завдань у Windows 8

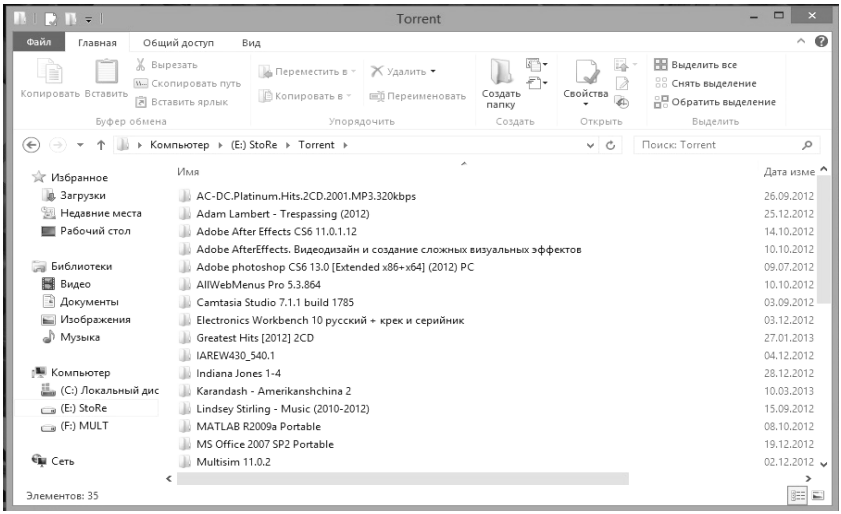



Рисунок 1.60. Проводник у Windows 8

### 1.6.3 Мережні операційні системи

**Операційна система комп'ютерної мережі** є комплексом взаємозалежних програм, який забезпечує зручність роботи користувачів та програмістів, надаючи їм деяку віртуальну систему й реалізуючи

ефективний спосіб поділу ресурсів між безліччю виконуваних у мережі процесів.

***Комп'ютерна мережа*** – це набір комп'ютерів, зв'язаних комунікаційною системою і оснащених відповідним програмним забезпеченням, яке уможливлює користувачам мережі доступ до ресурсів цього набору комп'ютерів.

Комунікаційна система може включати кабелі, повторювачі, комутатори, маршрутизатори та інші пристрої, які забезпечують передачу повідомлень між будь-якою парою комп'ютерів у мережі.

Комп'ютерна мережа надає можливість працювати із своїм комп'ютером як з автономним і додає до цього можливість доступу до інформаційних і апаратних ресурсів інших комп'ютерів мережі.

При організації мережної роботи операційна система екранує від користувача всі деталі низькорівневих програмно-апаратних засобів мережі.

### 1.6.3.1 Мережні й розподілені ОС

Залежно від того, який віртуальний образ створює операційна система для того, щоб підмінити ним реальну апаратуру комп'ютерної мережі, розрізняють мережні ОС і розподілені ОС.

**Мережна ОС** надає користувачеві якусь віртуальну обчислювальну систему, працювати з якою простіше, ніж з реальною мережною апаратурою. У той же час ця віртуальна система не повністю приховує розподілену природу реального прототипу, тобто є *віртуальною мережею*.

При використанні ресурсів комп'ютерів мережі користувач повинен враховувати те, що ресурси – мережні й для доступу до них варто виконати деякі особливі операції. Користувач мережної ОС звичайно повинен обізнантй, де зберігаються його файли, і використовувати явні команди для переміщення файлів з однієї машини на іншу.

При роботі в середовищі мережної ОС користувач може запустити завдання на будь-якій машині мережі, якщо знає, на якій машині це завдання повинно виконуватися.

За замовчуванням користувач виконує завдання на тій машині, на якій користувач виконав реєстрацію (вхід у систему).

Якщо користувач хоче виконати завдання на віддаленій машині, він повинен на ній зареєструватися (команда типу remote login), або в команді на виконання завдання прописати віддалений комп'ютер.

Магістральним напрямом розвитку мережних операційних систем є досягнення якомога вищого ступеня прозорості мережних ресурсів. В ідеальному випадку мережна операційна система повинна надавати користувачеві мережні ресурси у вигляді ресурсів єдиної централізованої віртуальної машини. Така операційна система називається **розподіленою ОС** або **істинно розподіленою ОС**.

Розподілена ОС змушує набір мережних машин працювати як віртуальний уніпроцесор. Користувач розподіленої ОС не має відомостей про те, на якій машині виконується його робота.

Розподілена ОС існує як єдина операційна система в масштабах обчислювальної системи. Кожний компонент мережі, що працює під керуванням розподіленої ОС, виконує частину функцій цієї глобальної ОС. Розподілена ОС поєднує всі комп'ютери мережі в тому розумінні, що вони працюють у тісній кооперації один з одним для ефективного використання всіх ресурсів комп'ютерної мережі.

У цей час практично всі мережні операційні системи далекі від ідеалу справжньої розподіленості. Ступінь автономності кожного комп'ютера в мережі, який працює під керуванням мережної операційної системи, значно вищий порівняно з комп'ютерами, що працюють під керуванням розподіленої ОС.

Таким чином, мережна ОС може розглядатися як набір операційних систем окремих комп'ютерів, які становлять мережу. На різних комп'ютерах мережі можуть використовувати однакові або різні ОС. У кожному випадку операційні системи комп'ютерів, що працюють у мережі, повинні включати взаємно погоджений набір комутаційних протоколів для організації взаємодії процесів, що виконуються на різних комп'ютерах мережі, і розподілу ресурсів цих комп'ютерів між користувачами мережі.

Якщо ОС окремого комп'ютера дозволяє йому працювати в мережі, то якщо надавати свої ресурси в загальне користування і/або споживати ресурси інших комп'ютерів мережі, то така операційна система окремого комп'ютера також називається мережною ОС.

Термін "мережна операційна система" використовується у двох значеннях:

- ✧ сукупність ОС усіх комп'ютерів мережі;
- ✧ операційна система окремого комп'ютера, здатного працювати в мережі.

### 1.6.3.2 Функціональні компоненти мережної ОС

Функціональні компоненти мережної ОС наступні:

засоби керування локальними ресурсами комп'ютера, які реалізують усі функції ОС автономного комп'ютера (розподіл оперативної пам'яті між процесами, планування й диспетчеризація процесів, керування процесами в мультипроцесорних машинах, керування зовнішньою пам'яттю, інтерфейс із користувачем тощо);

мережні засоби, які підрозділяються, у свою чергу, на наступні:

1) засоби надання локальних ресурсів і послуг у загальне користування – серверна частина ОС;

2) засоби запиту доступу до віддалених ресурсів і послуг – клієнтська частина ОС;
3) транспортні засоби ОС, які разом з комутаційною системою забезпечують передачу повідомлень між комп'ютерами мережі.

Клієнтська частина ОС не може отримати безпосередній доступ до ресурсів іншого комп'ютера. Вона може тільки "попросити" про це серверну частину ОС, що працює на тому комп'ютері, якому належать ресурси. Ця "прохання" виражається у вигляді повідомлень, переданих мережею. Повідомлення можуть містити не тільки команди на виконання деяких дій, але й дані, наприклад, вміст файлів.

Управляють передачею повідомлень між клієнтською й серверною частинами по комунікаційній системі мережі транспортні засоби. Транспортні засоби виконують такі дії, як форматування повідомлень, розбивка повідомлень на частини, перетворення імен комп'ютерів у числові адреси, організацію надійної доставки, визначення маршруту в складній мережі тощо.

Правила взаємодії комп'ютерів при передачі повідомлень мережею фіксуються в комунікаційних протоколах (Ethernet, Token Ring, IP, IPX та ін.). Щоб два комп'ютери змогли обмінюватися повідомленнями в мережі, транспортні засоби їх ОС повинні підтримувати деякий загальний набір комунікаційних протоколів. Комунікаційні протоколи переносять повідомлення клієнтських і серверних частин ОС по мережі, не вникаючи в їхній зміст.

Дуже зручною й корисною функцією клієнтської частини ОС є здатність відрізнити запит до віддаленого файлу від запиту до локального файлу. У цьому випадку додаток не повинен піклуватися про те, з локальним або віддаленим файлом він працює, тому що клієнтська частина сама розпізнає й перенаправляє (redirect) запит до віддаленої машини. Клієнтську частини служби називають редиректором. Якщо функція розпізнавання виділена в окремий модуль, то редиректором називається тільки цей модуль.

Клієнтські частини мережних ОС виконують перетворення форматів запитів до ресурсів із запиту у формі, прийнятій у локальній ОС, у запит у формі, що відповідає вимогам серверної частини ОС. Прийнята від сервера відповідь перетвориться в локальний формат.

**Мережні служби й мережні сервіси**

Сукупність серверної й клієнтської частин ОС, які надають доступ до конкретного типу ресурсу комп'ютера через мережу, називається **мережною службою**.

Мережна служба надає користувачам мережі деякий набір послуг. Ці послуги називаються **мережним сервісом** (слово "service" переводиться

як сервіс, послуга, служба). **Сервіс** – це інтерфейс між споживачем послуг і постачальником послуг.

Кожна служба пов'язана з певним типом мережних ресурсів і/або способам доступу до цих ресурсів. Наприклад, служба печатки забезпечує доступ користувачів мережі до поділюваних принтерів мережі й надає сервіс печатки, а поштова служба надає доступ до інформаційного ресурсу мережі – електронним листам.

Способом доступу відрізняється служба вилученого доступу, що надає користувачам комп'ютерної мережі до всіх її ресурсів через телефонні канали, що комутируються. Для одержання вилученого доступу до конкретного ресурсу, наприклад, до принтера, служба вилученого доступу взаємодіє зі службою друку.

Найбільш важливими для користувачів мережних ОС є файлова служба й служба друку.

Серед мережних служб є такі, які орієнтовані на адміністратора. Ці служби використовуються для організації мережі.

Позиція мережний ОС, у ряді інших мережних ОС, залежить від того, наскільки багатий набір послуг надає операційна система кінцевим користувачам.

Мережні служби за своєю природою є клієнт серверними. Оскільки при реалізації будь-якого мережного сервісу природно виникає джерело запитів (клієнт) і виконавець запитів (сервер), те й будь-яка мережна служба містить у своєму складі дві несиметричні частини – **клієнтську** й **серверну**. Мережна служба може бути представлена в операційній системі або обома частинами, або тільки однієї.

Сервер надає свої послуги клієнтові, а клієнт ними користується, однак, при наданні мережною службою деякої послуги використовуються ресурси не тільки сервера, але й клієнта. Клієнт може затрачати значну частину своїх ресурсів, наприклад, дискового простору, процесорного часу, на підтримку роботи мережної служби.

Принциповою різницею між клієнтом і сервером є те, що ініціатором роботи мережної служби завжди виступає клієнт, а сервер завжди перебуває в режимі пасивного очікування запитів.

Звичайна взаємодія клієнтської із серверної частинами стандартизується, так що один тип сервера може бути розрахований на роботу із клієнтами різного типу, реалізованими різними способами й різними виробниками. Єдина умова для цього – клієнти й сервер повинні підтримувати загальний стандартний протокол взаємодії.

**Вбудовані мережні служби й мережні оболонки**

На практиці заклалися кілька підходів до побудови мережних операційних систем, які розрізняються глибиною впровадження мережних служб в операційну систему:

- мережні служби глибоко **убудовані** в ОС;
- мережні служби об'єднані у вигляді деякого набору – **оболонки**;
- мережні служби виробляються й поставляються у вигляді **окремого продукту**.

Перші мережні ОС являли собою сукупність уже існуючої локальної ОС і надстроєної над нею мережної оболонки. При цьому в локальну ОС вбудовувався мінімум мережних функцій, необхідних для роботи мережної оболонки, що виконувала основні мережні функції.

Розроблювальні надалі мережні ОС проектувалися спеціально для роботи в мережі. Мережні функції в цих ОС глибоко вбудовуються в основні модулі системи, що забезпечує її логічну стрункість, простоту експлуатації й модифікації, а також високу продуктивність. При такому підході відсутня надмірність. Якщо всі мережні служби добре інтегровані, тобто розглядаються як невід'ємні частини ОС, то всі внутрішні механізми такої операційної системи можуть бути оптимізовані для виконання мережних функцій. До мережних ОС із убудованими мережними службами ставляться: Windows NT, всі сучасні версії Unix, NetWare, OS/2 Warp.

Інший варіант реалізації мережних служб – об'єднання їх у вигляді деякого набору (оболонки), при цьому всі служби цього набору повинні бути між собою погоджені, тобто у своїй роботі вони можуть звертатися друга один до одного, можуть мати у своєму складі загальні компоненти, наприклад, загальну підсистему аутентифікації користувачів або єдиний користувальницький інтерфейс. Для роботи оболонки необхідна наявність деякої локальної операційної системи, яка б виконувала звичайні функції, необхідні для керування роботою комп'ютера, і середовищу якої виконувалися б мережні служби, які становлять цю оболонку. *Оболонка представляє собою самостійний програмний продукт, що має назву, номер версії й інші характеристики*.

Мережні оболонки часто підрозділяються на **клієнтські** й **серверні**. ***Оболонка***, яка переважно містить клієнтські частини мережних служб, називається ***клієнтською***.

Відповідно, ***серверна мережна оболонка*** орієнтована на виконання серверних служб. Серверна оболонка як мінімум містить північні компоненти двох основних мережних служб – файлової служби й служби друку. Деякі оболонки містять настільки широкий набір служб, що їх називають ***мережними операційними системами***. Таким чином, ***термін "мережна операційна система"*** здобуває ще одне значення – набір мережних служб, здатних узгоджено працювати в загальному операційному середовищі.

З одним типом ресурсів можуть бути зв'язані різні служби, що відрізняються протоколом взаємодії клієнтських і серверних частин.

Мережні оболонки створюються як для локальних, так і для мережних операційних систем.

Існує й третій спосіб реалізації мережної служби – у вигляді окремого продукту.

**Однорангові й серверні мережні операційні системи**

Залежно від розподілу функцій між комп'ютерами мережі, ролі комп'ютерів наступні:

- комп'ютер, що займається винятково обслуговуванням запитів інших комп'ютерів, – ***виділений сервер мережі***;
- комп'ютер, що звертається із запитами до ресурсів іншої машини, – ***клієнтський вузол***;
- комп'ютер, що сполучає функції сервера й клієнта, – ***одноранговий вузол***.

Мережа, яка виправдує своє призначення й забезпечує взаємодію комп'ютерів, може бути побудована за однією з наступних трьох схем:

1) мережа на основі однорангових вузлів – ***одноранговa мережа***;
2) мережа на основі клієнтів і серверів – ***мережа з виділеними серверами***;
3) мережа, що включає вузли всіх типів, – ***гібридна мережа***.

Кожна схема має переваги й недоліки.

В однорангових мережах на всіх комп'ютерах встановлюють таку операційну систему, яка надає всім комп'ютерам у мережі потенційно рівні можливості. Мережні операційні системи такого типу називаються ***одноранговими мережними ОС***. Однорангові ОС повинні включати як серверні, так і клієнтські компоненти мережних служб**.**

Завжди в однорангових мережах спеціалізація ОС не залежить від того, яку функцію виконує комп'ютер: клієнта або сервера. Зміна ролі комп'ютера в одноранговій мережі досягається за рахунок того, що функції серверної або клієнтської частин просто не використовуються.

**Переваги**. Однорангові мережі простіші в експлуатації та організації, вони цілком придатні для роботи в невеликих мережах, де кількість комп'ютерів не перевищує 10-20. У таких мережах немає необхідності в застосуванні централізованих засобів адміністрування, користувачі самі домовляються про перелік ресурсів, що знаходяться в загальному користуванні і паролі для доступу до них.

У більших мережах засоби централізованого адміністрування, зберігання й обробки даних, а особливо захист даних стають необхідними, і такі можливості легше забезпечити в **мережах з виділеними серверами**.

У мережах з виділеними серверами використовуються спеціальні варіанти мережних ОС, пристосованих для роботи в ролі серверів, які називають ***серверними ОС***. Користувальницькі комп'ютери в цих мережах працюють під керуванням ***клієнтських ОС***.

Спеціалізація операційної системи для роботи в якості сервера є природним способом підвищення ефективності серверних операцій.

При наявності в мережі сотень і навіть тисяч користувачів інтенсивність запитів до спільно використовуваних ресурсів може бути дуже великою, і сервер повинен справлятися із цим потоком запитів без великих затримок.

Рішенням цієї проблеми є використання в якості сервера комп'ютера з потужною апаратною платформою й операційною системою, оптимізованою для серверних функцій.

Чим менше функцій виконує ОС, тим ефективніше можна їх реалізувати, тому із серверної ОС знімаються деякі функції універсальної ОС. Однак занадто вузька спеціалізація деяких серверних ОС є одночасно і їхньою слабкою стороною, тому розроблювачі багатьох серверних ОС відмовляються від функціональної обмеженості й включають до складу серверних ОС усі компоненти, які дозволяють використовувати їх як універсальний сервер і навіть у якості клієнтської ОС. Такі серверні ОС забезпечуються розвиненим графічним користувальницьким інтерфейсом і підтримують універсальний API. Це зближує їх з одноранговими операційними системами, але існують кілька відмінностей, що дозволяють віднести їх у класу серверних ОС:

- ✧ підтримка потужних апаратних платформ, у тому числі мультипроцесорних;
- ✧ підтримка великого числа одночасно виконуваних процесів і мережних з'єднань;
- ✧ включення до складу ОС компонентів централізованого адміністрування мережі, наприклад: довідкової служби, служби аутентифікації й авторизації користувачів мережі;
- ✧ ширший набір мережних служб.

Клієнтські операційні системи в мережах з виділеними серверами звичайно звільняють від серверних функцій, що значно спрощує їхню організацію. Розроблювачі клієнтських ОС основну увагу приділяють користувальницькому інтерфейсу і клієнтським частинам мережних служб. Найпростіші клієнтські ОС підтримують тільки базові мережні служби: файлову службу й службу друку. У той же час існують так звані універсальні клієнти, які підтримують широкий набір клієнтських частин, які дозволяють їм працювати практично з усіма серверами мережі.

Багато компаній, які розробляють мережні ОС, випускають два варіанти однієї операційної системи: один варіант – для роботи в якості серверної ОС, іншої – для клієнтської. Ці варіанти найчастіше засновані на тому самому базовому коді, але відрізняються набором служб і утиліт, а також параметрами конфігурації, деякі з яких установлюються за замовчуванням і не піддаються зміні.

### 1.6.4 Вимоги до ОС

Сьогодні до ОС пред'являється безліч вимог. Головними з них, звичайно ж, є виконання функцій ефективного керування ресурсами обчислювальної системи та забезпечення зручного інтерфейсу для користувача і прикладних програм. Крім того, можна виділити низку основних вимог, яким повинна задовольняти будь-яка сучасна ОС.

1. ***Продуктивність***. ОС повинна мати таку швидкодію і час реакції, наскільки це дозволяє апаратна платформа. На продуктивність ОС впливає безліч факторів, серед яких основними є архітектура ОС, різноманіття реалізованих нею функцій, кількість ресурсів, споживаних самою ОС для виконання поставлених перед нею завдань, якість програмного коду.
2. ***Надійність***. Це вимога ОС визначається архітектурними рішеннями, покладеними в її основу, а також якістю реалізації, обернено пропорційній кількості помилок у комплексі програм, що становлять ОС.
3. ***Захищеність***. Система повинна бути захищена як від внутрішніх, так і від зовнішніх помилок, збоїв і відмов. Її дії повинні бути завжди передбачувані, а додатки не повинні мати можливості завдавати шкоди. Сучасна ОС захищає дані та інші ресурси обчислювальної системи від несанкціонованого доступу і від спроб ненавмисного ушкодження цих даних.
4. ***Розширюваність***. **ОС** є **розширюваною**, якщо в неї можна вносити доповнення й зміни, не порушуючи цілісності системи. Розширюваність досягається за рахунок модульної структури ОС, при якій програми будуються з окремих модулів, взаємодіючих тільки через функціональний інтерфейс. Така архітектура дозволяє, при потребі, додавати нові або видаляти непотрібні компоненти. Однак простота, з якою користувач або системний програміст зможе робити такі функціональні зміни, визначається досконалістю і продуманістю застосовуваних при побудові системи рішень.
5. ***Мобільність***. В ідеальному випадку ОС повинна легко переноситися з одного типу апаратної платформи на іншу. Реально це досить не завжди швидко й легко здійсненне завдання. Як правило, ОС розробляється для певного типу апаратних платформ і перенесення її на платформу із принципово іншою будовою може стати важким завданням.
6. ***Сумісність***. ОС згодом завжди змінюються, і ці зміни більше значущі, ніж зміни апаратних засобів. Зміни ОС звичайно полягають у придбанні ними нових властивостей, додаванні нових і модифікації наявних функцій. Під вимогою сумісності розуміється збереження можливості використання прикладних програм, написаних для "старої" або взагалі іншої ОС, у новій ОС.

7. ***Зручність***. Засоби ОС повинні бути простими і гнучкими, а логіка її роботи ясна користувачеві. Сучасні ОС орієнтовані на забезпечення користувачеві максимально можливої зручності при роботі з ними. Необхідною умовою цього стала наявність в ОС графічного користувальницького інтерфейсу та різноманітних майстрів – програм, які автоматизують інсталяцію, налаштування та експлуатацію системи.

Залежно від галузі застосування конкретної ОС змінюється й склад вимог до неї. Часто деякі з вимог можуть суперечити одна одній. Наприклад, на здійснення необхідних перевірок і керування, пов'язаних із забезпеченням захисту, витрачається як процесорний час, так і деякі інші ресурси обчислювальної системи. Контроль захисту може збільшити час реакції системи і її продуктивність. Крім того, розроблювачеві ОС доводиться зіштовхуватися з рядом додаткових факторів, які сильно обмежують його можливості у виборі варіантів рішень. Так, граничні можливості електронних засобів, використовуваних у даному комп'ютері, визначають межі його застосовності. Економічні міркування при створенні системи також значно впливають на долю ОС. Незважаючи на це, будь-яка сучасна система повинна мати перераховані вище властивості.

## 1.7 Організація файлової системи. Захист від збоїв і несанкціонованого доступу

### 1.7.1 Логічна організація файлової системи

Диски є особливим видом периферійних пристроїв, на яких зберігається більша частина як користувальницьких, так і системних даних. Дані на дисках організують у файлові системи, і властивості файлової системи багато в чому визначають властивості самої операційної системи (ОС) – її відмовостійкість, швидкодію, максимальний обсяг збережених даних. Популярність файлової системи часто приводить до її міграції з "рідної" ОС в інші операційні системи – наприклад, файлова система FAT з'явилася спочатку в MS-DOS, потім була реалізована в OS/2, сімействі MS Windows та багатьох реалізаціях UNIX. Внаслідок цього підтримка декількох популярних файлових систем для підсистеми введення-виведення також важлива, як і підтримка широкого спектру периферійних пристроїв. Важливо також, щоб архітектура підсистеми введення-виведення дозволяла досить просто включати в її склад нові типи файлових систем, без необхідності переписування коду. Зазвичай в операційній системі є спеціальний шар програмного забезпечення, відповідальний за вирішення цієї задачі, наприклад шар **VFS (Virtual File System)** у версіях UNIX на основі коду **System V Release 4**.

Одним з основних завдань операційної системи є надання зручностей користувачеві при роботі з даними, що зберігаються на дисках. Для цього ОС підмінює фізичну структуру збережених даних деякою зручною для користувача логічною моделлю. **Логічну модель** файлової системи реалізують у вигляді дерева каталогів, виведеного на екран такими утилітами, як Norton Commander або Windows Explorer, у символьних складених іменах файлів, у командах роботи з файлами. Базовим елементом цієї моделі є файл, що так само, як і файлова система в цілому, може характеризуватися як логічною, так і фізичною структурою.

**Мета і завдання файлової системи**

**Файл** – *це поіменована область зовнішньої пам'яті, у яку можна записувати і з якої можна зчитувати дані*. Файли зберігаються в пам'яті, що не залежить від енергоживлення, зазвичай – на магнітних дисках. Виключенням є електронний диск, коли в оперативній пам'яті створюється структура, що імітує файлову систему.

Основними задачами застосування файлу є:

- ***довгострокове й надійне зберігання інформації***. Довгостроковість досягається за рахунок використання запам'ятовуючих пристроїв, що не залежать від живлення, а висока надійність визначається засобами захисту доступу до файлів і загальною організацією програмного коду ОС, при якій збої апаратури найчастіше не руйнують інформацію, що зберігається у файлах.
- ***спільне використання інформації***. Файли забезпечують природний і легкий спосіб поділу інформації між додатками й користувачами за рахунок наявності зрозумілого людині символьного імені і незмінності збереженої інформації і розташування файлу. Користувач повинен мати зручні засоби роботи з файлами, включаючи каталоги-довідники, що поєднують файли в групи, засоби пошуку файлів за ознаками, набір команд для створення, модифікації й видалення файлів. Файл може бути створений одним користувачем, а потім використовуватися зовсім іншим користувачем, при цьому творець файлу або адміністратор можуть визначити права доступу до нього іншим користувачам. Ці задачі реалізуються в ОС файловою системою.

**Файлова система** (ФС) – це частина операційної системи, що включає:

- *сукупність усіх файлів на диску*;
- *набори структур даних, використовуваних для керування файлами*, такі, наприклад, як каталоги файлів, дескриптори файлів, таблиці розподілу вільного й зайнятого простору на диску;

– *комплекс системних програмних засобів*, що реалізують різні операції над файлами, такі, як створення, знищення, читання, запис, іменування і пошук файлів.

Файлова система дозволяє програмам обходитися набором досить простих операцій для виконання дій над деяким абстрактним об'єктом, який представляє файл. При цьому програмістам не потрібно мати справу з деталями дійсного розташування даних на диску, буферизацією даних та іншими низькорівневими проблемами передачі даних з довгострокового запам'ятовуючого пристрою. Всі ці функції файлова система бере на себе. Файлова система розподіляє дискову пам'ять, підтримує іменування файлів, відображає імена файлів у відповідні адреси в зовнішній пам'яті, забезпечує доступ до даних, підтримує поділ, захист і відновлення файлів.

Таким чином, файлова система відіграє роль проміжного шару, що екранує всі складності фізичної організації довгострокового сховища даних і створює для програм більш просту логічну модель цього сховища, надаючи їм набір зручних у використанні команд для маніпулювання файлами.

Завдання, розв'язувані ФС, залежать від способу організації обчислювального процесу в цілому. Найпростіший тип – це ФС в однокористувальницьких й однопрограмних ОС, до числа яких відноситься, наприклад, MS-DOS. Основні функції в такій ФС націлені на вирішення таких завдань:

– іменування файлів;
– забезпечення програмного інтерфейсу для додатків;
– відображення логічної моделі файлової системи на фізичну організацію сховища даних;
– забезпечення стійкості файлової системи до збоїв живлення, помилок апаратних і програмних засобів.

Завдання ФС ускладнюються в операційних однокористу-вальницьких мультипрограмних ОС, які, хоч і призначені для роботи одного користувача, але дають йому можливість запускати одночасно кілька процесів. Однією з перших ОС цього типу стала OS/2. До перерахованого вище завдання додається нове завдання спільного доступу до файлу з декількох процесів. Файл у цьому випадку є поділюваним ресурсом, і файлова система повинна вирішувати весь комплекс проблем, пов'язаних з такими ресурсами. Зокрема, у ФС повинні бути передбачені засоби блокування файлу і його частин, запобігання гонок, виключення тупиків, узгодження копій тощо.

У багатокористувальницьких системах з'являється ще одне завдання: захист файлів одного користувача від несанкціонованого доступу іншого користувача.

Ще більш складними стають функції ФС, що працює у складі мережевої ОС.

**Типи файлів**

Файлові системи підтримують функціонально різні типи файлів, у число яких, зазвичай, входять *звичайні файли, файли-каталоги, спеціальні файли, іменовані конвеєри, відображувані на згадку файли* та ін.

**Звичайні файли**, або просто **файли**, містять інформацію довільного характеру, яку заносить у них користувач або яка утвориться в результаті роботи системних і користувальницьких програм. Більшість сучасних ОС (наприклад, UNIX, Windows, OS/2) ніяк не обмежує і не контролює вміст і структуру звичайного файлу. Зміст звичайного файлу визначається додатком, який з ним працює. Наприклад, текстовий редактор створює текстові файли, що складаються з рядків символів, представлених у якому-небудь коді. Це можуть бути документи, вихідні тексти програм та ін. Текстові файли можна прочитати на екрані й роздрукувати на принтері. Двійкові файли не використовують коди символів, вони часто мають складну внутрішню структуру, код, який виконується, наприклад, програми або архівні файли. Всі ОС повинні вміти розпізнавати хоча б один тип файлів – їхні власні файли, що виконуються.

**Каталоги** – це особливий тип файлів, які містять системну довідкову інформацію про набір файлів, згрупованих користувачами за якою-небудь неформальною ознакою (наприклад, в одну групу поєднуються файли, що містять документи одного договору, або файли, що становлять один програмний пакет). У багатьох ОС у каталог можуть входити файли будь-яких типів, у тому числі інші каталоги, за рахунок чого утвориться деревоподібна структура, зручна для пошуку. Каталоги встановлюють відповідність між іменами файлів і їхніми характеристиками, використовуваними файловою системою для керування файлами. У число таких характеристик входить, зокрема, інформація (або покажчик на іншу структуру, що містить ці дані) про тип файлу й розташування його на диску, правах доступу до файлу і датах його створення та модифікації. У всіх інших відношеннях каталоги розглядаються файловою системою як звичайні файли.

**Спеціальні файли** – це фіктивні файли, асоційовані із пристроями введення-виведення, які використовуються для уніфікації механізму доступу до файлів і зовнішніх пристроїв. Спеціальні файли дозволяють користувачеві виконувати операції введення-виведення за допомогою звичайних команд запису у файл або читання з файлу. Ці команди обробляються спочатку програмами файлової системи, а потім на деякому етапі виконання запиту перетворяться операційною системою в команди керування відповідним пристроєм.

**Ієрархічна структура файлової системи**

Користувачі звертаються до файлів за символьними іменами. Однак людська пам'ять обмежує кількість імен об'єктів, до яких користувач може

звертатися за іменем. Ієрархічна організація простору імен дозволяє значно розширити ці межі. Саме тому більшість файлових систем має ієрархічну структуру, в якій рівні утворюються за рахунок того, що каталог нижчого рівня може входити в каталог вищого рівня (рис. 1.61).

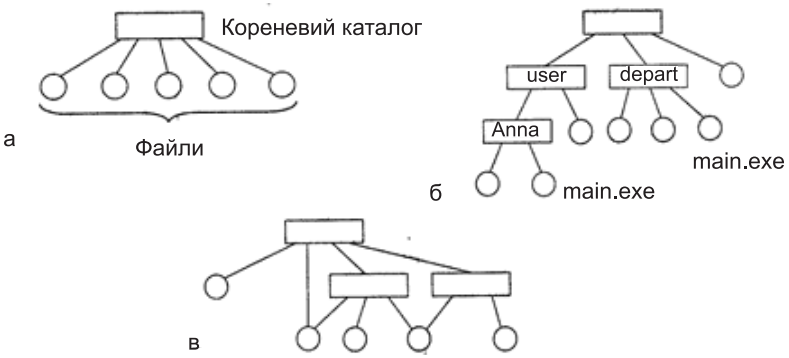



Рисунок 1.61. Ієрархія файлових систем: а – однорівнева організація; б – дерево; в – мережа

Граф, що описує ієрархію каталогів, може бути деревом або мережею. Каталоги утворять дерево, якщо файлу дозволено входити тільки в один каталог (рис. 1.61, б), і мережу – якщо файл може входити відразу в кілька каталогів (рис. 1.61, в). Наприклад, в MS-DOS і Windows каталоги утворять деревоподібну структуру, а в UNIX – мережеву. У деревоподібній структурі кожний файл є *аркушем*. Каталог самого верхнього рівня називається *кореневим каталогом*, або *коренем* (**root**).

При такій організації користувач звільнений від запам'ятовування імен усіх файлів, йому досить приблизно уявляти, до якої групи може бути віднесений той або інший файл, щоб шляхом послідовного перегляду каталогів знайти його. Ієрархічна структура зручна для багатокористувальницької роботи: кожний користувач із своїми файлами локалізується у своєму каталозі або піддереві каталогів, і разом з тим усі файли в системі логічно зв'язані.

Окремим випадком ієрархічної структури є однорівнева організація, коли всі файли входять в один каталог (рис. 1.61, а).

**Імена файлів**

Усі типи файлів мають символьні імена. В ієрархічно організованих файлових системах звичайно використовуються три типи імен файлів: прості, складені й відносні.

*Просте*, або *коротке*, *символьне* ім'я ідентифікує файл у межах одного каталогу. Прості імена присвоюють файлам користувачі і програмісти, при цьому вони повинні враховувати обмеження ОС як на номенклатуру

символів, так і на довжину імені. До недавнього часу ці межі були досить вузькими. Так, у файловій системі FAT довжина імен обмежувалися схемою 8.3 (8 символів – власне ім'я, 3 символи – розширення імені), а у файловій системі s5, підтримуваної багатьма версіями ОС UNIX, просте символьне ім'я не могло містити більше 14 символів. Однак користувачеві набагато зручніше працювати з довгими іменами, оскільки вони дозволяють дати файлам назви, які легко запам'ятовуються і з яких ясно видно, що міститься у цьому файлі. Тому сучасні файлові системи, а також удосконалені варіанти вже існуючих файлових систем, як правило, підтримують довгі прості символьні імена файлів. Наприклад, у файлових системах NTFS і FAT32, що входять до складу операційної системи Windows NT, ім'я файлу може містити до 255 символів.

Приклади простих імен файлів і каталогів:

**quest_ul.doc**
**task-entran.exe**
**додаток до CD 254L російською мовою.doc**
**installable filesystem manager.doc**

В ієрархічних файлових системах різні файли можуть мати однакові прості символьні імена за умови, що вони належать різним каталогам. Тобто тут працює схема "багато файлів – одне просте ім'я". Для однозначної ідентифікації файлу в таких системах використовується так зване повне ім'я.

**Повне ім'я файлу** є ланцюжком простих символьних імен усіх каталогів, через які проходить шлях від кореня до цього файлу. Таким чином, повне ім'я є складовим, у якому прості імена відділені один від одного прийнятим в ОС роздільником. Часто роздільником слугує прямий або зворотний слеш, при цьому прийнято не вказувати ім'я кореневого каталогу. На рис. 1.61, б два файли мають просте ім'я main.exe, однак їхні складені імена **/depart/main.exe** і **/user/anna/main exe** розрізняються.

У деревоподібній файловій системі між файлом і його повним ім'ям є взаємно однозначна відповідність "один файл – одне повне ім'я". У файлових системах, що мають мережеву структуру, файл може входити в кілька каталогів і мати кілька повних імен, тут справедлива відповідність "один файл – багато повних імен". В обох випадках файл однозначно ідентифікується повним ім'ям.

Файл може бути ідентифікований також відносним ім'ям. *Відносне* ім'я файлу визначається через поняття "поточний каталог". Для кожного користувача в кожний момент часу один з каталогів файлової системи є поточним, причому цей каталог вибирається самим користувачем командою ОС. Файлова система фіксує ім'я поточного каталогу, щоб потім використовувати його як доповнення до відносних імен для утворення повного

ім'я файлу. При використанні відносних імен користувач ідентифікує файл ланцюжком імен каталогів, через які проходить маршрут від поточного каталогу до даного файлу. Наприклад, якщо поточним каталогом є каталог **/user**, то відносне ім'я файлу **/user/anna/main.exe** виглядає в такий спосіб: **anna/main.exe.**

У деяких операційних системах дозволено присвоювати одному й тому самому файлу кілька простих імен, які можна інтерпретувати як псевдоніми. У цьому випадку, так само як у системі з мережевою структурою, установлюється відповідність "один файл – багато повних імен", тому що кожному простому імені файлу відповідає принаймні одне повне ім'я.

І хоч повне ім'я однозначно визначає файл, операційній системі простіше працювати з файлом, якщо між файлами і їхніми іменами є взаємно однозначна відповідність. З цією метою вона присвоює файлу *унікальне* ім'я, так що справедливо співвідношення "один файл – одне унікальне ім'я". Унікальне ім'я існує поряд з одним або декількома символьними іменами, що присвоюються файлу користувачами або додатками. Унікальне ім'я є числовим ідентифікатором і призначено тільки для операційної системи. Прикладом такого унікального імені файлу є номер індексного дескриптора в системі UNIX.

**Монтування**

У загальному випадку обчислювальна система може мати кілька дискових пристроїв. Навіть типовий персональний комп'ютер звичайно має один накопичувач на жорсткому диску, один накопичувач на гнучких дисках і накопичувач для компакт-дисків. Потужні ж комп'ютери, як правило, оснащені більшою кількістю дискових накопичувачів, на які встановлюються пакети дисків. Більше того, навіть один фізичний пристрій за допомогою засобів операційної системи може бути представлений у вигляді декількох логічних пристроїв, зокрема шляхом розбивки дискового простору на розділи. Виникає питання, яким чином організувати зберігання файлів у системі, що має кілька пристроїв зовнішньої пам'яті?

Перше рішення полягає в тому, що на кожному з пристроїв розміщується автономна файлова система, тобто файли, які є на цьому пристрої, описуються деревом каталогів, ніяк не пов'язаним з деревами каталогів на інших пристроях. У такому випадку для однозначної ідентифікації файлу користувач поряд із складеним символьним ім'ям файлу повинен указувати ідентифікатор логічного пристрою. Прикладом такого автономного існування файлових систем є операційна система MS-DOS, у якій повне ім'я файлу включає буквений ідентифікатор логічного диска. Так, при звертанні до файлу, розташованому на диску **A:**, користувач повинен указати ім'я цього диска: **A:\privat\letter\uni\let1.doc.**

Іншим варіантом є така організація зберігання файлів, при якій користувачеві надається можливість поєднувати файлові системи, які перебувають на різних пристроях, у єдину файлову систему, описувану єдиним деревом каталогів. Така операція називається **монтуванням**. Розглянемо, як здійснюється ця операція на прикладі ОС UNIX.

Серед усіх наявних у системі логічних дискових пристроїв операційна система виділяє один пристрій, який називають системним. Нехай є дві файлові системи, розташовані на різних логічних дисках (рис. 1.62), причому один з дисків є системним.

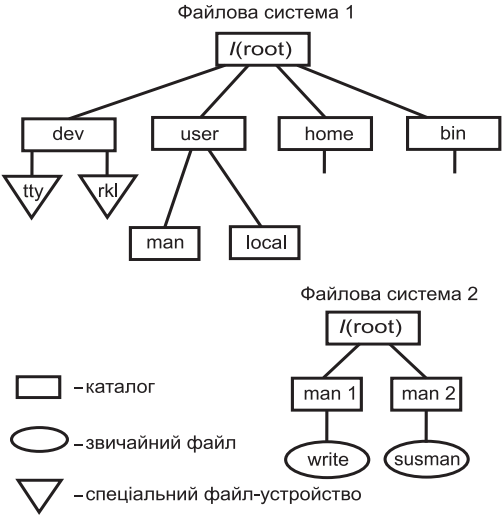



Рисунок 1.62. Дві файлові системи до монтування

Файлова система, розташована на системному диску, призначається кореневою. Для зв'язку ієрархій файлів у кореневій файловій системі вибирається деякий існуючий каталог, у даному прикладі – каталог **man**. Після виконання монтування обраний каталог **man** стає кореневим каталогом другої файлової системи. Через цей каталог монтована файлова система приєднується як піддерево до загального дерева (рис. 1.63).

Після монтування загальної файлової системи для користувача немає логічної різниці між кореневими і змонтованою файловими системами, зокрема, іменування файлів здійснюють так само, немов би вона з самого початку була єдиною.

**Атрибути файлів**

Поняття "файл" включає не тільки дані і присвоєні їм імена, але й атрибути. **Атрибути файлу** – *це інформація, що описує властивості файлу*.

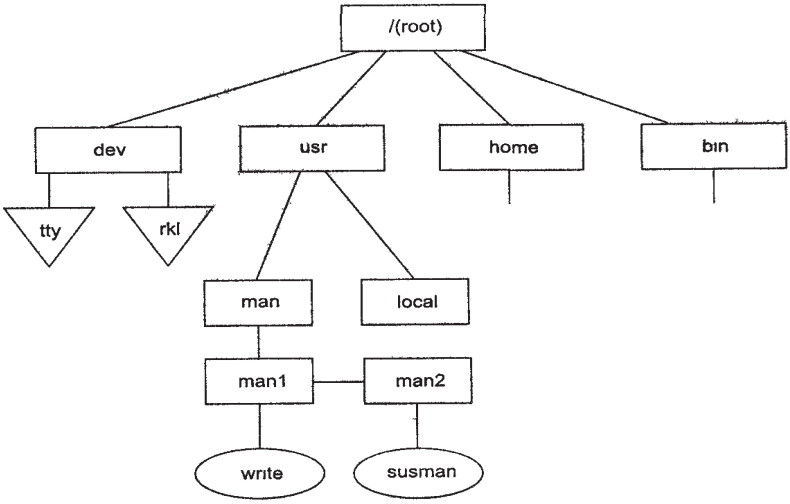



Рисунок 1.63. Загальна файлова система після монтування

Приклади можливих атрибутів файлу:

- тип файлу (звичайний файл, каталог, спеціальний файл тощо);
- власник файлу;
- творець файлу;
- пароль для доступу до файлу;
- інформація про дозволені операції доступу до файлу;
- часи створення, останнього доступу і останньої зміни;
- поточний розмір файлу;
- максимальний розмір файлу;
- ознака “тільки для читання”;
- ознака “прихований файл”;
- ознака “системний файл”;
- ознака “архівний файл”;
- ознака “двійковий/символьний”;
- ознака “тимчасовий” (видалити після завершення процесу);
- ознака блокування;
- довжина запису у файлі;
- покажчик на ключове поле в записуі;
- довжина ключа.

Набір атрибутів файлу визначається специфікою файлової системи: у файлових системах різного типу для характеристики файлів можуть використовуватися різні набори атрибутів. Наприклад, у файлових системах, що підтримують неструктуровані файли, немає необхідності використовувати

три останні атрибути у наведеному списку, зв'язані з структуризацією файлу. В однокористувальницькій ОС у наборі атрибутів будуть відсутні характеристики, що мають відношення до користувачів і захисту, такі як власник файлу, творець файлу, пароль для доступу до файлу, інформація про дозволений доступ до файлу.

Користувач може одержувати доступ до атрибутів, використовуючи засоби, надані для цієї мети файловою системою. Зазвичай дозволяється читати значення будь-яких атрибутів, а змінювати – тільки деякі. Наприклад, користувач може змінити права доступу до файлу (за умови, що він має необхідні для цього повноваження), але змінювати дату створення або поточний розмір файлу йому не дозволяється.

Значення атрибутів файлів можуть безпосередньо утримуватися в каталогах, як це зроблено у файловій системі MS-DOS (рис. 1.64, *а*). На рисунку представлена структура запису в каталозі, що містить просте символьне ім'я і атрибути файлу. Тут буквами позначені ознаки файлу: **R** – тільки для читання, **A** – архівний, **H** – прихований, **S** – системний.

Іншим варіантом є розміщення атрибутів у спеціальних таблицях, коли в каталогах містяться тільки посилання на ці таблиці. Такий підхід реалізований, наприклад, у файловій системі **ufs** ОС UNIX. У цій файловій системі структура каталогу дуже проста. Запис про кожний файл містить коротке символьне ім'я файлу і покажчик на індексний дескриптор файлу, так називають в ufs таблицю, у якій зосереджені значення атрибутів файлу (рис. 1.64, *б)*.

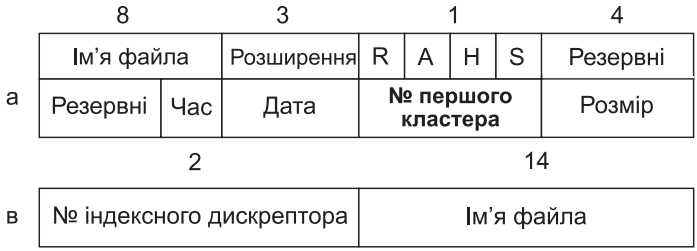



Рисунок 1.64. Структура каталогів:
а – структура запису каталогу MS-DOS (32 байта);
б – структура запису каталогу ОС UNIX

У тому чи іншому варіантах каталоги забезпечують зв'язок між іменами файлів і власне файлами. Однак підхід, коли ім'я файлу відділене від його атрибутів, робить систему більш гнучкою. Наприклад, файл може бути легко включений відразу в кілька каталогів. Записи про цей файл у різних каталогах можуть містити різні прості імена, але в полі посилання буде зазначений той самий номер індексного дескриптора.

**Логічна організація файлу**

У загальному випадку дані, що містяться у файлі, мають якусь логічну структуру. Ця структура є базою при розробці програми, призначеної для обробки цих даних. Наприклад, щоб текст міг бути правильно виведений на екран, програма повинна мати можливість виділити окремі слова, рядки, абзаци тощо. Ознаками, що відокремлюють один структурний елемент від іншого, можуть служити певні кодові послідовності або просто відомі програмі значення зсувів цих структурних елементів відносно початку файлу. Підтримка структури даних може бути або цілком покладеною на додаток, або в тому чи іншому ступені цю роботу може взяти на себе файлова система.

У першому випадку, коли всі дії, пов'язані із структуризацією та інтерпретацією вмісту файлу, повністю відносяться до ведення додатка, файл представляється ФС неструктурованою послідовністю даних. Додаток формулює запити до файлової системи на введення-виведення, використовуючи загальні для всіх додатків системні засоби, наприклад, указуючи зсув від початку файлу і кількість байт, які необхідно рахувати або записати. Потік байтів, який надходить до додатка, інтерпретується відповідно до закладеної у програмі логіки. Наприклад, компілятор генерує, а редактор зв'язків сприймає цілком певний формат об'єктного модуля програми. При цьому формат файлу, у якому зберігається об'єктний модуль, відомий тільки цим програмам.

Модель файлу, відповідно до якої вміст файлу представляється неструктурованою послідовністю (потоком) байт, стала популярною разом з ОС UNIX, а тепер вона широко використовується в більшості сучасних ОС, у тому числі – у Windows NT/2000. Неструктурована модель файлу дозволяє легко організувати поділ файлу між декількома додатками: різні додатки можуть по-своєму структурувати і інтерпретувати дані, що містяться у файлі.

Інша модель файлу, що застосовувалася в ОС OS/360, DEC RSX і VMS, а в цей час використовується досить рідко, – це структурований файл. У цьому випадку підтримку структури файлу доручають файловій системі. Файлова система сприймає файл як упорядковану послідовність логічних записів. Додаток може звертатися до ФС із запитами на введення-виведення на рівні записів, наприклад "зчитати запис 25 з файлу FILE.DOC". ФС повинна мати інформацію про структуру файлу, достатню для того, щоб виділити будь-який запис. ФС надає додатку доступ до запису, а вся подальша обробка даних, які містяться в цьому записі, виконується додатком. Розвитком цього підходу стали системи керування базами даних (СУБД), які підтримують не тільки складну структуру даних, але й взаємозв'язки між ними.

### 1.7.2 Фізична організація файлової системи

Уявлення користувача про файлову систему як про ієрархічно організовану множину інформаційних об'єктів має мало загального з порядком зберігання файлів на диску. Файл, що має образ цільного, неперервного набору байт, насправді дуже часто розкиданий "шматочками" по всьому диску, причому ця розбивка ніяк не пов'язана з логічною структурою файлу, наприклад, його окремий логічний запис може бути розташований у несуміжних секторах диска. Логічно об'єднані файли з одного каталогу зовсім не зобов'язані сусідити на диску. Принципи розміщення файлів, каталогів і системної інформації на реальному пристрої описуються фізичною організацією файлової системи. Очевидно, що різні файлові системи мають різну фізичну організацію.

**Диски, розділи, сектори, кластери**

Основним типом пристрою, що використовується в сучасних обчислювальних системах для зберігання файлів, є дискові накопичувачі. Ці пристрої призначені для зчитування і запису даних на тверді та гнучкі магнітні диски. Жорсткий диск складається з однієї або декількох скляних або металевих пластин, кожна з яких покрита з однієї або обох сторін магнітним матеріалом. Таким чином, диск у загальному випадку складається з пакету пластин (рис. 1.65).

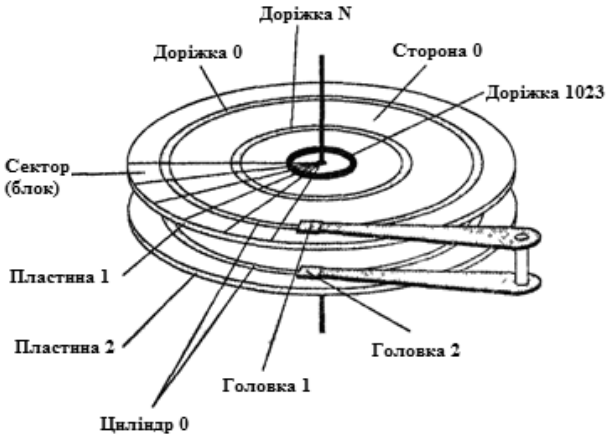



Рисунок 1.65. Схема устрою жорсткого диску

На кожній стороні кожної пластини розмічені тонкі концентричні кільця – *доріжки* (**traks**), на яких зберігаються дані. Кількість доріжок залежить від типу диска. Нумерація доріжок починається з 0 від зовнішнього краю до центра диска. Коли диск обертається, елемент, який називають го-

ловкою, зчитує двійкові дані з магнітної доріжки або записує їх на магнітну доріжку.

Головка може позиціонуватися над заданою доріжкою. Головки переміщаються над поверхнею диску дискретними кроками, кожний крок відповідає зсуву на одну доріжку. Запис на диск здійснюється завдяки здатності головки змінювати магнітні властивості доріжки. У деяких дисках уздовж кожної поверхні переміщається одна головка, а в інших – є по головці на кожну доріжку. У першому випадку для пошуку інформації головка повинна переміщатися по радіусу диска. Зазвичай усі головки закріплені на єдиному механізмі переміщення і рухаються синхронно. Тому, коли головка фіксується на заданій доріжці однієї поверхні, всі інші головки зупиняються над доріжками з такими ж номерами. Отже, коли на кожній доріжці є окрема головка, ніякого переміщення головок з однієї доріжки на іншу не потрібно, за рахунок цього заощаджується час, затрачуване на пошук даних.

Сукупність доріжок одного радіусу на всіх поверхнях усіх пластин пакету називається *циліндром* (**cylinder**). Кожна доріжка розбивається на фрагменти, так звані *сектори* (**sectors**), або блоки (**blocks**), так що всі доріжки мають однакове число секторів, у які можна максимально записати те саме число байт. Сектор має фіксований для конкретної системи розмір, що виражається ступенем двійки. Найчастіше розмір сектора становить 512 байт. З огляду на, що доріжки різного радіусу мають однакове число секторів, щільність запису стає тим вище, чим ближче доріжка до центра.

**Сектор** – *найменша адресована одиниця обміну даними дискового пристрою з оперативною пам'яттю*. Для того щоб контролер міг знайти на диску потрібний сектор, необхідно задати йому всі складові адреси сектора: номер циліндра, номер поверхні і номер сектора. Оскільки прикладній програмі в загальному випадку потрібний не сектор, а деяка кількість байт, не обов'язково кратна розміру сектора, то типовий запит включає читання декількох секторів, що містять необхідну інформацію, і одного або двох секторів, що містять поряд з необхідними надлишкові дані (рис. 1.66).

Операційна система при роботі з диском використовує, як правило, власну одиницю дискового простору, яку називають *кластером* (**cluster**). При створенні файлу місце на диску йому виділяється кластерами. Наприклад, якщо файл має розмір 2560 байт, а розмір кластера у файловій системі визначено у 1024 байти, то файлу буде виділено на диску 3 кластери.

Доріжки і сектори створюються в результаті виконання процедури *фізичного*, або *низькорівневого*, *форматування диска*, що передує використанню диска. Для визначення меж блоків на диск записується ідентифікаційна інформація. Низькорівневий формат диска не залежить від типу операційної системи, яка цей диск буде використовувати.

Розмітку диска під конкретний тип файлової системи виконують процедури *високорівневого*, або *логічного*, *форматування*. При високорівневому форматуванні визначається розмір кластера і на диск записується інформація, необхідна для роботи файлової системи, у тому числі інформація про доступний і невикористовуваний простір, про межі областей, відведених під файли і каталоги, інформація про ушкоджені області. Крім того, на диск записується завантажник операційної системи – невелика програма, що починає процес ініціалізації операційної системи після включення живлення або рестарту комп'ютера.

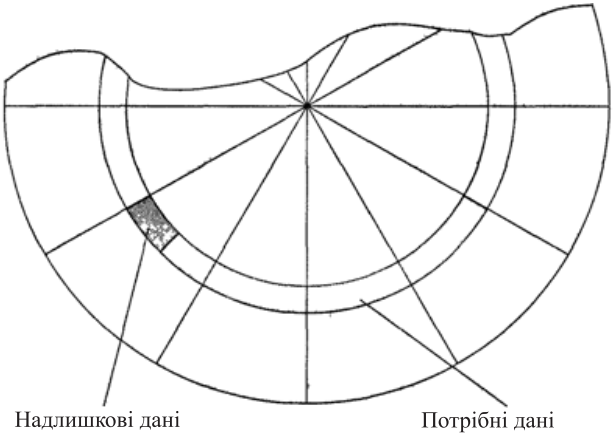



Рисунок 1.66. Зчитування надлишкових даних при обміні з диском

Перш ніж форматувати диск під певну файлову систему, його можна розбити на розділи. **Розділ** – *це неперервна частина фізичного диску, яку операційна система надає користувачеві як логічний пристрій (використовуються також назви логічний диск і логічний розділ)*. Логічний пристрій функціонує так, ніби це є окремий фізичний диск. Саме з логічними пристроями працює користувач, звертаючись до них за символьними іменами, використовуючи, наприклад, позначення **A:**, **B:**, **C:**, **SYS:** тощо. Операційні системи різного типу використовують єдине для всіх їх подання про розділи, але створюють на його основі логічні пристрої, специфічні для кожного типу ОС. Так само як файлова система, з якою працює одна ОС, у загальному випадку не може інтерпретуватися ОС іншого типу, логічні пристрої не можуть бути використані операційними системами різного типу. На кожному логічному пристрої може створюватися тільки одна файлова система.

В окремому випадку, коли весь дисковий простір охоплюється одним розділом, логічний пристрій представляє фізичний пристрій у цілому. Як-

що диск розбитий на кілька розділів, то для кожного з цих розділів може бути створений окремий логічний пристрій. Логічний пристрій може бути створений і на базі декількох розділів, причому ці розділи не обов'язково повинні належати одному фізичному пристрою. Об'єднання декількох розділів у єдиний логічний пристрій може виконуватися різними способами і переслідувати різні цілі, основні з яких: збільшення загального обсягу логічного розділу, підвищення продуктивності і відмовостійкості.

На різних логічних пристроях одного й того самого фізичного диску можуть розташовуватися файлові системи різного типу. На рис. 1.67 показано приклад диску, розбитого на три розділи, в яких установлені дві файлові системи NTFS (розділи **C:** і **E:**) і одна файлова система FAT (розділ **D:**).

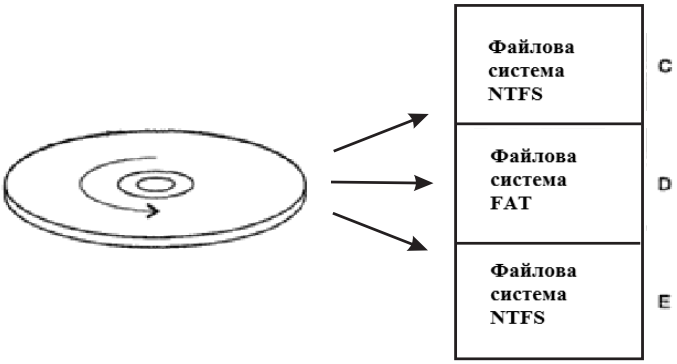



Рисунок 1.67. Розбивка диску на розділи

Усі розділи одного диску мають однаковий розмір блоку, визначений для даного диску в результаті низькорівневого форматування. Однак у результаті високорівневого форматування в різних розділах того самого диску, представлених різними логічними пристроями, можуть бути встановлені файлові системи, у яких визначені кластери інших розмірів.

Операційна система може підтримувати різні статуси розділів, особливим чином відзначаючи розділи, які можуть бути використані для завантаження модулів операційної системи, і розділи, в яких можна встановлювати тільки додатки і зберігати файли даних. Один з розділів диску позначається як завантажувальний (або активний). Саме з цього розділу зчитується завантажник операційної системи.

**Фізична організація й адресація файлу**

Важливим компонентом фізичної організації файлової системи є фізична організація файлу, тобто спосіб розміщення файлу на диску. Основними критеріями ефективності фізичної організації файлів є:

– швидкість доступу до даних;
– обсяг адресної інформації файлу;
– ступінь фрагментування дискового простору;
– максимально можливий розмір файлу.

Неперервне розміщення – найпростіший варіант фізичної організації (рис. 1.68, *а*), при якому файлу надається послідовність кластерів диску, що утворюють неперервну ділянку дискової пам'яті. Основним достоїнством цього методу є висока швидкість доступу, тому що витрати на пошук і зчитування кластерів файлу мінімальні. Також мінімальний обсяг адресної інформації – досить зберігати тільки номер першого кластера й обсяг файлу. Дана фізична організація максимально можливий розмір файлу не обмежує. Однак цей варіант має істотні недоліки, які утрудняють його застосовність на практиці, незважаючи на всю його логічну простоту. При більш пильному розгляді виявляється, що реалізувати цю схему не так вже й просто. Дійсно, якого розміру повинна бути неперервна область, виділювана файлу, якщо файл при кожній модифікації може збільшити свій розмір? Ще більш серйозною проблемою є фрагментація. Через деякий час після створення файлової системи в результаті виконання численних операцій створення й видалення файлів простір диска неминуче перетворюється в "клаптеву ковдру", що включає велику кількість вільних областей невеликого розміру. Як завжди буває при фрагментації, сумарний обсяг вільної пам'яті може бути дуже великим, а вибрати місце для розміщення файлу майже неможливо. Тому на практиці використовуються методи, за якими файл розміщається в декількох, у загальному випадку несуміжних областях диска.

Наступний спосіб фізичної організації – розміщення файлу у вигляді зв'язаного списку кластерів дискової пам'яті (рис. 1.68, *б)*. При такому способі на початку кожного кластера міститься покажчик на наступний кластер. У цьому випадку адресна інформація мінімальна: розташування файлу може бути задано одним числом – номером першого кластера. На відміну від попереднього способу кожний кластер може бути приєднаний до ланцюжка кластерів якого-небудь файлу, отже, фрагментація на рівні кластерів відсутня. Файл може змінювати свій розмір під час свого існування, нарощуючи число кластерів. Недоліком є складність реалізації доступу до довільно заданого місця файлу – щоб прочитати п'ятий кластер файлу, необхідно послідовно прочитати чотири перших кластери, простежуючи ланцюжок номерів кластерів. Крім того, при цьому способі кількість даних файлу, що містяться в одному кластері, не дорівнює ступеню двійки (одне слово витрачене на номер наступного кластера), а багато програм читають дані кластерами, розмір яких дорівнює ступеню двійки.

Популярним способом, застосовуваним, наприклад, у файловій системі FAT, є використання зв'язаного списку індексів (рис. 1.68, *в)*. Цей спосіб є деякою модифікацією попереднього. Файлу також виділяється

пам'ять у вигляді зв'язаного списку кластерів. Номер першого кластера запам'ятовується в запису каталогу, де зберігаються характеристики цього файлу. Інша адресна інформація відділена від кластерів файлу. З кожним кластером диска зв'язується деякий елемент – індекс. Індекси розташовуються в окремій області диску – в MS-DOS це таблиця FAT (**File Allocation Table**), що займає один кластер. Коли пам'ять вільна, всі індекси мають нульове значення. Якщо деякий кластер N призначений деякому файлу, то індекс цього кластера стає рівним або номеру M наступного кластера даного файлу, або приймає спеціальне значення, що є ознакою того, що цей кластер є для файлу останнім. Індекс же попереднього кластера файлу приймає значення N, указуючи на новопризначений кластер.

При такій фізичній організації зберігаються всі переваги попереднього способу: мінімальність адресної інформації, відсутність фрагментації, відсутність проблем при зміні розміру. Крім того, даний спосіб має додаткові переваги. По-перше, для доступу до довільного кластеру файлу не потрібно послідовно зчитувати його кластери, досить прочитати тільки сектори диска, що містять таблицю індексів, відрахувати потрібну кількість кластерів файлу по ланцюжку і визначити номер потрібного кластера. По-друге, дані файлу заповнюють кластер повністю, і тому мають обсяг, рівний ступеню двійки.

Ще один спосіб завдання фізичного розташування файлу полягає в простому перерахуванні номерів кластерів, які займає цей файл (рис. 1.68, *г)*. Цей перелік і служить адресою файлу. Недолік даного способу очевидний: довжина адреси залежить від розміру файлу і для великого файлу може скласти значну величину. Достоїнством же є висока швидкість доступу до довільного кластеру файлу, оскільки тут застосовується пряма адресація, що виключає перегляд ланцюжка покажчиків при пошуку адреси довільного кластеру файлу. Фрагментація на рівні кластерів у цьому способі також відсутня.

Останній підхід з деякими модифікаціями використовується в традиційних файлових системах ОС UNIX **s5** і **ufs**. Для скорочення обсягу адресної інформації прямий спосіб адресації сполучається з непрямим.

У стандартній на сьогоднішній день для UNIX файловій системі ufs використовується наступна схема адресації кластерів файлу. Для зберігання адреси файлу виділено 15 полів, кожне з яких складається з 4 байт (рис. 1.69). Якщо розмір файлу менше або дорівнює 12 кластерам, то номери цих кластерів безпосередньо перераховуються в перших дванадцяти полях адреси. Якщо кластер має розмір 8 Кбайт (максимальний розмір кластера, підтримуваного в **ufs**), то в такий спосіб можна адресувати файл розміром до 8192·12=98304 байт.

Якщо розмір файлу перевищує 12 кластерів, то наступне 13-е поле містить адресу кластера, в якому можуть бути розташовані номери наступ-

них кластерів файлу. Таким чином, 13-й елемент адреси використовується для непрямої адресації. При розмірі в 8 Кбайт кластер, на який указує 13-й елемент, може містити 2048 номерів наступних кластерів даних файлу і розмір файлу може зрости до 8192·(12+2048)=16875520 байт.

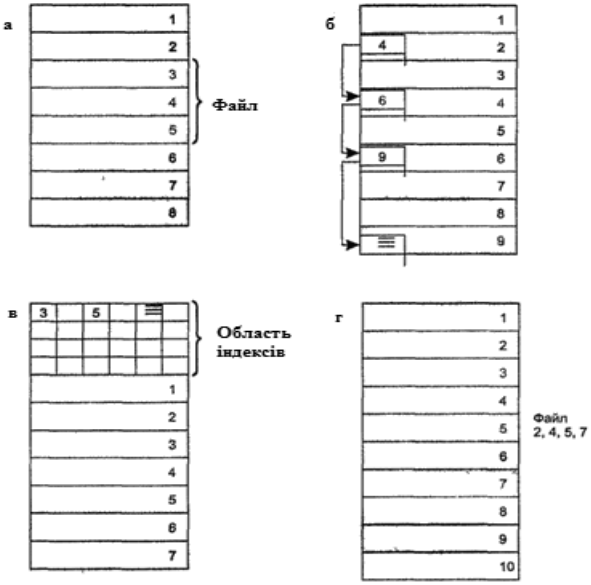



Рисунок 1.68. Фізична організація файлу:
а – неперервне розміщення; б – зв'язаний список кластерів;
в – зв'язаний список індексів; г – перелік номерів кластерів.

Якщо розмір файлу перевищує 12+2048=2060 кластерів, то використовується 14-е поле, у якому перебуває номер кластера, що містить 2048 номерів кластерів, кожний з яких зберігає 2048 номерів кластерів даних файлу. Тут застосовується вже подвійна непряма адресація. З її допомогою можна адресувати кластери у файлах, що містять до $8192\cdot(12+2048+2048^2)=3{,}43766\cdot10^{10}$ байт.

І нарешті, якщо файл включає більше $12+2048+2048^2=4196364$ кластерів, то використовується останнє 15-е поле для потрійної непрямої адресації, що дозволяє задати адресу файлу, що має наступний максимальний розмір: $8192\cdot(12+2048+2048^2+2048^3)=7{,}0403\cdot10^{13}$ байт.

Таким чином, файлова система ufs при розмірі кластера в 8 Кбайт підтримує файли, що складаються максимум з 70 трильйонів байт даних, що зберігаються в 8 мільярдах кластерів. Для задання адресної інформації про максимально великий файл потрібно (див. рис. 27): 15 елементів по 4 байти (60 байт) у центральній частині адреси плюс 1+(1+2048)+(1+204

$8+2048^2$)=4198403 кластери в непрямій частині адреси. Незважаючи на величезну величину, це число становить усього близько 0,05% від обсягу адресованих даних.

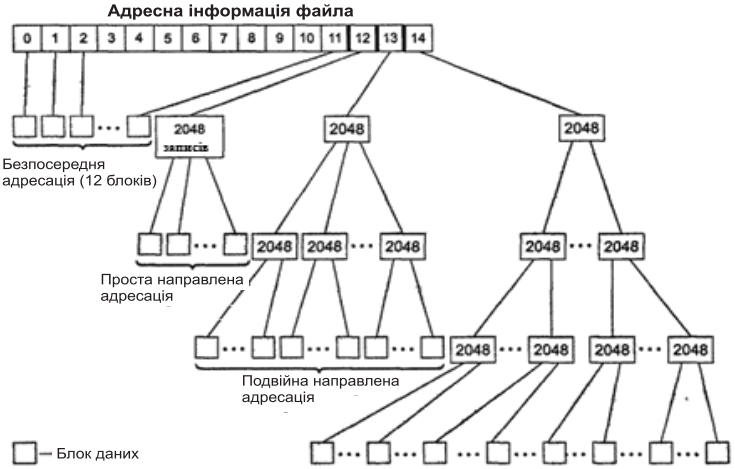



Рисунок 1.69. Схема адресації файлової системи ufs

Файлова система **ufs** підтримує дискові кластери і менших розмірів, при цьому максимальний розмір файлу буде іншим. Використовувана в більш ранніх версіях UNIX файлова система **s5** має аналогічну схему адресації, але вона розрахована на файли менших розмірів, тому в ній використовується 13 адресних елементів замість 15.

Метод перерахування адрес кластерів файлу задіяний і у файловій системі NTFS, яку використовує ОС Windows NT/2000. Тут він доповнений досить природним прийомом, що скорочує обсяг адресної інформації: адресуються не кластери файлу, а неперервні області, що складаються із суміжних кластерів диску. Кожна така область, яку називають *відрізком* (**run**), або *екстентом* (**extent**), описують за допомогою двох чисел: початкового номеру кластера і кількості кластерів у відрізку. Оскільки для скорочення часу операції обміну ОС намагається розмістити файл у послідовних кластерах диску, то у більшості випадків кількість послідовних областей файлу буде менше кількості кластерів файлу, і обсяг службової адресної інформації в NTFS скорочується порівняно із схемою адресації файлових систем **ufs/s5**.

Для того щоб коректно ухвалювати рішення щодо виділення файлу набору кластерів, файлова система повинна відслідковувати інформацію про стан усіх кластерів диску: вільний/зайнятий. Ця інформація може зберігатися як окремо від адресної інформації файлів, так і разом з нею.

**Фізична організація FAT**

Логічний розділ, відформатований під файлову систему FAT, складається з наступних областей (рис. 1.70):

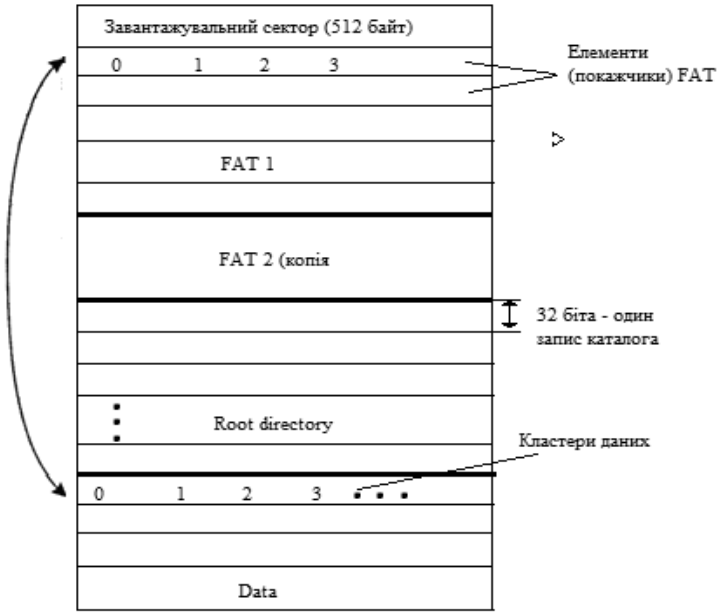



Рисунок 1.70. Фізична структура файлової системи FAT

- *Завантажувальний сектор* містить програму початкового завантаження операційної системи. Вид цієї програми залежить від типу операційної системи, що буде завантажуватися із цього розділу.
- *Основна копія* FAT містить інформацію про розміщення файлів і каталогів на диску.
- *Резервна копія* FAT.
- *Кореневий каталог* займає фіксовану область розміром у 32 сектори (16 Кбайт), що дозволяє зберігати 512 записів про файли й каталоги, тому що кожний запис каталогу складається з 32 байт.
- *Область даних* призначена для розміщення всіх файлів і всіх каталогів, крім кореневого каталогу.

Файлова система FAT підтримує всього два типи файлів: звичайний файл і каталог. Файлова система розподіляє пам'ять тільки з області даних, причому використовує в якості мінімальної одиниці дискового простору кластер.

Таблиця FAT (як основна копія, так і резервна) складається з масиву індексних покажчиків, кількість яких дорівнює кількості кластерів області даних. Між кластерами і індексними покажчиками є взаємно однозначна відповідність – нульовий покажчик відповідає нульовому кластеру і т.д.

Індексний покажчик може приймати наступні значення, що характеризують стан пов'язаного з ним кластера:

- кластер вільний (не використовується);
- кластер використовується файлом і не є останнім кластером файлу; у цьому випадку індексний покажчик містить номер наступного кластера файлу;
- останній кластер файлу;
- дефектний кластер;
- резервний кластер.

Таблиця FAT є загальною для всіх файлів розділу. У вихідному стані (після форматування) всі кластери розділу вільні і всі індексні покажчики (крім тих, які відповідають резервним і дефектним блокам) приймають значення "кластер вільний". При розміщенні файлу ОС переглядає FAT, починаючи з початку, і шукає перший вільний індексний покажчик. Після його виявлення у полі запису каталогу "номер першого кластера" фіксується номер цього покажчика. У кластер із цим номером записуються дані файлу, він стає першим кластером файлу. Якщо файл уміщається в одному кластері, то в покажчик, що відповідає даному кластеру, заноситься спеціальне значення "останній кластер файлу". Якщо ж розмір файлу більше одного кластера, то ОС продовжує перегляд FAT і шукає наступний покажчик на вільний кластер. Після його виявлення в попередній покажчик заноситься номер цього кластера, який тепер стає наступним кластером файлу. Процес повторюється доти, поки не будуть розміщені всі дані файлу. У такий спосіб створюється зв'язний список всіх кластерів файлу.

Досить наочно ідея файлової системи з використанням таблиці розміщення файлів FAT проілюстрована на рис. 1.71. Видно, що файл з ім'ям **MYFILE.TXT** розміщається, починаючи з восьмого кластеру. Усього файл **MYFILE.TXT** займає 12 кластерів. Ланцюжок кластерів для нашого прикладу може бути записаний у такий спосіб; 8, 9, 0A, 0У, 15, 16, 17, 19, 1A, 1У, 1C, 1D. Кластер з номером 18 позначений спеціальним кодом **F7** як поганий (**bad**), він не може бути використаний для розміщення даних. При форматуванні звичайно перевіряється поверхня магнітного диска, і ті сектори, при контрольному читанні з яких відбувалися помилки, позначаються в FAT як погані. Кластер **1D** позначений кодом FF як кінцевий (останній у ланцюжку) кластер, що належить даному файлу. Вільні (незайняті) кластери позначаються кодом 00; при виділенні нового кластера для запису файлу береться перший вільний кластер. Оскільки файли на диску змінюються – віддаляються, переміщаються, збільшуються або зменшуються, – то згадане правило виділення першого вільного кластера для нової

порції даних приводить до фрагментації файлів, тобто дані одного файлу можуть розташовуватися не в суміжних кластерах, а в дуже віддалених один від олного, формуючи складні ланцюжки. Природно, що це приводить до істотної затримки при роботі з файлами.

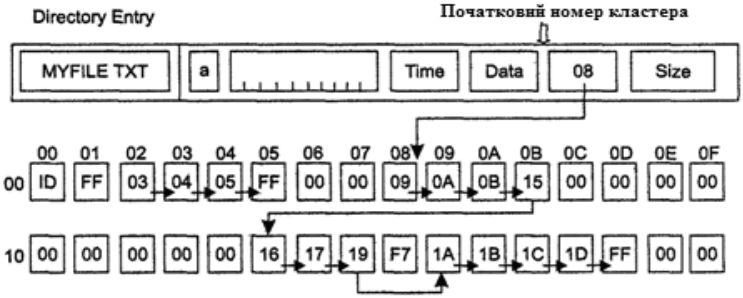



Рисунок 1.71. Основна концепція FAT

Розмір таблиці FAT і розрядність використовуваних у ній індексних покажчиків визначається кількістю кластерів в області даних. Для зменшення втрат через фрагментацію бажано кластери робити невеликими, а для скорочення обсягу адресної інформації і підвищення швидкості обміну навпаки – чим більше, тим краще. При форматуванні диску під файлову систему FAT звичайно вибирається компромісне рішення і розміри кластерів вибираються з діапазону від 1 до 128 секторів, або від 512 байт до 64 Кбайт.

Очевидно, що розрядність індексного покажчика повинна бути такою, щоб у ньому можна було задати максимальний номер кластера для диску певного обсягу. Існує кілька різновидів FAT, які відрізняються розрядністю індексних покажчиків, що і використовується як умовне позначення: FAT12, FAT16 і FAT32. У файловій системі FAT12 використовуються 12-розрядні покажчики, що дозволяє підтримувати до 4096 кластерів в області даних диска, в FAT16 – 16-розрядні покажчики для 65536 кластерів і в FAT32 – 32-розрядні для більш ніж 4 мільярдів кластерів.

Форматування FAT12 звичайно характерно тільки для невеликих дисків обсягом не більше 16 Мбайт, щоб не використовувати кластери більше 4 Кбайт. З цієї ж причини вважають, що FAT16 доцільніше для дисків з обсягом не більше 512 Мбайт, а для більших дисків краще підходить FAT32, що здатна використовувати кластери 4 Кбайт при роботі з дисками обсягом до 8 Гбайт і тільки для дисків більшого обсягу використовують 8, 16 і 32 Кбайт. Максимальний розмір розділу FAT16 обмежений 4 Гбайт, такий обсяг дає 65536 кластерів по 64 Кбайт кожний, а максимальний розмір розділу FAT32 практично не обмежений – $2^{32}$ кластерів по 32 Кбайт.

Таблиця FAT при фіксованій розрядності індексних покажчиків має змінний розмір, що залежить від обсягу області даних диска.

При видаленні файлу з файлової системи FAT у перший байт відповідного запису каталогу заноситься спеціальна ознака, що свідчить про те, що цей запис вільний, а в усі індексні покажчики файлу заноситься ознака "кластер вільний". Інші дані в запису каталогу, у тому числі номер першого кластера файлу, залишаються недоторканими, що залишає шанси для відновлення помилково вилученого файлу. Існує велика кількість утиліт для відновлення вилучених файлів FAT, що виводять користувачеві список імен вилучених файлів з відсутнім першим символом ім'я, затертим після звільнення запису. Очевидно, що надійно можна відновити тільки файли, які були розташовані в послідовних кластерах диску, оскільки при відсутності зв'язного списку виявити приналежність довільно розташованого кластера вилученому файлу неможливо.

Резервна копія FAT завжди синхронізується з основною копією при будь-яких операціях з файлами, тому резервну копію не можна використовувати для скасування помилкових дій користувача, що виглядали з погляду системи цілком коректними. Резервна копія може бути корисна тільки в тому випадку, коли сектори основної пам'яті виявляються фізично ушкодженими і не читаються.

Використовуваний у FAT метод зберігання адресної інформації про файли не відрізняється великою надійністю – при розриві списку індексних покажчиків в одному місці, наприклад через збій у роботі програмного коду ОС через зовнішні електромагнітні перешкоди, губиться інформація про всі наступні кластери файлу.

Файлові системи FAT12 і FAT16 оперували з іменами файлів, що складаються з 12 символів за схемою "8.3". У версії FAT16 операційної системи Windows NT був введений новий тип запису каталогу – "довге ім'я", що дозволяє використовувати імена довжиною до 255 символів, причому кожний символ довгого ім'я зберігається у двухбайтному форматі Unicode. Ім'я за схемою "8.3", назване тепер коротким (не потрібно плутати його з простим ім'ям файлу, яке іноді називають коротким), як і раніше зберігається у 12-байтовому полі імені файлу в запису каталогу, а довге ім'я міститься порціями по 13 символів в одному або кількох записах, що випливають безпосередньо за основним записом каталогу. Кожний символ у форматі Unicode кодується двома байтами, тому 13 символів займають 26 байт, а ще залишилися 6 відведені під службову інформацію. Таким чином, у файлу з'являються два імені – коротке, для сумісності із старими додатками, що не розуміють довгих імен в Unicode, і довге, зручне у використанні ім'я. Файлова система FAT32 також підтримує короткі й довгі імена.

Файлові системи FAT12 і FAT16 мають велике поширення завдяки їхньому застосуванню в операційних системах MS-DOS і Windows 3.x – са-

мих масових операційних системах першого десятиліття ери персональних комп'ютерів. З цієї причини ці файлові системи підтримуються сьогодні й іншими ОС, такими, як UNIX, OS/2, Windows NT/2000 і Windows 95/98. Однак через постійно зростаючі обсяги жорстких дисків, а також зростаючі вимоги до надійності ці файлові системи швидко витісняються як системою FAT32, що вперше з'явилася в Windows 95 OSR2, так і файловими системами інших типів.

**Фізична організація s5 і ufs.**

Файлові системи **s5** (отримали назву від System V, родового імені декількох версій ОС UNIX, розроблених в Bell Labs компанії AT&T) і **ufs** (**UNIX File System**) використовують дуже близьку фізичну модель. Це не дивно, тому що система ufs є розвитком системи **s5**. Файлова система **ufs** розширює можливості **s5** з підтримки великих дисків і файлів, а також підвищує її надійність.

Розташування файлової системи **s5** на диску ілюструє рис. 1.72. Розділ диску, де розміщується файлова система, ділиться на чотири області:

- ✧ *завантажувальний блок*;
- ✧ *суперблок* (**superblock**) містить загальну інформацію про файлову систему: розмір файлової системи, розмір області індексних дескрипторів, число індексних дескрипторів, список вільних блоків і список вільних індексних дескрипторів, а також іншу адміністративну інформацію;
- ✧ *область індексних дескрипторів* (**mode list**), порядок розташування індексних дескрипторів у якій відповідає їхнім номерам;
- ✧ *область даних*, у якій розташовані як звичайні файли, так і файли-каталоги, у тому числі й кореневий каталог; спеціальні файли представлені у файловій системі тільки записами у відповідних каталогах і індексних дескрипторах спеціального формату, але місця в області даних не займають.

Основною особливістю фізичної організації файлової системи **s5** є відділення імені файлу від його характеристик, що зберігаються в окремій структурі, яку називають *індексним дескриптором* (**inode**). Індексний дескриптор в **s5** має розмір 64 байти і містить дані про тип файлу, адресну інформацію, привілеї доступу до файлу і деяку іншу інформацію, а саме:

- ✧ ідентифікатор власника файлу;
- ✧ тип файлу; файл може бути файлом звичайного типу, каталогом, спеціальним файлом, а також конвеєром або символьним зв'язком;
- ✧ права доступу до файлу;
- ✧ тимчасові характеристики: час останньої модифікації файлу, час останнього звертання до файлу, час останньої модифікації індексного дескриптора;

- ✧ число посилань на даний індексний дескриптор, дорівнює кількості псевдонімів файлу;
- ✧ адресна інформація;
- ✧ розмір файлу в байтах.

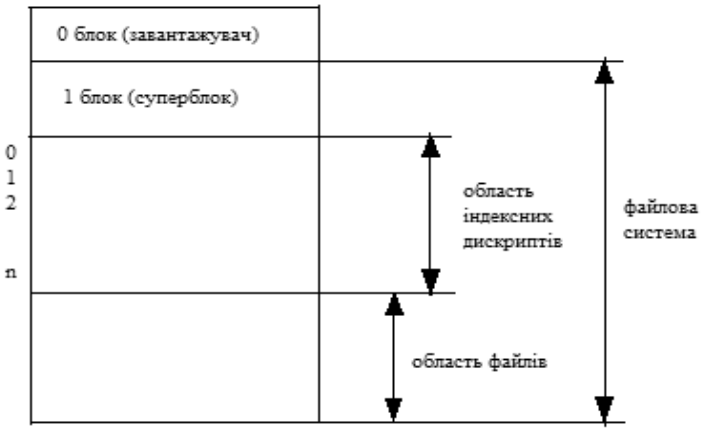



Рисунок 1.72. Розташування файлової системи s5 на диску

Кожний індексний дескриптор має номер, що одночасно є унікальним ім'ям файлу. Індексні дескриптори розташовані в особливій області диску в строгій відповідності до своїх номерів. Відповідність між повними символьними іменами файлів і їхніми унікальними іменами установлюється за допомогою ієрархії каталогів. Система веде список номерів вільних індексних дескрипторів. При створенні файлу йому виділяється номер із цього списку, а при знищенні файлу номер його індексного дескриптора повертається в список.

Запис про файл у каталозі складається всього із двох полів: символьного імені файлу і номера індексного дескриптора. Наприклад, на рис. 1.73 показана інформація, що міститься в каталозі **/user**.

| Ім'я файлу | № індексного дескриптора |
|---|---|
| Prog 1 | 23 |
| firolights | 126 |
| doc_23 txt | 51 |
| glaring txt | 17 |
| lambada_good | |
| | |
| | |

Рисунок 1.73. Структура каталогу у файловій системі s5

Файлова система не накладає особливих обмежень на розмір кореневого каталогу, оскільки він розташований в області даних і може збільшуватися як звичайний файл.

Доступ до файлу здійснюється шляхом послідовного перегляду всього ланцюжка каталогів, що входять у повне ім'я файлу, і відповідних їм індексних дескрипторів. Пошук завершується після одержання всіх характеристик з індексного дескриптора заданого файлу. Розглянемо цю процедуру на прикладі файлу **/bin/my_shell/print**, що входить до складу файлової системи, зображеної на рис. 1.74.

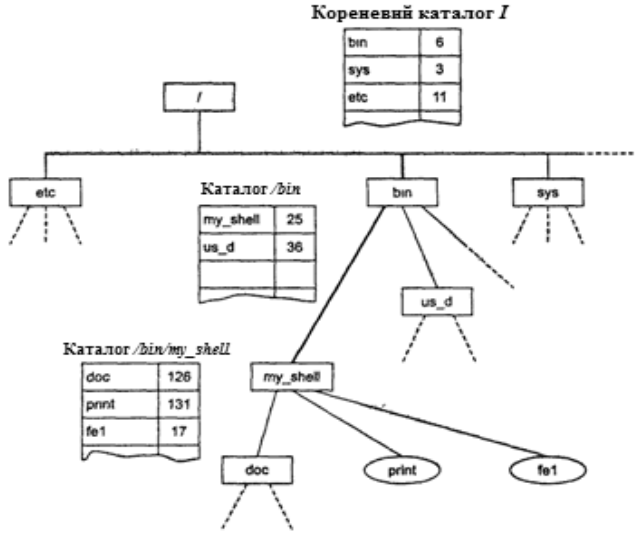



Рисунок 1.74. Пошук адреси файлу за його символьним ім'ям

Визначення фізичної адреси цього файлу включає наступні етапи:

1. Насамперед проглядається кореневий каталог з метою пошуку першої складової символьного імені – **bin**. Визначається номер (у даному прикладі – 6) індексного дескриптора каталогу, що входить у кореневий каталог. Адреса кореневого каталогу відома системі.
2. З області індексних дескрипторів зчитується дескриптор з номером 6. Початкова адреса дескриптора визначається на підставі відомих системі номера початкового сектора області індексних дескрипторів і розміру індексного дескриптора. З індексного дескриптора 6 визначається фізична адреса каталогу **/bin**.
3. Проглядається каталог **/bin** з метою пошуку другої складової символьного імені **my_shell**. Визначається номер індексного дескриптора каталогу **/bin/my_shell** (у нашому випадку – 25).

4. Зчитується індексний дескриптор: 25, визначається фізична адреса **/bin/my_shell**.
5. Проглядається каталог **/bin/my_shell**, визначається номер індексного дескриптора файлу **print** (у нашому випадку – 131).
6. З індексного дескриптора 131 визначаються номери блоків даних, а також інші характеристики файлу **/bin/my_shell/print**.

Ця процедура вимагає в загальному випадку декількох звертань до диска, пропорційно числу складових у повному імені файлу. Для зменшення середнього часу доступу до файлу його дескриптор копіюється в спеціальну системну область оперативної пам'яті. Копіювання індексного дескриптора входить у процедуру відкриття файлу.

Фізична організація файлової системи ufs відрізняється від описаної фізичної організації файлової системи **s5** тим, що розділ складається з повторюваної кілька разів послідовності областей " завантажник-суперблок-блок групи циліндрів-область індексних дескрипторів" (рис. 1.75).

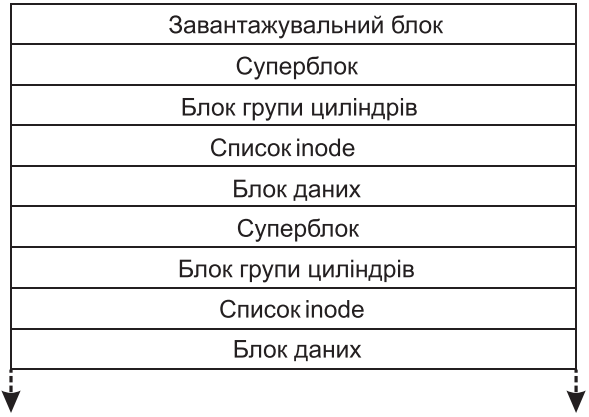



Рисунок 1.75. Фізична організація файлової системи ufs

У цих повторюваних послідовностях областей суперблок є резервною копією основної першої копії суперблоку. При ушкодженні основної копії суперблоку може бути використана резервна копія суперблоку. Області ж блоку групи циліндрів і індексних дескрипторів містять індивідуальні для кожної послідовності значення. Блок групи циліндрів описує кількість індексних дескрипторів і блоків даних, розташованих на даній групі циліндрів диска. Таке угруповання робиться для прискорення доступу, щоб перегляд індексних дескрипторів і даних файлів, описуваних цими дескрипторами, не приводив до занадто великих переміщень головок диску.

Крім того, в **ufs** імена файлів можуть мати довжину до 255 символів (кодування **ASCII**, по одному байту на символ), у той час як в **s5** довжина ім'я не може перевищувати 14 символів.

**Фізична організація NTFS**

Файлова система NTFS була розроблена як основна файлова система для ОС Windows NT на початку 90-х років з урахуванням досвіду розробки файлових систем FAT і HPFS (основна файлова система для OS/2), а також інших існуючих у той час файлових систем. Основними відмітними властивостями NTFS є:

- ✧ підтримка великих файлів і великих дисків обсягом до $2^{64}$ байт;
- ✧ відновлюваємість після збоїв і відмов програм і апаратури керування дисками;
- ✧ висока швидкість операцій, у тому числі й для великих дисків;
- ✧ низький рівень фрагментації, у тому числі й для великих дисків;
- ✧ гнучка структура, що допускає розвиток за рахунок додавання нових типів записів і атрибутів файлів із збереженням сумісності з попередніми версіями ФС;
- ✧ стійкість до відмов дискових накопичувачів;
- ✧ підтримка довгих символьних імен;
- ✧ контроль доступу до каталогів і окремих файлів.

***Структура тому NTFS***. На відміну від розділів FAT і **s5**/**ufs** весь простір тому NTFS є є або файлом, або частиною файлу. Основою структури тому NTFS є *головна таблиця файлів* (**Master File Table**, **MFT**), що містить принаймні один запис для кожного файлу тому, включаючи один запис для самої себе. Кожний запис MFT має фіксовану довжину, що залежить від обсягу диску, – 1, 2 або 4 Кбайт.

Усі файли на томі NTFS ідентифікуються номером файлу, що визначається позицією файлу в MFT. Цей спосіб ідентифікації файлу близький до способу, використовуваному у файлових системах **s5** і **ufs**, де файл однозначно ідентифікується номером його запису в області індексних дескрипторів.

Весь том NTFS складається з послідовності кластерів, що відрізняє цю файлову систему від розглянутих раніше, де на кластери ділилася тільки область даних. Порядковий номер кластера в томі NTFS називається *логічним номером кластера* (**Logical Cluster Number**, **LCN**). Файл NTFS також складається з послідовності кластерів, при цьому порядковий номер кластера усередині файлу називається *віртуальним номером кластера* (**Virtual Cluster Number**, **VCN**).

Базова одиниця розподілу дискового простору для файлової системи NTFS – неперервна область кластерів, яку називають *відрізком*. Як адресу відрізка NTFS використовує логічний номер його першого кластера, а

також кількість кластерів у відрізку **k**, тобто пари (LCN, k). Таким чином, частина файлу, поміщена у відрізок **i** починається з віртуального кластера VCN, характеризується адресою, що складається із трьох чисел (VCN, LCN, k).

Для зберігання номера кластера в NTFS використовуються 64-розрядні покажчики, що дає можливість підтримувати томи й файли розміром до $2^{64}$ кластерів. При розмірі кластера в 4 Кбайт це дозволяє використовувати томи й файли, що складаються з 64 мільярдів кілобайт.

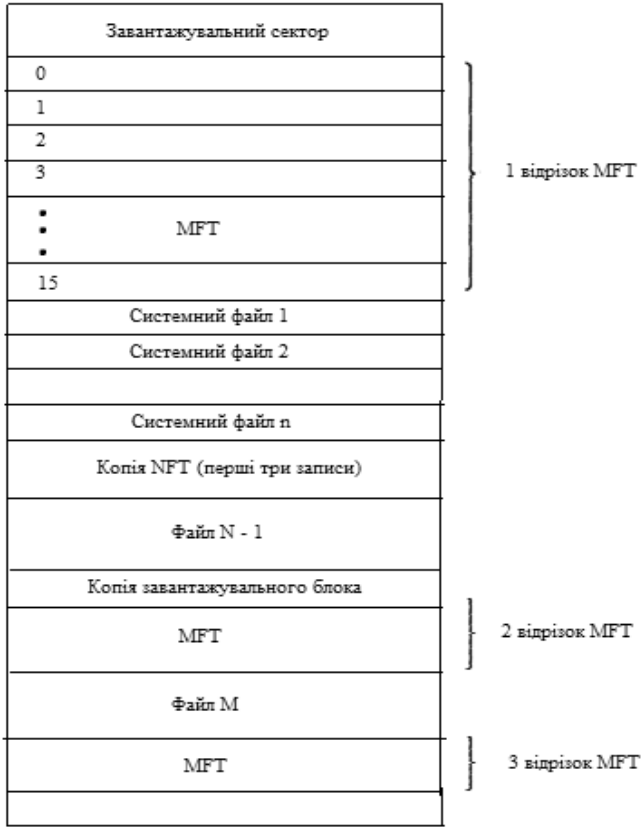



Рисунок 1.76. Структура тому NTFS

Структура тому NTFS показана на рис. 1.76. Завантажувальний блок тому NTFS розташовується на початку тому, а його копія – у середині тому. Завантажувальний блок містить стандартний блок параметрів BIOS, кількість блоків у томі, а також початковий логічний номер кластера основної копії MFT і дзеркальну копію MFT.

Далі розташовується перший відрізок MFT, що містить 16 стандартних, створюваних при форматуванні записів про системні файли, NTFS.

В NTFS файл цілком розміщається в запису таблиці MFT, якщо це дозволяє зробити його розмір. У тому ж випадку, коли розмір файлу більше розміру запису MFT, у запису розміщують тільки деякі атрибути файлу, а інша частина файлу розміщається в окремому відрізку тому (або декількох відрізках). Частина файлу, розташовувана в запису MFT, називається *резидентною* частиною, а інші частини – *нерезидентними*. Адресна інформація про відрізки, що містять нерезидентні частини файлу, розміщається в атрибутах резидентної частини.

Деякі системні файли є повністю резидентними, а деякі мають і нерезидентні частини, які розташовуються після першого відрізка MFT.

Нульовий запис MFT містить опис самої MFT, у тому числі й такий її важливий атрибут, як адреси всіх її відрізків. Після форматування MFT складається з одного відрізка, але після створення першого ж несистемного файлу для зберігання його атрибутів потрібно ще один відрізок, тому що споконвічно безперервна послідовність кластерів MFT уже завершена системними файлами.

З наведеного опису видно, що сама таблиця MFT розглядається як файл, до якого застосуємо метод розміщення в томі у вигляді набору довільно розташованих декількох відрізків.

***Структура файлів NTFS***. Кожний файл і каталог у томі NTFS складається з набору атрибутів. Ім'я файлу і його дані також розглядаються як атрибути файлу, тобто в трактуванні NTFS, крім атрибутів у файлу немає ніяких інших компонентів.

Кожний атрибут файлу NTFS складається з полів: тип атрибута, довжина атрибута, значення атрибута і, можливо, ім'я атрибута. Тип атрибута, довжина та ім'я утворять заголовок атрибута.

Є системний набір атрибутів, обумовлених структурою тому NTFS. Системні атрибути мають фіксовані імена і коди їхнього типу, а також певний формат. Можуть застосовуватися також атрибути, обумовлені користувачами. Їхні імена, типи й формати задаються винятково користувачем. Атрибути файлів упорядковані за зменшенням коду атрибута, причому атрибут того самого типу може повторюватися кілька разів. Існують два способи зберігання атрибутів файлу – резидентне зберігання в записах таблиці MFT і нерезидентне зберігання поза нею, у зовнішніх відрізках. Таким чином, резидентна частина файлу складається з резидентних атрибутів, а нерезидентна – з нерезидентних атрибутів. Сортування може здійснюватися тільки за резидентними атрибутами.

Перелічимо деякі атрибути із системного набору:

- ***Attribute List*** *(список атрибутів)* – список атрибутів, з яких складається файл; містить посилання на номер запису MFT, де

розташований кожний атрибут; цей рідко використовуваний атрибут потрібний тільки в тому випадку, якщо атрибути файлу не вміщаються в основному запису і займають додаткові записи MFT;

- ***File Name*** *(ім'я файлу)* – цей атрибут містить довге ім'я файлу у форматі Unicode, а також номер входу в таблиці MFT для батьківського каталогу; якщо цей файл утримується в декількох каталогах, то в нього буде кілька атрибутів типу File Name*;* цей атрибут завжди повинен бути резидентним;
- ***DOS Name*** *(ім'я* MS-DOS*)* – цей атрибут містить ім'я файлу у форматі 8.3;
- ***Security Descriptor*** *(дескриптор безпеки)* – цей атрибут містить інформацію про захист файлу: список прав доступу і поле аудиту, що визначає, якого роду операції над цим файлом потрібно реєструвати;
- ***Data*** *(дані)* – містить звичайні дані файлу;
- ***MFT bitmap*** *(бітова карта* MFT*)* – цей атрибут містить карту використання блоків у томі;
- **Standard Information** *(стандартна інформація)* – цей атрибут зберігає всю іншу стандартну інформацію про файл, що важко зв'язати з яким-небудь із інших атрибутів файлу, наприклад, час створення файлу, час відновлення та ін.

Файли NTFS залежно від способу розміщення діляться на невеликі, великі, дуже великі і надвеликі.

*Невеликі файли* (**small**) – якщо файл має невеликий розмір, то він може цілком розташовуватися усередині одного запису **MFT**, що має, наприклад, розмір 2 Кбайт. Невеликі файли **NTFS** складаються принаймні з наступних атрибутів (рис. 1.77):

- стандартна інформація (**SI – standard information**);
- ім'я файлу (**FN – file name**);
- дані (**Data**);
- дескриптор безпеки (**SD-Security descriptor**).

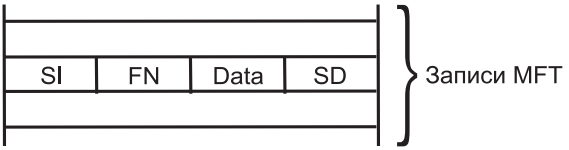



Рисунок 1.77. Невеликий файл NTFS

*Великі файли* (**large**) – якщо дані файлу не вміщуються в один запис **MFT**, то цей факт відбивається в заголовку атрибута **Data**, що містить ознаку того, що цей атрибут є нерезидентним, тобто перебуває у відрізках поза

таблицею MFT. У цьому випадку атрибут **Data** містить адресну інформацію (LCN, VCN, k) кожного відрізка даних (рис. 1.78).

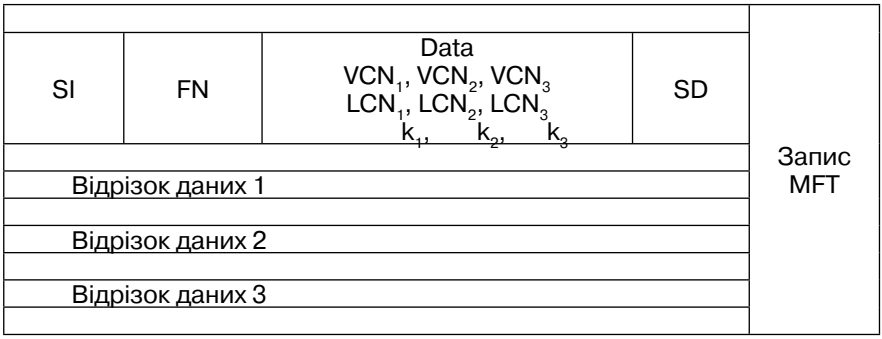



Рисунок 1.78. Великий файл

*Надвеликі файли* (**extremely huge**) – для надвеликих файлів в атрибуті **Attribute List** можна вказати кілька атрибутів, розташованих у додаткових записах MFT (рис. 1.79). Крім того, можна використовувати подвійну непряму адресацію, коли нерезидентний атрибут буде посилатися на інші нерезидентні атрибути, тому в NTFS не може бути атрибутів занадто великої для системи довжини.

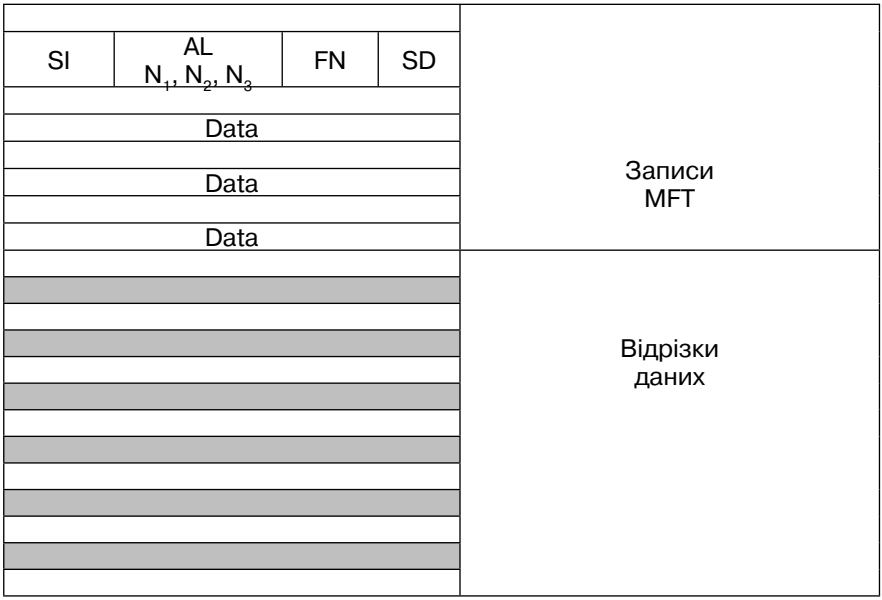



Рисунок 1.79. Надвеликий файл

*Дуже великі файли* (**huge**). Якщо файл настільки великий, що його атрибут даних, що зберігає адреси нерезидентних відрізків даних, не вміщуються в одному запису, то цей атрибут розміщують в інший запис MFT, а посилання на такий атрибут розміщують в основний запис файлу (рис. 1.80). Це посилання міститься в атрибуті **Attribute List**. Сам атрибут даних як і раніше містить адреси нерезидентних відрізків даних.

<table>
<tr><td colspan="4"></td><td rowspan="12">Записи MFT</td></tr>
<tr><td>SI</td><td>AL<br>N=106</td><td>FN</td><td>SD</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="4">$VCN_1$, $LCN_1$, $k_1$; $VCN_2$, $LCN_2$, $k_2$; $VCN_3$, $LCN_3$, $k_3$; $VCN_m$, $LCN_m$, $k_m$<br>Data...</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="4">Відрізок даних 1</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="4">Відрізок даних 2</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="4">Відрізок даних 3</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="4">Відрізок даних m</td></tr>
<tr><td colspan="4"></td></tr>
</table>

Рисунок 1.80. Дуже великий файл

## КОНТРОЛЬНІ ПИТАННЯ

1. Розкажіть про інформаційні революції в історії розвитку цивілізації.
2. Зіставте процеси, що відбуваються в історії розвитку ЕОМ, з останньою інформаційною революцією.
3. Як ви собі уявляєте інформаційне суспільство?
4. У чому полягає процес інформатизації?
5. Як і для чого з'явилася інформатика?
6. Розкажіть про інформатику як про галузь, як про науку, як про прикладну дисципліну. Опишіть цілі і завдання.
7. Дайте визначення терміну "інформація". Які підходи у визначенні цього терміну вам відомі?
8. Чим інформація відрізняється від даних? Як пов'язано між собою ці два поняття?
9. Що таке адекватність інформації"? Які види адекватності ви знаєте? Коротко опишіть їх.
10. Яким чином вимірюють інформацію? Які заходи інформації вам відомі?
11. Як пов'язані між собою кількість інформації і міра невизначеності стану системи?
12. Чим визначається семантична міра інформації? Що таке "тезаурус"? Яким відношенням пов'язані між собою об'єм тезауруса і об'єм сприйманої інформації?
13. Що таке "прагматична міра інформації"?
14. Якими поняттями визначається якість інформації?
15. Що таке "інформаційна система"?
16. Намалюйте схему передачі інформації і поясните призначення її елементів.
17. Що таке "кодування" і "декодування"?
18. Які типи шифрування вам відомі?
19. Що таке "класифікація"? Що таке "реквізит", "класифікатор"?
20. Які методи класифікації вам відомі?
21. У чому полягає фасетный принцип класифікації?
22. У яких випадках застосовується дескрипторний метод класифікації?
23. За якими ознаками можна класифікувати інформацію?
24. Які системи кодування інформації застосовуються при класифікації?
25. Що таке "інформаційна система"?
26. Як ви розумієте інформаційну технологію?
27. У чому полягає різниця між комп'ютерами і інформаційними системами?
28. Як розвивалися інформаційні системи?
29. Розкажіть про піраміду рівнів управління у фірмі.

30. Чому при створенні інформаційної системи слід враховувати вплив зовнішнього середовища?
31. Наведіть приклади інформаційних систем, що підтримують діяльність фірми.
32. Які завдання стоять при створенні інформаційної системи?
33. Розкажіть про основні функціональні інформаційні системи.
34. Наведіть приклади інформаційних систем, що забезпечують ефективність роботи.
35. Як ви представляєте структуру інформаційної системи?
36. Розкажіть про піраміду інформаційних систем у фірмі, де використовується функціональна ознака класифікації.
37. Які роль і функції ІС оперативного рівня, ІС для фахівців, для менеджерів середньої ланки, стратегічних ІС?
38. Приведіть класифікацію інформаційних систем по характеру використання інформації, по мірі автоматизації, по сфері застосування.
39. У чому схожість і в чому відмінність інформаційної технології і технології матеріального виробництва?
40. Відобразите інформаційну технологію у вигляді ієрархічної структури і наведіть приклади її складових.
41. Викладете вимоги, яким повинна відповідати інформаційна технологія.
42. Як співвідносяться інформаційна технологія і інформаційна система?
43. Охарактеризуйте методологію використання інформаційної технології.
44. Дайте загальне уявлення про інформаційні технології обробки даних, управління, автоматизації офісу, підтримки ухвалення рішень. Назвіть їх основні компоненти.
45. З яких елементів складається процесор комп'ютера?
46. Які типи моніторів вам відомі?
47. Які типи принтерів вам відомі?
48. Які способи збільшення продуктивності роботи центрального процесора?
49. Яке призначення материнської плати?
50. Що таке "чіпсет"?
51. Перерахуєте усі відомі вам пристрої для введення даних в комп'ютер.
52. Перерахуєте відомі вам пристрої для виведення даних.
53. Які типи пам'яті вам відомі?
54. Чи являються поняття "інформація" і "дані" синонімами? Дайте визначення тому і іншому поняттю.
55. Чи Вірне висловлювання: "інформація в комп'ютері завжди представлена у бінарному виді"?
56. У якому виді зберігаються в комп'ютері негативні цілі числа?
57. У якому виді зберігаються в комп'ютері дійсні числа?
58. Які проблеми можуть виникнути при збереженні результатів деяких арифметичних операцій?

59. Як вирішується проблема наявності різних алфавітів в різних мовах при кодуванні і зберіганні символьної інформації?
60. Що таке "ASCII"?
61. Дайте визначення комп'ютеру.
62. Які основні етапи в розвитку обчислювальної техніки?
63. Як виглядає машина Т'юринга? Чи була вона виготовлена?
64. Як влаштована машина фон Неймана?
65. Перерахуєте основні покоління комп'ютерів.
66. Які основні ознаки четвертого покоління комп'ютерів?
67. Які види класифікації комп'ютерів ви знаєте?
68. Як комп'ютери можна класифікувати за призначенням?
69. Що таке "мейнфрейм"?
70. Які тенденції і шляхи розвитку мікропроцесорів?
71. Назвіть основні відмінності ноутбука від КПК.
72. Які перспективи розвитку комп'ютерів і обчислювальних систем?
73. Що таке "система BIOS", які її функції?
74. Яку роль в операційній системі грає ядро?
75. Що таке "драйвер"?
76. Чим служби (сервіси) операційної системи відрізняються від прикладних програм?
77. Для чого потрібні системні оболонки?
78. Що таке "операційна система реального часу"?
79. Які мають бути функції операційної системи, якщо розглядати її як набір необхідних функцій?
80. За допомогою якого інструменту робиться налаштування параметрів безпеки в Windows XP?
81. Які застосування поставляються у складі Windows XP?
82. Що таке "мікшування"?
83. За допомогою яких програм робиться управління звуковими каналами?
84. Які програми для редагування оцифрованого звуку вам відомі?
85. Які програми для створення MIDI-композиций ви знаєте?
86. У чому відмінність растрового формату від векторного?
87. Чи є надмірність в цифровому коді, що представляє зображення?
88. За яким принципом здійснюється стискування зображення?
89. Які програми для створення і редагування векторних зображень вам відомі?
90. Які програми для створення і редагування растрових зображень вам відомі?

## ТЕСТИ ДЛЯ САМОКОНТРОЛЮ

1. Як давно з'явився термін "інформатика"?
   - ✧ В XIX сторіччі
   - ✧ На початку XX сторіччя
   - ✧ В 60-х роках XX сторіччя

2. Скільки інформаційних революцій спостерігалось на протязі існування людства?
   - ✧ Одна
   - ✧ Дві
   - ✧ Три
   - ✧ Чотири
   - ✧ П'ять

3. Що характеризує поняття "економічна інформація"?
   - ✧ виробничі відносини у суспільстві
   - ✧ рівень заробітної плати оператора ПК
   - ✧ чисельність населення країни
   - ✧ рівень освіти громадян окремих країн

4. Як в інформатиці називається найменша одиниця вимірювання даних?
   - ✧ Binary digit
   - ✧ Біт
   - ✧ Байт
   - ✧ Число

5. Існують дві принципово різні системи кодування економічної інформації – класифікаційна та реєстраційна. До якої з них відноситься порядкова система кодування?
   - ✧ До реєстраційної
   - ✧ До классифікаційної
   - ✧ Такої системи не існує
   - ✧ До обох

6. Що таке система числення?

   Це правила обробки чисел за допомогою персонального комп'ютера

   Це спосіб назв і зображень чисел за допомогою символів, які мають певні кількісні значення

   Це певний алгоритм, виконання якого дозволить отримати результат обчислення

7. В якому діапазоні лежать значення цифр у шестирічній системі числення?

✧ Від 1 до 6
✧ Від 0 до 5
✧ Такої системи взагалі не існує
✧ Тільки 1 і 6
✧ Тільки 0 і 6

8. Скільки значень кодів закріплено в таблиці кодування ASCII?

128
256
1024

9. Модем... (відмітити правильні твердження)

✧ це пристрій, що конвертує цифрові сигнали в аналогові
✧ це скручена пара дротів
✧ це сукупність формальних правил та методів, які визначають обмін даними між сполученими обчислювальними ресурсами
✧ це супутниковий зв'язок

10. Тип файлу характеризується (вибрати чим саме):

✧ розширенням файлу
✧ ім'ям файлу
✧ повним шляхом до файлу
✧ розміром файлу
✧ маю власний варіант відповіді

11. До основних складових базової (стандартної) конфігурації сучасного персонального комп'ютера відносять:

✧ системний блок, клавіатуру, мишу, дисплей
✧ системний блок, клавіатуру, монітор
✧ системний блок, клавіатуру, мишу
✧ процесор, монітор, клавіатуру, мишу, принтер

12. Звукова карта персонального комп'ютера – це

✧ пристрій, який виконує обчислювальні операції, пов'язані з обробкою звуку зовнішні звукові колонки
✧ сукупність мікрофона, звукових колонок та інших пристроїв для відтворення звуку
✧ пристрій, за допомогою якого здійснюється показ відеопродукції

13. Яка максимальна довжина назви файлів у Windows?
    - ✧ У Windows обмежень на довжину файла немає
    - ✧ 8 символів
    - ✧ 16 символів
    - ✧ 255 символів

14. Яка організація управляє стандартом Unicode?
    - ✧ ISO (International Organization for Standardization)
    - ✧ IETF (Internet Engineering Task Force)
    - ✧ Unicode Consortium
    - ✧ ANSI (American National Standards Institute)

# Розділ 2.
# МЕРЕЖНІ ТЕХНОЛОГІЇ ТА ОРГАНІЗАЦІЯ КОМП'ЮТЕРНОЇ БЕЗПЕКИ, ЗАХИСТУ ІНФОРМАЦІЇ В ЕКОНОМІЦІ

## 2.1 Основні поняття комп'ютерних мереж

### 2.1.1 Поняття та призначення комп'ютерної мережі

На всіх етапах розвитку людство для вирішення будь-яких проблем використовувало інформацію. Інформація завжди виступає носієм знань, засоби зв'язку – каналами її передачі, а обчислювальна техніка – засобом обробки та подання її в обраній формі. Створення комп'ютерних мереж привело до появи нових форм інформаційного простору як окремих ланок суспільства, так і людства в цілому. Поняття інформаційного простору не є щось новим. Інформаційний простір існує з моменту спілкування людей. Тільки історично відповідно до розвитку людства змінювались якісні та кількісні показники цього простору. Так само і вплив показників інформаційного простору на якість діяльності суспільства невпинно зростав. Натепер рівень впливу досяг такого значення, що нехтування ним веде до спроби затримання об'єктивного розвитку суспільства.

Під єдиним інформаційним простором тут і далі розуміють сукупність банків даних, технологій їх ведення та використання, інформаційних телекомунікаційних систем та мереж, функціонуючих на основі єдиних принципів та технологій, які забезпечують інформаційну взаємодію в межах їх можливостей для задоволення споживачів цього простору.

Для створення єдиного інформаційного простору країни необхідно вирішити низку проблем:

- створення єдиної інформаційної структури;
- функціональне об'єднання сукупності всіх інформаційних ресурсів;
- забезпечення всіх користувачів засобами інформаційної взаємодії;
- введення в дію державної організаційно-правової бази, яку складає система законодавчих, виконавчих та організаційних документів та заходів для встановлення стандартів, протоколів і правил використання інформаційного простору суспільства;
- організація загальної системи наукової, конструкторсько-дослідницької роботи з розвитку інформаційного простору;

- створення системи професійної підготовки фахівців для ефективного використання інформаційного простору країни.

Комплексне вирішення цих проблем і складає поняття процесу інформатизації.

**Телематика** – це нова науково-технічна дисципліна, предметом якої є методи й засоби передачі інформації на відстані, істотно перевищуючі лінійні розміри площі, займаної учасниками зв'язку. Назва дисципліни відбулася із частин слів "телекомунікації" й "інформатика".

Створення сучасної єдиної інформаційної структури базується на впровадженні комп'ютерних мереж.

Бурхливий розвиток засобів обчислювальної техніки, програмних засобів та їх широке впровадження в усі сфери діяльності суспільства привели до необхідності створення комп'ютерних мереж. Це обумовлено такими чинниками:

комп'ютерні мережі дозволяють у короткі терміни часу здійснювати обмін інформацією в значних обсягах між досить віддаленими кореспондентами;

застосування обчислювальної техніки дозволяє створювати та використовувати значні за обсягом бази даних;

організація роботи в мережах сприяє скороченню кількості необхідного коштовного обладнання (принтерів, сканерів та ін.) за рахунок його колективного використання.

**Комп'ютерна мережа** – це система розподіленої обробки інформації, яка є комплексним об'єднанням комп'ютерів засобами передачі даних.

З системної точки зору, таке об'єднання надає нові можливості:

a) в мережах може використовуватись весь наявний парк обчислювальної техніки всіх поколінь.
b) об'єднуються всі ресурси складових комп'ютерної мережі, забезпечуючи підвищення загальної ефективності її роботи.
c) впроваджується принцип колективної роботи всіх користувачів мережі для вирішення проблем в реальному масштабі часу незалежно від територіального розміщення елементів мережі.
d) забезпечується колективне використання єдиного банку даних.
e) комп'ютерні мережі можна застосовувати як автоматизовані системи управління в ієрархічних структурах будь-якого масштабу.

Створення комп'ютерних мереж призводить і до появи нових проблем, без вирішення яких неможливе їх ефективне впровадження та використання:

- значно ускладнюються питання технічного та програмного забезпечення;
- виникає необхідність чіткої організації доступу до інформаційних ресурсів мережі, надійного контролю та їх захисту;

- зростають вимоги до рівня надійності елементів мережі, які визначають якість її функціонування;
- збільшуються вимоги до професійного рівня всіх користувачів мережі;
- постійно ускладнюється інформаційний простір та організація його нормального функціонування в межах суспільства.

**Комунікаційна мережа** – система, що складається з об'єктів, що здійснюють функції генерації, перетворення, зберігання й споживання продукту, називаних пунктами (вузлами) мережі, і ліній передачі (зв'язків, комунікацій, з'єднань), що здійснюють передачу продукту між пунктами.

**Відмінна риса комунікаційної мережі** – більші відстані між пунктами в порівнянні з геометричними розмірами ділянок простору, займаних пунктами. Як продукт можуть фігурувати інформація, енергія, маса, і відповідно розрізняють групи мереж інформаційних, енергетичних, речовинних. У групах мереж можливий поділ на підгрупи. Так, серед речовинних мереж можуть бути виділені мережі транспортні, водопровідні, виробничі й ін. При функціональному проектуванні мереж вирішуються завдання синтезу топології, розподілу продукту по вузлах мережі, а при конструкторському проектуванні виконуються розміщення пунктів у просторі й проведення (трасування) з'єднань.

**Інформаційна мережа** – комунікаційна мережа, у якій продуктом генерування, переробки, зберігання й використання є інформація.

**Обчислювальна мережа** – інформаційна мережа, до складу которой входить обчислювальне встаткування. Компонентами обчислювальної мережі можуть бути ЕОМ і периферійні пристрої, що є джерелами й приймачами даних, переданих по мережі. Ці компоненти становлять ***крайове устаткування даних*** (КУД, або DTE – Data Terminal Equipment). У якості КУД можуть виступати ЕОМ, принтери, плоттери й інше обчислювальне, вимірювальне й виконавче встаткування автоматичних й автоматизованих систем.

**Термінал** – пристрій, призначений для взаємодії користувача з обчислювальною системою або мережею ЕОМ. Складається із пристрою уведення (найчастіше це клавіатура) і одного або декількох пристроїв висновку (дисплей, принтер і т.д.).

Під вузлом будемо розуміти компоненти обчислювальної мережі, що є джерелами й приймачами даних, переданих по мережі.

Властиво пересилання даних відбувається за допомогою середовищ і засобів, поєднуваних за назвою середовище передачі даних. Об'єднання в мережу ЕОМ здійснюється за допомогою кабелів різного типу. Кабелі приєднуються до комп'ютера через спеціальний пристрій, називаний мережною картою (мережним адаптером), що уставляється в слот розширення

на материнській платі комп'ютера. Для побудови мережі використовуються й інші мережні пристрої різної складності.

Підготовка даних, переданого або одержуваних КУД від середовища передачі даних, здійснюється функціональним блоком, званим ***апаратурою закінчення каналу даних*** (АКД, або DCE – Data Circuit-Terminating Equipment). АКД може бути конструктивно окремим або убудованим в КУД блоком. КУД й АКД разом являють собою станцію даних, що часто називають вузлом мережі. Прикладом АКД може служити модем.

### 2.1.2 Класифікація комп'ютерних мереж

Обчислювальні мережі класифікуються по ряду ознак. Залежно від відстаней між зв'язують вузлами, що, розрізняють обчислювальні мережі:

**територіальні** – що охоплюють значний географічний простір; серед територіальних мереж можна виділити мережі регіональні й глобальні, що мають відповідно регіональні або глобальні масштаби; регіональні мережі іноді називають мережами MAN (Metropolitan Area Network), а загальна англомовна назва для територіальних мереж – WAN (Wide Area Network);

**локальні** (ЛОМ) – що охоплюють обмежену територію (звичайно в межах далекості станцій не більше ніж на кілька десятків або сотень метрів друг від друга, рідше на 1...2 км); локальні мережі позначають LAN (Local Area Network);

**корпоративні** (масштабу підприємства) – сукупність зв'язаних між собою ЛОМ, що охоплюють територію, на якій розміщене одне підприємство або установа в одному або декількох близько розташованих будинках.

Особливо виділяють єдину у своєму роді **глобальну** мережу Internet. Це всесвітня комп'ютерна мережа, мережа мереж, що поєднує за допомогою міжмережевих інтерфейсів багато мереж. Internet охоплює американський континент, Європу, Азію. Деякі мережі, що входять до складу Internet, самі по собі великі, інші мають свої підмережі. У цей час мережа Internet поєднує більше 4 млд. комп'ютерів багатьох країн миру й доступна декільком десяткам мільйонів користувачів.

З технічної точки зору Internet – об'єднання транснаціональних комп'ютерних мереж, що працюють по найрізноманітніших протоколах, що зв'язує всілякі типи комп'ютерів, фізично передавальні дані по телефонним дротам й оптоволокну, через супутники й радіомодеми. Координацію мережі здійснює Центр інформаційних мереж при Стенфордском дослідницькому інституті в Менло Парк, Каліфорнія.

Розрізняють інтегровані мережі, неінтегровані мережі й підмережі. Інтегрована обчислювальна мережа (інтермережа) являє собою взаємозалежну сукупність багатьох обчислювальних мереж, які в інтермережі називаються підмережами.

В автоматизованих системах великих підприємств підмережі включають обчислювальні засоби окремих проектних підрозділів. Інтермережі потрібні для об'єднання таких підмереж, а також для об'єднання технічних засобів автоматизованих систем проектування й виробництва в єдину систему комплексної автоматизації. Звичайно інтермережі пристосовані для різних видів зв'язку: телефонії, електронної пошти, передачі відеоінформації, цифрових даних і т.п., і в цьому випадку вони називаються мережами інтегрального обслуговування.

Розвиток інтермереж полягає в розробці засобів сполучення різнорідних підмереж і стандартів для побудови підмереж, споконвічно пристосованих до сполучення.

Підмережі в інтермережах поєднуються відповідно до обраної топології за допомогою блоків взаємодії. ***Топологія*** – геометричне відображення відносин у мережі. Залежно від топології з'єднань вузлів розрізняють мережі шинної (магістральної), кільцевої, зоряної, ієрархічної, довільної структури. Топології мереж будуть розглянуті пізніше

Залежно від способу керування розрізняють мережі:

"клієнт/сервер" – у них виділяється один або кілька вузлів (сервери), що виконують у мережі керуючі або спеціальні обслуговуючі функції, а інші вузли (клієнти) є термінальними, у них працюють користувачі;

однорангові – у них всі вузли рівноправні; оскільки в загальному випадку під клієнтом розуміється об'єкт (пристрій або програма), що запитує деякі послуги, а під сервером – об'єкт, що надає ці послуги, те кожен вузол в однорангових мережах може виконувати функції й клієнта, і сервера.

Нарешті з'явилася мережевоцентрічна концепція, відповідно до якої користувач має лише дешеве встаткування для звертання до вилучених комп'ютерів, а мережа обслуговує замовлення на виконання обчислень й одержання інформації, тобто користувачеві не потрібно здобувати програмне забезпечення для рішення прикладних завдань, йому потрібно лише платити за виконані замовлення. Подібні комп'ютери називають тонкими клієнтами або мережними комп'ютерами.

Залежно від того, однакове або неоднакові ЕОМ застосовують у мережі, розрізняють мережі однотипних ЕОМ, називані однорідними, і різнотипних ЕОМ – неоднорідні (гетерогенні). У великих автоматизованих системах, як правило, мережі виявляються неоднорідними.

Залежно від прав власності на мережі останні можуть бути мережами загального користування (public) або приватная (private). Серед мереж загального користування виділяють телефонні мережі Тфоп (PSTN – Public Switched Telephone Network) і мережі передачі даних (PSDN – Public Switched Data Network).

Мережі також розрізняють залежно від використовуваних у них протоколів і по способах комутації.

### 2.1.3 Архітектура комп'ютерних мереж

Кожна мережа має відповідну архітектуру (рис.2.1), яка відображає прийняті топологію, протоколи, інтерфейси, технічні та програмні засоби.

***Топологія*** мережі розкриває структуру зв'язків між її елементами.

***Протоколи*** встановлюють правила та обмеження функціонування структурних елементів мережі.

***Інтерфейс*** – це система уніфікованих апаратних, конструктивних та програмних засобів, необхідних для організації надійної взаємодії елементів мережі. Основу його складає комплекс електронних і часових параметрів, необхідних управляючих сигналів та команд, набір конструктивних та системотехнічних засобів та рішень, які відображають структуру та порядок обміну інформацією згідно з вибраним протоколом обміну даними (стандарт обміну).

***Технічні засоби*** мережі забезпечують об'єднання комп'ютерів в єдину систему. До них належать сервери, робочі станції, адаптери мережі, концентратори, мости, маршрутизатори, елементи ліній зв'язку і таке інше.

***Програмні засоби*** мережі призначені для управління роботою мережі та забезпечують необхідний інтерфейс з користувачами. До них включають операційні системи мереж та додаткові програмні засоби.

Показники складових архітектури визначають як тип мережі, так і її можливості.




Рисунок 2.1. Архітектура мережі

Структурно будь-яка комп'ютерна мережа складається з робочих станцій (PC), серверів (у серверних мережах), виділених каналів зв'язку та додаткового обладнання.

### 2.1.4 Топологія обчислювальних мереж

Розглянемо деякі топології ЛОМ.

**Топологія "зірка".**

**Зіркоподібна топологія мережі** – різновид мережі, де кожен термінал з'єднаний із центральною станцією (рис.2.2).

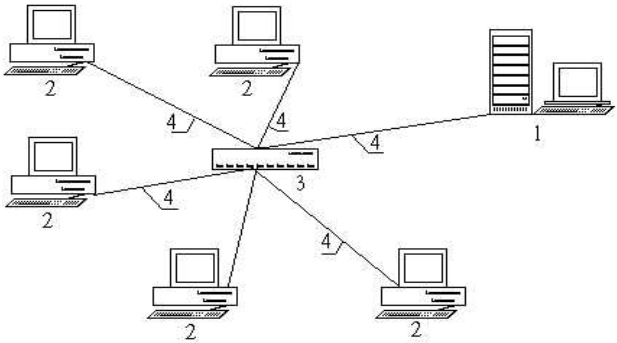



Рис 2.2. Зіркоподібна топологія мережі: 1 – сервер; 2 – робоча станція; 3 – концентратор; 4 – канали передачі даних

Ця топологія взята з області більших електронних обчислювальних машин. Тут файловий сервер перебуває в "центрі".

***Переваги***:

- ушкодження кабелю є проблемою для одного конкретного комп'ютера й у цілому не позначається на роботі мережі;
- просто виконується підключення, тому що робоча станція повинна з'єднуватися тільки із сервером;
- механізми захисту проти несанкціонованого доступу оптимальна;
- висока швидкість передачі даних від робочої станції до сервера, тому що обоє ПК безпосередньо з'єднані один з одним.

***Недоліки***:

- у той час як передача даних від робочої станції до сервера (і назад) відбувається швидко, швидкість передачі даних між окремими робочими станціями мала;
- потужність всієї мережі залежить від можливостей сервера, якщо він недостатньо оснащений або погано конфігурований, те буде гальмом для всієї системи;
- неможлива комунікація між окремими робочими станціями без допомоги сервера.

Топологія із сервером у центрі практично не реалізується, тому що в цьому випадку сервер повинен мати багато мережних адаптерів, робочі станції підключаються до концентратора (хабу).

**Кільцева топологія**

**Мережа типу кільце** – різновид мережі, у якій кожен термінал підключений до двох інших сусідніх терміналів кільця (рис.2.3).

У цьому випадку всі робочі станції й сервер з'єднані один з одним по кільцю, по якому посилає інформація, постачена адресою одержувача. Робочі станції одержують відповідні дані, аналізуючи адресу посланого повідомлення.

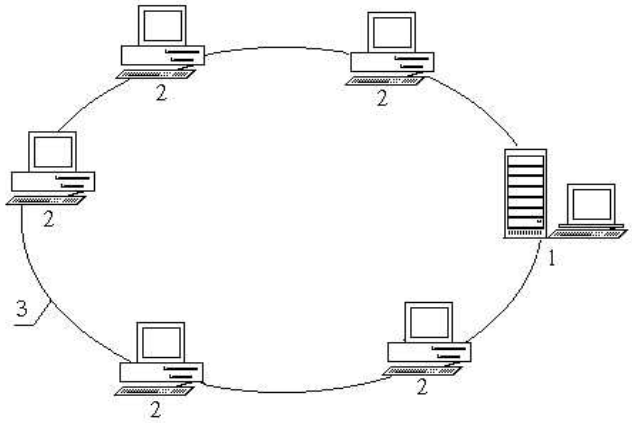



Рисунок 2.3. Кільцеподібна топологія мережі: 1 – сервер; 2 – робоча станція; 3 – канали передачі даних

***Переваги***:

– робітники станції можуть комутируватися один з одним без допомоги сервера.

***Недоліки***:

– час передачі даних збільшується пропорційно числу з'єднаних у кільце комп'ютерів;
– кожна робоча станція причетна до передачі даних, вихід з ладу однієї станції може паралізувати всю мережу, якщо не використаються спеціальні перехідні з'єднання;
– при підключенні нових робочих станцій мережа повинна бути короткочасно виключена.

**Шинна топологія**

**Мережа шинної топології** – різновид мережі, що має центральну лінію, до якої підключений сервер й окремі робочі станції. Шинна топологія мала широке поширення в колишні роки, що, насамперед , можна пояснити невеликими потребами в кабелі (рис.2.4).

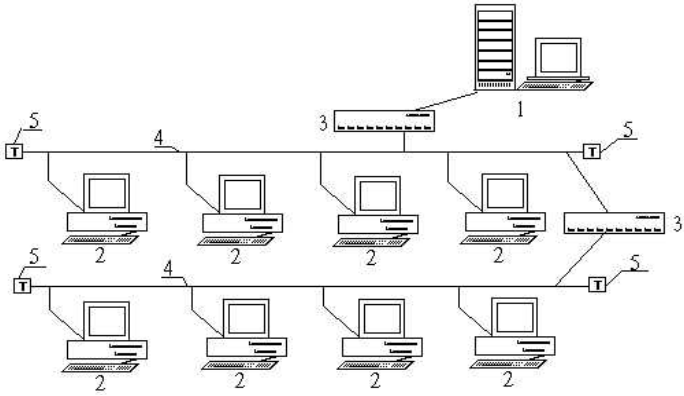



Рисунок 2.4. Шинна топологія мережі: 1 – сервер; 2 – робоча станція; 3 – концентратор; 4 – канали передачі даних; 5 – термінатори

***Переваги***:
- невеликі витрати на кабелі;
- робочі станції в будь-який момент часу можуть бути встановлені або відключені без переривання роботи всієї мережі;
- робочі станції можуть комутируватися один з одним без допомоги сервера.

***Недоліки***:
- при обриві кабелю виходить із ладу вся ділянка мережі від місця розриву;
- можливість несанкціонованого підключення до мережі, оскільки для збільшення числа робочих станцій немає необхідності в перериванні роботи мережі.

**Комбінована структура локальної обчислювальної мережі**

Поряд з відомими топологиями обчислювальних мереж кільце, зірка й шина, на практиці застосовується й **комбінована** (рис.2.5). Вона утвориться в основному у вигляді комбінацій вищезгаданих топологий обчислювальних мереж.

Обчислювальні мережі з комбінованою структурою застосовуються там, де неможливо безпосереднє застосування базових мережних структур у чистому виді. Для підключення великої кількості робочих станцій застосовують мережні підсилювачі й/або комутатори. Комутатор, що володіє одночасно й функціями підсилювача, називають активним концентратором.

Пасивний концентратор звичайно використають як розгалуджувач. Він не має потреби в підсилювачі. Передумовою для підключення пасивного концентратора є те, що максимальна можлива відстань до робочої станції не повинне перевищувати декількох десятків метрів.

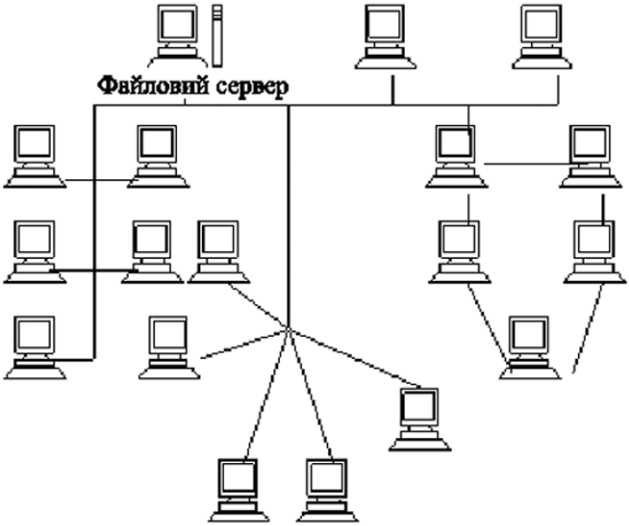



Рисунок 2.5. Комбінована структура

Комп'ютерні мережі стали логічним результатом еволюції комп'ютерних і телекомукційних технологій. З одного боку, вони є окремим випадком розподілених комп'ютерних систем, а з іншого боку, можуть розглядатися як засіб передачі інформації на великі відстані, для чого в них застосовуються методи кодування і мультиплексування даних, що одержали розвиток в різних телекомунікаційних системах.

## 2.2 Основи мереж передачі даних

### 2.2.1 Способи комутації. Класифікація способів комутації даних

Система протоколів, стандартів, правил та форматів встановлює порядок створення мереж, організацію їх функціонування, методи обробки та передачі інформації в рамках локальних мереж та встановлює порядок роботи із зовнішнім середовищем (іншими мережами). Принципи, правила, стандарти та обмеження реалізуються в технічних засобах (адаптерах, мостах, маршрутизаторах, концентраторах та ін.), системо-технічних вузлах та пристроях мережі, а також у програмних засобах у вигляді операційних систем мереж і драйверів їх елементів. Приймані при створенні мережі протоколи, стандарти та формати в цілому визначають архітектуру локальної мережі, типи засобів для її реалізації, формати циркулюючої інформації в мережі та необхідні програмні засоби, встановлюють порядок організації обробки інформації і обміну нею між користувачами цієї мережі.

Під **комутацією даних** розуміється їхня передача, при якій канал передачі даних може використатися попеременно для обміну інформацією між різними пунктами інформаційної мережі на відміну від зв'язку через не комутирують канали, що, звичайно закріплені за певними абонентами. Розрізняють наступні способи комутації даних:

*комутація каналів* – здійснює з'єднання ООД двох або більше станцій даних і забезпечує монопольне використання каналу передачі даних доти , поки з'єднання не буде розімкнуто; Комутація каналів може бути просторової й тимчасовий. У цей час переважно використається тимчасова або змішана комутація.

*комутація повідомлень* – характеризується тим, що створення фізичного каналу між оконечными вузлами необов'язково й пересилання повідомлень відбувається без порушення їхньої цілісності; замість фізичного каналу є віртуальний канал, що складається з фізичних ділянок, і між ділянками можлива буферизация повідомлення; Первісними видами повідомлень можуть бути голос, зображення, текст, дані. Для передачі звуку традиційно використається телефон, зображень – телебачення, тексту – телеграф (телетайп), даних – обчислювальні мережі. Передача документів (тексту) може бути кодової або факсимільної. Для передачі в єдиному середовищі звуку, зображень і даних застосовують мережі, називані мережами інтегрального обслуговування.

Кодова передача повідомлень між накопичувачами, що перебувають у вузлах інформаційної мережі, називається *телетексом* (на відміну від телекса – телетайпного зв'язку), а факсимільний зв'язок називається *телефаксом*. Види телетекса: електронна пошта (E-mail) – обмін повідомленнями між двома користувачами мережі, обмін файлами, "дошка оголошень" і телеконференції – широкомовна передача повідомлень. Установлення з'єднання між відправником й одержувачем з можливістю обміну повідомленнями без помітних тимчасових затримок характеризує **режим роботи on-line** ("на лінії"). При істотних затримках із запам'ятовуванням інформації в проміжних вузлах маємо **режим off-line** ("поза лінією").

*комутація пакетів* – повідомлення передається по віртуальному каналі, але воно розділяється на пакети, при цьому канал передачі даних зайнятий тільки під час передачі пакета (без порушення його цілісності) і по її завершенні звільняється для передачі інших пакетів.

У мережах комутації пакетів розрізняють два режими роботи:

- ✧ режим віртуальних каналів (інша назва – зв'язок із установленням з'єднання);
- ✧ дейтаграмний режим (зв'язок без установлення з'єднання).

У *дейтаграмному режимі* повідомлення ділиться на дейтаграми. **Дейтаграма** – частина інформації, передана незалежно від інших частин того самого повідомлення в обчислювальних мережах з комутацією пакетів.

Зв'язок може бути однобічної (симплексної), з поперемінной передачею інформації в обох напрямках (напівдуплексної) або одночасної в обох напрямках (дуплексної).

### 2.2.2 Протоколи мережі. Базова еталонна модель взаємозв'язку відкритих систем

2.2.2.1 Протоколи мережі

**Протокол** – це сукупність угод щодо способу подання даних, що забезпечує їхню передачу в потрібних напрямках і правильній інтерпретації даних всіма учасниками процесу інформаційного обміну.

Протоколи визначають порядок обміну інформацією між мережними об'єктами. Вони дозволяють взаємодіючим ЕОМ посилати один одному виклики, інтерпретувати дані, обробляти помилкові ситуації й виконувати безліч інших різних функцій. Суть протоколів полягає в регламентованих обмінах точно специфікованими командами й відповідями на них (наприклад, призначення фізичного рівня зв'язку – передача блоків даних між двома пристроями, підключеними до одного фізичного середовища).

Протокол передачі даних вимагає наступної інформації.

*Синхронізація*. Під синхронізацією розуміють механізм розпізнавання початку блоку даних і його кінця.

*Ініціалізація*. Під ініціалізацією розуміють установлення з'єднання між взаємодіючими партнерами. За умови, що приймач і передавач використають той самий протокол, синхронізація встановлюється автоматично.

*Блокування*. Під блокуванням розуміють розбивка переданої інформації на блоки даних строго певної максимальної довжини (включаючи розпізнавальні знаки початку блоку і його кінця).

*Адресація*. Адресація забезпечує ідентифікацію різного використовуваного встаткування, що обмінюється один з одним інформацією під час взаємодії.

*Виявлення помилок*. Під виявленням помилок розуміють установку й перевірку контрольних бітів.

*Нумерація блоків*. Поточна нумерація блоків дозволяє встановити помилково передану або інформацію, що втратилася.

*Керування потоком даних*. Керування потоком даних служить для розподілу й синхронізації інформаційних потоків. Так, наприклад, якщо не вистачає місця в буфері пристрою даних або дані недостатньо швидко обробляються в периферійних пристроях, повідомлення й/або запити накопичуються.

*Методи відновлення*. Після переривання процесу передачі даних використають методи відновлення, щоб повернутися до певного положення для повторної передачі інформації.

*Дозвіл доступу*. Розподіл, контроль і керування обмеженнями доступу до даних ставляться в обов'язок пункту дозволу доступу (наприклад, "тільки передача" або "тільки прийом").

Оскільки інформаційний обмін – процес багатофункціональний, то протоколи діляться на рівні. Кожен рівень підрозділяється на дві частини:

– специфікацію послуг;
– специфікацію протоколу.

Специфікація послуг визначає, що робить рівень, а специфікація протоколу – як він це робить. Причому, кожен конкретний рівень може мати більше одного протоколу.

Сукупність правил взаємодії пристроїв і програм між собою й засобів, що реалізують ця взаємодія, називається інтерфейсом. Поняття інтерфейсу містить у собі як самі апаратні й програмні засоби, що зв'язують різні пристрої або програми між собою, так і правила й алгоритми, на основі яких ці засоби створені.

Для правильної взаємодії вузлів різних обчислювальних мереж їхня архітектура повинна бути відкритої. Цим цілям служать уніфікація й стандартизація в області телекомунікацій й обчислювальних мереж.

Уніфікація й стандартизація протоколів виконуються рядом міжнародних організацій, що разом з різноманітністю типів мереж породило велике число різних протоколів. Найбільше широко розповсюдженими є протоколи, розроблені для мережі ARPANET і застосовувані в глобальній мережі Internet, протоколи відкритих систем Міжнародної організації по стандартизації (ISO – International Standard Organization), протоколи Міжнародного телекомунікаційного союзу (International Telecommunication Union – ITU, що раніше називалися CCITT) і протоколи Інституту інженерів по електротехніці й електроніці (IEEE – Institute of Electrical and Electronics Engineers). Протоколи мережі Internet поєднують за назвою TCP/IP (Transmission Control Protocol/Internet Protocol – протокол керування передачею /міжмірежевий протокол). Відповідно до протоколу TCP/IP працює апаратне й програмне забезпечення, що реалізує передачу даних по Інтернету. Протоколи ISO є семирівневими й відомі як протоколи базової еталонної моделі взаємозв'язку відкритих систем (EMBOC).

2.2.2.2 Еталонна модель взаємозв'язку відкритих систем

**Базова еталонна модель взаємозв'язку відкритих систем (EMBOC, OSI – Open System Interconnection model)** – це модель, прийнята ISO для опису загальних принципів взаємодії інформаційних систем. Модель OSI визнана всіма міжнародними організаціями як основа для стандартизації протоколів інформаційних мереж.

У моделі OSI інформаційна мережа розглядається як сукупність функцій, які діляться на групи, звані рівнями. Розділення на рівні дозволяє

вносити зміни в засоби реалізації одного рівня без перебудови засобів інших рівнів, що значно спрощує й здешевлює модернізацію засобів у міру розвитку техніки.

Модель OSI містить сім рівнів (рис.2.6).

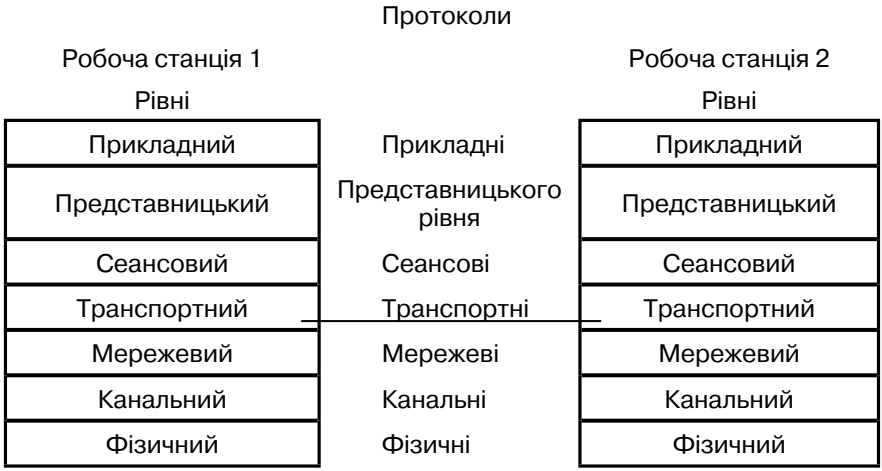



Рис. 2.6. Семирівнева модель протоколів взаємодії ЛОМ

***7-й рівень*** – прикладної (Application): включає засобу керування прикладними процесами; ці процеси можуть поєднуватися для виконання поставлених завдань, обмінюватися між собою даними. Інакше кажучи, на цьому рівні визначаються й оформляються в блоки ті дані, які підлягають передачі по мережі. Рівень включає, наприклад, такі засоби для взаємодії прикладних програм, як прийом і зберігання пакетів в "поштових скриньках" (mail-box).

***6-й рівень*** – представницький (Presentation): реалізуються функції подання даних (кодування, форматування, структурування). Наприклад, на цьому рівні виділені для передачі дані перетворяться з коду EBCDI (Extended Binary Coded Decimal Interchange Code – символи кодуються вісьма бітами) в ASCII (American Standards Committee for Information Interchange – семибітовий двійковий код) і т.п. (табл. 2.1).

***5-й рівень*** – сеансовий (Session): призначений для організації й синхронізації діалогу, що ведеться об'єктами (станціями) мережі. На цьому рівні визначаються тип зв'язку (дуплекс або напівдуплекс), початок і закінчення завдань, послідовність і режим обміну запитами й відповідями взаємодіючих партнерів.

***4-й рівень*** – транспортний (Transport): призначений для керування наскрізними каналами в мережі передачі даних; на цьому рівні забезпечується

зв'язок між крайовими пунктами (на відміну від наступного мережного рівня, на якому забезпечується передача даних через проміжні компоненти мережі). До функцій транспортного рівня ставляться мультиплексування й демультиплексування (зборка-розбирання пакетів), виявлення й усунення помилок у передачі даних, реалізація замовленого рівня послуг (наприклад, замовленої швидкості й надійності передачі). На транспортному рівні пакети звичайно називають сегментами.

Таблиця 2.1

**Протоколи рівнів OSI**

| Рівень OSI | Протоколи |
|---|---|
| Прикладний | HTTP, gopher, Telnet, DNS, SMTP, SNMP, CMIP, FTP, TFTP, SSH, IRC, AIM, NFS, NNTP, NTP, SNTP, XMPP, FTAM, APPC, X.400, X.500, AFP, LDAP, SIP, ITMS, Modbus TCP, BACnet IP, IMAP, POP3, SMB, MFTP, BitTorrent, e2k, PROFIBUS |
| Представницький | HTTP/HTML, ASN.1, XML, TDI, XDR, SNMP, FTP, Telnet, SMTP, NCP, AFP |
| Сеансовий | ASP, ADSP, DLC, Named Pipes, NBT, NetBIOS, NWLink, Printer Acces Protocol, Zone Information Protocol, SSL, TLS, SOCKS |
| Транспортний | TCP, UDP, NetBEUI, AEP, ATP, IL, NBP, RTMP, SMB, SPX, SCTP, DCCP, RTP, STP, TFTP |
| Мережевий | IP, IPv6, ICMP, IGMP, IPX, NWLink, NetBEUI, IPSec, ARP, RARP, DHCP, BootP, SKIP |
| Канальний (ланка даних) | ARCnet, ATM, DTM, SLIP, SMDS, Ethernet, FDDI, Frame Relay, LocalTalk, Token ring, PPP, PPPo, StarLan, WiFi, PPTP , L2F, L2TP, PROFIBUS |
| Фізичний | RS-232, RS-422, RS-423, RS-449, RS-485, ITU-T, xDSL, ISDN, T-carrier (T1, E1), модифікації стандарту Ethernet: 10BASE-T, 10BASE2, 10BASE5, 100BASE-TX, 100BASE-FX, 100BASE-T, 1000BASE-T, 1000BASE-TX, 1000BASE-SX |

***3-й рівень*** – мережної (Network): на цьому рівні відбувається керування передачею пакетів через проміжні вузли й мережі, контроль навантаження на мережу з метою запобігання перевантажень, що негативно впливають на роботу мережі, маршрутизація пакетів. Маршрутизація пакетів – визначення й реалізація маршрутів, по яких передаються пакети. Маршрутизація зводиться до визначення логічних каналів. Логічним каналом називається віртуальне (не обов'язково фізичне) з'єднання двох або більше об'єктів мережного рівня, при якому можливий обмін даними між цими об'єктами. Поняттю логічного каналу необов'язкова відповідність якогось фізичного

з'єднання ліній передачі даних між зв'язують пунктами, що. Це поняття уведене для абстрагування від фізичної реалізації з'єднання.

***2-й рівень*** – канальний (Link, рівень ланки даних): надає послуги по обміні даними між логічними об'єктами попереднього мережного рівня й виконує функції, пов'язані з формуванням і передачею кадрів, виявленням і виправленням помилок, що виникають на наступному, фізичному рівні. Кадром називається пакет канального рівня, оскільки пакет на попередніх рівнях може складатися з одного або багатьох кадрів. У ЛОМ функції канального рівня підрозділяють на два підрівні: керування доступом до каналу (MAC – Medium Access Control) і керування логічним каналом (LLC – Logical Link Control). До підрівню LLC ставиться частина функцій канального рівня, не пов'язаних з особливостями передавального середовища. На підрівні MAC здійснюється доступ до каналу передачі даних.

***1-й рівень*** – фізичний (Physical): надає механічні, електричні, функціональні й процедурні засоби для встановлення, підтримки й роз'єднання логічних з'єднань між логічними об'єктами канального рівня; реалізує функції передачі бітів даних через фізичні середовища. Саме на фізичному рівні здійснюються подання інформації у вигляді електричних або оптичних сигналів, перетворення форми сигналів, вибір параметрів фізичних середовищ передачі даних (табл.2.2).

Таблиця 2.2

**Аналогія рівнів OSI**

| Рівень | Аналогія |
|---|---|
| Прикладний рівень | Лист написано на папері. Визначено його зміст. |
| Представницький рівень | Лист запечатано в конверт. Конверт підписано. Наклеєно марку. Клієнтом виконані необхідні умови протоколу доставки. |
| Сеансовий рівень | Лист опущено в поштову скриньку. Вибрана служба доставки (лист можна було б запечатати в пляшку і кинути в річку, але вибрана інша служба). |
| Транспортний рівень | Лист доставлено на поштамт. Його відділили від листів, з доставкою яких місцева поштова служба справилась би самостійно. |
| Мережевий рівень | Після сортування лист покладено в мішок. З'явилася нова одиниця доставки – мішок. |
| Канальний рівень | Мішки листів покладені у вагон. З'явилася нова одиниця доставки – вагон. |
| Фізичний рівень | Вагон причеплено до локомотива. З'явилася нова одиниця доставки – потяг. За доставку взялось інше відомство, яке діє за іншими протоколами. |

У конкретних випадках може виникати потреба в реалізації лише частини названих функцій, тоді відповідно в мережі є лише частина рівнів. Так, у прості (нерозгалужених) ЛОМ відпадає необхідність у засобах мережного й транспортного рівнів.

Передача даних через розгалужені мережі відбувається при використанні інкапсуляції/декапсуляції порцій даних. Так, повідомлення, що прийшло на транспортний рівень, ділиться на сегменти, які одержують заголовки й передаються на мережний рівень. На мережному рівні сегмент може бути розділений на частині (пакети), якщо мережа не підтримує передачу сегментів цілком. Пакет забезпечується своїм мережним заголовком (тобто відбувається інкапсуляція сегментів у пакети). При передачі між вузлами проміжної ЛОМ може знадобитися поділ пакетів на кадри (тобто інкапсуляція пакетів у кадри). У прийомному вузлі сегменти декапсулюються й відновлюється вихідне повідомлення.

**Базові технології локальних мереж**

Архітектури або технології локальних мереж можна розділити на два покоління. До першого покоління ставляться архітектури, що забезпечують низьку й середню швидкість передачі інформації: Ethernet 10 Мбит/с), Token Ring (16 Мбит/с) і ARC net (2,5 Мбит/с).

Для передачі даних ці технології використовують кабелі з мідною жилою. До другого покоління технологій ставляться сучасні високошвидкісні архітектури: FDDI (100 Мбит/с), ATM (155 Мбит/с) і модернізовані версії архитектур першого покоління (Ethernet): Fast Ethernet (100 Мбит/с) і Gigabit Ethernet (1000 Мбит/с).

Удосконалені варіанти архитектур першого покоління розраховані як на застосування кабелів з мідними жилами, так і на волоконно-оптичні лінії передачі даних

Нові технології (FDDI й ATM) орієнтовані на застосування волоконно-оптичних ліній передачі даних і можуть використовуватися для одночасної передачі інформації різних типів (відеозображення, голосу й даних).

Мережна технологія – це мінімальний набір стандартних протоколів і їхніх програмно-апаратних засобів, що реалізують, достатній для побудови обчислювальної мережі. Мережні технології називають базовими технологіями. У цей час налічується величезну кількість мереж, що мають різні рівні стандартизації, але широке поширення одержали такі відомі технології, як Ethernet, Token- Ring, Arcnet, FDDI.

**Методи доступу до мережі**

**Ethernet** є методом множинного доступу із прослуховуванням несучої й дозволом колізій (конфліктів). Перед початком передачі кожна робоча станція визначає, вільні канал або зайнятий. Якщо канал вільний, станція починає передачу даних. Реально конфлікти приводять до зниження швидкодії мережі тільки в тому випадку, коли працюють 80-100 станцій.

Метод доступу **Arcnet**. Цей метод доступу одержав широке поширення в основному завдяки тому, що встаткування Arcnet дешевше, ніж

устаткування Ethernet або Token-Ring. Arcnet використовується в локальних мережах з топологією "зірка".

Один з комп'ютерів створює спеціальний маркер (спеціальне повідомлення), що послідовно передається від одного комп'ютера до іншого. Якщо станція повинна передати повідомлення, вона, одержавши маркер, формує пакет, доповнений адресами відправника й призначення. Коли пакет доходить до станції призначення, повідомлення "відчіплюється" від маркера й передається станції.

Метод доступу **Token Ring**. Цей метод розроблений фірмою IBM; він розрахований па кільцеву топологію мережі. Даний метод нагадує Arcnet, тому що теж використовує маркер, переданий від однієї станції до іншої. На відміну від Arcnet при методі доступу Token Ring передбачена можливість призначати різні пріоритети різним робочим станціям.

**Базові технології ЛОМ**

Технологія **Ethernet**. У класичній мережі Ethernet застосовується стандартний коаксіальний кабель двох видів (товстий і тонкий). Однак все більше поширення одержала версія Ethernet, що використовує як середовище передачі кручені пари, тому що монтаж і обслуговування їх набагато простіше. Застосовуються топології типу "шина" і типу "пасивна зірка".

Стандарт визначає чотири основних типи середовища передачі.

- ✧ 10BASE5 (товстий коаксіальний кабель);
- ✧ 10BASE2 (тонкий коаксіальний кабель);
- ✧ 10 BASE-T (кручена пари);
- ✧ 10 BASE-F (оптоволоконный кабель).

**Fast Ethernet** – високошвидкісний різновид мережі Ethernet, що забезпечує швидкість передачі 100 Мбит/с. Мережі Fast Ethernet сумісні з мережами, виконаними по стандарті Ethernet. Основна топологія мережі Fast Ethernet - пасивна зірка.

Стандарт визначає три типи середовища передачі для Fast Ethernet:

- ✧ 100 BASE-T4 (зчетверена кручена пари);
- ✧ 100 BASE-TX (здвоєна кручена пари);
- ✧ 100 BASE-FX (оптоволоконный кабель).

**Gigabit Ethernet** – високошвидкісний різновид мережі Ethernet, що забезпечує швидкість передачі 1000 Мбит/с. Стандарт мережі Gigabit Ethernet у цей час містить у собі наступні типи середовища передачі:

- ✧ 1000 BASE-SX – сегмент на мультимодовом оптоволоконном кабелі з довжиною хвилі світлового сигналу 850 нм.
- ✧ 1000 BASE-LX – сегмент на мультимодовом і одномодовом оптоволоконном кабелі з довжиною хвилі світлового сигналу 1300 нм.
- ✧ 1000 BASE-CX – сегмент на електричному кабелі (екранована кручена пари).

- ✧ 1000 BASE-T – сегмент на електричному кабелі (зчетверена неекранована кручена пари).

У зв'язку з тим, що мережі сумісні, легко й просто з'єднувати сегменти Ethernet, Fast Ethernet і Gigabit Ethernet у єдину мережу.

Мережа **Token-Ring** запропонована фірмою IBM.Token-Ring призначалася для об'єднання в мережу всіх типів комп'ютерів, що випускаються IBM (від персональних до більших). Мережа Token-Ring має зірково-кільцеву топологію.

Мережа **Arcnet** – це одна з найстарших мереж. Як топологія мережа Arcnet використовує "шину" і "пасивну зірку". Мережа Arcnet користувалася великою популярністю. Серед основних переваг мережі Arcnet можна назвати високу надійність, низьку вартість адаптерів і гнучкість. Основним недолікам мережі є низька швидкість передачі інформації (2,5 Мбит/с).

**FDDI** (Fiber Distributed Data Interface) – стандартизована специфікація для мережної архітектури високошвидкісної передачі даних по оптоволоконним лініях. Швидкість передачі – 100 Мбит/с.

Основні технічні характеристики мережі FDDI наступні:

- максимальна кількість абонентів мережі – 1000;
- максимальна довжина кільця мережі – 20 км;
- максимальна відстань між абонентами мережі – 2 км;
- середовище передачі – оптоволоконний кабель;
- метод доступу – маркерний;
- швидкість передачі інформації – 100 Мбит/с.

### 2.2.3 Апаратне забезпечення комп'ютерних мереж

Склад технічних засобів у конкретній локальній мережі визначається вибраною топологією, використовуваним програмним забезпечення мережі та плановими витратами на її створення і підтримку. До технічних засобів належать: сервери, робочі станції, адаптери мережі, канали передачі даних та доповнююче обладнання. Особливе місце серед технічних засобів займають системи безперебійного живлення.

**Комп'ютерна мережа** – це складний комплекс взаємозалежних і узгоджено функціонуючих програмних і апаратних компонентів.

Комп'ютерну мережу можна представити багатошаровою моделлю, що складається із шарів:

- комп'ютери;
- комунікаційне встаткування;
- операційні системи;
- мережні додатки.

2.2.3.1 Комп'ютери

Основою будь-якої локальної мережі є ПК, які підключаються до мережі за допомогою мережної карти. Всі комп'ютери локальних мереж можна розділити на два класи: сервери й робочі станції.

***Сервер***

Центральним вузлом локальної мережі, як правило, є комп'ютер спеціального призначення – **сервер** для управління мережею та сервісного обслуговування її клієнтів. Функціонально сервісне обслуговування передбачає розділений доступ користувачів до носіїв інформації, файлових структур, банків даних, кінцевих пристроїв (принтерів, сканерів, модемів і інше) та електронної пошти, встановлює правила доступу до інформації в мережі, проводить резервне копіювання даних та забезпечує встановлений рівень захисту інформації тощо.

Тому що сервер є центральним елементом мережі, то вимоги відповідно надійності, швидкодії до його програмних та функціональних можливостей мають бути досить високими.

Іноді в мережі встановлюють декілька серверів, які розподіляють функції обробки інформації в мережі між собою. Ними можуть бути файл-сервер, сервер друку, сервер банків даних, сервер електронної пошти та інші. Назви вказують на їх головне функціональне призначення. Це так звані виділені сервери, які виконують тільки окрему функцію і не можуть виконувати функції робочої станції.

Відповідно до вимог це мають бути найпотужніші, високопродуктивні та надійні комп'ютери. Показники ефективності їхнього функціонування визначають рівень інформаційних, технічних та програмних ресурсів всієї мережі.

Серверу в ієрархічні (централізованих) мережах належить центральна роль. Отже, він повинен бути й добре оснащений. Його оснащення залежить від числа підключених робочих станцій.

Більше старі версії мереж пропонували можливість використання сервера в невиділеному режимі (non dedicated). У цьому випадку сервер функціонує не тільки як центральний ПК, але може використатися як звичайна робоча станція. Це вигідно в змісті ціни, тому що заощаджується одна робоча станція, але через сервер "відверненого" на рішення завдань користувача, зазнає труднощів вся мережа, тому рекомендується використати сервер тільки у виділеному режимі (dedicated).

Залежно від способів використання сервера в ієрархічних ЛОМ розрізняють сервери наступних типів.

**Файловий сервер**. У цьому випадку на сервері перебувають спільно оброблювані файли й спільно використовувані програми.

**Сервер баз даних**. На сервері розміщається мережна база даних. База даних на сервері може поповнюватися з різних робочих станцій і видавати інформацію із запитів з робочих станцій.

**Сервер доступу** – виділений комп'ютер у локальній мережі для виконання вилученої обробки завдань. Сервер виконує завдання, отримане з вилученої робочої станції, і результати направляє на вилучену робочу станцію. Іншими словами сервер призначений для вилученого доступу (наприклад, з мобільного ПК) до ресурсів локальної мережі.

**Сервер – друку**. До комп'ютера невеликої потужності підключається досить продуктивний принтер, на якому може бути роздрукована інформація відразу з декількох робочих станцій. Програмне забезпечення організує чергу завдань на печатку.

**Поштовий сервер**. На сервері зберігається інформація, що відправляється й одержувана як по локальній мережі, так і ззовні по модему. Користувач може переглянути інформацію, що надійшла на його ім'я, або відправити через поштовий сервер свою інформацію.

***Робоча станція***

**Робоча станція** – персональний комп'ютер, що працює під керуванням власної операційної системи. Однак на відміну від автономного персонального комп'ютера робоча станція містить плату мережного інтерфейсу й фізично з'єднана кабелем із сервером. Крім того, робоча станція повинна мати мережну операційну систему або спеціальну програму, називану оболонкою мережі, при використанні немережевий ОС, що дозволяє їй обмінюватися інформацією із сервером, іншими робітниками станціями й іншими пристроями мережі. Оболонка дозволяє робочій станції використати файли й програми, що зберігаються на сервері, так само легко, як і перебувають на її власних дисках.

Оснащення окремих робочих станцій усередині мережі дуже сильно залежить від оснащення сервера. Якщо серверу виділена центральна роль, то робочі станції можуть оснащуватися встаткуванням у меншому ступені.

По-іншому виглядає однорангова мережа, у якій відсутній сервер. Тут чим краще окремі станції, тим краще розподіл ресурсів усередині всієї мережі. Дорогі периферійні пристрої, такі як сканер, модем, змінні жорсткі диски й т.п., необхідно встановлювати лише на одній робочій станції, тому що в мережі ресурси доступні всім користувачам.

2.2.3.2 Комунікаційне встаткування

Мережні карти

Мережна карта встановлюється в один з вільних слотов материнської плати. При цьому сервер не обов'язково повинен мати кращу карту, чим робочі станції.

Мережні карти є посередниками між ПК і мережею й передають мережні дані по системі шин до ЦП й ОЗУ сервера або робочої станції.

Мережна карта обладнана власним процесором і пам'яттю, що звичайно має обсяг 8-16 Кбайт. Більшість мережних плат передбачає використання мікросхеми ROM вилученого завантаження. Це потрібно для бездискових станцій (не мають ні вінчестера, ні дисководів для дискет). Завантаження операційної системи на згадку таких ПК відбуваються через мережу. Ця мікросхема ініціює процес завантаження. Щоб комп'ютер знав її місцезнаходження, треба на мережній карті встановити для неї базова адреса. На зовнішній стороні карти є рознімання для підключення кабелів.

Як і будь-які інші карти, мережні карти бувають 8-, 16- і 32-розрядними й можуть мати виконання для різних комп'ютерних архітектур: ISA, EISA, VESA, PCI, MCA.

**Мережний адаптер (МА)** – це спеціальний пристрій, який призначен для сполучення комп'ютера з локальною мережею й для організації двунаправленного обміну даними в мережі. Мережна карта вставляється у вільний слот розширення на материнській платі й обладнана власним процесором і пам'яттю, а для підключення до мережі має рознімання типу RJ- 45. Найпоширеніші карти типу PCI, які вставляються в слот розширення PCI на материнській платі. Залежно від застосовуваної технології Ethernet, Fast Ethernet або Gigabit Ethernet і мережної карти швидкість передачі даних у мережі може бути: 10, 100 або 1000 Мбит/с.

Основні функції адаптерів й їхніх технічних характеристик визначаються підтримуваним рівнем протоколу ЛОМ відповідно до архітектури семиуровневой еталонної моделі мережі.

По виконуваних функціях МА розділяються на дві групи:

1) реалізуючі функції фізичного й канального рівнів. Вони застосовуються в мережах із простою топологією;

2) реалізуючі функції перших чотирьох рівнів моделі мережі – фізичного, канального, мережного й транспортного. Такі адаптери застосовуються в більше складних ЛОМ.

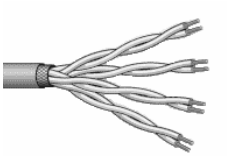
Адаптери орієнтовані на певну архітектуру локальної мережі і її технічні характеристики, тому по топології ЛОМ адаптери розділяються на наступні групи: підтримуючу шинну топологію, кільцеву, зіркоподібну, деревоподібну, комбіновану (зірково-кільцеву, зірково-шинну).

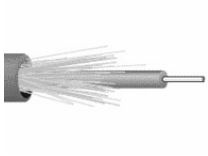
**Мережні кабелі**

У мережі дані циркулюють по кабелях, що з'єднує окремі комп'ютери різним образом залежно від обраної топології, виду мережі (Ethernet, Arcnet, Token Ring). Більшість збоїв і помилок усередині мережі відбувається через неякісний або дефектний кабель або кабельне рознімання. Залежно від топології пошук несправності може бути досить трудомісткий.

Як кабелі з'єднуючих окремі ПК і комунікаційне встаткування в локальних мережах застосовуються:

- Кручена пари – передавальна лінія зв'язку, що являє собою два проведення, перекручених один з одним з певним кроком з метою зниження впливу електромагнітних полів.
- Коаксіальний кабель – кабель, що складається з одного центрального провідника в ізоляторі й другого провідника розташованого поверх ізолятора.
- Оптичний кабель – це кабель, у якому носієм інформації є світловий промінь, що поширюється по оптичному волокну (рис.2.6).







| Кабель на витій парі | Оптоволоконний кабель | Коннектори RJ-45 для кабеля на витій парі |
|---|---|---|

Рисунок 2.6. Типи кабелів

Крім того, як передавальне середовище в бездротових локальних мережах використовуються радіохвилі в мікрохвильовому діапазоні.

До комунікаційного встаткування локальних мереж ставляться: трансиверы, повторювачі, концентратори, мости, комутатори, маршрутизатори й шлюзи.

***Частина встаткування*** (приемопередатчики або трансиверы, повторювачі або репитеры й концентратори або hubs) служить для об'єднання декількох комп'ютерів у необхідну конфігурацію мережі. З'єднані з концентратором ПК утворять один сегмент локальної мережі, тобто концентратори є засобом фізичної структуризації мережі, тому що, розбиваючи мережу на сегменти, спрощують підключення до мережі великого числа ПК.

***Друга частина встаткування*** (мости, комутатори) призначені для логічної структуризації мережі. Тому що локальні мережі є широкомовними (Ethernet і Token Ring), те зі збільшенням кількості комп'ютерів у мережі, побудованої на основі концентраторів, збільшується час затримки доступу комп'ютерів до мережі й виникнення колізій. Тому в мережах побудованих на хабах установлюють мости або комутатори між кожними трьома або чотирма концентраторами, тобто здійснюють логічну структуризацію мережі з метою недопущення колізій.

***Третя частина встаткування*** призначена для об'єднання декількох локальних мереж у єдину мережу: **маршрутизатори (routers)**, **шлюзи (gateways)**. До цієї частини встаткування можна віднести й **мости (bridges)**, а також **комутатори (switches)**.

**Повторювачі (repeater)** – пристрою для відновлення й посилення сигналів у мережі, що служать для збільшення її довжини. За допомогою цих пристроїв можна об'єднати кілька сегментів мережі із шинною топологією, збільшуючи в такий спосіб загальну довжину мережі.

**Приемопередатчики (трансиверы)** – це пристрою, призначені для прийому пакетів від контролера робочих станцій мережі й передачі їх у мережу. **Трансиверы (конвертори)** можуть перетворювати електричні сигнали в інші види сигналів (оптичні або радіосигнали) з метою використання інших середовищ передачі інформації.

**Концентратори** або **хабы (Hub)** (рис. 2.7) – пристрої множинного доступу, які поєднує в одній крапці окремі фізичні відрізки кабелю, утворять загальне середовище передачі даних або сегменти мережі, тобто хабы використовуються для створення сегментів і є засобом фізичної структуризації мережі. Випускається ряд типів концентраторів, вони відрізняються по кількості, типу й довжині кабелів, що підключають, і можуть автоматично управляти приєднаними сегментами (включати й виключати їх у випадку виявлення збоїв й обривів).

**Мости (bridges)** – це програмно – апаратні пристрої, які забезпечують з'єднання декількох локальних мереж між собою. Мости призначені для логічної структуризації мережі або для з'єднання в основному ідентичних мереж, що мають деякі фізичні розходження на фізичному й канальному рівнях.

**Комутатори (switches)** – програмно – апаратні пристрої є швидкодіючим аналогом мостів, які ділять загальне середовище передачі даних на логічні сегменти. Логічний сегмент утвориться шляхом об'єднання декількох фізичних сегментів за допомогою одного або декількох концентраторів. Кожний логічний сегмент підключається до окремого порту комутатора. При надходженні даних з комп'ютера – відправника на який-небудь із портів комутатор передасть ці дані, але не на всі порти, як у концентраторі, а тільки на той порт, до якого підключений сегмент, що містить комп'ютер – одержувач даних(рис. 2.7).



Концентратор на 24 порти



Комутатор на 12 портів

Рисунок 2.7. Приклади концентратора та комутатора

**Маршрутизатори (routers)**. Ці пристрої забезпечують вибір маршруту передачі даних між декількома мережами, що мають різну архітектуру або протоколи, тобто встановлюють з'єднання на 4-м (транспортному) рівні, при цьому верхні рівні мережі (5, 6 й 7) повинні бути однакові. Вони забезпечують складний рівень сервісу, тому що можуть виконувати "інтелектуальні" функції: вибір найкращого маршруту для передачі повідомлення, адресованого іншої мережі; керування балансованим навантаженням у мережі шляхом рівномірного розподілу потоків даних; захист даних; буферизацию переданих даних; різні протокольні перетворення. Такі можливості маршрутизаторів особливо важливі при побудові базових мереж великих організацій.

**Шлюзи (gateway)** – пристрої (комп'ютер), призначені для об'єднання різнорідних мереж з різними протоколами обміну. Шлюзи виконують протокольне перетворення для всіх семи рівнів мережі, зокрема – маршрутизацію пакетів, перетворення повідомлення з одного формату в іншій або з однієї системи кодування в іншу.

**Модеми й факси-модеми**. Модем, забезпечуючи узгодження цифрових сигналів комп'ютера з аналоговими сигналами телефонної лінії, при передачі даних здійснює модулювання аналогових сигналів цифровою інформацією, а при прийомі – демодулювання. Факси-модеми забезпечують швидкісну передачу даних тільки в одному напрямку й використають свої власні стандарти. Вони краще справляються з передачею інформації, чим із прийомом. У цей час випускаються й комбіновані модеми (модем даних/факс-модем).

**Аналізатори ЛОМ**. Це потужний діагностичний інструмент, призначений для контролю якості функціонування мережі.

**Мережні тестери**. Це прилади, що входять до складу контрольно-вимірювальних апаратур, що полегшує установку й технічне обслуговування локальних мереж. Тестери ліній передачі є гарним засобом перевірки нового кабелю й відшукання несправностей у системі встановлених кабелів. Вони здатні не тільки виявляти несправність, але й подавати відомості про її характер і місце розташування.

### 2.2.4 Програмне забезпечення мережі

Програмне забезпечення складається з операційної системи мережі, прикладних програм та доповнюючих програмних засобів.

**Операційна система мережі (ОСМ)** – програма, яка забезпечує управління нею. За її допомогою користувачі мережі підтримують взаємозв'язок, розподіляють між собою роботу з файлами та дисками, роздільно використовують периферійні засоби обробки інформації в межах наданих прав та обов'язків. Найбільшого поширення у локальних мережах

набули такі версії ОСМ: NetWare, Windows for Workgroups та мережні засоби Windows.

**ОСМ NetWare фірми Novell**. Широке застосування цієї ОСМ обумовлене такими перевагами:

- ✧ в середовищі NetWare можуть працювати більше додатків, чим у будь-якій іншій ОСМ;
- ✧ ОСМ NetWare сумісна з програмним забезпеченням типу DOS, Windows, OC/2, Unix, Windows NT, Mac System 7 та іншими, встановленими на робочих станціях мережі;
- ✧ для роботи ОСМ NetWare можуть використовуватись адаптери мережі будь-якого типу;
- ✧ ОСМ NetWare не має обмежень по кількості робочих станцій мережі;
- ✧ практичний досвід експлуатації показує високу надійність її роботи;
- ✧ застосовувані заходи та засоби захисту даних в ОСМ достатні для більшості локальних мереж.

**ОСМ Windows for Workgroups** призначена для забезпечення роботи однорангових локальних мереж в яких чисельність користувачів (РС) не більше 10. За рахунок встановлення даної ОСМ є можливість розподілити дисковий простір, встановити порядок використання файлових структур РС, роботи кінцевих пристроїв (принтерів, сканерів та інших), застосування електронної пошти, організовувати колективну роботу з прикладними програмами з використанням інформації банків даних, забезпечувати відносний захист та збереження даних.

**Мережні засоби операційної системи Windows** дозволяють без їх спеціального встановлення при фізичному підключенні комп'ютера до мережі ініціювати та налаштовувати роботу в локальній або іншій мережі.

Прикладне програмне забезпечення мереж дозволяє проводити роботу всіх користувачів в межах однієї прикладної програми. Такими програмами є програми управління базами даних, електронних таблиць та інші.

Крім ОСМ, для роботи мережі на всіх адаптерах необхідно встановлювати додаткове програмне забезпечення, яким є драйвери елементів мережі.

Застосування мережного програмного забезпечення не виключає використання програмного забезпечення та прикладних програм, встановлених на кожній робочій станції. Вони можуть, за необхідності, використовуватись приватно кожним користувачем.

Окремим питанням є організація та робота електронної пошти в локальній мережі. В найпростішому варіанті електронна пошта є засобом обміну текстовими файлами. В мережі це можна забезпечити простим копіюванням текстових файлів в поштову директорію адресата на сервері. Але такий підхід не гарантує захисту переданої інформації від

несанкціонованого доступу та її руйнування. Для цього використовується ціла низка заходів. Виконання функцій організації адресного обміну даними здійснюється спеціальними програмними засобами електронної пошти.

Для організації роботи пошти на сервері в базу даних заносять адреси та імена всіх користувачів мережі. Спеціально створена додаткова база даних пошти зберігає індивідуальні поштові повідомлення та зв'язані з ними файли. Залежно від адреси та імені за командою відправника програмні засоби електронної пошти здійснюють передачу інформації до робочої станції отримувача в директорію поштових повідомлень. При передачі даних поза межі мережі повідомлення відправляються таким же чином за встановленими на сервері маршрутами зовнішнього середовища за адресою отримувача через модем (міст) та засоби зв'язку. Організація роботи електронної пошти, організація доступу до змісту повідомлень та захист даних в них залежить від впроваджених засобів мережі та її топології.

### 2.2.5 Пропозиції зі створення та організації роботи в локальних комп'ютерних мережах

Для впровадження локальних мереж в роботу практичні рекомендації можна звести до пропозицій відносно планування та створення локальної мережі, організації роботи в ній та нарощування її можливостей.

Для створення мережі в будь-якій структурі необхідно визначити:

- її функціональне призначення та обсяг задач, які необхідно виконувати на даному рівні управління;
- структуру мережі та ступінь залучення елементів її структури для виконання задач;
- наявність технічних, програмних та фінансових можливостей для створення, використання та нарощування мережі як елемента більш складної системи управління;
- готовність особового складу до роботи в мережі і виконання відповідних функціональних посадових обов'язків;
- наявність необхідного комплексу службових та правових документів, які регламентують організацію і роботу в мережі, права, обов'язки та ступінь відповідальності всіх її користувачів, порядок захисту та збереження даних в ній.

Функціональне призначення системи управління та задачі, які вона виконує, визначають структуру мережі та її складові елементи. При цьому першочергову увагу необхідно звернути на ті елементи системи, робота яких визначає якість виконання задач.

Архітектура мережі може передбачати різні варіанти топології, але з урахуванням подальшої розбудови мережі слід рекомендувати:

- мережу необхідно створювати на базі виділених серверів. Одноранговa мережа значно дешевша, але її можливості дуже обмежені, особливо відносно створення потужного банку даних з розподіленим доступом до нього колективу користувачів. Колективна робота з таким банком в одноранговій мережі дуже проблематична;
- мережа із зіркоподібною топологією має відносно більшу надійність. Вихід з ладу будь-якого променю зіркоподібної топології не призводить до відмови всієї системи. Необхідно передбачити застосування багатопортового концентратора як розподіляючого елемента, що надасть в майбутньому можливість простого переходу від зіркоподібної топології до деревоподібної;
- для головних елементів мережі – серверів – необхідно передбачати застосування найбільш потужних комп'ютерів. Це зумовлено необхідністю забезпечення достатнього рівня загальних показників ефективності мережі: швидкодії, обсягу оперативної пам'яті та загального обсягу зовнішніх носіїв пам'яті;
- при організації роботи з віддаленими робочими станціями через модем необхідно передбачити застосування закритих виділених ліній зв'язку;
- з точки зору якості передачі даних кращі показники мають оптиковолоконні кабелі, але, враховуючи їх вартість та порівняльні характеристики ліній зв'язку, більш придатними є екрановані витi пари. Вони забезпечують досить високу швидкість передачі даних, роздільне використання каналів обміну (в різних парах), значну захищеність і надійність передачі інформації;
- досвід експлуатації локальних мереж показує, що для них доцільнішим є використання операційної системи мережі **NetWare 4.0**, переваги якої наведені раніше.

Організація роботи локальної мережі залежить від багатьох чинників, але її ефективність визначається, в першу чергу, рівнем використання загального ресурсу мережі, рівнем кваліфікації адміністратора, користувачів та суворим дотриманням всім особовим складом дисципліни та встановленого порядку роботи в мережі.

В організаційних заходах необхідно передбачити:

- визначення відповідальних осіб за планування, створення та налаштування локальної мережі, термінів виконання етапів, передбачення необхідного матеріального, технічного, програмного та фінансового забезпечення всього комплексу робіт;
- встановлення та затвердження порядку прийому та перевірки на функціональну придатність створеної мережі для виконання визначених задач управління;

- призначення адміністратора (адміністративної групи) мережі та групи технічного забезпечення для організації її роботи та підтримки відповідної функціональної готовності, встановлення порядку та періодичності технічного обслуговування і заходів матеріально-технічного забезпечення поточної роботи;
- організацію робіт з визначення необхідного ступеня захисту інформації, встановлення категорії мережі та рівня обмежень при виконанні робіт в ній;
- розробку перспективного плану розвитку та визначення основних напрямків подальшого технічного, програмного і комунікаційного удосконалення мережі, планування заходів з підвищення професіональної підготовки особового складу.

Організація управління мережею має базуватись на її плані, на якому відображаються відомості про кабельну мережу, схемне рішення її сполучень, її протяжність, стандарти протоколів та обладнання, дані про робочі станції, технології збору, обробки та подання інформації в межах мережі та при обміні з зовнішнім середовищем. До плану залучають, крім того, інформацію про засоби та методи контролю і управління мережею.

Управління мережею включає:

- аналіз роботи мережі в цілому та окремих її складових щодо продуктивності, обсягів та швидкості обробки інформації, що циркулює в мережі, кількості та характеру помилок в роботі, які свідчать про ступінь сумісності технічного та програмного рівня елементів мережі;
- своєчасне внесення коректив при залученні в мережу нових елементів та відповідне до цього обновлення програмних засобів;
- постійний контроль за роботою користувачів, періодична зміна порядку доступу до інформаційних ресурсів, аналіз стану та удосконалення системи заходів із захисту інформації та запобігання втрати даних;
- систематичний облік помилок роботи мережі, їх аналіз та впровадження заходів щодо запобігання їх появи або зменшення їх впливу на функціональні можливості мережі;
- контроль розподілу та підтримка потрібного рівня ресурсів мережі за рахунок нарощування технічних можливостей, видалення зайвої та надмірної інформації.

Розглянутий перелік заходів та задач зі створення та організації роботи мережі в органах управління різних рівнів потребує від посадових осіб глибоких знань у вказаній предметній області та цілеспрямованої роботи з обґрунтування вимог, широкого впровадження та ефективного використання інформаційно-розрахункових систем на основі локальних мереж.

## 2.3 Загальні відомості та характеристика мережі Internet

**6 лютого**
**День безпечного Інтернету (Safer Internet day)**
Відзначається за ініціативою Європейської комісії з 2004 р. Оновлюємо антивірус, якщо немає те його ставимо. Испольузем найсвіжіший софтвер і ставимо усі оновлення безпеки. Кращий день для того що б зробити свою роботу за комп'ютером безпечної для своїх даних і грошей.

### 2.3.1 Історія створення Інтернету

Internet – це величезна мережа комп'ютерів, у якій працюють мільйони користувачів в усьому світі. Internet поєднує в собі безліч інтерактивних і поштових служб, найвідомішою з яких є World Wide Web – інтерактивна мережна служба, що дозволяє передавати через мережу графіку, анімацію і мультимедиа. Крім цієї служби Internet надає послуги електронної пошти, передачі файлів, пошуку інформації, служб новин і багато інших.

Windows містить програмні компоненти, що забезпечують підтримку більшості служб Internet.

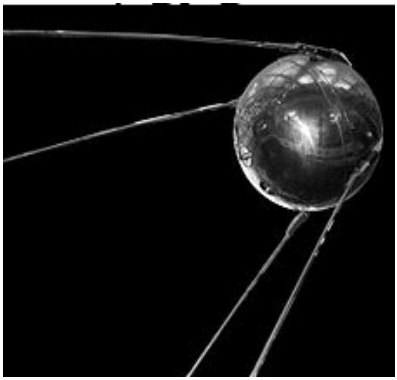


**Рисунок 2.8. Супутник-1, перший штучний супутник Землі, був запущений на орбіту в СРСР 4 жовтня 1957 року**



**Рисунок 2.9. Модель Супутника-2 у Політехнічному музеї Росії, другий космічний апарат, запущений на орбіту Землі 3 листопада 1957, уперше вивів в космос живу істоту – собаку Лайку. Офіційно супутник був запущений у рамках Міжнародного геофизичного року**

Наприклад, програма, що виконує обмін інформацією зі службою World Wide Web – **броузер (оглядач) Internet Explorer** вбудований у Windows. Аналогічно інтегрована в систему і **програма для роботи з електронною поштою – Outlook Express**. Також є програми для організації мережних конференцій, ведення віщальних передач, участі в інтерактивних бесідах, і т.д. Однак, найбільше часто користувачами застосовуються програми для огляду World Wide Web і обміну електронною поштою.

Ранні експерименти по передачі і прийому інформації за допомогою комп'ютерів почалися ще в 50-х роках і мали лабораторний характер. У США рішення про створення першої глобальної мережі національного масштабу було прийняте в 1958 р. Воно стало реакцією на запуск в СРСР першого штучного супутника Землі (рис.2.8, 2.9). Дата запуску вважається початком космічної ери людства.

Непрямим поштовхом створення Інтернет з'явився запуск в Радянському Союзі в 1957 році першого штучного супутника. Вже почалася Холодна Війна і Сполучені Штати, розуміючи, що супутник – це лише вершина айсберга радянських наукових досліджень у військовій області, побачили для себе загрозу використання ракет для нанесення ядерного удару по США. У цьому ж 1957 році, при департаменті оборони США було створене Агентство за науково-дослідними проектами – ARPA.

Приводом для створення глобальної комп'ютерної мережі стала розробка Пентагоном глобальної системи раннього сповіщення про пуски ракет (NORAD – North American Aerospace Defence Command). Станції системи NORAD простягнулися через північ Канади від Аляски до Гренландії, а підземний командний центр розташувався поблизу міста Колорадо-Спрингс в надрах гори Шайенн. Центр управління був введений в дію в 1964 р., і, власне, з того часу можна говорити про роботу першої глобальної комп'ютерної мережі, хоча і відомчої. З середини 60-х років до неї стали підключатися авіаційні, метеорологічні і інші військові і цивільні служби.

Курируванням роботи мережі займалася спеціальна організація – Управління перспективних розробок міністерства оборони США (DARPA – Defense Advanced Research Project Agency) (1957р.). Основним недоліком централізованої мережі була недостатня стійкість, пов'язана з тим, що при виході з ладу якого-небудь з вузлів повністю виходив з ладу і увесь сектор, що знаходився за ним, а при виході з ладу центру управління виходила з ладу уся мережа. За часів ядерного протистояння наддержав цей недолік був критичним.

Вирішення проблеми стійкості і надійності мережі було доручене управлінню DARPA. Основними напрямами досліджень стали пошук нових протоколів обслуговування мережі і нових принципів мережевої архітектури. Полігоном для випробувань нових принципів стали найбільші

університетські і наукові центри США, між якими були прокладені лінії комп'ютерного зв'язку. З боку міністерства оборони роботи курирувалися тим же управлінням DARPA, і перша позавідомча національна комп'ютерна мережа дістала назву ARPANET. Її впровадження відбулося в 1969 р. (рис. 2.10).

Початок існування Internet – 2 січня 1969г., коли група учених із США почала дослідження в області об'єднання комп'ютерів в мережу. 29 жовтня 1969 р. була зроблена найперша, правда, що не цілком вдалася, спроба дистанційного підключення до комп'ютера, що знаходився в дослідницькому центрі Стэнфордского університету (SRI), з іншого комп'ютера, який стояв в Каліфорнійському університеті в Лос-Анджелесі (UCLA). Віддалені один від одного на відстань 500 кілометрів, SRI і UCLA стали першими вузлами майбутньої мережі ARPANet.

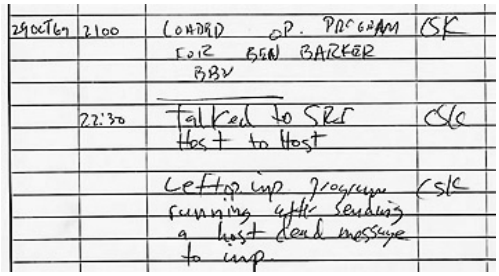

29 Oct 69 2100 (CONTD OP. PROGRAM CSK
FOR BEN BARKER
BBN
22:30 Talked to SRI CSK
Host to Host
Left IMP program CSK
running while sending
a host dead message
to IMP

Рисунок 2.10. Фрагмент робочого журналу
(запис зроблений 29 жовтня 1969 р.)

Випробування першої черги ARPANet зайняли усю осінь 1969 р. Потім до мережі підключили ще два вузли: Каліфорнійський університет Санта-Барбари (UCSB) і Університет штату Юта (UTAH). Саме ці чотири організації розподілили між собою основні функції по створенню компонентів першої в історії Wide Area Network (2.11):

- UCLA – проведення вимірювальних випробувань;
- SRI – створення інформаційного центру;
- UCSB – розробка математичного апарату;
- UTAH – перші роботи по тривимірній графіці.

Проведення випробувань стало можливе завдяки тому, що до 1 вересня 1969 р. компанія BBN (Bolt, Baranek і Newman) виготовила перші екземпляри пристрою IMP (Interface Message Processor), що забезпечує зв'язок між комп'ютерами по телефонних каналах.

Перед ученими було поставлене завдання створення комп'ютерної мережі, який могли б користуватися військові при ядерному нападі на країну. Була запропонована концепція мережі, що базується на двох основних ідеях:

– відсутність центрального комп'ютера – всі комп'ютери мережі рівноправні;
– пакетний спосіб передачі файлів по мережі.

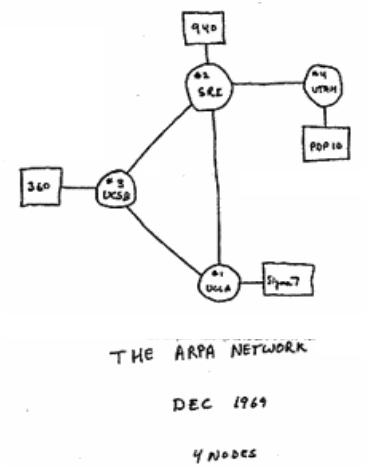



Рисунок 2.11. Документальний ескіз ARPANet, що складається з чотирьох вузлів

Цю концепцію в 1962 році запропонував Підлогу Бэрен, що використала теорію пакетного пересилання файлів, висунуту Леонардом Клейнроком в 1961 році (рис. 2.12).



Рисунок 2.12. Леонард Клейнрок

Ще одним теоретичним джерелом створення мережі стала концепція "Галактичної мережі" Джозефа Ликлайдера.

Згідно цієї концепції за допомогою мережі будь-яка людина з будь-якої точки Землі може отримувати інформацію і обмінюватися файлами з будь-якою іншою людиною. Сьогодні можна сказати, що ця концепція утілилася в сучасній мережі Інтернет.

У 1966 році було почато створення комп'ютерної мережі Арпанет. У жовтні 1967 року англійський вчений Дональд Девіс, що дослідив питання пакетної пересилки файлів, уперше застосував термін "пакет".

У жовтні 1967 року для створення Арпанет було вирішено використати концепції П. Бэрена й Дж. Ликлайдера.

Суть ідеї П. Бэрена полягає в тому, що файл, що потрібно передати по мережі, розбивається на кілька частин – пакетів. Кожен пакет передається незалежно від інших. На кінцевому пункті в комп'ютері всі пакети збира-



Рисунок 2.13.
У 1962 році у рамках Агентства ARPA були початі роботи по комп'ютерних проектах. Керівником комп'ютерної програми був призначений Джозеф Ликлайдер (1915-1990)

ються в один файл. Тому що пакети передаються незалежно, те кожен пакет може дійти до кінцевого комп'ютера по своєму шляху.

Щоб мережа, що складається з рівноправних комп'ютерів, працювала, кожному комп'ютеру привласнюється ім'я, і в кожен комп'ютер записується таблиця імен всіх комп'ютерів мережі й таблиця з'єднань. Завдяки цим відомостям кожен комп'ютер "знає", по якому шляху направити пакет. Спочатку перевіряється найкоротший шлях, якщо він зайнятий або зруйнований, то перевіряється наступний найбільш короткий шлях і т.д. Після того, як пакети потраплять на оконечный комп'ютер, перевіряється наявність всіх пакетів, що становлять файл. Якщо якого-небудь пакета не вистачає, комп'ютер надсилає запит на комп'ютер-відправник і повідомляє який пакет відсутній. Потрібний пакет заново посилає адресатові. Всі правила кодування й пересилання файлів записуються в мережному протоколі.

Для мережі Арпанет був створений протокол IP, що дозволяв ділити файли на пакети й передавати пакети від вузла до вузла.

К январю 1971 года сеть Арпанет насчитывала 13 компьютеров, а к апрелю 1972 года – 23 компьютера.

Другою датою народження Інтернету прийнято вважати 1983 р. Цього року сталися революційні зміни в програмному забезпеченні комп'ютерного зв'язку. Проблема стійкості глобальної мережі була розв'язана впровадженням протоколу TCP/IP, що лежить в основі всесвітньої мережі по нинішній день. Вирішивши, нарешті, це завдання, управління DARPA припинило свою участь в проекті і передало управління мережею Національному науковому фонду (NSF), який в США виконує роль нашої Академії наук. Так в 1983 р. утворилася глобальна мережа NSFNET. В середині 80-х до неї почали активно підключатися академічні і наукові мережі інших країн, наприклад академічна мережа Великобританії JANET (Joined Academic Network).

Для об'єднання мереж, що працюють по протоколі IP і мереж, що працюють по інших протоколах, необхідно було створити спеціальний межсетевой протокол. Цей протокол був створений Винсентом Серфом і Робертом Каном (рис. 2.14) в 1974 році й названий TCP. Протокол TCP, забезпечував передачу пакетів між комп'ютером-відправником і комп'ютером-приймачем, цей протокол дозволяв також досилати загублені

пакети. Всі ці властивості протоколу TCP дозволили використати його для міжмережевого обміну файлами.

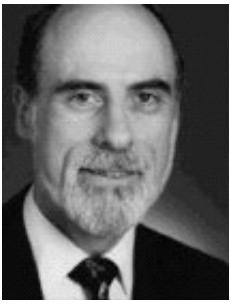




Рисунок 2.14. Винсент Сёрф, "батько Internet" і Роберт Кан

**Internet** – сукупність телекомунікаційних, комп'ютерних мереж, з'єднаних між собою протоколом IP й інших, аналогічних йому протоколів. Протокол обміну даними – набір правил й угод, використовуваних для взаємодії між комп'ютерами, підключеними до мереж передачі даних.

**Стік протоколів TCP/IP** – це первинний протокол глобальних мереж, використовуваний в Інтернет, і на сьогоднішній день один з найпоширеніших стеков транспортних протоколів обчислювальних мереж. Основними протоколами стека, що дали йому назва, є протоколи TCP й IP. TCP – протокол транспортного рівня із установленням з'єднання, а IP – межсетевой протокол Інтернет.

Після об'єднання в 1982 році двох протоколів TCP і IP в один протокол TCP/IP, він став стандартним протоколом об'єднаної мережі – Інтернет. У цьому ж році Серф і його колеги ввели термін "Інтернет". Сьогодні Вінсента Серфа називають "Батьком Інтернету".

Необхідно підкреслити кілька найважливіших подій в історії Internet, які, видимо, і визначили її бурхливий ріст і зростаюче значення в житті сучасного суспільства.

Ріст числа комп'ютерних мереж поза ARPANET створив більші труднощі, що стосуються з'єднання однієї мережі з інший, не через розходження в апаратній частині, а в силу несумісності протоколів комунікації. Для їхнього подолання був створений новий стандарт комунікацій, названий TCP/IP (Transmission Control Protocol/Internet Protocol – протокол керування передачею /межсетевой протокол), що дозволив різним мережам з'єднуватися один з одним.

Іншою важливою подією в історії Internet з'явилося створення "Всесвітньої павутини" (World Wide Web, або WWW, або W3). В процесі роботи Роберт Каллио прийшов до необхідності використовувати гіпертекст, а далі до WWW залишався один крок. І він був зроблений разом з Бернерсом-Ли (Tim Berners – Lee) (рис.2.15), фізиком Європейської організації ядерних досліджень (ЦЕРН) у Женеві (1989 рік). Ідея полягала в тім, щоб дозволити фізикам й іншому вченим користуватися розподіленою в мережі Internet інформацією більше простим способом. Існуючі тоді засобу Internet жадали від користувачів чималих знань про мережі, тому



Рисунок 2.16.
Зліва направо: А. В. Тихонравов,
В. В. Воеводин, Р. Калліо



Рисунок 2.15.
Тим Бернерс-Ли

був розроблений новий метод передачі й відображення інформації. Після опублікування ЦЕРН специфікацій (набору правил і вимог) для WWW користувачі стали писати програмне забезпечення для клієнтів і серверів WWW, що привело до створення "Всесвітньої павутини" у тім виді, у якому ми її знаємо сьогодні (принцип гипермедиа).

Назву загальному дітищу – World Wide Web ("Всесвітня павутина") – придумав Тім, а Роберту належить вибір зеленого кольору в логотипі WWW.

І, нарешті, третьою важливою подією в історії Internet була розробка групою програмістів з Національного центра комп'ютерних додатків (NCSA) на чолі з Марком Андриссеном спеціальної програми "Мозаїка", що остаточно зробила Internet загальнодоступної. "Мозаїка" дозволила користувачам одержувати й відображати документи простим натисканням кнопки "миші". Відпала необхідність піклуватися про перемикання програм і перетворення файлів – "Мозаїка" обробляла документи, графіки, зображення й звуки автоматично, забезпечуючи легкий доступ до "Павутини". Крім того, "Мозаїка" забезпечувала доступ і до інших ресурсів Internet.

На початку 1980 р. стали з'являтися нові мережі, обслуговуючі інші співтовариства, групи і організації. Наприклад, мережа BITnet – мережа національного масштабу для академічних кругів або мережа Csnet, що об'єднала дослідників по обчислювальній техніці і програмуванню. Спочатку ці мережі не були частиною Internet, але з часом були приєднані до Internet, що дозволило користувачам цих мереж обмінюватися інформацією.

Система ARPAnet пов'язувала науковців з видаленими комп'ютерними центрами. У 1983 році ARPAnet розділилася на дві мережі – ARPAnet і MILnet. Система MILnet була зарезервована для військового використання, тоді як система ARPAnet використовувалася для мирних (дослідницьких)

цілей. Існувала можливість обміну інформацією між двома мережами, і це об'єднання стало відомо під назвою Internet.

1986 рік був особливо важливим для розвитку Internet. Саме цього року Національний Науковий фонд зв'язав учених на всій території США з п'ятьма суперкомп'ютерними центрами. Високошвидкісні мережі, сполучаючі ці суперкомп'ютери, утворюють базову мережу, відому під назвою NSFnet. Вона є главою частиною Internet і здійснює основну роботу в мережі по передачі повідомлень і даних. Інші країни одночасно розвивали свої комп'ютерні мережі і мали можливість з'єднання з NSFnet/Internet.

В даний час "хребтом" Internet є наступна група базових мереж: MBONE і ANSnet в США, а в Європі – NORDUnet, EUROPAnet і EUNET.

### 2.3.2 Структура Internet. Способи доступу до Internet

#### 2.3.2.1. Структура Internet

В кінці 70-х років досягла свого розквіту технологія локальних мереж. На ринку з'явилася величезна кількість різних типів локальних мереж. Це привело до того, що багато хто з них був просто несумісний.

Паралельно технології локальних мереж в той же період одержала помітний розвиток технологія мереж широкої зони (WAN-Wide Area Network). На відміну від локальної мережі мережа широкої зони ставила своїм завданням з'єднати комп'ютери, що знаходяться географічно далеко один від одного. Кожний з комутованих комп'ютерів був пов'язаний із спеціальним комп'ютером, який мав вихід в мережу. Ситуація в технологіях широких мереж складалася аналогічно ситуації з локальними мережами.

Було вирішено знайти спосіб, що дозволяє адаптувати один до одного об'єднані мережі. Цей проект одержав назву Internet.

У вузькому значенні слова інтернет – це об'єднання мереж. Проте останніми роками у цього слова з'явився і ширший сенс: всесвітня комп'ютерна мережа. Інтернет можна розглядати у фізичному сенсі як мільйони комп'ютерів, пов'язаних один з одним всілякими лініями зв'язку, проте такий "фізичний" погляд на Інтернет занадто вузький. Краще розглядати Інтернет як деякий інформаційний простір.

Інтернет – це не сукупність прямих з'єднань між комп'ютерами. Так, наприклад, якщо два комп'ютери, що знаходяться на різних континентах, обмінюються даними в Інтернеті, це зовсім не означає, що між ними діє одне пряме з'єднання. Дані, які вони посилають один одному, розбиваються на пакети, і навіть в одному сеансі зв'язку різні пакети одного повідомлення можуть пройти різними маршрутами. Якими б маршрутами не рухалися пакети даних, вони все одно досягнуть пункту призначення і будуть зібрані разом в цілісний документ. При цьому дані, відправлені пізніше, можуть

приходити раніше, але це не перешкодить правильно зібрати документ, оскільки кожен пакет має свою маркіровку.

Internet використовує спеціальні комп'ютери для з'єднання мереж. Звичайне це цілком стандартні комп'ютери з процесором, пам'яттю і комутаційною платнею. Єдина відмінність полягає в тому, що на них встановлені спеціальні програми, відповідальні за зв'язок між мережами. Одержавши пакет інформації від однієї мережі, цей комутаційний комп'ютер передає її іншої мережі. Цей спеціальний комп'ютер називається ***маршрутизатором*** (router).

Важливо, щоб передавана інформація підкорялася певним стандартам. Без них маршрутизатор був би не в змозі зрозуміти і переробити інформацію, що поступає.

Проте більшість людей не можуть собі дозволити прямого підключення в одну з мереж Internet, оскільки це дорого, тому для виходу в Internet персональний комп'ютер повинен мати ***модем***, який під'єднується до звичайної телефонної мережі, і спеціальну програму для управління модемом.

Хоча Internet і не є єдиною глобальною мережею, в практиці спілкування поняття Internet означає взагалі всі мережі (а швидше – ресурси) за межами нашої локальної мережі. Тому ми говоритимемо про сервіси глобальних мереж загалом, незалежно від того, де фізично знаходиться потрібний нам ресурс чи сервер (рис.2.17).

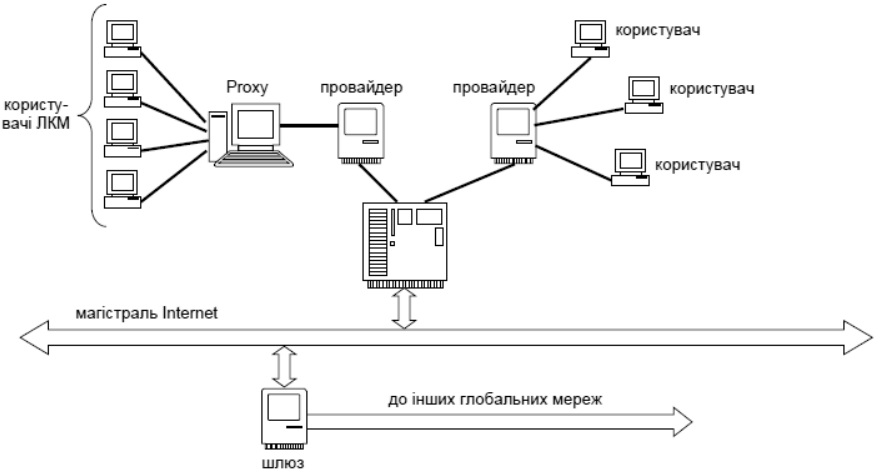



Рисунок 2.17. Структура підключення користувачів до мережі Internet

Магістраль – це лінії зв'язку, які можуть бути кабельними (здебільшого так воно і є), чи супутниковими радіолініями. Зі схеми

рисунку видно, що магістраль з'єднує між собою вузли мережі (хости). До хостів підключаються сервери провайдерів – організацій, які надають послуги з доступу до Internet. До серверів провайдерів підключаються або індивідуальні користувачі (на схемі – справа), або через власні проксі – сервери (proxy-server) – колективні користувачі (локальні чи корпоративні мережі). Доступ до ресурсів інших глобальних мереж (FidoNET, BitNET, CompuServe тощо) здійснюється через спеціальні комп'ютери-шлюзи.

З появою недорогих і могутніх персональних комп'ютерів стан справ істотно змінився, тому що тепер персональні комп'ютери можуть самі безпосередньо підключатися до мережі, отримувати з неї потрібні початкові дані, а обчислення проводити самостійно. Потрібно відмітити, що під обчисленнями розуміє будь-яка робота комп'ютера, навіть гра. У спадок від тих часів, коли обчисленнями займалася головна обчислювальна машина, до якої зверталися через термінал, залишилися терміни термінал (terminal) і хост (host), або головна обчислювальна машина. Проте тепер в терміни "хост" і "термінал" укладається дещо інше значення. **Хост** – це будь-який комп'ютер, який є постійною частиною Internet, тобто сполучений по Internet – протоколу з іншим хостом, який у свою чергу, сполучений з іншими, і так далі. **Термінал** – це комп'ютер, який не є постійною частиною Internet, але який може під'єднуватися до хосту.

Як термінал можна використовувати персональний комп'ютер, запустивши на нім спеціальну програму, яка примушує його сприймати ті ж команди і посилати ті ж у відповідь сигнали, що і справжній термінал. Така програма називається програма-емулятор (terminal emulator). Емулятором також називається будь-яка програма (чи пристрій), яка моделює в повному або неповному об'ємі реакції якого-небудь іншого пристрою на сигнали, що управляють. У Internet і в програмах емуляції найчастіше використовується тип терміналу VT100 і ANSI. Вони були найпопулярнішими терміналами за часів народження Internet, і їх можливості підтримуються більшістю емуляторів.

### 2.3.2.2 Способи доступу до Internet

Для роботи в Internet у першу чергу необхідно мати фізичне (провідне) з'єднання з іншими комп'ютерами цієї мережі. Таке з'єднання може бути організовано одним з наступних способів:

**Через телефонну мережу за допомогою модему**. Цей спосіб найчастіше застосовується для підключення комп'ютерів, що знаходяться в малих офісах і будинках. Для реалізації такого доступу використовується утиліта **Віддалений доступ до мережі (Dial-Lp Networking).** Зв'язок через телефонний канал може бути погодинний (при встановленні зв'язку необхідно щораз додзвонитися на сервер по звичайній телефонній лінії за

допомогою модему) і по виділеному каналу (телефонній лінії, зв'язок по якій від абонента до провайдера встановлений постійно).

**Через лінії зв'язку нових типів**. Цей спосіб передбачає з'єднання із сервером через лінії ISDN, AUSL чи супутниковий канал.

**Через одну з комерційних мережних служб**, наприклад, CompuServe, America Online чи Microsoft Network. Ці служби надають вихід ("шлюз") у Internet, скористатися яким можна за допомогою ключового слова "Interne" у меню цих служб.

**Через локальну мережу організації**. Якщо локальна мережа підключена до Internet, то цим підключенням може скористатися будь-який комп'ютер з цієї мережі (якщо це не заборонено адміністратором мережі). Щоб довідатися, як вийти в Internet з локальної мережі, необхідно звернутися до мережного адміністратора.

У двох останніх випадках користувач не повинний виконувати ніяких дій по підключенню, оскільки з'єднання з Internet вже існує. У першому випадку необхідно спочатку підключитися до мережі.

Доступ до Internet здійснюється за допомогою фірми, що займається підключенням користувачів до Internet постачальника (провайдера) послуг Internet (ISP – Internet Service Provider). Провайдери звичайно пропонують декілька наборів послуг, що різняться за швидкістю з'єднання і можливістю роботи з різними службами. Набір цих умов заноситься в реєстраційний запис клієнта, що носить назву **облікового запису** або **рахунку (account).**

Найбільш загальними типами рахунків є рахунок з наданням доступу до інтерактивних (on-line, онлайнових) служб, при роботі з якими взаємодія користувача зі службою відбувається в реальному часі, і не інтерактивними (off-line, автономними) службами, спілкування з якими відбувається у формі переписування. World Wide Web є інтерактивною службою, а електронна пошта – не інтерактивною. Рахунок на онлайповий доступ завжди дорожче, але надає більш широкі можливості. Крім того, онлайновий рахунок звичайно містить у собі також послуги електронної пошти.

Перед установкою доступу до Internet через телефонну лінію необхідно придбати, установити і настроїти модем. З цього питання найкраще звернутися до книги Д. Кірсанова "Факс-модем: від покупки і підключення до виходу в Internet". Після цього необхідно встановити зв'язок з фірмою-провайдером і укласти договір про надання послуг. Після цього фірма повинна надати дані, необхідні для подальшого конфігурування з'єднання користувачем, або навіть виконати ці роботи самостійно з виїздом фахівця (як правило, за додаткову плату).

**День домена. UA**
1 грудня 1992 року – дата офіційного делегування Україні домена .UA.

### 2.3.3 Системи адресації в internet. Поняття IP-адреси та доменної адреси, протоколу DNS

Для того, щоб вирішити проблему обліку мільйонів ПК, сполучених в єдину Мережу, Internet використовує унікальні коди – число і ім'я, які привласнюються кожному комп'ютеру. Звичайне це імена, але іноді зустрічаються і числа (рис. 2.18).

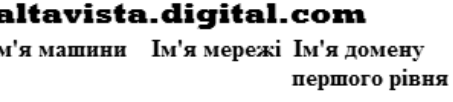



Рисунок 2.18. Приклад імені домена в мережі

Як вже було сказано, TCP/IP – це мережевий протокол, необхідний для повного використання можливостей Internet і єдиний спосіб для доступу до них. Кожен комп'ютер в мережі TCP/IP повинен мати унікальний ідентифікаційний код. Таким чином, кожен комп'ютер в Internet ідентифікується за унікальним числовим кодом, який називається IP-номер. Цей номер складається з чотирьох чисел із значеннями від 0 до 255, розділених крапками – XXX.XXX.XXX.XXX. Така схема нумерації дозволяє мати в Мережі більше чотирьох мільярдів комп'ютерів.

Коли локальна мережа або окремий комп'ютер вперше приєднуються до Internet, спеціальна організація привласнює їм IP-номери. Це гарантує унікальність IP-номерів і правильність підключення.

Спочатку підключення до інших хостів в Internet здійснювалося по IP-номерах, але коли в 1984 р. число машин перевалило за 1000 і цей спосіб став незручним, був прийнятий метод зв'язку імен і IP-номерів, який називається сервер імені домена (Domain Name Server, DNS). Сервер DNS підтримує список імен локальних мереж і комп'ютерів і відповідних їм IP-номерів. Оскільки серверу DNS доводиться працювати із списком тільки локальних мереж і хостів, цей список виходить набагато меншим, ніж у випадку, якщо б сервер зберігав інформацію про кожен комп'ютер Internet. Завдяки цій угоді про імена запам'ятати адресу комп'ютера в Internet набагато легше, ніж його IP-номер. Типове ім'я домена (domain-група ресурсів, керованих одним вузлом) складається з декількох частин, розташованих

в певному порядку і розділених крапками. На рис. 2.18 показані приклади імені домена і IP-номера.

DNS не ввів нову систему адрес, а встановив відповідність між наявними числовими IP – адресами і новими буквеними DNS – іменами, які стали рівноправними. Тепер з'явилася можливість давати осмислені імена в Інтернеті.

Розглянемо структуру DNS - імені.

У системі DNS вводиться поняття домену.

Доменом називається частина мережі, що складається з сервера і підключених до нього комп'ютерів-клієнтів.

Вводиться також поняття домена вищого рівня (по-англійськи TLD – Top Level Domain).

Домен вищого рівня – це частина мережі, логічно об'єднана за якою-небудь ознакою.

Домени вищого рівня діляться на два види – тематичні і домени кодів країн. Коди країн доменів вищого рівня складаються з двох букв, наприклад, код Росії .ru.

Ім'я домена верхнього рівня указує на домен конкретного комп'ютера. Це ім'я звичайно вибирають так, щоб відобразити тип організації, в якій встановлений цей комп'ютер. І хоча дотримуватися вимоги про імена необов'язково, це вважається хорошим тоном. До 2001 року імена тематичних доменів вищого рівня, або просто імена доменів вищого рівня, складалися з трьох букв.

Основні імена доменів верхнього рівня:

com – commercial (комерційні);

edu – educational (освітні);

gov – goverment (урядові);

mil – military (військові);

net – network (організації, що забезпечують роботу мережі);

org – organization (некомерційні організації).

В 2001 було прийнято рішення зняти це обмеження і ввести нові імена доменів вищого рівня.

У 2002 році почали діяти 7 нових імен:

| | |
|---|---|
| aero | Індустрія авіаційного транспорту |
| biz | Бізнес |
| coop | Кооперативні організації |
| info | Інформаційні ресурси |
| museum | Музеї |
| name | Особисті, іменні ресурси |
| pro | Професійні ресурси |

Слід зазначити, що ці імена доменів використовуються тільки в США; у інших країнах використовується небагато інша структура найменування – замість типу організації ім'я домена верхнього рівня означає країну, тип організації указується в секції зліва від імені верхнього рівня, причому в різних країнах світу прийняті різні імена для типів організацій. Деякі імена доменів верхнього рівня для країн:

au – Australia (Австралія);
by – Belarus (Білорусія);
ca – Canada (Канада);
de – Germany (Германія);
dk – Denmark;
fr – France (Франція);
il – Israel (Ізраїль);
jp – Japan (Японія);
ru – Russia (Росія);
su – Soviet Union (Радянський Союз – підтримується, але не розподіляється);
ua – Ukraine (Україна);
uk – United Kingdom (Сполучене Королівство Великобританія / Англія).

Домени вищого рівня включають інші домени. Наприклад, домен вищого рівня com містить у своєму складі усі домени, що відносяться до комерційних організацій. Ці домени називаються доменами першого рівня. У адресі сервера, вони записуються зліва від імені домену вищого рівня і відділяються точкою.

Домени першого рівня також можуть мати у своєму складі домени – домени другого рівня і так далі

Таким чином, доменна структура – це логічна матрьошка: кожен наступний домен знаходиться усередині попереднього.

Розглянемо приклади доменних імен:

1) Ім'я intel.com Ім'я домену вищого рівня – com, ім'я домену першого рівня – intel

2) Ім'я rnd.runnet.edu Ім'я домену вищого рівня – edu, ім'я домену першого рівня – runnet, ім'я домену другого рівня – rnd

3) fbm.ru Код країни домену вищого рівня – ru, ім'я домену першого рівня – fbm




Розподілом доменних імен вищого рівня займається некомерційна організація ICANN (Internet Corporation for Assigned Names and Numbers). Керує організацією "батько Інтернету" Вінсент Серф.

Ім'я мережі кожен обчислювальний центр вибирає для себе самостійно, а потім реєструє його в тій орга-

нізації Internet, яка забезпечує підключення. Це ім'я звичайно містить інформацію про обчислювальний центр і в межах домена верхнього рівня повинно бути унікальним. Наприклад, на рис. 2.18 ім'я мережі digital означає приналежність до відомої комп'ютерної корпорації DEC (Digital Equipment Corporation).

Імена, що знаходяться зліва від імені першої мережі, можуть бути призначені адміністратором локальної мережі. Якщо доступ до якого-небудь хосту здійснюється через іншу мережу, ім'я цієї мережі розташовується зліва від першої мережі. Ім'я машини, що займає крайнє ліве положення, – це ім'я конкретного хосту. Приклад розширеного імені приведений на рис. 2.19.

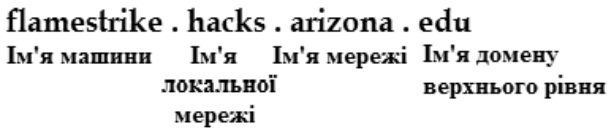



Рис. 2.19. Розширене ім'я домена

Для кінцевого користувача існує в основному два способи підключення до Internet – через електронні дошки оголошень (BBS – Bulletin Board System) і повне підключення по Internet-протоколу. BBS - звичайно складається з ПК, модему з комунікаційною програмою, приєднаних до телефонної лінії і здатних передавати і приймати повідомлення.

Повне підключення по IP-протоколу – це кращий варіант для домашнього комп'ютера. Для того, щоб встановити цей тип з'єднання, необхідно домовитися з організацією-провайдером послуг Internet. Провайдер (provider – постачальник мережевих послуг) – особа або організація, що надає послуги з підключення до комп'ютерних мереж. Провайдер перенаправлятиме інформацію за допомогою мережевого протоколу TCP/IP по телефонних лініях на ваш комп'ютер. Такий спосіб доступу називається Dial-Up і дозволяє повністю управляти виконуваними програмами. При цьому пропускна спроможність даного способу обмежена швидкістю використовуваного модему і якістю телефонних ліній від користувача до провайдера Internet.

Щоб пропустити протокол TCP/IP по стандартному телефонному каналу, інформаційний потік необхідно перетворити у форму, яку можна передати через модем. Для вирішення цього завдання існують два способи. Перший, старіший, називається SLIP (Serial Line Internet Protocol) – протокол Internet для послідовних ліній. Новіший метод – PPP (Point to Point Protocol) – протокол передачі "пункт-пункт" – набагато ефективніше для сучасних телефонних систем і модемів. Існує безліч пакетів програм для

мережі TCP/IP (їх часто називають Internet-пакетами – Internet packages), які підтримують використання SLIP і PPP.

Окрім Dial-Up, способу повного підключення до Internet, існує і доступ по виділених каналах, який використовується звичайно тільки компаніями для підключення своїх локальних мереж до Internet, використовуючи постійні канали зв'язку, що орендуються (leased line). Хоча в порівнянні з Dial-Up, доступом по звичайних телефонних лініях, цей спосіб не дає нових можливостей, його перевагами є швидший зв'язок і наявність постійного і безперервного з'єднання.

Процес установки на вашому комп'ютері програмного забезпечення може показатися дуже складним, але насправді це не так. Звичайно провайдери Internet при реєстрації нового абонента видають йому спеціально написаний пакет програм, який автоматично інсталює (встановлює) необхідні програми на машину абонента. При цьому всі настройки вже встановлені провайдером або також відбуваються автоматично.

При реєстрації ви одержите від свого провайдера настановні параметри:

IP-адрес (IP-address);

Мережева маска (Netmask);

Сервер імені (Nameserver);

Шлюз за умовчанням (Default Gateway),

Сервер часу (Jimeserver);

Суфікс (ім'я) домена (Domain Suffix);

Номер телефону модему провайдера;

Швидкість роботи модему (Baud Rate);

Послідовність входу в систему.

Розглянемо ці параметри докладніше.

IP-адреса – це дійсна адреса вашого комп'ютера в Internet, задана у формі XXX.XXX.XXX.XXX, де кожна група XXX може бути числом від 0 до 255. Проте останнім часом провайдери Internet все частіше використовують так званий DHCP (Dynamic Host Configuration Protocol) – протокол динамічної конфігурації хосту – спосіб отримання яким-небудь пристроєм змінної IP-адреси від сервера. Він дозволяє управляти розподілом IP-адрес і дуже гнучкий – пристрій в одному сеансі може одержати одну IP-адресу, а в іншому сеансі – інший.

Мережева маска (Netmask). З її допомогою TCP/IP визначає розмір локальної мережі, що дозволяє при необхідності розповсюдити по ній які-небудь дані. Маска показує, в якому діапазоні лежать IP-адреси комп'ютерів, що входять в дану локальну мережу, і, отже, здатних прийняти ці дані. Звичайно мережева маска задається у форматі 255.255.255.0 або 255.255.0.0, але конкретні числа можуть бути будь-якими від 0 до 255.

Сервер імені (name server) – це IP-адреса комп'ютера, який переводить текстові адреси Internet в числову форму. Наприклад, адреса

d85.sitek.net він перетворить в 195.212.225.85. Internet використовує ці числові значення, оскільки кожній включеній в Мережу машині привласнюється однина.

Сервер часу (timeserver) – це IP-адреса комп'ютера, що містить точний час. Деякі мережеві програми можуть запрошувати цей сервер, а потім встановлювати одержаний час на внутрішньому годиннику персонального комп'ютера. В цьому випадку, поле може залишитися порожнім.

Суфікс (ім'я) домена (Domain Suffix) – це частина вашої адреси електронної пошти, що стоїть після імені користувача, символу @ і імені хосту. Наприклад, якщо адреса електронної пошти – askor@mastak.sitek.ru, то суфікс домена – sitek.ru.

Номер телефону модему провайдера – це телефон, по якому ваш модем може подзвонити (Dial-Up), з'єднатися з модемом провайдера і підключитися до Internet. Звичайно він вже встановлений провайдером у ваше програмне забезпечення.

Швидкість роботи модему (Baud Rate) – це максимальна швидкість, з якою модем може передавати дані, звичайно указувана на корпусі модему або його упаковці. У принципі, в цьому полі можна виставити максимально допустиму даною системою або програмою швидкість. У будь-якому випадку реальний зв'язок встановиться з максимально можливою за даних умов швидкістю залежно від шуму телефонної лінії (line noise) між вами і провайдером і швидкостей ваших модемів.

Послідовність входу в систему. Подразумеваются визначення імені користувача (userid або user name або name account або login), пароля для входу (password) в систему, використовуваного протоколу (SLIP або PPP) і інші відповіді на можливі питання системи. Звичайно провайдером пишеться спеціальний сценарій (script) входу в систему, в який користувачу необхідно внести тільки свої ім'я і пароль. Цей сценарій можна редагувати звичайним текстовим редактором типу Блокнот (Notepad) з пакету Windows.

## 2.4 Основні послуги Internet

19 сентября
**День народження "смайла"**
19 вересня 1982 р., професор Університету Карнеги-Меллона Скотт Фалман (Scott E. Fahlman) уперше запропонував використовувати три символи, що йдуть підряд, – двокрапку, дефіс і закриваючу дужку :-) – для позначення "усміхненого обличчя". Тепер – за допомогою смайликов в звичайному тексті – стало можливо передавати свої емоції.

### 2.4.1 Електронна пошта

До основних сервісів телекомунікаційних технологій відносять електронну пошту, передачу файлів, телеконференції, довідкові служби (дошки оголошень), відеоконференції, доступ до інформаційних ресурсів (інформаційним базам) мережевих серверів, мобільний стільниковий зв'язок, комп'ютерну телефонію.

**Електронна пошта** *(electronic mail, email)* – це метод передачі поштових повідомлень (листів) електронним способом. За допомогою електронної пошти можна передати повідомлення в будь-який кінець світу не більше ніж за добу. Щоб використовувати можливості email, необхідно мати доступ до машини, включеної в Internet. Це можна зробити за допомогою комунікаційних програм через поштову BBS (електронну дошку оголошень) або через домашній комп'ютер, підключений до Internet в режимі DIAL-UP.

Існують два різних методах пересилки пошти. Частина електронної пошти передається з використанням методу з проміжним зберіганням даних (store-and-forvard). Цей метод звичайно використовується, коли ***хост*** підключається до Internet періодично. Повідомлення чекає наступного підключення ***хосту*** на вашому поштовому сервері Internet. Якщо це відбувається, проводиться обмін вхідними і витікаючими повідомленнями, і весь процес повторюється. При використанні цього методу повідомлення може йти до адресата годинник і навіть дні.

Швидший метод полягає у використанні поштовими серверами для передачі пошти протоколу SMPT (Simple Mail Transfer Protocol – простий протокол передачі пошти). В цьому випадку при створенні поштового повідомлення поштовий сервер відразу ж вступає в контакт з адресатом і відправляє повідомлення. Електронна пошта, що посилається таким методом, може обійти земну кулю за секунди. Метод SMTP використовується вже досить давно, але існує ще безліч поштових серверів, які застосовують

метод з проміжним зберіганням. Наприклад, в Росії велика частина поштових серверів працює таким чином. Насправді програми, що застосовують SMTP, можуть використовувати обидва методи передачі пошти.

Для відправки повідомлення по електронній пошті вам необхідно знати тільки одне – адреса одержувача. Як і будь-яка інша адреса, адреса в електронній пошті полягає з двох частин: "кому" і "куди". "Кому" – це ідентифікатор (ім'я) користувача, "куди" – ім'я домена його комп'ютера. Для розділення цих двох частин використовується символ @ (вимовляється "ет", іноді його називають собакою). Якщо ви відправляєте лист комунебудь, хто підключений до того ж хосту, що і ви, то ім'я домена указувати не обов'язково, оскільки хост автоматично доставить повідомлення локальному користувачу із заданим ім'ям. Нижче приведений приклад стандартного формату адреси електронної пошти в Internet:

**ім'я_користувача@машина.ім'я_мережі.ім'я_домену_верхнього_рівня.код_країни**

name@root.icenet.com.ru

Останнім часом набули широкого поширення безкоштовні поштові сервери, реєструється на яких, можна одержати в своє розпорядження поштова скринька і використовувати його на свій розсуд. Як приклади можна привести наступні служби:

www.mail.ru;
www.hotbox.ru;
www.inbox.ru;
www.hotmail.com та інш.

Для роботи з електронною поштою можна використовувати спеціальні поштові програми, наприклад, The Bat, Microsoft Outlook, Outlook Express.

### 2.4.2 Мережеві новини Usenet

Іншим корисним ресурсом Internet є *мережеві телеконференції Usenet ("юзнет")*. Це величезна електронна дошка оголошень, в якій може брати участь будь-який користувач Internet, що базується на повідомленнях. Ця система дозволяє користувачам, що живуть в різних частинах миру, брати участь в дискусіях по тисячах різних тем. Ця технологія аналогічна електронній пошті, де адресатом є не конкретний користувач, а *телеконференція (newsgroup)* на задану тему.

Звичайно доступ до новин *Usenet* здійснюється в режимі підписки, тобто ви посилаєте по спеціальній email-адресі запит на сервер, використовуючи всього одну або дві команди. Наприклад, команда LIST примусить сервер прислати вам список доступних телеконференцій, а команда HELP – список команд даного сервера, команд підписки на

(або відписки від) необхідну конференцію. Після того, як ви пошлете серверу команду "підписатися" на задану конференцію, він буде у міру надходження в конференцію нових статей посилати їх у вашу поштову скриньку. Так само по електронній пошті і ви можете послати свою статтю на конференцію. Цей спосіб достатньо громіздкий і вимагає деяких знань, тому в даний час доступ до Usenet можна одержати при повному підключенні до Internet по TCP/IP-протоколу. Приєднавшись до Мережі, слідує потім підключитися до сервера *Usenet* і запустити на своєму ПК спеціальну програму для читання повідомлень. Сервер *Usenet* спілкується з іншими серверами *Usenet* і обмінюється з ними будь-якими новими вхідними і витікаючими повідомленнями. Програма читання повідомлень показує вам список конференцій, до яких ви приєдналися або з яких вже вийшли, і відзначає, чиї повідомлення ви вже переглянули. Потім програма відображає повідомлення на екрані і дозволяє вам відповісти на них. Відмітимо, що повідомлення *Usenet* завжди загальнодоступні: мільйони людей у всьому світі можуть прочитати ваше повідомлення в тій конференції, в яку ви його послали.

*Телеконференція* – це своєрідний журнал, що публікує виключно листи читачів. Його автори є також його читачами. Звичайно, виходить це видання без цензури і без редакторської правки. Правда, існують так звані *модератори (moderator)* – добровольці, які перевіряють листи на відповідність заданій темі конференції і перенаправляють їх. Автором може стати будь-який. Середньостатистичному користувачу Internet доступні 2-3 тисячі різних телеконференцій, а всього їх близько 10 тисяч. Але якщо серед різноманіття тим немає тієї, єдиної, то можна організувати її самостійно, адже один з головних моральних принципів Internet звучить приблизно так: "Індивідуалізм вітається". А щоб однодумцям було легше знайти один одного, конференції діляться на декілька основних категорій:

- Alt – альтернативні теми;
- Comp – обчислювальна техніка и суміжні області;
- Misk – всяка всячина;
- News – корисна інформація для новачків в Usenet;
- Rec – розважальна (музика, спорт, ігри);
- Soc – соціальні питання (культура, релігія и т.д.);
- Talk – дискурси на загальнолюдські теми;
- Sci – наука і техніка (окрім комп'ютерної сфери).

Кожна з цих широких тем розбивається на декілька вужчих напрямів, що відповідають якимсь конкретним інтересам, наприклад REC.MUSIC. ABBA. У *Usenet* існують і інші теми, але з їх назв важко зрозуміти, чому вони присвячені. Конкретні дії при приєднанні до якої-небудь конференції залежать від використовуваної програми, але найчастіше це пункт меню

"приєднатися до конференції (join newsgroup)", або "додати конференцію (Add newsgroup)", або команда серверу "підписатися на конференцію (subscribe to newsgroup)". На відміну від інших конференцій групи ALT ніхто не обслуговує. Будь-який обчислювальний центр, що має доступ до *Usenet*, може створити групу ALT. Через відсутність контролю і величезної кількості передаваних даних (більше 45 Мбайт щодня) багато обчислювальних центрів блокують пересилку всіх груп ALT.

### 2.4.3 Протокол передачі файлів (FTP)

Ще одним ресурсом Internet є *FTP (File Transfer Protocol – протокол передачі файлів),* що дозволяє користувачам копіювати файли з одного приєднаного до Internet комп'ютера в інший. Програмне забезпечення FTP розділене на дві частини: одна частина виконується на комп'ютері, який містить файли (FTP-сервері), а інша на комп'ютері, якому ці файли потрібні (клієнті). Клієнтом може бути будь-яка приєднана до Internet BBS або навіть вашого локального персонального комп'ютера. Дві частини програми спілкуються один з одним і пересилають файли. Щоб використовувати FTP, ви повинні видати на FTP-сервер команди UNIX (UNIX – це операційна система, використовувана на багатьох великих комп'ютерах). FTP – дуже могутній засіб, але ви не зможете ним користуватися, не реєструється на FTP-сервері. Проте існує анонімний FTP. Для того, щоб з'єднатися з сервером анонімного FTP, вам досить увійти з ім'ям *anonimous.* Пароль вводити не потрібно, тому, коли система попросить його ввести, бажано набрати свою email-адресу (це вважається хорошим тоном). Хоча анонімний доступ популярний, його робота не гарантується. Багато обчислювальних центрів обмежують кількість анонімних користувачів, які одночасно можуть увійти до системи, щоб забезпечити вільний доступ до сервера для локальних користувачів.

### 2.4.4 Мурашка (Archie)

Кількість файлів, доступних для FTP, – величезно і постійно росте. Зараз можна одержати мільйони файлів – від дослідницького звіту до кулінарного рецепту, тому знайти те, що вам потрібне, може виявитися непростим завданням. У цьому і покликаний допомогти засіб, званий "Мурашка" (Archie – вимовляється "арчі"). Archie – програма, яка каталогізує файли на більш ніж 1000 серверах Internet. Archie-сервери створюють списки файлів, доступних для FTP, і постійно їх оновлюють. Системою Archie можна користуватися або підключившись до Archie-сервера, або запустити на своєму комп'ютері локальну версію Archie, яка спілкуватиметься з Archie-сервером. *Archie-програми* шукають файли за допомогою *регулярних виразів* – методу пошуку, використовуваного в UNIX.

### 2.4.5 Telnet

Після підключення до Internet ви теоретично можете під'єднатися до будь-якого іншого комп'ютера в Мережі. Для цього необхідно мати дозвіл і видалений комп'ютер повинен підтримувати це з'єднання. На відміну від FTP, який дозволяє тільки копіювати файли з сервера і на нього, Telnet – це протокол емуляції терміналу, який дозволяє підключатися до видаленого комп'ютера по Internet і управляти ним. Він дозволяє з'єднуватися з видаленим комп'ютером так, як ніби ви працюєте безпосередньо з ним. З погляду видаленого комп'ютера ваш персональний комп'ютер аналогічний безпосередньо підключеному терміналу.

### 2.4.6 Гіпертекстова система Gopher

*Gopher* – це ієрархічна система меню, що дозволяє легко знаходити інформацію, що є на сервері або на декількох серверах.

*Gopher* – це керована за допомогою меню система доступу в Internet, яка була розроблена в університеті штату Міннесота в 1989 р. У основі Gopher лежить тільки меню, тому DOS-орієнтований текстовий Gopher ні в чому не поступається Windows-орієнтованому графічному. Раніше користувачам необхідно було пам'ятати, який `хост` зберігав ту або іншу базу даних або колекцію файлів. Тепер ці відомості пам'ятають тільки ті люди, які обслуговують Gopher-сервери. Вони пов'язують ці ресурси з пунктами меню, які доступні решті користувачів. Все, що повинен зробити користувач, це вибрати потрібний пункт меню. Після цього він буде сполучений з комп'ютером (який може знаходитися на іншій півкулі), чия інформація йому необхідна. За простим інтерфейсом Gopher ховається дуже складна система доступу до інформації. Вибір пункту меню може привести до підміню, текстовому файлу, бінарному файлу або безпосередньо до іншого Gopher-сервера на іншій машині. Завдяки цьому можна легко проглядати Internet, не запам'ятовуючи команди, приховані за пунктами меню.

### 2.4.7 WAIS – інформаційна система широкого користування

*WAIS (Wide Area Information System)* була створена для забезпечення нескладного послідовного інтерфейсу для пошуку інформації в базах даних, доступних в Internet, – від тих, що зберігають безкоштовну інформацію до спеціалізованих і дорогих. На відміну від Gopher, де ви повинні проглянути декілька меню, перш ніж дістанетеся до хоста, що дозволяє звернутися до потрібної бази даних, WAIS дозволяє підключитися безпосередньо до WAIS-сервера, причому він працюватиме тільки з інформацією, що надається іншими WAIS-серверами. Кінцевий WAIS-сервер підключиться до вибраної вами бази даних, виконає ваш запит і поверне результати по-

шуку. WAIS краще використовувати тоді, коли ви точно знаєте, якого роду інформація вам потрібна. Якщо ви не цілком упевнені, простіше буде скористатися Gopher.

### 2.4.8 IRC (Internet Relay Chat) – "Базікання по Internet"

"*Базікання по Internet" (IRC)* – можливість розрахованого на багато користувачів діалогу по Internet в режимі реального часу за допомогою текстових повідомлень, що набирають на клавіатурі. Набираючий на клавіатурі текст відправляється в деяку область Мережі, звану *каналом (channel*). Цей текст стає доступним для читання всім користувачам, які в даний момент теж приєдналися до цього каналу. Відповідно текст, набраний іншими користувачами в цьому каналі, доступний для читання вам. Ці канали існують на спеціальних IRC-серверах і до них можна підключитися за допомогою безкоштовних програм, які можна одержати через "Всесвітню павутину". Каналів на сервері досить багато – можна вибрати будь-який і приєднатися до нього, але там завжди "багатолюдно". Якщо ви хочете поспілкуватися без перешкод, то можна створити свій власний канал, а потім домовитися з респондентом (по електронній пошті) про назву IRC-сервера, назві каналу і часу підключення до нього.

Останнім часом з'явилася можливість використовувати Internet для прямої розмови (не через текст на дисплеї, як в IRC, а за принципом телефону). Ця можливість стримувалася до недавнього часу низькими швидкостями модемів (тут потрібний не менше 14,4 Кбіт/с), якістю телефонних ліній, а також оперативною пам'яттю комп'ютера (в даному випадку не менше 16 Мбайт). Але тепер ці труднощі подолані, існують також необхідні програми.

### 2.4.9 "ВСЕСВІТНЯ ПАВУТИНА" (WORLD WIDE WEB)



"**ВСЕСВІТНЯ ПАВУТИНА" (WORLD WIDE WEB** або **WWW** – глобальна система гіпертекстових документів, пов'язаних один з одним по Internet.

Служба WWW, звана також веб-сервером, обслуговує запити клієнтів і передає їм інформацію. Використовує протокол HTTP – Hyper Text Transfer Protocol (Гіпер Текстовий Трансфертний Протокол).

"Павутина" містить в собі всі ресурси, про які мовилося раніше. При цьому ви можете працювати тільки з однією *програмою-клієнтом* на своєму персональному комп'ютері, званою *браузером, або переглядачем* (по-англійськи *browser* означає "переглядач"). Це програмний інструмент, що дозволяє працювати зі всіма ресурсами Internet простим і зручним способом. Браузери здатні зв'язуватися зі всіма ресурсами Мережі через єдиний

*призначений для користувача інтерфейс (common user interface, CUI) і* перетворювати блукання по Internet в чудову гру. Комп'ютерні фірми розробили достатньо велику кількість програм- браузеров для Internet. Серед них Netscape *Navigator, MS Internet Explorer, Mosaic, Tango, Ariadna, Cello, Lynx.*

"*Всесвітня павутина",* або *World Wide Web,* починає свою історію з набору правил і вимог, розроблених Європейською організацією ядерних досліджень (ЦЕРН). Щоб уникнути труднощів, що виникали тоді при роботі з Internet (користувач повинен був бути добре підготовленим програмістом), був розроблений новий метод передачі і відображення інформації. Цей метод називається *HTTP (Hyper Text Transfer Protocol – протокол передачі гіпертекстів)* і є надбудовою над TCP/IP – стандартним мережевим протоколом Internet. HTTP працює за стандартною для Internet схемою, коли сервер видає дані, а програма-клієнт їх відображає або виконує.

Концепція гіпертексту, яка лежить в основі "Всесвітньої павутини", була запропонована в 60-х роках Тедом Нельсоном. **Гіпертекст** (Hypertext) – це багатовимірний текст, тобто така організація документів, при якій один документ або текст може включати різноспрямовані посилання або покажчики (адреси) на інші документи і посилання. Такі покажчики і посилання, включені в гіпертекст, називаються *гіпертекстовими посиланнями* або *гіперпосиланнями (Hypertext links, hyperlinks).* ***Гіперпосилання*** – посилання на інший документ – картинка, кнопка, виділене слово в початковому документі, які можуть бути вибрані читачем для отримання більшої інформації. Ці посилання дозволяють проглядати документ в будь-якому необхідному порядку, тобто не обов'язково читати його з початку і до кінця. Таким чином, об'єднання, або сукупність, одновимірних текстів з включеними в них посиланнями один на одного і називатиметься гіпертекстом.

Найпростішим і відомішим прикладом гіпертексту є система контекстної допомоги *Microsoft Windows.* У цій системі користувачі можуть перескакувати з однієї теми в іншу, відзначаючи клацанням миші виділені слова або фрази. Інше клацання миші повертає у вихідну позицію або відводить далі по іншому посиланню.

***Гіпертекст*** – це абсолютно нова форма в порівнянні з традиційними формами документів, наприклад книга, в якій єдиним зв'язком між схожими темами є наочний покажчик. А гіпертекст у багатьох випадках є єдиним зручним способом осмислено вивчити документ. Якщо, наприклад, за допомогою MS Windows використовувати віддруковане на папері керівництво, то доведеться паралельно читати текст і відстежувати посилання індексу (допустимо, що він є), тобто перегортати керівництво до того місця, куди показує індекс, залишивши закладку на первинній сторінці, і повертатися назад, прочитавши роз'яснення по посиланню.

Розвиток ідеї гіпертексту знайшов своє віддзеркалення в новій формі організації документів, названої *гіпермедіа (hypermedia).* Гіпермедіа до-

зволяє зв'язувати не тільки слова, але і малюнки, звуки або файли будь-якого типу даних, які можна зберігати в комп'ютері. Наприклад, якщо ви клацнули по виділеній фразі *My image* на гіпермедіа-документі, залежно від контексту вам покажуть або фотографію автора цієї сторінки, або прокрутять відеоролик з його ж фізіономією, що говорить "Привіт! Я – листоноша Печкін".

***Гіпермедіа*** – це метод з'єднання один з одним елементів даних (незалежно від їх формату). І якщо ваша програма для роботи з "Всесвітньою павутиною" знатиме, що їй робити з одержаними даними, гіпермедіа може стати процесом з майже безмежними можливостями. Гіпермедіа-метод в середовищі "Всесвітньої павутини" дозволив сформувати безліч цікавих і різноманітних додатків – від журналів до фотографій і відеороликів з метеосупутників. Тепер WWW можна порівняти з телевізором, що приймає мільйони телевізійних каналів.

Для того, щоб створювати гіпертекстові або гіпермедіа-документи, була розроблена спеціальна *мова гіпертекстової розмітки (HTML – Hypertext Markup Language).* **HTML** – мова, використовувана для розмітки текстових файлів для використання в WWW. HTML-документи є звичайним текстовою ASCII-файлом, який містить як стандартний текст, так і особливі коди форматування або розмітки. Ці коди указують, яким чином повинен відображатися даний документ, і для правильного висновку на екран WWW-клієнт повинен їх прочитати.

Гіпермедіа-посилання, які присутні на сторінці "Всесвітньої павутини", описують місцезнаходження документа, який ваша програма-браузер повинна відобразити на екрані. Такий опис місцезнаходження джерела називають *уніфікованим покажчиком ресурсу (URL – Uniform Resource Locator).* URL дозволяє браузеру перейти безпосередньо до файлу, що знаходиться на будь-якому сервері Мережі. Система іменування URL – дуже проста, але могутня. Стандартний URL складається з чотирьох частин: формат передачі, ім'я хоста, на якому знаходиться запрошуваний файл, шлях до цього файлу і, звичайно, ім'я самого файлу. Втім, ім'я файлу необов'язково. Для стандартних сторінок "Всесвітньої павутини" форматом передачі служить HTTP-протокол, тому покажчики URL на ці сторінки починаються з букв *http.* Щоб відокремити формат передачі від імені хоста, використовується двокрапка і два слэша (://). Для частини URL, яка описує ім'я хоста, використовується стандартна угода про імена в Internet. Шлях до файлу відповідає угоді про імена каталогів UNIX. Приклад покажчика URL(рис. 2.20):




Рисунок 2.20. Приклад покажчика URL

Користуватися WWW виключно легко і зручно. Досить набрати на клавіатурі якої-небудь URL (наприклад *http://www.name.com/homepage.html*) – і ви потрапляєте на відповідну цій адресі так звану "домашню сторінку" *(home page*), яка є написаним з використанням мови HTML повідомленням, яке хоче зробити та або інша організація, служба, база даних, компанія або приватна особа. Деякі слова тексту виділені. Це гіперпосилання. Клацнувши клавішею миші на такому слові, можна одержати коментар до нього, текст, картинку, фотографію або уривок з музичного твору, а то і все перераховане відразу. У цьому коментарі теж можуть бути виділені слова. І так далі... Те, що сподобалося, можна завантажити у власний комп'ютер або роздрукувати.

Останнім часом стало престижним мати сторіночку в Internet. Віртуальні офіси створюються або беруться в оренду навіть невеликими фірмами. В цьому випадку перша сторіночка гіпертексту деколи буває і останньої, а для отримання додаткової інформації пропонується користуватися електронною поштою.

## 2.5 Введення в захист інформації. Джерела, класифікація погроз і форми атак на інформацію

Безпечна інформаційна система – це система, яка, по-перше, захищає дані від несанкціонованого доступу, по-друге, завжди готова надати їх своїм користувачам, а по-третє, надійно зберігає інформацію і гарантує незмінність даних. Таким чином, безпечна система за визначенням має властивості конфіденційності, доступності і цілісності.

***Захист інформації (ЗІ)*** – дисципліна, що динамічно розвивається, у якій надзвичайно високу роль грають наукові дослідження. У цей час такі дослідження ведуться по двох напрямках. *Перший напрямок* складається в розкритті природи явищ, що приводять до порушення таких характеристик інформації, як цілісність, доступність, конфіденційність, вірогідність і т.п. *Другий* – у розробці практичних методів захисту інформації від зазначених вище явищ. З метою забезпечення фактичної бази, необхідної для розвитку обох напрямків, серйозно вивчаються статистика й причини порушень, особистості порушників, суть застосовуваних порушниками прийомів, обставини, при яких було виявлене порушення. З іншого боку, для визначення необхідних і достатніх умов захищеності інформації ведуться інтенсивні роботи з моделювання системи захисту інформації (СЗІ).

**30 листопада**
**Міжнародний день захисту інформації (Computer Security Day)**
Відзначається з 30 листопада 1988 р. за ініціативою американської Асоціації комп'ютерного устаткування, яка хотіла нагадати усім про необхідність захисту комп'ютерної інформації, звернувши увагу виробників і користувачів апаратних і програмних засобів на проблему безпеки. У 1988 р. була зафіксована перша масова епідемія черв'яка, що дістав назву по імені свого "творця", – Моріса.

### 2.5.1 Захист комп'ютерної інформації: основні поняття й визначення

***Модель погроз інформації*** – це опис методів і засобів здійснення погроз для інформації в конкретних умовах функціонування автоматизованої інформаційної системи.

Формування моделі погроз інформації є одним з головних факторів забезпечення її безпеки, оскільки безпосередньо пов'язане з вибором необхідного переліку послуг безпеки, заходів і засобів захисту, що реалізуються в АС. Докладність і повнота аналізу погроз дає необхідну впевненість, що враховано всі істотні погрози безпеки інформації.

**Безпека інформації –** ступінь (міра) захищеності інформації, збереженої й оброблюваної в автоматизованій системі (АС), від негативного впливу на неї з погляду порушення її фізичної й логічної цілісності (знищення, перекручування) або несанкціонованого використання.

*Автоматизована система* – організована сукупність засобів, методів і заходів, використовуваних для регулярної обробки інформації в процесі рішення певного кола прикладних завдань.

**Захищеність інформації** – підтримка на заданому рівні тих параметрів інформації, що перебуває в автоматизованій системі, які забезпечують установлений статус її зберігання, обробки й використання.

**Захист інформації (ЗІ)** – процес створення й використання в автоматизованих системах спеціальних механізмів, що підтримують установлений статус її захищеності.

**Комплексний захист інформації** – цілеспрямоване регулярне застосування в автоматизованих системах засобів і методів захисту інформації, а також здійснення комплексу заходів з метою підтримки заданого рівня захищеності інформації із всієї сукупності показників і умов, що є істотно значимими з погляду забезпечення безпеки інформації.

**Безпечна інформаційна система** – це система, що, по-перше, захищає дані від несанкціонованого доступу, по-друге, завжди готова надати їх своїм користувачам, по-третє, надійно зберігає інформацію й гарантує

незмінність даних. Таким чином, безпечна система по визначенню має властивості конфіденційності, доступності й цілісності.

**Конфіденційність** – гарантія того, що секретні дані будуть доступні тільки тим користувачам, яким цей доступ дозволений (такі користувачі називаються авторизованими).

**Доступність** – гарантія того, що авторизовані користувачі завжди одержать доступ до даних.

**Цілісність** – гарантія схоронності даними правильних значень, що забезпечується забороною для неавторизованих користувачів яким-небудь образом змінювати, модифікувати, руйнувати або створювати дані.

Вимоги безпеки можуть мінятися залежно від призначення системи, характеру використовуваних даних і типу можливих погроз. Важко представити систему, для якої були б не важливі властивості цілісності й доступності, але властивість конфіденційності не завжди є обов'язковим.

Поняття конфіденційності, доступності й цілісності можуть бути визначені не тільки стосовно інформації, але й до інших ресурсів обчислювальної мережі, наприклад зовнішнім пристроям або додаткам. Існує безліч системних ресурсів, можливість "незаконного" використання яких може привести до порушення безпеки системи. Наприклад, необмежений доступ до пристрою печатки дозволяє зловмисникові одержувати копії документів, що роздруковуються, змінювати параметри настроювання, що може привести до зміни черговості робіт і навіть до висновку пристрою з ладу. Властивість конфіденційності, застосована до пристрою печатки, можна інтерпретувати так: доступ до пристрою мають ті й тільки ті користувачі, яким цей доступ дозволений, причому вони можуть виконувати тільки ті операції із пристроєм, які для них визначені. Властивість доступності пристрою означає його готовність до використання щораз , коли в цьому виникає необхідність, а властивість цілісності може бути визначене як властивість незмінності параметрів настроювання даного пристрою.

Важливим для розуміння подальших визначень є такі поняття, як об'єкт, суб'єкт, *доступ*.

Об'єкт – пасивний компонент системи, що зберігає, приймає або передає інформацію.

Суб'єкт – активний компонент системи, звичайно представлений у вигляді користувача, процесу або пристрою, якому може знадобитися звертання до об'єкта або системи.

***Доступ*** – взаємодія між суб'єктом і об'єктом, що забезпечує передачу інформації між ними, або зміну стану системи.

**Доступ до інформації** – забезпечення суб'єктові можливості ознайомлення з інформацією і її обробки, зокрема, копіювання, модифікації або знищення інформації.

**Ідентифікація** – присвоєння суб'єктам і об'єктам доступу унікального ідентифікатора у вигляді номера, шифру, коду й т. інш. з метою одержання доступу до інформації.

**Аутентифікація** – перевірка дійсності суб'єкта по пред'явленому їм ідентифікатору для ухвалення рішення про надання йому доступу до ресурсів системи.

**Загроза** – будь-яка дія, спрямована на порушення конфіденційності, цілісності й/або доступності інформації, а також на нелегальне використання інших ресурсів інформаційної системи (ІС).

Загрози можуть бути розділені на *навмисні* й *ненавмисні*.

*Ненавмисні загрози* викликаються помилковими діями лояльних співробітників, стають наслідком їхньої низької кваліфікації або безвідповідальності. Крім того, до такого роду загроз відносяться наслідки ненадійної роботи програмних і апаратних засобів системи. Так, наприклад, через відмову диску, контролера диску або всього файлового серверу можуть виявитися недоступними дані, критично важливі для роботи підприємства. Тому питання безпеки так тісно переплітаються з питаннями надійності, відмовостійкості технічних засобів. Загрози безпеки, які випливають із ненадійності роботи програмно-апаратних засобів, запобігають шляхом їхнього вдосконалювання, використання резервування на рівні апаратури (**RAID**-масиви, багатопроцесорні комп'ютери, джерела безперебійного живлення, кластерні архітектури) або на рівні масивів даних (тиражування файлів, резервне копіювання).

*Навмисні загрози* можуть обмежуватися або пасивним читанням даних чи моніторингом системи, або містити в собі активні дії, наприклад, порушення цілісності й доступності інформації, приведення в неробочий стан додатків і пристроїв. Так, навмисні загрози виникають у результаті діяльності хакерів і явно спрямовані на завдання збитків.

В обчислювальних мережах можна виділити наступні типи навмисних загроз:

- незаконне проникнення в один з комп'ютерів мережі під видом легального користувача;
- руйнування системи за допомогою програм-вірусів;
- нелегальні дії легального користувача;
- "підслуховування" внутрімережевого трафіку.

*Незаконне проникнення* може бути реалізоване через уразливі місця в системі безпеки з використанням недокументованих можливостей операційної системи. Ці можливості можуть дозволити зловмисникові "обійти" стандартну процедуру, що контролює вхід у мережу.

Іншим способом незаконного проникнення в мережу є використання "чужих" паролів, отриманих шляхом підглядання, розшифровки файлу паролів, підбору паролів або одержання пароля шляхом аналізу

мережного трафіку. Особливо небезпечне проникнення зловмисника під ім'ям користувача, наділеного великими повноваженнями, наприклад адміністратора мережі. Для того щоб заволодіти паролем адміністратора, зловмисник може спробувати ввійти в мережу під ім'ям простого користувача. Тому дуже важливо, щоб всі користувачі мережі зберігали свої паролі в таємниці, а також вибирали їх так, щоб максимально утруднити вгадування.

Ще один спосіб одержання пароля – це впровадження в чужий комп'ютер "троянського коня". Так називають резидентну програму, що працює без відома хазяїна даного комп'ютера і виконує дії, задані зловмисником. Зокрема, такого роду програма може зчитувати коди пароля, який вводиться користувачем під час логічного входу в систему.

Програма "троянський кінь" завжди маскується під яку-небудь корисну утиліту або гру, а робить дії, які руйнують систему. За таким принципом діють і програми-*віруси*, відмінною рисою яких є здатність "заражати" інші файли, впроваджуючи в них свої власні копії. Найчастіше віруси вражають файли, які виконуються. Коли такий виконуваний код завантажується в оперативну пам'ять для виконання, разом з ним одержує можливість виконати свої шкідницькі дії вірус. Віруси можуть привести до ушкодження або навіть повної втрати інформації.

*Нелегальні дії легального користувача* – цей тип загроз виходить від легальних користувачів мережі, які, використовуючи свої повноваження, намагаються виконувати дії, що виходять за рамки їхніх посадових обов'язків. Наприклад, адміністратор мережі має практично необмежені права на доступ до всіх мережевих ресурсів. Однак на підприємстві може бути інформація, доступ до якої адміністраторові мережі заборонений. Для реалізації цих обмежень можуть бути вжиті спеціальні заходи, такі, наприклад, як шифрування даних, але й у цьому випадку адміністратор може спробувати одержати доступ до ключа. Нелегальні дії може спробувати почати й звичайний користувач мережі. Існуюча статистика говорить про те, що чи не половина всіх спроб порушення безпеки системи виходить від співробітників підприємства, які саме і є легальними користувачами мережі.

*"Підслуховування" внутрімережевого трафіку* – це незаконний моніторинг мережі, захват і аналіз мережевих повідомлень. Існує багато доступних програмних і апаратних аналізаторів трафіку, які роблять це завдання досить тривіальним.

Для забезпечення безпеки інформації в комп'ютерах, особливо в офісних системах і комп'ютерних мережах, проводяться різні заходи, об'єднувані поняттям "система захисту інформації".

**Система захисту інформації** – це сукупність організаційних (адміністративних) і технологічних заходів, програмно-технічних засобів,

правових і морально-етичних норм, спрямованих на протидію загрозам порушників з метою звести до мінімуму можливий збиток користувачам і власникам системи.

На практиці при побудові системи захисту інформації склалися два підходи: фрагментарний і комплексний. У першому випадку заходи щодо захисту спрямовані на протидію певним погрозам при строго певних умовах, наприклад обов'язкова перевірка носіїв антивірусними програмами, застосування криптографічних систем шифрування тощо. При комплексному підході різні заходи протидії загрозам поєднуються, формуючи так звану архітектуру безпеки систем.

**Архітектура безпеки** – комплексне вирішення питань безпеки обчислювальної системи, де виділяються загрози безпеки, служби безпеки і механізми забезпечення безпеки.

З огляду на важливість, масштабність і складність рішення проблеми схоронності й безпеки інформації, рекомендується розробляти архітектуру безпеки за кілька етапів:

- аналіз можливих загроз;
- розробка системи захисту;
- реалізація системи захисту;
- супровід системи захисту.

Етап розробки системи захисту інформації передбачає використання комплексів заходів організаційно-адміністративного, організаційно-технічного, програмно-апаратного, технологічного, правового, морально-етичного характеру та ін.

*Організаційно-адміністративні засоби захисту* зводяться до регламентації доступу до інформаційних і обчислювальних ресурсів, функціональним процесам систем обробки даних, до регламентації діяльності персоналу та ін. Їхня мета – найбільшою мірою утруднити або виключити можливість реалізації загроз безпеки. Найбільш типові організаційно-адміністративні заходи – це:

- створення контрольно-пропускного режиму на території, де розташовуються ЕОМ та інші засоби обробки інформації;
- допуск до обробки й передачі конфіденційної інформації тільки перевірених посадових осіб; зберігання магнітних та інших носіїв інформації, що представляють певну таємницю, а також реєстраційних журналів у сейфах, недоступних для сторонніх осіб;
- організація захисту від установки прослуховуючої апаратури у приміщеннях, пов'язаних з обробкою інформації тощо.

*Технічні засоби захисту* – системи охорони територій і приміщень за допомогою екранування машинних залів і організації контрольно-пропускних систем.

Технічні засоби покликані створити деяке фізично замкнуте середовище навколо об'єкта і елементів захисту. У цьому випадку використовуються такі заходи, як:

- установка засобів фізичної перешкоди захисного контуру приміщень, де ведеться обробка інформації (кодові замки; охоронна сигналізація – звукова, світлова, візуальна без запису та із записом на відеоплівку);
- обмеження електромагнітного випромінювання шляхом екранування приміщень, де відбувається обробка інформації, аркушами з металу або спеціальної пластмаси тощо.

*Програмні засоби й методи захисту* активніше й ширше інших застосовуються для захисту інформації в персональних комп'ютерах і комп'ютерних мережах, реалізуючи такі функції захисту, як розмежування і контроль доступу до ресурсів; реєстрацію і аналіз процесів, що протікають, подій, користувачів; запобігання можливих руйнівних впливів на ресурси; криптографічний захист інформації, тобто шифровку даних; ідентифікацію і аутентифікацію користувачів і процесів та ін.

Для надійного захисту інформації і виявлення випадків неправомірних дій проводиться реєстрація роботи системи: створюються спеціальні щоденники й протоколи, у яких фіксуються всі дії, що мають відношення до захисту інформації в системі. Фіксуються час надходження заявки, її тип, ім'я користувача і терміналу, з якого ініціюється заявка.

Використовуються також спеціальні програми для тестування системи захисту. Періодично або у випадково обрані моменти часу вони перевіряють працездатність апаратних і програмних засобів захисту.

*Технологічні засоби захисту інформації* – це комплекс заходів,, що вбудовуються органічно в технологічні процеси перетворення даних. Серед них:

- створення архівних копій носіїв;
- ручне або автоматичне збереження оброблюваних файлів у зовнішній пам'яті комп'ютера;
- реєстрація користувачів комп'ютерних засобів у журналах;
- автоматична реєстрація доступу користувачів до тих або інших ресурсів та ін.

До *правових і морально-етичних заходів і засобів захисту* відносяться чинні в країні закони, нормативні акти, що регламентують правила обігу інформації і відповідальність за їхнє порушення; норми поводження (що склалися у вигляді статуту або неписані), дотримання яких сприяє захисту інформації.

**Уразливість інформації** – можливість виникнення на якому-небудь етапі життєвого циклу автоматизованої системи такого її стану, при якому створюються умови для реалізації погроз безпеки інформації.

**Атака** – реалізована погроза.

**Ризик** – це імовірнісна оцінка величини можливого збитку, що може понести власник інформаційного ресурсу в результаті успішно проведеної атаки. Значення ризику тим вище, ніж більше уразливої є існуюча система безпеки й чим вище ймовірність реалізації атаки.

**Споконвічно захищена інформаційна технологія** – інформаційна технологія, що споконвічно містить всі необхідні механізми для забезпечення необхідного рівня захисту інформації.

***Якість інформації*** – сукупність властивостей, що обумовлюють придатність інформації задовольняти певні потреби відповідно до її призначення.

**19 грудня**
**День галузевого фахівця з безпеки**

19 грудня 2009 року було засновано галузеве свято – День фахівця з безпеки України. Цей день проводитиметься не лише як святкова подія, але і як підведення підсумків року, організація конкурсів, рейтингів і всіляких акцій. Ініціатором події виступив Security.ua – галузевий портал по безпеці України. У 2009 році під час урочистої установи Дня фахівця з безпеки України був сформований оргкомітет з питання внесення дати святкування Дня в державний реєстр свят України.

### 2.5.2 Класифікація погроз безпеки інформації

Згідно з нормативними документами системи ТЗИ (НД ТЗИ 1.1-002-99, НД ТЗИ 2.5-004-99) за результатом впливу на інформацію й систему її обробки погрози підрозділяються на чотири класи:

- порушення конфіденційності інформації (одержання інформації користувачами або процесами всупереч установленим правилам доступу);
- порушення цілісності інформації (повне або часткове знищення, скривлення, модифікація, нав'язування помилкової інформації);
- порушення доступності інформації (втрата часткова або повна захистоспроможності системи, блокування доступу до інформації);
- втрата спостерігатимості або керованості системи обробки (порушення процедур ідентифікації та аутентифікації користувачів і процесів, надання їм повноважень, здійснення контролю за їхньою діяльністю, відмова від одержання або пересилання повідомлень).

Для забезпечення конфіденційності, цілісності й доступності інформації, а також керованості системою (спостерігатимость) необхідно захищати інформацію не тільки від витоку технічними каналами й несанкціонованим

доступом, але й виключати можливість негативного впливу на інформацію, втручання в процес її обробки, порушення працездатності системи. Таким чином, захищати необхідно все компоненти АС: апаратуру й устаткування, програми, дані, персонал.

Відомо велика кількість різнопланових погроз безпеки інформації різного походження. Як критерії розподілу безлічі погроз на класи можуть використовуватися види породжуваних небезпек, ступінь злого наміру, джерела прояву погроз і т. інш. Все різноманіття існуючих класифікацій може бути зведене до деякої системної класифікації, наведеної в таблиці 2.3.

Таблиця 2.3

**Системна класифікація погроз безпеки інформації**

| | Критерії класифікації | Значення критеріїв | Зміст значення критеріїв |
|---|---|---|---|
| 1. | Види погроз | Фізична цілісність<br>Логічна структура<br>Зміст<br>Конфіденційність<br>Право власності | Знищення (перекручування).<br>Перекручування структури.<br>Несанкціонована модифікація.<br>Несанкціоноване одержання.<br>Присвоєння чужого права. |
| 2. | Природа походження погроз | Випадкова<br><br>Навмисна | Відмови, збої, помилки, стихійні лиха, побічні впливи.<br>Злочинні дії людей. |
| 3. | Передумови появи погроз | Об'єктивні<br><br><br><br>Суб'єктивні | Кількісна недостатність елементів системи, якісна недостатність елементів системи.<br>Розвідоргани іноземних держав, промислове шпигунство, карні елементи, несумлінні співробітники. |
| 4. | Джерела погроз | Люди<br><br>Технічні пристрої<br><br>Моделі, алгоритми, програми<br>Технологічні схеми обробки<br>Зовнішнє середовище | Сторонні особи, користувачі, персонал.<br>Реєстрації, передачі, зберігання, переробки, видачі.<br>Загального призначення, прикладні, допоміжні.<br>Ручні, інтерактивні, внутрімашинні, мережні.<br>Стан атмосфери, побічні шуми, побічні сигнали. |

Дамо короткий коментар до використаного в табл. 2.3 параметрам класифікації, їхнім значенням і змісту.

1. *Види погроз*. Даний критерій є основним, визначальним цільову спрямованість захисту інформації.

2. *Походження погроз.* У таблиці виділено два значення даного критерію: випадкове й навмисне. Під *випадковим* розуміється таке походження погроз, що обумовлюється спонтанними й не залежними від волі людей обставинами, що виникають в АС у процесі її функціонування. Найбільш відомими подіями даного плану є відмови, збої, помилки, стихійні лиха й побічні впливи. Сутність перерахованих подій (крім стихійних лих, сутність яких ясна) визначається в такий спосіб:

- **відмова** – порушення працездатності якого-небудь елемента системи, що приводить до неможливості виконання їм основних своїх функцій;
- **збій** – тимчасове порушення працездатності якого-небудь елемента системи, наслідком чого може бути неправильне виконання їм у цей момент своєї функції;
- **помилка** – неправильне (разове або систематичне) виконання елементом однієї або декількох функцій, що відбуває внаслідок специфічного (постійного або тимчасового) його стану;
- **побічний вплив –** негативний вплив на систему в цілому або окремі її елементи, надаване якими-небудь явищами, що відбуваються усередині системи або в зовнішнім середовищі.

*Навмисне походження погрози* обумовлюється злочинними діями людей.

3. *Передумови появи погроз.* У таблиці наведені два можливі різновиди передумов: об'єктивні (кількісна або якісна недостатність елементів системи) і суб'єктивні (діяльність розвідорганів іноземних держав, промислове шпигунство, діяльність карних елементів, дії несумлінних співробітників системи).

Перераховані різновиди передумов інтерпретуються в такий спосіб:

- *кількісна недостатність* – фізична недостача одного або декількох елементів системи, що викликає порушення технологічного процесу обробки даних і/або перевантаження наявних елементів;
- *якісна недостатність* – недосконалість конструкції (організації) елементів системи, у силу чого можуть з'являтися можливості випадкового або навмисного негативного впливу на оброблювану або збережену інформацію;
- *діяльність розвідорганів іноземних держав* – спеціально організуємая діяльність державних органів, професійно орієнтованих на добування необхідної інформації будь-якими способами й засобами. До основних видів розвідки ставляться агентурна (несанкціонована діяльність професійних розвідників,

завербованих агентів і так званих доброзичливців) і технічна, що включає радіорозвідку (перехоплення радіоелектронними засобами інформації, що циркулює в телекомунікаційних каналах), радіотехнічну розвідку (реєстрацію спеціальними засобами електромагнітних випромінювань технічних систем) і космічну розвідку (використання космічних кораблів і штучних супутників Землі для спостереження за територією, її фотографування, реєстрації радіосигналів і одержання корисної інформації будь-якими іншими доступними способами);

– *промислове шпигунство* – негласна діяльність організації (в особі її представників) по добуванню інформації, спеціально охоронюваної від несанкціонованого її витоку або розкрадання, з метою створення для себе сприятливих умов і одержання максимальних вигід (несумлінна конкуренція);
– *злочинні дії карних елементів* – розкрадання інформації або комп'ютерних програм з метою наживи;
– *дії несумлінних співробітників* – розкрадання (копіювання) або знищення інформаційних масивів і/або програм по егоїстичних або корисливих мотивах, а також у результаті недотримання встановленого порядку роботи з інформацією.

4. *Джерела погроз*. Під джерелом погроз розуміється безпосередній їхній генератор або носій. Таким джерелом можуть бути люди, технічні засоби, моделі (алгоритми), програми, технологічні схеми обробки, зовнішнє середовище.

**Методи й засоби несанкціонованого одержання інформації з автоматизованих систем**

***Соціальний інжиніринг*** – це використання психології для потайливого добування критичної з погляду доступу до АС інформації (як правило – паролів, імен, кодів доступу й т. інш.) у її носіїв. Це гуманітарна складова – розглядатися не буде.

Методи й засоби несанкціонованого одержання інформації з АС можна класифікувати, виходячи з різних ознак: по виду доступу, за рівнем доступу, по характері дій зловмисника, по багаторазовості доступу, по спрямованості дій зловмисника, по вазі наслідків (рис. 2.21).

По ***виду доступу*** всі методи й засоби можна розділити на дві більші групи. До першої групи ставляться методи й засоби, використовувані при *локальному* (фізичному) *доступі* до АС. а до другого – методи й засоби, використовувані при *вилученому доступі* (по комп'ютерній мережі). Як правило, будь-яка, навіть сама надійна АС при наявності в зловмисника локального доступу, достатніх сил і засобів і достатнього часу, не зможе забезпечити схоронності інформації. При вилученому доступі АС може бути досить надійно захищена, але, з іншого боку, абсолютної безпеки

АС, що має фізичне підключення до мереж передачі даних, гарантувати також не можна.

За ***рівнем доступу*** методи й засоби несанкціонованого одержання інформації звичайно розділяють на методи й засоби *гостьового*, *користувальницького*, *адміністративного*, *системного* й *необмеженого* рівня. У багатьох сучасних операційних системах є убудовані облікові записи, що надають їхніми власниками гостьової (Guest у системах Windows NT/2000/XP), адміністративний (Administrator в Windows NT/2000/XP, root в Unix-Системах), системний (SYSTEM в Windows 2000/XP) або необмежений (адміністратор підприємства в Windows 2000/XP) доступ. При створенні додаткових облікових записів у більшості сучасних операційних систем можна вказати будь-який рівень доступу, але змінити його для убудованих облікових записів найчастіше неможливо.




Рисунок 2.21. Класифікація методів і засобів несанкціонованого одержання інформації з АС

По ***характері дій зловмисника*** використовувані їм методи й засоби можуть бути спрямовані на *копіювання*, *модифікацію*, *знищення* або *впровадження* інформації. В останньому випадку проявляється особливість АС, відсутня в традиційних засобів нагромадження інформації, пов'язана

з тим, що в АС зберігаються не тільки дані, але й програмні засоби, що забезпечують їхню обробку й обмін інформацією. Ця особливість інтенсивно використовується зловмисниками, які часто прагнуть одержати доступ до тої або інший АС не заради несанкціонованого доступу до інформації, що зберігається в ній, а для впровадження програмної закладки, тобто для несанкціонованого створення в АС нової інформації, що представляє собою активний компонент самої АС, або для потайливого зберігання власної інформації без ведена власника АС.

По ***багаторазовості доступу*** виділяють методи й засоби, спрямовані на *разове одержання* несанкціонованого доступу й *багаторазове*. У першому випадку завдання попередження несанкціонованих дій зловмисника значно ускладнюється, однак часто, оскільки останній не піклується про приховання факту таких дій, трохи полегшується завдання виявлення таких дій. У другому випадку завдання попередження спрощується, але ускладнюється завдання виявлення, оскільки основна увага зловмисник, що планує багаторазово проникати в АС, зосереджує на прихованні всіх ознак такого проникнення.

По ***спрямованості дій*** зловмисника методи й засоби санкціонованого одержання інформації з АС підрозділяються на методи й засоби, спрямовані на *одержання системної інформації* (файли паролів, ключів шифрування, переліки облікових записів, схеми розподілу мережних адрес і т. інш.) і *властиво прикладної інформації*. Багатьох зловмисників, що проникають в АС, підключеним до глобальних мереж, взагалі не цікавить прикладна інформація, що зберігається в цих АС, або цікавить лише в тім ступені, у якій вона дозволяє одержати доступ до системної інформації. Звичайно такі зловмисники використовують подібні АС або як проміжні вузли для проникнення в інші АС, тому що для несанкціонованого зберігання власної інформації.

***По вазі наслідків*** використовувані зловмисниками методи й засоби несанкціонованого одержання інформації можна розділити на *безпечні* (сканування портів, спроби встановлення з'єднань і т.інш.), *потенційно небезпечні* (одержання доступу до вмісту підсистем зберігання даних, спроби підбора паролів і т.інш.), *небезпечні* (одержання доступу з високим рівнем повноважень, модифікація інформації в АС, копіювання системної й прикладної інформації, створення власної інформації й т.інш.) і *надзвичайно небезпечні* (знищення інформації, блокування доступу легальних користувачів до АС і т.інш.).

Спробуємо тепер, опираючись на наведену системну класифікацію погроз безпеки інформації, визначити повна безліч погроз, потенційно можливих у сучасних автоматизованих системах. При цьому ми повинні врахувати не тільки всі відомі (раніше проявлялися) погрози, але й такі погрози, які раніше не проявлялися, але потенційно можуть виникнути при

застосуванні нових концепцій архітектурної побудови АС і технологічні схеми обробки інформації.

Класифікуємо всі можливі **канали витоку інформації** (КВІ) по двох критеріях: по необхідності доступу (фізичного або логічного) до елементів АС для реалізації того або іншого КВІ й по залежності появи КВІ від стану АС.

За першим критерієм КВІ можуть бути розділені на *не потребуючого доступу*, тобто що дозволяють одержувати необхідну інформацію дистанційно (наприклад, шляхом візуального спостереження через вікна приміщень АС), и *потребуючого доступу* в приміщення АС (таблиця 2). У свою чергу КВІ, скористатися якими можна тільки одержавши доступ у приміщення АС, діляться на КВІ, що *не залишають сліди* в АС (наприклад, візуальний перегляд зображень на екранах моніторів або документів на паперових носіях), і на КВІ, використання яких *залишає ті або інші сліди* (наприклад, розкрадання документів або машинних носіїв інформації).

За другим критерієм КВІ діляться на *потенційно існуючі не завищимо от стани* АС(наприклад, викрадати носії інформації можна незалежно від того, у робочому стані перебувають засоби чи АС ні) і *існуючі тільки в робочому стані* АС (наприклад, побічні електромагнітні випромінювання й наведення).

Таблиця 2.4

**Класифікаційна структура каналів витоку інформації**

| Залежність від доступу до елементів системи | Відношення до обробки інформації | |
|---|---|---|
| | що виявляються безвідносно до обробки | що виявляються в процесі обробки |
| Що не вимагають доступу | 1-й клас. Загальнодоступні постійні | 2-й клас. Загальнодоступні функціональні |
| Що не вимагають доступу без зміни елементів системи | 3-й клас. Вузькодоступні постійні без залишення слідів | 4-й клас. Вузькодоступні функціональні без залишення слідів |
| Що вимагають доступу із зміною елементів системи | 5-й клас. Вузькодоступні постійні з складанням слідів | 6-й клас. Вузькодоступні функціональні з складанням слідів |

Приведемо орієнтовну характеристику каналів несанкціонованого одержання інформації виділених нами класів(табл. 2.4).

КВІ 1-го класу – канали, що проявляються безвідносно до обробки інформації й без доступу зловмисника до елементів системи. Сюди може бути віднесене підслуховування розмов, а також провокування на розмови осіб, що мають відношення до АС, і використання зловмисником

візуальних, оптичних і акустичних засобів. Даний канал може виявитися й шляхом розкрадання носіїв інформації в момент їхнього знаходження за межами приміщення, де розташований АС.

КВІ 2-го класу – канали, що проявляються в процесі обробки інформації без доступу зловмисника до елементів АС. Сюди можуть бути віднесені електромагнітні випромінювання різних пристроїв ЕОМ, апаратури й ліній зв'язку, паразитні наведення в ланцюгах харчування, телефонних мережах, системах теплопостачання, вентиляції й каналізації, шинах заземлення, підключення до інформаційно-обчислювальної мережі генераторів перешкод і апаратури, що реєструє. До цього ж класу може бути віднесений огляд відходів виробництва, що попадають за межі контрольованої зони.

КВІ 3-го класу – канали, що проявляються безвідносно до обробки інформації з доступом зловмисника до елементів АС, але без зміни останніх. До них ставляться всілякі види копіювання носіїв інформації й документів, а також розкрадання виробничих відходів.

КВІ 4-го класу – канали, що проявляються в процесі обробки інформації з доступом зловмисника до елементів АС, але без зміни останніх. Сюди може бути віднесені запам'ятовування й копіювання інформації в процесі обробки, використання програмних пасток, недоліків мов програмування й операційних систем, маскування під зареєстрованого користувача.

КВІ 5-го класу – канали, що проявляються безвідносно до обробки інформації з доступом зловмисника до елементів АС і зі зміною програмних або апаратних засобів. Серед цих каналів: підміна й розкрадання носіїв інформації й апаратури, включення в програми блоків типу троянський кінь, комп'ютерний хробак і т.п., читання залишкової інформації, що втримується в пам'яті, після виконання санкціонованих запитів.

КВІ 6-го класу – канали, що проявляються в процесі обробки інформації з доступом зловмисника до елементів АС і зі зміною останніх. Сюди може бути віднесене незаконне підключення до апаратури й ліній зв'язку, а також зняття інформації на шинах харчування різних елементів АС.

### 2.5.3 Форми атак на об'єкти інформаційних систем

#### 2.5.3.1 Форми атак

У загальному випадку програмне забезпечення будь-якої інформаційної системи складається із трьох основних компонентів: операційної системи, мережного програмного забезпечення (СПО) і системи керування базами даних (СУБД). Тому всі спроби злому захисту ІС на рівні програмного забезпечення можна розділити на три групи:

атаки на рівні систем керування базами даних;
атаки на рівні операційної системи;
атаки на рівні мережного програмного забезпечення.

**Атаки на рівні систем керування базами даних**

Захист СУБД є однієї з найпростіших завдань. Це пов'язане з тим, що СУБД мають строго певну внутрішню структуру й операції над елементами СУБД задані досить чітко. Є чотири основних дії – пошук, вставка, видалення й заміна елемента. Інші операції є допоміжними й застосовуються досить рідко. Наявність строгої структури й чітко певних операцій спрощує рішення завдання захисту СУБД. У більшості випадків зловмисники воліють зламувати захист інформаційної системи на рівні операційної системи й одержувати доступ до файлів СУБД за допомогою засобів операційної системи. Однак у випадку, якщо використовується СУБД, що не має досить надійних захисних механізмів, або погано протестована версія СУБД, що містить помилки, або якщо при визначенні політики безпеки адміністратором СУБД були допущені помилки, то стає цілком імовірним подолання захисту, реалізованої на рівні СУБД.

Крім того, є два специфічних сценарії атаки на СУБД, для захисту від яких потрібно застосовувати спеціальні методи. У першому випадку результати арифметичних операцій над числовими полями СУБД округляються в меншу сторону, а різниця підсумується в деякому іншому записі СУБД (як правило, цей запис містить особистий рахунок зловмисника в банку, а заокруглювані числові поля ставляться до рахунків інших клієнтів банку). У другому випадку зловмисник одержує доступ до полів записів СУБД, для яких доступної є тільки статистична інформація Ідея атаки на СУБД – так хитро сформулювати запит, щоб безліч записів, для якого збирається статистика, складалося тільки з одного запису.

**Атаки на рівні операційної системи**

Захищати операційну систему, на відміну від СУБД, набагато складніше. Справа в тому, що внутрішня структура сучасних операційних систем надзвичайно складна, і тому дотримання адекватної політики безпеки є значно більше важким завданням.

Успіх реалізації того або іншого алгоритму атаки на практиці в значній мірі залежить від архітектури й конфігурації конкретної операційної системи, що є об'єктом цієї атаки. Однак є атаки, яким може бути піддана практично будь-яка операційна система.

1) Крадіжка пароля:

– підглядання за користувачем, коли той уводить пароль, що дає право на роботу з операційною системою (навіть якщо під час уведення пароль не висвічується на екрані дисплея, зловмисник може легко довідатися пароль, просто стежачи за переміщенням пальців користувача по клавіатурі);

– одержання пароля з файлу, у якому цей пароль був збережений користувачем, що не бажає утрудняти себе уведенням пароля при підключенні до мережі (як правило, такий пароль зберігається у файлі в незашифрованому виді);

– пошук пароля, що користувачі, щоб не забути, записують на календарях, у записних книжках або на зворотному боці комп'ютерних клавіатур (особливо часто подібна ситуація зустрічається, якщо адміністратори змушують користувачів застосовувати паролі, що запам'ятовуються важко);

– крадіжка зовнішнього носія парольної інформації (дискети або електронного ключа, на яких зберігається пароль користувача, призначений для входу в операційну систему);

– повний перебір всіх можливих варіантів пароля;

– підбор пароля по частоті зустрічальності символів, за допомогою словників найбільше часто застосовуваних паролів, із залученням знань про конкретного користувача – його ім'я, прізвища, номера телефону, дати народження й т.д., з використанням відомостей про існування еквівалентних паролів, при цьому з кожного класу випробується всього один пароль, що може значно скоротити час перебору.

2) Сканування жорстких дисків комп'ютера (зловмисник послідовно намагається звернутися до кожного файлу, збереженому на жорстких дисках комп'ютерної системи, якщо обсяг дискового простору досить великий, можна бути цілком упевненим, що при описі доступу до файлів і каталогів адміністратор допустив хоча б одну помилку, у результаті чого всі такі каталоги й файли будуть прочитані зловмисником).

3) Складання "сміття" (якщо засобу операційної системи дозволяють відновлювати раніше вилучені об'єкти, зловмисник може скористатися цією можливістю, щоб одержати доступ до об'єктів, вилученим іншими користувачами, наприклад, переглянувши вміст їх "сміттєвих" кошиків).

4) Перевищення повноважень (використовуючи помилки в програмному забезпеченні або в адмініструванні операційної системи, зловмисник одержує повноваження, що перевищують повноваження, надані йому відповідно до діючої політики безпеки):

– запуск програми від імені користувача, що має необхідні повноваження, або як системна програма (драйвера, сервісу, і т.д.);

– підміна бібліотеки, що завантажується динамічно, використовуваної системними програмами, або зміна змінні середовища, що описують шлях до таких бібліотек;

– модифікація коду або даних підсистем захисту самої операційної системи.

5) Відмова в обслуговуванні (метою цієї атаки є частковий або повний висновок з ладу операційної системи):

– захоплення ресурсів (програма зловмисника робить захоплення всіх наявних в операційній системі ресурсів, а потім входить у нескінченний цикл);

– бомбардування запитами (програма зловмисника постійно направляє операційній системі запити, реакція на які вимагає залучення значних ресурсів комп'ютера);

– використання помилок у програмному забезпеченні або адмініструванні.

Якщо в програмному забезпеченні комп'ютерної системи немає помилок і її адміністратор строго дотримує політики безпеки, рекомендовану розроблювачами операційної системи, то атаки всіх перерахованих типів малоефективні. Додаткові заходи, які повинні бути початі для підвищення рівня безпеки, у значній мірі залежать від конкретної операційної системи, під керуванням якої працює дана комп'ютерна система. Проте доводиться визнати, що поза залежністю від вжитих заходів повністю усунути погрозу злому комп'ютерної системи на рівні операційної системи неможливо. Тому політика забезпечення безпеки повинна проводитися так, щоб, навіть перебороввши захист, створюваний засобами операційної системи, хакер не зміг нанести серйозного збитку.

**Атаки на рівні мережного програмного забезпечення**

Мережеве програмне забезпечення (МПЗ) є найбільш уразливим, тому що канал зв'язку, по якому передаються повідомлення, найчастіше не захищене, і всякий, хто має доступ до цього каналу, може перехоплювати повідомлення й відправляти свої власні. Тому на рівні МПЗ можливі наступні атаки:

– прослуховування сегмента локальної мережі (у межах того самого сегмента локальної мережі будь-який підключений до нього комп'ютер у стані приймати повідомлення, адресовані іншим комп'ютерам сегмента, а отже, якщо комп'ютер зловмисника приєднаний до деякого сегмента локальної мережі, то йому стає доступний весь інформаційний обмін між комп'ютерами цього сегмента);

– перехоплення повідомлень на маршрутизаторі (якщо зловмисник має привілейований доступ до мережного маршрутизатора, то він одержує можливість перехоплювати всі повідомлення, що проходять через цей маршрутизатор, і хоча тотальне перехоплення неможливе через занадто великий обсяг, надзвичайно привабливим для зловмисника є вибіркове перехоплення повідомлень, що містять паролі користувачів і їхню електронну пошту);

– створення помилкового маршрутизатора (шляхом відправлення в мережу повідомлень спеціального виду зловмисник домагається, щоб його комп'ютер став маршрутизатором мережі, після чого одержує доступ до всім минаючим через нього повідомленням);

– нав'язування повідомлень (відправляючи в мережу повідомлення з помилковою зворотною мережною адресою, зловмисник перемикає на свій комп'ютер уже встановлені мережні з'єднання й у результаті дістає права користувачів, чиї з'єднання шляхом обману були переключені на комп'ютер зловмисника);

– відмова в обслуговуванні (зловмисник відправляє в мережу повідомлення спеціального виду, після чого одна або кілька комп'ютерних систем, підключених до мережі, повністю або частково виходять із ладу).

2.5.3.2. Програмні закладки

Сучасна концепція створення комп'ютерних систем припускає використання програмних засобів різного призначення в єдиному комплексі. Головною умовою правильного функціонування такої комп'ютерної системи є забезпечення захисту від втручання в процес обробки інформації тих програм, присутність яких у комп'ютерній системі не обов'язково. Серед подібних програм, у першу чергу, варто згадати комп'ютерні віруси. Однак є шкідливі програми ще одного класу. Від них, як і від вірусів, треба з особливою старанністю очищати свої комп'ютерні системи. Це програмні закладки, які можуть виконувати хоча б одне з перерахованих нижче дій:

- вносити довільні перекручування в коди програм, що перебувають в оперативній пам'яті комп'ютера (програмна закладка першого типу);
- переносити фрагменти інформації з одних областей оперативної або зовнішньої пам'яті комп'ютера в інші (програмна закладка другого типу);
- спотворювати виведену на зовнішні комп'ютерні пристрої або в канал зв'язку інформацію, отриману в результаті роботи інших програм (програмна закладка третього типу).

**Програмна закладка** – несанкціоноване впроваджена програма, що здійснює погрозу інформації. Програмні закладки можна класифікувати й по методу їхнього впровадження в комп'ютерну систему:

- програмно-апаратні закладки, асоційовані з апаратними засобами комп'ютера (їхнім середовищем перебування, як правило, є BIOS – набір програм, записаних у вигляді машинного коду в ПЗП);
- завантажувальні закладки, асоційовані із програмами початкового завантаження, які розташовуються в завантажувальних секторах (із цих секторів у процесі виконання початкового завантаження комп'ютер зчитує програму, що бере на себе керування для наступного завантаження самої операційної системи);
- драйверні закладки, асоційовані із драйверами (файлами, у яких утримується інформація, необхідна операційній системі для керування підключеними до комп'ютера периферійними пристроями);
- прикладні закладки, асоційовані із прикладним програмним забезпеченням загального призначення (текстові редактори, утиліти, антивірусні монітори й програмні оболонки);

- виконуються закладки, що, асоційовані із програмними модулями, що виконуються, утримуючий код цієї закладки (найчастіше ці модулі являють собою пакетні файли, тобто файли, які складаються з команд операційної системи, виконуваних одна за однією, як якби їх набирали на клавіатурі комп'ютера);
- закладки-імітатори, інтерфейс яких збігається з інтерфейсом деяких службових програм, що вимагають уведення конфіденційної інформації (паролів, криптографічних ключів, номерів кредитних карток);
- замасковані закладки, які маскуються під програмні засоби оптимізації роботи комп'ютера (файлові архіватори, дискові дефрагментатори) або під програми ігрового й розважального призначення.

Щоб програмна закладка могла зробити які-небудь дії стосовно інших програм або стосовно даних, процесор повинен приступитися до виконання команд, що входять до складу коду програмної закладки. Це можливо тільки при одночасному дотриманні наступних умов:

- програмна закладка повинна потрапити в оперативну пам'ять комп'ютера (якщо закладка ставиться до першого типу, то вона повинна бути завантажена до початку роботи іншої програми, що є метою впливу закладки, або під час роботи цієї програми);
- робота закладки, що перебуває в оперативній пам'яті, починається при виконанні ряду умов, які мають назву активізуючи.

З урахуванням зауваження про те, що програмна закладка повинна бути обов'язково завантажена в оперативну пам'ять комп'ютера, можна виділити *резидентні* закладки (вони перебувають в оперативній пам'яті постійно, починаючи з деякого моменту й до закінчення сеансу роботи комп'ютера, тобто до його перезавантаження або до вимикання харчування) і *нерезидентні* (такі закладки попадають в оперативну пам'ять комп'ютера аналогічно резидентним, однак, на відміну від останніх, вивантажуються після закінчення деякого часу або при виконанні особливих умов).

У всіх програмних закладок є одна важлива загальна риса: вони обов'язково виконують операцію запису в оперативну або зовнішню пам'ять системи. При відсутності даної операції ніякого негативного впливу програмна закладка зробити не може. Для цілеспрямованого впливу вона повинна виконувати й операцію читання, інакше в ній може бути реалізована тільки функція руйнування (наприклад, видалення або заміна інформації в певних секторах жорсткого диска).

Виявлення впровадженого коду програмної закладки полягає у виявленні ознак його присутності в комп'ютерній системі. Ці ознаки можна розділити на наступні два класи:

– якісні й візуальні;
– виявляються засобами тестування й діагностики.

До якісних і візуальних ознак ставляться відчуття й спостереження користувача комп'ютерної системи, що відзначає певні відхилення в її роботі (змінюється склад і довжини файлів, старі файли кудись пропадають, а замість них з'являються нові, програми починають працювати повільніше або закінчують свою роботу занадто швидко, або взагалі перестають запускатися). Незважаючи на те що судження про наявність ознак цього класу здається занадто суб'єктивним, проте вони часто свідчать про наявність неполадок у комп'ютерній системі й, зокрема, про необхідність проведення додаткових перевірок присутності програмних закладок.

Ознаки, що виявляються за допомогою засобів тестування й діагностики, характерні як для програмних закладок, так і для комп'ютерних вірусів. Наприклад, завантажувальні закладки успішно виявляються антивірусними програмами, які сигналізують про наявність підозрілого коду в завантажувальному секторі диска. З ініціюванням статичної помилки на дисках добре справляється Disk Doctor, що входить у розповсюджений комплект утиліт Norton Utilities, а засобу перевірки цілісності даних на диску типу Adinf дозволяють успішно виявляти зміни, внесені у файли програмними закладками.

Конкретний спосіб видалення впровадженої програмної закладки залежить від методу її впровадження в комп'ютерну систему:

– якщо це програмно-апаратна закладка, те варто перепрограмувати ПЗП комп'ютера.
– якщо це завантажувальна, драйверна, прикладна, замаскована закладка або закладка-імітатор, то можна замінити їх на відповідний завантажувальний запис, драйвер, утиліту, прикладну або службову програму, отриману від джерела, що заслуговує довіри.
– якщо це програмний модуль, що виконується, то можна спробувати добути його вихідний текст, забрати з нього наявні закладки або підозрілі фрагменти, а потім заново відкомпілювати.

2.5.3.3 Моделі впливу програмних закладок на комп'ютери

**Перехоплення**

У моделі *перехоплення* програмна закладка впроваджується в ПЗП, системне або прикладне програмне забезпечення й зберігає всю або обрану інформацію, що вводиться із зовнішніх пристроїв комп'ютерної системи або виведену на ці пристрої, у схованій області пам'яті локальної або вилученої комп'ютерної системи. Об'єктом збереження, наприклад, можуть служити символи, уведені із клавіатури (всі повторювані два рази

послідовності символів), або електронні документи, що роздруковуються на принтері.

Дана модель може бути **двоступінчастої**.

На **першому етапі** зберігаються тільки, наприклад, імена або початку файлів.

На **другому** накопичені дані аналізуються зловмисником з метою ухвалення рішення про конкретні об'єкти подальшої атаки.

Модель типу перехоплення може бути ефективно використана при атаці на захищену операційну систему Windows NT. Після старту Windows NT на екрані комп'ютерної системи з'являється запрошення нажати клавіші <Ctrl>+<Alt>+<Del>. Після їхнього натискання завантажується динамічна бібліотека MSGINA.DLL, що здійснює прийом пароля, що вводиться, і виконання процедури його перевірки (аутентификации). Також існує простий механізм заміни вихідної бібліотеки MSGINA.DLL на користувальницьку (для цього необхідно просто зі спеціальний рядок до реєстру операційної системи Windows NT і вказати місце розташування користувальницької бібліотеки). У результаті зловмисник може модифікувати процедуру контролю за доступом до комп'ютерної системи, що працює під керуванням Windows NT.

**Спотворення**

У моделі *спотворення* програмна закладка змінює інформацію, яка записується в пам'ять комп'ютерної системи в результаті роботи програм, або придушує/ініціює виникнення помилкових ситуацій в комп'ютерній системі.

Можна виділити **статичне** і **динамічне** спотворення. *Статичне спотворення* відбувається всього один раз. При цьому модифікуються параметри програмного середовища комп'ютерної системи, щоб згодом в ній виконувалися потрібні зловмисникові дії. До статичного спотворення відноситься, наприклад, внесення змін до файлу AUTOEXEC.BAT операційної системи Windows 95/98, які приводять до запуску заданої програми, перш ніж будуть запущені всі інші, перераховані в цьому файлі.

*Динамічне спотворення* полягає в зміні яких-небудь параметрів системних або прикладних процесів за допомогою заздалегідь активізованих закладок. Динамічне спотворення можна умовно розділити на:

- спотворення *на вході* (коли на обробку потрапляє вже спотворений документ);
- спотворення *на виході* (коли спотворюється інформація, що відображається для сприйняття людиною або призначена для роботи інших програм).

В рамках моделі "спотворення" також реалізуються програмні закладки, дія яких ґрунтується на ***ініціації*** або ***придушенні*** сигналу про

виникнення помилкових ситуацій в комп'ютерній системі, тобто тих, які приводять до відмінного від нормального завершенню виконуваної програми (наказаного відповідною документацією).

Для *ініціації статичної помилки* на пристроях зберігання інформації створюється область, при зверненні до якої (читання, запис, форматування і тому подібне) виникає помилка, що може утруднити або блокувати деякі небажані для зловмисника дії системних або прикладних програм (наприклад, не дозволяти коректно знищити конфіденційну інформацію на жорсткому диску).

При *ініціації динамічної помилки* для деякої операції генерується помилкова помилка з числа тих помилок, які можуть виникати при виконанні даної операції. Наприклад, для блокування прийому або передачі інформації в комп'ютерній системі може постійно ініціюватися помилкова ситуація "МОДЕМ ЗАЙНЯТИЙ".

Щоб маскувати помилкові ситуації, зловмисники зазвичай використовують придушення статичної або динамічної помилки. Метою такого придушення часто є прагнення блокувати нормальне функціонування комп'ютерної системи або бажання змусити її неправильно працювати. Надзвичайно важливо, щоб комп'ютерна система адекватно реагувала на виникнення всіх без виключення помилкових ситуацій, оскільки відсутність належної реакції на будь-яку помилку еквівалентно її придушенню і може бути використано зловмисником.

Різновидом спотворення є також модель типу *троянський кінь*. В цьому випадку програмна закладка вбудовується в постійно використовуване програмне забезпечення і по деякій активізуючій події викликає виникнення збійної ситуації в комп'ютерній системі.

**Прибирання сміття**

Як відомо, при зберіганні комп'ютерних даних на зовнішніх носіях прямого доступу виділяється декілька рівнів ієрархії сектора, кластери і файли. Сектори є одиницями зберігання інформації на апаратному рівні. Кластери складаються з одного або декількох підряд секторів, що йдуть. ***Файл*** – це безліч кластерів, зв'язаних по певному закону.

Робота з конфіденційними електронними документами зазвичай зводиться до послідовності наступних маніпуляцій з файлами:

- створення;
- зберігання;
- корекція;
- знищення.

Для захисту конфіденційної інформації зазвичай використовується шифрування. Основна загроза виходить зовсім не від використання нестійких алгоритмів шифрування і "поганих" криптографічних ключів, а

від звичайних текстових редакторів і баз даних, вживаних для створення і корекції конфіденційних документів.

Річ у тому, що подібні програмні засоби, як правило, в процесі функціонування створюють в оперативній або зовнішній пам'яті комп'ютерної системи тимчасові копії документів, з якими вони працюють. Природно, всі ці тимчасові файли випадають з поля зору будь-яких програм шифрування і можуть бути використані зловмисником для того, щоб скласти уявлення про зміст конфіденційних документів, що зберігаються в зашифрованому вигляді.

Важливо пам'ятати і про те, що при записі відредагованої інформації меншого об'єму в той же файл, де зберігалася початкова інформація до початку сеансу її редагування, утворюються так звані "хвостові" кластери, в яких ця початкова інформація повністю зберігається. І тоді "хвостові" кластери не тільки не піддаються дії програм шифрування, але і залишаються незачепленими навіть засобами гарантованого стирання інформації. Звичайно, рано чи пізно інформація з "хвостових" кластерів затирається даними з інших файлів, проте по оцінках фахівців з "хвостових" кластерів через добу можна витягувати до 85%, а через десять діб – до 25–40% початкової інформації.

**Спостереження і компрометація**

Окрім перерахованих, існують і інші моделі дії програмних закладок на комп'ютери. Зокрема, при використанні моделі типу *спостереження* програмна закладка вбудовується в мережеве або телекомунікаційне програмне забезпечення. Користуючись тим, що подібне програмне забезпечення завжди знаходиться в стані активності, упроваджена в нього програмна закладка може стежити за всіма процесами обробки інформації в комп'ютерній системі, а також здійснювати установку і видалення інших програмних закладок.

Модель типу *компрометація* дозволяє отримувати доступ до інформації, перехопленої іншими програмними закладками. Наприклад, ініціюється постійне звернення до такої інформації, що приводить до зростання співвідношення сигнал/шум. А це, у свою чергу, значно полегшує перехоплення побічних випромінювань даної комп'ютерної системи і дозволяє ефективно виділяти сигнали, що згенерували закладкою типу компрометація, із загального фону випромінювання, витікаючого від устаткування.

2.5.3.4 Троянські програми

**Троянською програмою** (Троєю, або троянським конем) називається програма, яка, будучи частиною іншої програми з відомими користувачеві функціями, здатна таємно від нього виконувати деякі додаткові дії з метою спричинення йому певного збитку.

Таким чином, ***троянська програма*** – це особливий різновид програмної закладки. Вона додатково наділена функціями, про існування яких користувач навіть не підозрює. Коли троянська програма виконує ці функції, комп'ютерній системі завдається певного збитку.

Більшість троянських програм призначена для збору конфіденційної інформації. Їх завдання найчастіше полягає у виконанні дій, що дозволяють дістати доступ до даних, які не підлягають широкому розголосу. До таких даних відносяться призначені для користувача паролі, реєстраційні номери програм, відомості про банківські рахунки і так далі Решта Трої створюється для спричинення прямого збитку комп'ютерній системі, приводячи її в непрацездатний стан.

До останніх можна віднести, наприклад, троянську програму PC CYBORG, яка заманювала нічого непідозрюючих користувачів обіцянками надати їм новітню інформацію про боротьбу з вірусом, що викликає синдром придбаного імунодефіциту (СНІД). Проникнувши в комп'ютерну систему, PC CYBORG відлічувала 90 перезавантажень цієї системи, а потім ховала всі каталоги на її жорсткому диску і шифрувала файли, що знаходяться там.

В даний час троянські програми можна відшукати практично де завгодно. Вони написані для всіх без виключення операційних систем і для будь-яких платформ. Не рахуючи випадків, коли троянські програми пишуться самими розробниками програмного забезпечення, Троя розповсюджується тим же способом, що і комп'ютерні віруси. Тому найпідозрілішими на предмет присутності в них Трої, насамперед, є безкоштовні і умовно-безкоштовні програми, викачані з Internet, а також програмне забезпечення, поширюване на піратських компакт-дисках.

В даний час існує цілий ряд троянських програм, які можна абсолютно вільно викачати, підключившись до мережі Internet. Найбільшу популярність серед них здобули Троя Back Orifice, Net Bus і SubSeven.

### 2.5.3.5 Клавіатурні шпигуни

Один з найбільш поширених різновидів програмних закладок – *клавіатурні шпигуни.* Такі програмні закладки націлені на перехоплення паролів користувачів операційної системи, а також на визначення їх легальних повноважень і прав доступу до комп'ютерних ресурсів.

Типовий клавіатурний шпигун обманним шляхом заволодіває призначеними для користувача паролями, а потім переписує ці паролі туди, звідки їх може без особливих зусиль витягувати зловмисник. Відмінності між клавіатурними шпигунами стосуються тільки способу, який застосовується ними для перехоплення призначених для користувача паролів. Відповідно всі клавіатурні шпигуни діляться на три типи – ***імітаторів***, ***фільтри*** і ***заступників***.

**Імітатори**

Клавіатурні шпигуни цього типу працюють по наступному алгоритму. Зловмисник упроваджує в операційну систему програмний модуль, який імітує запрошення користувачеві реєструватися для того, щоб увійти до системи. Потім упроваджений модуль (у прийнятій термінології – імітатор) переходить в режим очікування введення призначеного для користувача ідентифікатора і пароля. Після того, як користувач ідентифікує себе і введе свій пароль, імітатор зберігає ці дані там, де вони доступні зловмиснику. Далі імітатор ініціює вихід з системи (що в більшості випадків можна зробити програмним шляхом), і в результаті перед очима ні у чого непідозрюючого користувача з'являється ще одне, але цього разу вже справжнє запрошення для входу в систему. Деякі імітатори для переконливості видають на екран монітора правдоподібне повідомлення про нібито досконалій користувачем помилці. Наприклад, таке: "НЕВІРНИЙ ПАРОЛЬ. СПРОБУЙТЕ ЩЕ РАЗ".

Написання імітатора не вимагає від його творця яких-небудь особливих навиків. Зловмисникові, що уміє програмувати на одній з універсальних мов програмування (наприклад, на мові BASIC), знадобиться на це лічений годинник. Єдина трудність, з якою він може зіткнутися, полягає в тому, щоб відшукати в документації відповідну програмну функцію, що реалізовує вихід користувача з системи.

Перехоплення пароля часто полегшують самі розробники операційних систем, які не утрудняють себе створенням ускладнених формою запрошень користувачеві реєструватися для входу в систему. Подібне зневажливе відношення характерне для більшості версій операційної системи UNIX, в яких реєстраційне запрошення складається з двох текстових рядків, що видаються по черзі на екран терміналу:

login:

password:

Проте само по собі ускладнення зовнішнього вигляду запрошення не створює для зловмисника, що задумав упровадити в операційну систему імітатор, яких-небудь непереборних перешкод. Для цього потрібно удатися до складніших і витонченіших заходів захисту. Як приклад операційної системи, в якій такі заходи в достатньо повному об'ємі реалізовані на практиці, можна привести Windows NT.

Системний процес WinLogon, що відповідає в операційній системі Windows NT за аутентифікацію користувачів, має свій власний робочий стіл – сукупність вікон, одночасно видимих на екрані дисплея. Цей робочий стіл називається *столом аутентифікації.* Ніякий інший процес, у тому числі і імітатор, не має доступу до робочого столу аутентифікації і не може розташувати на нім своє вікно.

Після запуску Windows NT на екрані комп'ютера виникає початкове вікно робочого столу аутентифікації, що містить вказівку натиснути на клавіатурі клавіші <Ctrl>+<Alt>+<Del>. Повідомлення про натиснення цих клавіш передається тільки системному процесу WinLogon, а для решти процесів, зокрема для всіх прикладних програм, їх натиснення відбувається абсолютно непомітно. Далі проводиться перемикання на інше, так зване *реєстраційне* вікно робочого столу аутентифікації. У нім-то якраз і розміщується запрошення користувачеві ввести своє ідентифікаційне ім'я і пароль, які будуть сприйняті і перевірені процесом WinLogon.

Для перехоплення призначеного для користувача пароля упроваджений в Windows NT імітатор обов'язково повинен уміти обробляти натиснення користувачем клавіш <Ctrl>+<Alt>+<Del>. Інакше відбудеться перемикання на реєстраційне вікно робочого столу аутентифікації, імітатор стане неактивним і не зможе нічого перехопити, оскільки всі символи пароля, введені користувачем, минуть імітатор і стануть надбанням виключно системного процесу WinLogon. Як вже мовилося, процедура реєстрації в Windows NT влаштована таким чином, що натиснення клавіш <Ctrl>+<Alt>+<Del> проходить безслідно для всіх процесів, окрім WinLogon, і тому призначений для користувача пароль поступить саме йому.

Звичайно, імітатор може спробувати відтворити не початкове вікно робочого столу аутентифікації (у якому висвічується вказівка користувачеві одночасно натиснути клавіші <Ctrl>+<Alt>+<Del>), а реєстраційне (де міститься запрошення ввести ідентифікаційне ім'я і пароль користувача). Проте за відсутності імітаторів в системі реєстраційне вікно автоматично замінюється на початкове після короткого проміжку часу (залежно від версії Window NT він може продовжуватися від 30 з до 1 міни), якщо протягом цього проміжку користувач не робить ніяких спроб реєструватися в системі. Таким чином, сам факт дуже довгої присутності на екрані реєстраційного вікна повинен насторожити користувача Windows NT і змусити його ретельно перевірити свою комп'ютерну систему на предмет наявності в ній програмних закладок.

Підводячи підсумок сказаному, можна відзначити, що ступінь захищеності Windows NT від імітаторів достатньо високий. Розгляд захисних механізмів, реалізованих в цій операційній системі, дозволяє сформулювати ***дві необхідні умови***, дотримання яких є обов'язковим для забезпечення надійного захисту від імітаторів:

- системный процес, який при вході користувача в систему отримує від нього відповідні реєстраційне ім'я і пароль, повинен мати свій власний робочий стіл, недоступний іншим процесам;
- перемикання на реєстраційне вікно робочого столу аутентифікації повинне відбуватися абсолютно непомітно для прикладних програм, які до того ж ніяк не можуть вплинути на це перемикання (наприклад, заборонити його).

К сожалению, эти два условия ни в одной из операционных систем, за исключением Windows NT, не соблюдаются. Поэтому для повышения их защищенности от имитаторов можно порекомендовать воспользоваться административными мерами. Например, обязать каждого пользователя немедленно сообщать системному администратору о том, что вход в систему оказывается невозможен с первого раза, несмотря на корректно заданное идентификационное имя и правильно набранный пароль.

**Фільтри**

*Фільтри "*полюють" за всіма даними, які користувач операційної системи вводить з клавіатури комп'ютера. Самі елементарні фільтри просто скидають перехоплене клавіатурне введення на жорсткий диск або в якесь інше місце, до якого має доступ зловмисник. Витонченіші програмні закладки цього типу піддають перехоплені дані аналізу і фільтрують інформацію, що має відношення до призначених для користувача паролів.

Фільтри є резидентними програмами, що перехоплюють одне або декілька переривань, які пов'язані з обробкою сигналів від клавіатури. Ці переривання повертають інформацію про натиснуту клавішу і введений символ, яка аналізується фільтрами на предмет виявлення даних, що мають відношення до пароля користувача.

Виготовити подібного роду програмну закладку не складає великої праці. У операційних системах Windows 95/98 передбачений спеціальний програмний механізм, за допомогою якого в них вирішується ряд завдань, пов'язаних з діставанням доступу до введення з клавіатури, у тому числі і проблема підтримки національних розкладок клавіатур. Наприклад, будь-який клавіатурний русифікатор для Windows є як там не є справжнісіньким фільтром, оскільки покликаний перехоплювати всі дані, що вводяться користувачем з клавіатури комп'ютера. Неважко "допрацювати" його так, щоб разом зі своєю основною функцією (підтримка національної розкладки клавіатури) він заразом виконував би і дії з перехоплення паролів. При цьому завдання створення фільтру стає такий простій, що не вимагає наявності яких-небудь спеціальних знань у зловмисника. Йому залишається тільки непомітно упровадити виготовлену їм програмну закладку в операційну систему і уміло замаскувати її присутність.

У загальному випадку можна стверджувати, що якщо в операційній системі дозволяється перемикати клавіатурну розкладку під час введення пароля, то для цієї операційної системи можливе створення фільтру. Тому, щоб забезпечити її від фільтрів, необхідно забезпечити виконання наступних трьох ***умов:***

- під час введення пароля перемикання розкладок клавіатури не вирішується;
- конфігурувати ланцюжок програмних модулів, що беруть участь в роботі з паролем користувача, може тільки системний адміністратор;

- доступ до файлів цих модулів має виключно системний адміністратор.

Дотримати перше з цих умов в локалізованих для України версіях операційних систем принципово неможливо. Річ у тому, що засоби створення облікових призначених для користувача записів російською мовою є невід'ємною частиною таких систем. Тільки у англомовних версіях систем Windows NT і UNIX передбачені можливості, що дозволяють підтримувати рівень безпеки, при якому дотримуються всі три перераховані умови.

**Заступники**

*Заступники* повністю або частково підміняють собою програмні модулі операційної системи, що відповідають за аутентифікацію користувачів. Подібного роду клавіатурні шпигуни можуть бути створені для роботи в середовищі практично будь-якої многопользовательской операційної системи. Трудомісткість написання заступника визначається складністю алгоритмів, що реалізовуються підсистемою аутентифікації, і інтерфейсів між її окремими модулями. Також при оцінці трудомісткості слід брати до уваги ступінь документированности цієї підсистеми. В цілому можна сказати, що завдання створення заступника значно складніше за завдання написання імітатора або фільтру. Тому фактів використання подібного роду програмних закладок зловмисниками поки відмічено не було. Проте у зв'язку з тим, що в даний час всього більшого поширення набуває операційна система Windows NT, що має могутні засоби захисту від імітаторів і фільтрів, в найшвидшому майбутньому від хакерів слід чекати активнішого використання заступників в цілях діставання несанкціонованого доступу до комп'ютерних систем.

Оскільки заступники беруть на себе виконання функцій підсистеми аутентифікації, перш ніж приступити до перехоплення призначених для користувача паролів вони повинні виконати наступні дії:

- подібно до комп'ютерного вірусу упровадитися в один або декілька системних файлів;
- використовувати інтерфейсні зв'язки між програмними модулями підсистеми аутентифікації для вбудовування себе в ланцюжок обробки введеного користувачем пароля.

Для того, щоб захистити систему від впровадження заступника, її адміністратори повинні строго дотримувати адекватну політику безпеки. І що особливо важливе, підсистема аутентифікації має бути одним з найзахищеніших елементів операційної системи. Проте, як показує практика, адміністратори, подібно до всіх людей, схильні до здійснення помилок. А отже, дотримання адекватної політики безпеки протягом необмеженого періоду часу є нездійсненним завданням. Крім того, як тільки заступник потрапив в комп'ютерну систему, будь-які заходи захисту від впровадження програмних закладок перестають бути адекватними, і тому

необхідно передбачити можливість використання ефективних засобів виявлення і видалення упроваджених клавіатурних шпигунів. Це означає, що адміністратор повинен вести найретельніший контроль цілісності виконуваних системних файлів і інтерфейсних функцій, використовуваних підсистемою аутентифікації для вирішення своїх завдань.

**Як захистити систему від клавіатурних шпигунів**

Клавіатурні шпигуни представляють реальну загрозу безпеці сучасних комп'ютерних систем. Щоб відвести цю загрозу, потрібно реалізувати цілий комплекс адміністративних мерів і програмно-апаратних засобів захисту. Надійний захист від клавіатурних шпигунів може бути побудована тільки тоді, коли операційна система володіє певними можливостями, що утрудняють роботу клавіатурних шпигунів. Оскільки вони були детально описані вищим, не має сенсу знову на них зупинятися. Проте необхідно ще раз відзначити, що єдиною операційною системою, в якій побудова такого захисту можлива, є Windows NT. Так і то з обмовками, оскільки все одно її доведеться забезпечити додатковими програмними засобами, що підвищують ступінь її захищеності. Зокрема, в Windows NT необхідно ввести контроль цілісності системних файлів і інтерфейсних зв'язків підсистеми аутентифікації.

Крім того, для надійного захисту від клавіатурних шпигунів адміністратор операційної системи повинен дотримувати політику безпеки, при якій тільки адміністратор може:

- конфігурувати ланцюжки програмних модулів, що беруть участь в процесі аутентифікації користувачів;
- здійснювати доступ до файлів цих програмних модулів;
- конфігурувати саму підсистему аутентифікації.

### 2.5.4 Аналіз погроз і каналів просочування інформації

Аналіз потенційно можливих погроз інформації є одним з перших і обов'язковим етапом розробки будь-якої захищеної ІС. При цьому складається якомога повніша сукупність погроз, аналізується ступінь ризики при реалізації тієї або іншої загрози, після чого визначаються напрями захисту інформації в конкретній ІС.

Різноманітність потенційних погроз інформації в ІС таке велике, що не дозволяє передбачити кожну з них, тому аналізовані характеристики погроз слід вибирати з позицій здорового глузду, одночасно виявляючи не тільки власне погрози, вірогідність їх здійснення, масштаб потенційного збитку, але і їх джерела.

Ідентифікація погроз припускає розгляд дій і наслідків реалізації погроз. Проблеми, що виникли після реалізації загрози, зводяться до розкриття джерел витоку, модифікації захисту, руйнування інформації або від-

мови в обслуговуванні. Значніші довготривалі наслідки реалізації загрози приводять до втрати управління, порушення таємниці, втрати адекватності даних або інших довготривалих ефектів.

Краще всього аналізувати наслідки реалізації погроз ще на стадії проектування локальної мережі, робочого місця або інформаційної системи, щоб заздалегідь визначити потенційні втрати і встановити вимоги до заходів забезпечення безпеки. Вибір захисних і контрольних заходів на ранній стадії проектування ІС вимагає набагато менших витрат, чим виконання подібної роботи на експлуатованій ІС. Але і в останньому випадку аналіз небезпек допомагає виявити вразливі місця в захисті системи, які можна усунути розумними засобами.

В більшості випадків проведення аналізу можливих небезпек дозволяє персоналу краще усвідомити проблеми, які можуть виявитися під час роботи. Раніше за розробку заходів щодо захисту інформації зазвичай відповідав менеджер системи інформації і управління або автоматизованої обробки даних. Тепер застосовується інший підхід, при якому в кожній організації відповідальність за виконання аналізу небезпек і розробку методики їх виключення покладається на декілька груп службовців.

У робочу групу, покликану аналізувати можливі небезпечні ситуації, рекомендується включати наступних фахівців:

- *аналітика* по небезпечних ситуаціях (особа, призначена для проведення аналізу), відповідального за збір початкових даних і за представлення керівництву інформації, сприяючої кращому вибору захисних заходів;
- *користувачів*, відповідальних за надання аналітикові точної інформації про вживані застосування;
- *службовців*, що займаються експлуатацією будівлі, працівників відділу кадрів, охорону і інших співробітників, які можуть надати інформацію про зовнішні небезпеки і проблеми, пов'язані з навколишнім середовищем;
- *персонал супроводу мережі,* на який покладається відповідальність за надання інформації про апаратне і програмне забезпечення, а також про вживані процедури по обмеженню прав доступу і заміні (зміні) паролів;
- *представників керівництва*, відповідальних за забезпечення захисту цінностей організації.

**Неформальна модель порушника**

**Порушник** – це особа, що зробила спробу виконання заборонених операцій (дій) або помилково і незнанню, або усвідомлено із злим наміром (з корисливих інтересів) або без такого (ради гри або задоволення, з метою самоствердження і тому подібне) з використанням для цього різних мож-

ливостей, методів і засобів. Зловмисником називатимемо порушника, що навмисно йде на порушення з корисливих мотивів.

Неформальна модель порушника відображає його практичні і теоретичні можливості, апріорні знання, час і місце дії і тому подібне Для досягнення своїх цілей порушник повинен докласти зусилля, витратити певні ресурси. Дослідивши причини порушень, можна або вплинути на ці причини (якщо можливо), або точніше визначити вимоги до системи захисту від даного виду порушень або злочинів.

У кожному конкретному випадку виходячи з існуючої технології обробки інформації може бути визначена модель порушника, яка має бути адекватна реальному порушникові для даної ІС.

При розробці моделі порушника формуються:

- припущення про категорії осіб, до яких може належати порушник;
- припущення про мотиви (цілях) порушника;
- припущення про кваліфікацію порушника і його технічної оснащеності;
- обмеження і припущення про характер можливих дій порушників.

По відношенню до ІС порушники бувають **внутрішніми** (з числа персоналу) або **зовнішніми** (сторонні особи).

*Внутрішніми* порушниками можуть бути:

- користувачі (оператори) системи;
- персонал, обслуговуючий технічні засоби (інженери, техніки);
- співробітники відділів розробки і супроводу ПО (прикладні і системні програмісти);
- технічний персонал, обслуговуючий будівлі (прибиральники, електрики, сантехніки і інші співробітники, що мають доступ в будівлі і приміщення, де розташовані компоненти ІС);
- співробітники служби безпеці ІС;
- керівники різних рівнів посадової ієрархії.

*Сторонніми* порушниками можуть бути:

- клієнти;
- випадкові відвідувачі;
- представники підприємств, що забезпечують енерго-, водо- і теплопостачання організації;
- представники конкуруючих організацій (іноземних спецслужб) або особи, що діють за їх завданням;
- будь-які особи за межами контрольованої території.

Можна виділити три основні мотиви порушень безпеки ІС з боку користувачів:

- безвідповідальність;
- самоствердження;
- корисливий інтерес.

При порушеннях, викликаних *безвідповідальністю*, користувач цілеспрямовано або випадково проводить які-небудь руйнівні дії, не пов'язані проте із злим наміром. В більшості випадків це наслідок некомпетентності або недбалості.

Деякі користувачі рахують діставання доступу до системних наборів даних крупним успіхом, затіваючи свого роду гру "користувач – проти системи" ради *самоствердження* або у власних очах, або в очах колег.

Порушення безпеки ІС може бути викликано і *корисливим інтересом* користувача системи. В цьому випадку він цілеспрямовано намагатиметься подолати систему захисту для доступу до інформації в ІС, що зберігається, передається і обробляється. Навіть якщо ІС має засоби, що надзвичайно ускладнюють проникнення, повністю захистити її від цього практично неможливо.

## 2.6 Підходи до створення комплексної системи захисту інформації

Для розгляду проблеми захисту інформації (ЗІ) в загальному вигляді виділимо в її наочній області три наступні ієрархії: структурну, причинно-наслідкову і функціональну.

Способи ЗІ залежать від типу інформації, форми її зберігання, обробки і передачі, типу носія інформації, а також передбачуваного способу нападу і наслідків його по впливу на інформацію (копіювання, спотворення, знищення).

В основному власник інформації не знає де, коли і яким чином буде здійснено напад, тому йому необхідно виявити сам факт нападу.

### 2.6.1 Підходи до створення комплексної системи захисту інформації

Визначення потенційної цінності інформації дозволяє подумати насамперед про безпеку найбільш важливих секретів, витік яких здатний завдати збитку.

Досвід застосування систем ЗІ (СЗІ) показує, що ефективною може бути лише комплексна система захисту інформації (КСЗІ).

#### 2.6.1.1 Структура системи захисту інформації

Для забезпечення ефективності захисту інформації всі використовувані засоби і заходи доцільно об'єднати в систему захисту інформації, яка має бути функціонально самостійною підсистемою ІС. Головна властивість системи захисту – адаптивність її при зміні структури технологічних схем або умов функціонування ІС. Іншими принципами можуть бути:

- ✧ мінімізація витрат, максимальне використання серійних засобів;
- ✧ забезпечення вирішення необхідної сукупності завдань захисту;
- ✧ комплексне використання засобів захисту, оптимізація архітектури;
- ✧ зручність для персоналу;
- ✧ простота експлуатації.

Систему захисту інформації доцільно будувати у вигляді взаємозв'язаних підсистем: криптографічного захисту; забезпечення юридичної значущості електронних документів; захисту від несанкціонованого доступу (НСД); організаційно-правового захисту; управління системою захисту інформації (СЗІ).

Побудова системи захисту інформації у такому вигляді дозволить забезпечити комплексність процесу захисту інформації ІС, керованість процесу і можливість адаптації при зміні умов функціонування ІС.

*Підсистема криптографічного захисту* об'єднує засоби такого захисту інформації і по ряду функцій кооперується з підсистемою захисту від НСД.

*Підсистема забезпечення юридичної значущості електронних документів* служить для додання юридичного статусу документам в електронному уявленні і є визначальним моментом при переході до безпаперової технології документообігу. Дану підсистему зручно і доцільно розглядати як частина підсистеми криптографічного захисту.

*Підсистема захисту від НСД* запобігає доступу несанкціонованих користувачів до ресурсів ІС.

*Підсистема управління СЗІ* призначена для управління ключовими структурами підсистеми криптографічного захисту, а також контролю і діагностування програмно-апаратних засобів і забезпечення взаємодії всіх підсистем СЗІ.

*Підсистема організаційно-правового захисту* призначена для регламентації діяльності користувачів ІС і є впорядкованою сукупністю організаційних рішень, нормативів, законів і правил, що визначають загальну організацію робіт по захисту інформації в ІС.

Обгрунтований вибір необхідного рівня захисту інформації є системообразующей завданням, оскільки як заниження, так і завищення рівня неминуче веде до втрат. При цьому останнім часом роль даного питання різко зросла у зв'язку з тим, що, по-перше, тепер в число що захищаються окрім військових, державних і відомчих, включені також секрети промислові, комерційні і навіть особисті, а по-друге, сама інформація все більше ставати товаром.

2.6.1.2 Політика інформаційної безпеки

**Політика інформаційної безпеки** (ПІБ) – сукупність законів, правив, практичних рекомендацій і практичного досвіду, що визначають управлінські і проектні рішення в області захисту інформації. На основі ПІБ будується управління, захист і розподіл критичної інформації в системі. Вона повинна охоплювати всі особливості процесу обробки інформації, визначаючи поведінку ІС в різних ситуаціях.

Для конкретної ІС політика безпеки має бути індивідуальною. Вона залежить від технології обробки інформації, використовуваних програмних і технічних засобів, структури організації і так далі При розробці і проведенні політики безпеки в життя доцільно дотримувати наступні принципи:

1) неможливість минути захисні засоби;
2) посилення найслабкішої ланки;
3) неприпустимість переходу у відкритий стан;
4) мінімізація привілеїв;
5) розділення обов'язків;
6) багаторівневий захист;
7) різноманітність захисних засобів;
8) простота і керованість інформаційної системи;
9) забезпечення загальної підтримки мерів безпеки.

Основу політики безпеки складає спосіб управління доступом, що визначає порядок доступу суб'єктів системи до об'єктів системи. Назва цього способу, як правило, визначає назву політики безпеки.

Для вивчення властивостей способу управління доступом створюється його формальний опис – математична модель. При цьому модель повинна відображати стани всієї системи, її переходи з одного стану в інше, а також враховувати, які стани і переходи можна вважати за безпечні в сенсі даного управління.

В даний час краще всього вивчено два види політики безпеки: *виборча* і *повноважна (мандатна)*, засновані відповідно на *виборчому* і *повноважному* способах *управління доступом*.

**Виборче управління доступом** – метод управління доступом суб'єктів системи до об'єктів, заснований на ідентифікації і пізнанні користувача, процесу і/або групи, до якої він належить.

**Мандатне управління доступом** – концепція доступу суб'єктів до інформаційних ресурсів по грифові секретності дозволеної до користування інформації, визначуваному влучній секретності. Крім того, існує набір вимог, що підсилює дію цих політик і призначений для управління інформаційними потоками в системі.

Засоби захисту, призначені для реалізації якого-небудь з названих способу управління доступом, тільки надають можливості надійного управління доступом або інформаційними потоками. Визначення прав

доступу суб'єктів до об'єктів і/або інформаційних потоків (повноважень суб'єктів і атрибутів об'єктів, привласнення міток критичності і так далі) входить в компетенцію адміністрації системи.

Основою виборчої політики безпеки є *виборче управління доступом*, яке має на увазі, що:

- ✧ всі суб'єкти і об'єкти системи мають бути ідентифіковані;
- ✧ права доступу суб'єкта до об'єкту системи визначаються на підставі деякого правила (властивість вибірковості).

Для опису властивостей виборчого управління доступом застосовується модель системи на основі матриці доступу (МД), іноді її називають матрицею контролю доступу. Така модель отримала назву *матричної*.

Матриця доступу є прямокутною матрицею, в якій об'єкту системи відповідає рядок, а суб'єктові стовпець. На перетині стовпця і рядка матриці указується тип дозволеного доступу суб'єкта до об'єкту. Зазвичай виділяють такі типи доступу суб'єкта до об'єкту, як "доступ на читання", "доступ на запис", "доступ на виконання" і ін. *Матриця доступу* – найбільш простий підхід до моделювання систем доступу.

Виборча політика безпеки найширше застосовується в комерційному секторі, оскільки її реалізація на практиці відповідає вимогам комерційних організацій по розмежуванню доступу і підзвітності, а також має прийнятну вартість і невеликі накладні витрати.

Основу повноважної політики безпеки складає повноважне управління доступом, яке має на увазі, що:

- ✧ всі суб'єкти і об'єкти системи мають бути однозначно ідентифіковані;
- ✧ кожному об'єкту системи привласнена мітка критичності, що визначає цінність інформації, що міститься в нім;
- ✧ кожному суб'єктові системи привласнений рівень прозорості, що визначає максимальне значення мітки критичності об'єктів, до яких суб'єкт має доступ.

Коли сукупність міток має однакові значення, говорять, що вони належать до одного рівня безпеки. Організація міток має ієрархічну структуру і, таким чином, в системі можна реалізувати ієрархічно висхідний потік інформації (наприклад, від рядових виконавців до керівництва). Чим важливіше об'єкт або суб'єкт, тим вище його мітка критичності. Тому найбільш захищеними виявляються об'єкти з найбільш високими значеннями мітки критичності.

Кожен суб'єкт, окрім рівня прозорості, має поточне значення рівня безпеки, яке може змінюватися від деякого мінімального значення до значення його рівня прозорості.

Основне призначення повноважної політики безпеки – регулювання доступу суб'єктів системи до об'єктів з різним рівнем критичності і за-

побігання просочуванню інформації з верхніх рівнів посадової ієрархії в нижние, а також блокування можливого проникнення з нижніх рівнів у верхні. При цьому вона функціонує на тлі виборчої політики, додаючи її вимогам ієрархічно впорядкований характер (відповідно до рівнів безпеки).

### 2.6.2 Методи й засоби захисту інформації в інформаційно-обчислювальних мережах

2.6.2.1 Класифікація методів і способів захисту інформації

***Методи захисту*** можна розділити на *організаційних, технічних, криптографічних* і *програмних.*

***Засоби захисту*** у свою чергу можна розділити на *напади, що постійно діють* і *включаються при виявленні спроби.* По **активності** вони діляться на *пасивних, напівактивних* і *активних.* По **рівню забезпечення ЗІ** засобу захисту підрозділяються на 4 класи: *системи слабкого захисту* (1 клас), *системи сильного захисту, системи дуже сильного захисту, системи особливого захисту.*

В рамках системи множина і різноманітність видів захисту інформації визначається способами дії на дестабілізуючі чинники або причини, що породжують їх, на елементи ІС, інформацію, що захищається, і навколишнє середовище у напрямі підвищення показників захищеності інформації. Ці способи можуть бути класифіковані таким чином: морально-етичні і законодавчі, адміністративні і психологічні, захисні можливості програмних і апаратних засобів.

До *морально-етичних засобів захисту* можна віднести всілякі норми, які склалися у міру розповсюдження обчислювальних засобів в тій або іншій країні. Наприклад, подібно до того як в боротьбі проти піратського копіювання програм в даний час в основному використовуються заходи виховного плану, необхідно упроваджувати в свідомість людей аморальність усіляких замахів на порушення конфіденційності, цілісності і доступності чужих інформаційних ресурсів.

*Законодавчі засоби захисту* – це закони, ухвали Уряду і укази Президента, нормативні акти і стандарти, якими регламентуються правила використання і обробки інформації обмеженого доступу, а також вводяться заходи відповідальності за порушення цих правил. Правова регламентація діяльності в області захисту інформації має на меті захист інформації, складовій державну таємницю, забезпечення прав споживачів на отримання якісних продуктів, захист конституційних прав громадян на збереження особистої таємниці, боротьбу з організованою злочинністю.

*Адміністративні заходи* – це дії, підприємства, що робляться керівництвом, або організації для забезпечення інформаційної безпеки. До таких заходів відносяться конкретні правила роботи співробітників підприємства,

наприклад режим роботи співробітників, їх посадові інструкції, строго визначальні порядок роботи з конфіденційною інформацією на комп'ютері. До адміністративних заходів також відносяться правила придбання підприємством засобів безпеки. Представники адміністрації, які несуть відповідальність за захист інформації, повинні з'ясувати, наскільки безпечним є використання продуктів, придбаних у зарубіжних постачальників. Особливо це стосується продуктів, пов'язаних з шифруванням. У таких випадках бажано перевірити наявність у продукту сертифікату, виданого російськими тестуючими організаціями.

*Психологічні заходи* безпеки можуть грати значну роль в зміцненні безпеки системи. Зневага обліком психологічних моментів в неформальних процедурах, пов'язаних з безпекою, може привести до порушень захисту. Розглянемо, наприклад, мережу підприємства, в якій працює багато видалених користувачів. Час від часу користувачі повинні міняти паролі (звичайна практика для запобігання їх підбору). У даній системі вибір паролів здійснює адміністратор. У таких умовах зловмисник може подзвонити адміністраторові по телефону і від імені легального користувача спробувати отримати пароль. При великій кількості видалених користувачів не виключено, що такий простій психологічний прийом може спрацювати.

До *фізичних засобів захисту* відносяться екранування приміщень для захисту від випромінювання, перевірка апаратури, що поставляється, на відповідність її специфікаціям і відсутність апаратних "жучків", засоби зовнішнього спостереження, пристрої, блокуючі фізичний доступ до окремих блоків комп'ютера, різні замки і інше устаткування, що захищають приміщення, де знаходяться носії інформації, від незаконного проникнення і так далі

*Технічні засоби* інформаційної безпеки реалізуються програмним і апаратним забезпеченням обчислювальних мереж. Такі засоби, звані також службами мережевій безпеці, вирішують найрізноманітніші завдання по захисту системи, наприклад контроль доступу, що включає процедури аутентифікації і авторизації, аудит, шифрування інформації, антивірусний захист, контроль мережевого графіка і багато інших завдань. Технічні засоби безпеки можуть бути або вбудовані в програмне (операційні системи і додатки) і апаратне (комп'ютери і комунікаційне устаткування) забезпечення мережі, або реалізовані у вигляді окремих продуктів, створених спеціально для вирішення проблем безпеки.

### 2.6.2.2 Комп'ютерна інформація – об'єкт правовідносин

К правовым средствам защиты принадлежат действующие в стране законы, указы и прочие нормативные акты, которые регламентируют правила пользования информацией и ответственность за их нарушение. Они являются сдерживающим фактором для потенциальных нарушителей.

Відповідно до ст.41 Конституції, інформація є предметом державної охорони, яка забезпечується **Законом** *"Про інформацію"* від 2 жовтня 1992 року, **Законом** *"Про захист інформації в автоматизованих системах"* та ст. 361[1] кримінального кодексу України *"Про незаконне втручання в роботу електронно-обчислювальних машин (комп'ютерів), систем і комп'ютерних мереж"*.

Закон України "Про інформацію" закріплює право громадян України на інформацію, закладає правові основи інформаційної діяльності. Ґрунтуючись на Декларації про державний суверенітет України та Акті проголошення її незалежності. Закон стверджує інформаційний суверенітет України і визначає правові форми міжнародного співробітництва в галузі інформації. Серед найважливіших досягнень, що закріплюються ним, слід назвати : визначення поняття інформації (ст.1) та права власності на інформацію (ст.38), встановлення основних принципів інформаційних відносин (ст.5) та визначення державної інформаційної політики (ст.6), одним з головних напрямків якої є створення загальної системи охорони інформації.

До принципових визначень, що безпосередньо стосуються безпеки інформації, належить також визначення режиму доступу до інформації (ст.28) та встановлення його видів (ст. 30). Стаття 47 встановлює відповідальність за порушення законодавства про інформацію та містить перелік випадків її настання.

Закон України "Про захист інформації в автоматизованих системах" встановлює основи регулювання правових відносин щодо захисту інформації в автоматизованих системах за умови дотримання права власності громадян України і юридичних осіб на інформацію та права доступу до неї, права власника інформації на її захист, а також встановленого чинним законодавством обмеження на доступ до інформації.

Дія Закону поширюється на будь-яку інформацію, що обробляється в автоматизованих системах. Цим Законом встановлюються об'єкти правового захисту – інформація, що обробляється в АС, права власників цієї інформації та власників АС, права користувача (ч.І ст.2). Зазначено, що захисту підлягає будь-яка інформація в АС, необхідність захисту якої визначається її власником або чинним законодавством (ч.2 ст.2).

Безпосередньо встановлюються загальні вимоги щодо захисту інформації (ст.11) та обов'язок винних осіб понести дисциплінарну, адміністративну, кримінальну чи матеріальну відповідальність за порушення вимог закону про захист інформації (ст.17).

Саме цією нормою забезпечується легітимність введення статті 361[1] кримінального кодексу (далі – КК), якою здійснюється кримінально-правова охорона зазначеного кола суспільних відносин.

Стаття 361[1] КК передбачає відповідальність за дві самостійні форми злочинних дій:

1) **умисне втручання у роботу автоматизованих систем**, *що призвело до перекручення чи знищення інформації або носіїв інформації;*

2) **розповсюдження програмних і технічних засобів**, *призначених для незаконного проникнення в автоматизовані системи і здатних спричинити перекручення або знищення інформації чи то носіїв інформації.*

В Статье 361 Уголовного кодекса Украины "О незаконном вмешательстве в работу электронно-вычислительных машин (компьютеров), систем и компьютерных сетей" указано, что "незаконное вмешательство в работу автоматизированных электронно-вычислительных машин, их систем или компьютерных сетей, что привело к ущербу или уничтожению компьютерной информации, или носителей такой информации, а также распространению компьютерного вируса путем применения программных и технических средств, предназначенных для незаконного проникновения в эти машины, системы или компьютерные сети и способных нанести ущерб, или уничтожение компьютерной информации или носителей такой информации – *наказываются штрафом до семидесяти не обложенными налогами, минимумов доходов граждан или исправительными роботами на срок до двух лет, или ограничением воли на один и тот же срок".*

*Одни и те же действия, если они нанесли важный вред или содеянные повторно или за предыдущим заговором группой лиц. – наказываются ограничением воли на срок до пяти лет или лишением свободы на срок от трех до пяти лет.*

Предметом цих злочинів є:

1) **автоматизовані системи (АС)** – системи, що здійснюють автоматизовану обробку даних і до складу яких входять технічні засоби їх обробки (засоби обчислювальної техніки і зв'язку), а також методи і процедури, програмне забезпечення (ст. І Закону від 5 липня 1994 р. "Про захист інформації в автоматизованих системах");

2) **носії інформації** – фізичні об'єкти, поля і сигнали, хімічні середовища, нагромаджувачі даних в інформаційних системах (ст. 24 Положення про технічний захист інформації в Україні, затвердженого постановою Кабінету Міністрів України від 9 вересня 1994 р. );

3) **інформація, яка використовується в АС**, – сукупність усіх даних і програм, які використовуються в АС незалежно від засобу їх фізичного та логічного представлення (ст. 1 Закону від 5 липня 1994 р.);

4) **програмні і технічні засоби**, призначені для незаконного проникнення в автоматизовані системи.

Об'єктивна сторона злочинів характеризується двома видами дій:

1) втручанням в роботу АС;

2) розповсюдженням програмних і технічних засобів, призначених для незаконного проникнення в АС і здатних спричинити перекручення або знищення інформації чи носіїв інформації.

***Розповсюдження програмних і технічних засобів, призначених для незаконного проникнення в автоматизовані системи і здатних спричинити перекручення або знищення інформації чи носіїв інформації, відноситься до формальних складів злочинів***, оскільки об'єктивна сторона цього злочину виражається в самих діях, незалежно від того, спричинили вони чи ні наслідки у вигляді перекручення або знищення інформації чи носіїв інформації.

Під **втручанням у роботу АС** законодавець пропонує розуміти будь-які *дії винного, що впливають на обробку АС інформації, яка в ній зберігається, або яка вводиться чи передається для обробки в АС, тобто дії, що впливають на всю сукупність операцій (зберігання, введення, записування, перетворення, зчитування, зберігання, знищення, реєстрація), що здійснюються за допомогою технічних і програмних засобів, включаючи обмін по каналах передачі даних*. При втручанні в роботу АС здійснюється порушення її роботи, яке спричиняє спотворення процесу обробки інформації, внаслідок чого перекручується або знищується сама інформація чи її носії.

**Знищення інформації** – *це її втрата, коли інформація в АС перестає існувати для фізичних і юридичних осіб, які мають право власності на неї в повному чи обмеженому обсязі*. Як знищення, втрату інформації треба розглядати і її блокування, тобто припинення доступу до інформації користувачам АС. Втручання в роботу АС може бути і в формі впливу на канали передачі інформації як між технічними засобами ім обробки і зберігання всередині АС, так і між окремими АС, внаслідок чого інформація, що передається для обробки, знищується чи перекручується. Такі дії можуть виражатись, наприклад, в електромагнітному, лазерному і іншому впливі на носії інформації, в яких вона матеріалізується, або по яких вона передається; в формуванні сигналів полів засобів і блоків програм, вплив яких на інформацію, її носії і засоби технічного захисту викликає порушення цілісності інформації, її знищення чи спотворення; у включенні до бібліотек програм спеціальних програмних блоків, зміни програмного забезпечення і інших подібних діях, що призводять до порушення цілісності інформації.

**Перекручення інформації** – *це зміна її змісту, порушення її цілісності, в тому числі і часткове знищення.*

**Під розповсюдженням** програмних і технічних засобів, призначених для незаконного проникнення в АС і здатних спричинити перекручення або знищення інформації чи носіїв інформації, треба розуміти:

1) їх передачу будь-яким способом і на будь-яких підставах (продаж, дарування, обмін, надання можливості скопіювати тощо) з метою їх використання для несанкціонованого доступу до інформації особами, які згідно з правилами розмежування доступу до інформації, встановленими

власником інформації чи уповноваженою ним особою, не мають права доступу до такої інформації;

2) їх "закладку" в АС на стадії її виготовлення, ремонту, реалізації, користування з метою використання в майбутньому для здійснення несанкціонованого доступу до інформації;

3) ознайомлення інших осіб із змістом програмних засобів чи технічними характеристиками або технологією виготовлення та використання технічних засобів для незаконного проникнення в АС.

**Програмні засоби, призначені для незаконного проникнення в АС**, – це спеціальні комп'ютерні програми (програмні блоки, програмне забезпечення), з допомогою яких можна здійснити несанкціонований доступ до інформації, яка зберігається чи обробляється в АС, і які здатні спотворити або знищити інформацію (її носії) шляхом спотворення процесу обробки інформації.

**Технічні засоби, призначені для незаконного проникнення в АС**, – це різного роду прилади, обладнання, устаткування тощо, з допомогою яких можливе або безпосереднє підключення до АС чи каналів передачі даних, або які здатні шляхом формування сигналів, полів, середовищ створити умови для несанкціонованого доступу до інформації з метою ознайомлення з такою інформацією особами, які не мають права доступу до неї, або з метою впливу на процес обробки інформації в АС, порушення роботи АС, перекручення або знищення інформації чи її носіїв.

### 2.6.3 Основні відомості про криптографію. Підсистема криптографічного захисту

#### 2.6.3.1 Основні поняття

Наукою, що вивчає математичні методи захисту інформації шляхом її перетворення, є **криптологія**. Кріптология розділяється на два напрями – криптографію і криптоаналіз.




**Криптографія** – наука, що вивчає методи перетворення інформації, що забезпечують її конфіденційність і автентичність. Нагадаємо, що під конфіденційністю розуміють неможливість отримання інформації з перетвореного масиву без знання додаткової інформації. Автентичність інформації полягає в достовірності авторства і цілісності.

**Криптоаналіз** об'єднує математичні методи порушення конфіденційності і автентичності інформації без знання ключів.

Існує ряд суміжних, але таких, що не входять в криптологию галузей знання. Так, забезпеченням скритності інформації в інформаційних масивах займається *стеганографія*. Забезпечення цілісності інформації в умовах випадкової дії знаходиться у веденні теорії *перешкодостійкого кодування*. Нарешті, існують математичні методи стискування інформації.

Сучасна криптографія включає чотири крупні розділи:

1) симетричні криптосистеми;

2) криптосистеми з відкритим ключем;

3) системи електронного підпису;

4) управління ключами.

Основні напрями використання криптографічних методів – передача конфіденційній інформації по каналах зв'язку (наприклад, електронна пошта), встановлення достовірності повідомлень, що передаються, зберігання інформації (документів, баз даних) на носіях в зашифрованому вигляді.

Як інформація, що підлягає шифруванню і расшифрованию, а також електронному підпису розглядатимуться тексти, побудовані на деякому алфавіті.

*Алфавіт* – кінцева безліч використовуваних для кодування інформації знаків.

*Текст* – впорядкований набір з елементів алфавіту.

Як приклади алфавітів, використовуваних в сучасних інформаційних системах, можна привести наступні:

- ✧ алфавіт Z33 – 32 букви російського алфавіту (виключаючи "е") і пропуск;
- ✧ алфавіт Z256 – символи, що входять в стандартні коди ASCII і KOI-8;
- ✧ двійковий алфавіт – Z2 = {0,1};
- ✧ вісімковий або шістнадцятиричний алфавіти.

*Зашифрування* – процес перетворення відкритих даних в зашифрованих за допомогою шифру. Замість терміну "відкриті дані" часто уживаються терміни "відкритий текст" і "початковий текст", а замість терміну "зашифровані дані" – "шифрований текст".

*Розшифрування* – процес, зворотний зашифруванню, тобто процес перетворення зашифрованих даних у відкритих за допомогою ключа. У деяких вітчизняних джерелах окремо виділяють термін дешифровка, маючи на увазі під цим відновлення початкового тексту на основі шифрованого без знання ключа, тобто методами криптоаналізу. Надалі рахуватимемо расшифрування і дешифровку синонімами.

Під *шифруванням* розуміється процес зашифрування або расшифрування. Також термін "шифрування" (у вузькому сенсі) використовується як синонім зашифрування. Проте невірно як синонім шифрування використовувати термін "кодування" (а замість шифру – код), оскільки під

кодуванням зазвичай розуміють представлення інформації у вигляді знаків (букв алфавіту).

*Криптографічна система*, або *шифр,* є сімейство Т оборотних перетворень відкритого тексту в шифрований. Члени цього сімейства індексуються або позначаються символом $Д_о$; параметр до зазвичай називається *ключем*. Перетворення $Т_k$ визначається відповідним алгоритмом і значенням ключа $Д_о$.

**Ключ** – конкретне значення деяких параметрів алгоритму криптографічного перетворення, що забезпечує вибір одного перетворення з сімейства. Секретність ключа повинна забезпечувати неможливість відновлення початкового тексту по шифрованому.

Простір ключів $Д_о$ – набір можливих значень ключа. Зазвичай ключем є послідовний ряд букв алфавіту. Слід відрізняти поняття "ключ" і "пароль". *Пароль* також є секретною послідовністю букв алфавіту, проте використовується не для шифрування (як ключ), а для ідентифікації суб'єктів.

Криптосистеми підрозділяються на симетричні і асиметричні (або з відкритим (публічним) ключем). У симетричних криптосистемах для зашифрування і расшифрування використовується один і той же ключ. У системах з відкритим ключем використовуються два ключі – відкритий (публічний) і закритий (секретний), які математично зв'язані один з одним. Інформація зашифровується за допомогою відкритого ключа, який доступний всім охочим, а розшифровується за допомогою закритого ключа, відомого тільки одержувачеві повідомлення.

Терміни "розподіл ключів" і "управління ключами" відносяться до процесів системи обробки інформації, змістом яких є вироблення і розподіл ключів між користувачами.

**Електронним (цифровий) підписом** називається приєднуване до тексту його криптографічне перетворення, яке дозволяє при отриманні тексту іншим користувачем перевірити авторство і достовірність повідомлення.

Спроба прочитання або підробки зашифрованого повідомлення, обчислення ключа методами криптоаналізу називається **криптоатакой**, або *атакою на шифр*. Вдалу криптоатаку називають *зломом*.

**Кріптостойкостью** називається характеристика шифру, визначальна його стійкість до расшифруванию без знання ключа (тобто криптоатаке). Показник криптостойкости – головний параметр будь-якої криптосистеми.

*Імітозащита* – захист системи шифрованого зв'язку від нав'язування помилкових даних.

### 2.6.3.2 З історії криптографії

Проблема захисту інформації шляхом її перетворення, що виключає її прочитання сторонньою особою хвилювала людський розум з давніх часів. Історія криптографії – ровесниця історії людської мови. Більш того, спо-

чатку писемність сама по собі була криптографічною системою, оскільки в стародавніх суспільствах нею володіли тільки вибрані. Священні книги Давнього Єгипту, Стародавньої Індії тому приклади.

Довгий час заняття криптографією було долею одинаків. Серед них були обдаровані учені, дипломати і священнослужителі. Відомі випадки, коли криптографію вважали навіть за чорну магію. Цей період розвитку криптографії, як мистецтва, тривав з незапам'ятних часів до початку XX століття, коли з'явилися перші шифрувальні машини. Розуміння математичного характеру вирішуваних криптографічних завдань прийшло тільки в середині XX століття, після робіт видатного американського ученого До. Шенона.

Свій слід в історії криптографії залишив багато хто добре відомі історичні особи.

Перші відомості про використання шифрів у військовій справі пов'язані з ім'ям спартанського полководця Лісандра (шифр "Сциталь", V століття д.н.э). Цезарь використовував в листуванні шифр, який увійшов до історії як "шифру Цезаря". У стародавній Греції був винайдений вид шифру, який надалі називався "Квадрат Політія". Братерство франкмасонов з моменту свого виникнення (VIII століття) розробило і використовувало цілу систему особливих шифрів.

Одну з перших книг по криптографії написав абат І. Трітемій (1462-1516 рр.) що жив в Германії. У 1566 р. відомий механік і математик Д. Кардано опублікував роботу з описом винайденої ним системи шифрування ("грати Кардано"). Франція XVI століття залишила в історії криптографії шифри короля Генріха IX і Рішельє. У Росії найбільш відомим шифром є "цифрова азбука" 1700 року, автором якої був Петро І.

Деякі відомості про властивості шифрів і їх застосування могло знайти в художній літературі і кіно. Хороше і докладне пояснення одне з простих шифрів – шифру заміни і методів його розтину міститься в двох відомих розповідях: "Золотий жук" Е. По і "Танцюючі чоловічки" А. Конан-дойля.

З широким розповсюдженням писемності криптографія почала формуватися як самостійна наука.

Бурхливий розвиток криптографічні системи отримали в роки першою і другою світових воєн. Починаючи з післявоєнного часу і по нинішній день поява обчислювальних засобів прискорила розробку і вдосконалення криптографічних методів.

В даний час розроблена велика кількість різних методів шифрування, створені теоретичні і практичні основи їх застосування. Що пригнічують число цих методів може бути успішно використано і для закриття інформації.

Отже, криптографія дає можливість перетворити інформацію таким чином, що її прочитання (відновлення) можливе тільки при знанні ключа.

Як інформація, що підлягає шифруванню і дешифровці, розглядатимуться тексти, побудовані на деякому алфавіті (рис. 2.22).

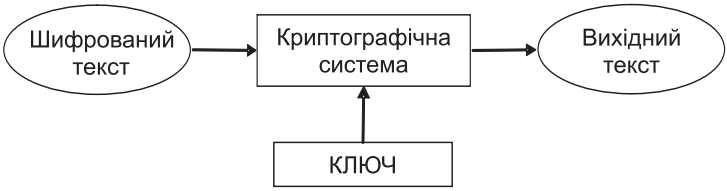



Рисунок 2.22. Процедура шифрування файлів

Розглянемо детальніше деякі приклади.

***Шифр "Сциталь".*** Цей шифр відомий з часів війни Спарти і Персії проти Афін. Спартанський полководець Лісандр підозрював персів в зраді, але не знав їх таємних планів. Його агент в стані персів прислав шифроване повідомлення, яке дозволило Лісандру випередити персів і розгромити їх. Шифроване повідомлення було написане на поясі офіційного гінця від персів таким чином: агент намотав пояс на сциталь (дерев'яний циліндр певного діаметру) і написав на поясі повідомлення уподовж сциталя; потім він розмотав пояс і вийшло, що упоперек поясу безладно написані букви. Гонець не здогадався, що узор на його красивому поясі насправді містить зашифровану інформацію. Лісандр узяв сциталь такого ж діаметру, акуратно намотав на нього пояс і уподовж сциталя прочитав повідомлення від свого агента.

Відзначимо, що в цьому шифрі перетворення відкритого тексту в шифрований полягає в певній перестановці букв відкритого тексту. Тому клас шифрів, до яких відноситься і шифр "Сциталь", – це ***перестановочні шифри.***

***Шифр Цезаря.*** Цей шифр реалізує наступні перетворення відкритого тексту: кожна буква відкритого тексту замінюється третьою після неї буквою в алфавіті, який вважається по кругу, тобто після букви "я" слідує буква "а". Тому клас шифрів, до яких відноситься і шифр Цезаря, – це ***підстановлювальні шифри.***

Наприклад, відкритий текст КРИПТОГРАФІЯ при такому способі шифрування перетвориться в шифртекст НУЛТХСЕУГЧЛВ.

Відзначимо, що Цезарь замінював букву третьою після неї буквою, але можна замінювати і п'ятою, і будь-який інший. Головне, щоб той, кому посилається шифроване повідомлення, знав цю величину зрушення.

***Шифр Віженера.*** Цей шифр відноситься до сімейства *полиалфавитных підстановлювальних шифрів.* Його найзручніше уявити, як шифр Цезаря із змінною величиною зрушення. Щоб знати, на скільки зрушувати чергову букву відкритого тексту, заздалегідь обмовляється спосіб запам'ятовування зрушень. Для цієї мети використовується ключове слово,

кожна буква якого своїм номером в алфавіті указує величину зрушення. Ключове слово повторюється стільки раз, скільки потрібно для заміни всіх букв відкритого тексту. Наприклад, якщо ключове слово ВАЗА, а відкритий текст – КРИПТОГРАФІЯ, значить, ключове слово дасть наступну послідовність зрушень букв відкритого тексту

319131913191.

При такому способі шифрування відкритий текст перетвориться в шифротекст

НССРХПЛСГХСА.

Подальший розвиток ідеї ключового слова, а саме ідея запам'ятовувати спосіб перетворення відкритого тексту за допомогою якої-небудь книги, привів до виникнення різних видів так званих книжкових шифрів.

Результати криптографічних досліджень реалізуються зараз у вигляді шифруючих пристроїв, вбудованих в сучасні системи зв'язку. Тому криптографи обмежені у вибиранні засобів тим рівнем техніки і технології, який досягнутий на даний момент. Така залежність відбивається і на виборі використовуваного в криптографії математичного апарату.

Умовно можна виділити три принципово разных етапу в розвитку математичного апарату криптографії.

До 40-х років XX століття застосовувалися тільки електромеханічні шифромашины, тому і спектр математичних перетворень був обмежений, в основному, методами комбінаторного аналізу і теорії вірогідності.

Після появи електронної техніки, а тим більше комп'ютерів, сильно змінився і математичний апарат криптографії. Отримали розвиток прикладні ідеї і методи теорії інформації, алгебри, теорії кінцевих автоматів.

Роботи Діффі і Хеллмана (70-і роки) послужили поштовхом для бурхливого розвитку нових напрямів математики: теорії односторонніх функцій, доказів з нульовим розголошуванням. У наш час прогрес саме в цих напрямах визначає практичні можливості криптографії.

Проте для того, щоб криптографічні методи перетворення забезпечили ефективний захист інформації, вони повинні задовольняти ряду вимог. У стислому вигляді їх можна сформулювати таким чином:

- складність і стійкість криптографічного закриття повинні вибиратися залежно від об'єму і ступеня секретності даних;
- надійність закриття має бути такій, щоб секретність не порушувалася у тому випадку, коли зловмисникові стає відомий метод закриття;
- метод закриття, набір використовуваних ключів і механізм їх розподілу не можуть бути дуже складними;
- виконання процедур прямого і зворотного перетворень має бути формалізованим. Ці процедури не повинні залежати від довжини повідомлень;

– помилки, що виникають в процесі виконання перетворення, не повинні розповсюджуватися на текст повною мірою і по системі;
– що вноситься процедурами захисту надмірність має бути мінімальною.

2.6.3.3 Підсистема криптографічного захисту

Підсистема об'єднує засоби криптографічного захисту інформації. По ряду функцій підсистема кооперується з іншими підсистемами . Структурно підсистема складається з:

- ✧ програмних засобів симетричного шифрування даних;
- ✧ програмно-апаратних засобів цифрового підпису електронних документів.

Функції підсистеми передбачають:

- ✧ забезпечення цілісності того що передається по каналах зв'язку і інформації, що зберігається;
- ✧ имитозащиту повідомлень, яки передаются по каналах зв'язку;
- ✧ утаєння смислового змісту конфіденційних повідомлень, яки передаются по каналах зв'язку і зберігаються на носіях;
- ✧ забезпечення аутентифікації джерела даних.

Функції підсистеми направлені на ліквідацію найбільш поширених погроз повідомленням в автоматизованих системах:

- ✧ погроз, направлених на несанкціоноване ознайомлення з інформацією;
- ✧ несанкціонованого читання інформації на машинних носіях ЕОМ;
- ✧ незаконного підключення до апаратури і ліній зв'язку;
- ✧ перехоплення ЕМІ з ліній зв'язку;
- ✧ погроз, направлених на несанкціоновану модифікацію (порушення цілісності) інформації;
- ✧ зміни службовій або змістовній частині повідомлення;
- ✧ підміни повідомлення;
- ✧ вилучення (знищення) повідомлення і так далі.

2.6.3.4 Класифікація криптографічних методів

В даний час не існує закінченої і загальноприйнятої класифікації криптографічних методів, оскільки багато хто з них знаходиться у стадії розвитку і становлення. Найбільш доцільною представляється класифікація, представлена на рис. 2.23.

Під *шифруванням* в даному випадку розуміється такий вид криптографічного закриття, при якому перетворенню піддається кожен символ повідомлення, що захищається. Всі відомі способи шифрування розбиті на п'ять груп: *підстановка* (заміна), *перестановка, аналітичне перетворення,*

*гаммирование* і *комбіноване шифрування.* Кожен з цих способів може мати декілька різновидів.

Під *кодуванням* розуміється такий вид криптографічного закриття, коли деякі елементи даних (не обов'язково окремі символи), що захищаються, замінюються заздалегідь вибраними кодами (цифровими, буквеними, буквено-цифровими поєднаннями і так далі). Цей метод має два різновиди: смислове і символьне кодування. При *смисловому кодуванні* кодовані елементи мають цілком певний сенс (слова, пропозиції, групи пропозицій). При *символьному кодуванні* кодується кожен символ тексту, що захищається. Символьне кодування по суті збігається з підстановлювальним шифруванням.

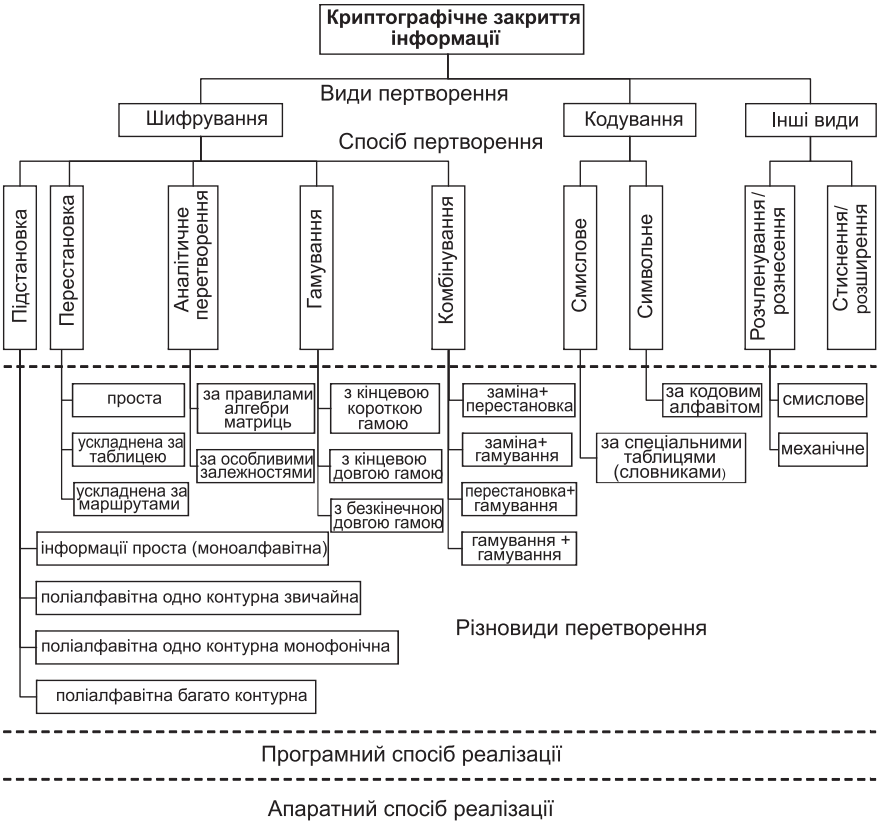



Рис. 2.23 Класифікація криптографічних методів

До окремих видів криптографії відносяться методи *розчленування-рознесення* і *стиснення-розширення даних*. Розчленування-рознесення

полягає в тому, що масив даних, що захищаються, ділиться (розчленовуються) на такі елементи, кожен з яких окремо не дозволяє розкрити зміст інформації, яка захищається. Виділені таким чином елементи даних розносяться по різних зонах пам'яті або розташовуються на різних носіях. Стиснення даних є заміною однакових рядків даних або послідовностей однакових символів, що часто зустрічаються, деякими заздалегідь вибраними символами.

Методи перестановки і підстановки характеризуються хорошими ключами, а їх надійність пов'язана з складним алгоритмом перетворення. При аддитивних методах користуються простими алгоритмами перетворення, забезпечуючи надійність за допомогою ключів великого об'єму.

Іноді говорять про *блокові методи,* маючи на увазі перші дві групи, в яких алгоритм працює відразу над великим блоком інформації, і про *потокові методи,* де шифрування відбувається знак за знаком. Проте при використанні аддитивних методів перетворення може здійснюватися відразу над цілим машинним словом і метод набуває ознак блокового.

Все різноманіття існуючих криптографічних методів можна звести до наступних класів перетворень (рис.2.24):

Багатоалфавітна підстановка – найбільш простий вид перетворень, що полягає в заміні символів початкового тексту на інших (того ж алфавіту) за більш менш складним правилом. Для забезпечення високої криптостойкости потрібне використання великих ключів.

Перестановки – нескладний метод криптографічного перетворення. Використовується, як правило, у поєднанні з іншими методами.




Рис. 2.24. Класи перетворень симметричних криптосистем

Гаммірованіє – цей метод полягає в накладенні на початковий текст деякої псевдовипадкової послідовності, що генерується на основі ключа.

Блокові шифри є послідовністю (з можливим повторенням і чергуванням) основних методів перетворення, вживаною до блоку (частини) шифрованого тексту. Блокові шифри на практиці зустрічаються частіше, ніж "чисті" перетворення того або іншого класу через них вищій крипто-

стойкости. Російський і американський стандарти шифрування засновані саме на цьому класі шифрів.

Криптосистеми розділяються на *симетричних* і *асиметричних (з відкритим ключем)*.

Під **симетричними криптографічними системами** розуміються такі криптосистеми, в яких для шифрування і дешифровки використовується один і той же ключ.

Для користувачів це означає, що, перш ніж почати використовувати систему, необхідно отримати загальний секретний ключ, щоб виключити до нього доступ потенційного зловмисника.

Достоїнства:

- *Простота.*
- *Нізькі обчислювальні потужності.*

Основні недоліки систем шифрування з симетричним – секретним ключем:

- ✧ необхідність розповсюдження секретних ключів;
- ✧ захист від підробки повідомлень за допомогою имитовставки, отриманою на симетричному ключі, не забезпечує захист інформації від підробки одним з власників секретного ключа. Це приводить до того, що недобросовісний учасник інформаційного обміну може сфабрикувати нібито отримане повідомлення або відмовитися від повідомлення, посланого їм самим.

Всі ці проблеми дозволяє вирішити криптографія з відкритим ключем.

**Асиметричні** (або **з відкритим ключем**) **криптографічні системи** – криптосистеми, в яких для шифрування і дешифровки використовуються різні ключі, зв'язані між собою деякою залежністю. При цьому дана залежність така, що встановити один ключ, знаючи інший, з обчислювальної точки зору дуже важко. Один з ключів (наприклад, ключ шифрування) може бути зроблений загальнодоступним, і в цьому випадку проблема отримання загального секретного ключа для зв'язку відпадає. Якщо зробити загальнодоступним ключ дешифровки, то на базі отриманої системи можна побудувати систему аутентифікації передаваних повідомлень. Оскільки в більшості випадків один ключ з пари делается загальнодоступним, такі системи і отримали назву криптосистем з відкритим ключем.

Шифрують за допомогою відкритого ключа. Дешифрують за допомогою закритого ключа. У основі асиметричного шифрування лежать:

**1. Необоротна функція вигляду**

$$y = F(x), \tag{2.1}$$

де:

F – алгоритм шифрування.

x – повідомлення.

Необоротна функція називається функція вигляду y = F(x), яка задовольняє двом умовам:

– *За будь-яким аргументом x ми можемо обчислити значення.*
– x = F–1(y) *інвертувати x з обчислювальної точки зору не можливо.*

**2. Функція з секретом вигляду**

$$\mathbf{y = Fk(x),} \qquad (2.2)$$

де:
k – секрет.

Ця функція відповідає наступним умовам:

– *По відомому* ***x*** *при будь-якому* k *легко обчислюється функція* ***y***.
– *При відомому* k *і відомому* **y** *за значенням* ***x = F–1k(y)***.
– *При невідомому* k *ми ні в якому випадку не можемо інвертувати (відновити) цю функцію.*

**Третій спосіб шифрування.**

**Змішаний метод шифрування** – це коли ключ шифрується асиметричним методом і передається партнеру, а сама інформація передається симетричним методом шифрування.

Терміни розподіл ключів і управління ключами відносяться до процесів системи обробки інформації, змістом яких є складання і розподіл ключів між користувачами.

Одній з основних проблем, зв'язаних з використанням криптосистем з відкритим ключем, є проблема забезпечення достовірності передаваних відкритих ключів.

**Електронні цифрові підписи**

***Цифровий підпис*** (ЦП) є рядком даних, який залежить від деякого секретного параметра (ключа), відомого тільки підписуючій особі, і від змісту підписуваного повідомлення, представленого в цифровому вигляді. Таким чином, цифровий підпис пов'язує повідомлення з деяким об'єктом, що породжує або підписуючим його.

*Електронним (цифровий) підписом* називається приєднуване до тексту його криптографічне перетворення, яке дозволяє при отриманні тексту іншим користувачем перевірити авторство і достовірність повідомлення.

Завдання цифрового підпису полягає в тому, щоб забезпечити захист електронного документа не тільки від підробки стороннім зловмисником, але і від дій недобросовісних учасників інформаційного обміну. Вимоги, що пред'являються до цифрового підпису, можна сформулювати таким чином:

1. Цифровий підпис доводить, що той, що підписав згоден із змістом документа.

2. Вона є невід'ємною частиною документа і не може бути перенесена на інший документ.

3. Після того, як документ підписаний, його неможливо змінити.

4. Від підпису не можна відмовитися, тобто той, що підписав не зможе згодом стверджувати, що документ підписаний без його відома.

Механізм цифрового підпису може забезпечувати перевірку достовірності документів навіть надійніше, ніж звичайний підпис під паперовим документом. Очевидно, проте, що цифровий підпис не є повним аналогом звичайного підпису. Так, наприклад, можливість постановки підпису авторучкою належить конкретній фізичній особі і не може бути передана або загублена. Що стосується цифрового підпису, то секретний ключ, що дозволяє підписувати електронні документи, може бути переданий іншій особі, а також втрачений або вкрадений.

Ще одна особливість цифрового підпису полягає в тому, що вона не пов'язана з єдиним екземпляром документа. Знищивши підписаний екземпляр паперового документа, можна бути упевненим, що другого такого самого більше немає. Що стосується знищення електронного документа з цифровим підписом, то це ні про що не говорить, оскільки може існувати необмежена кількість нічим копій цього документа, що не відрізняються.

Щоб обійти дану проблему, підписують не само повідомлення, а значення хэш-функции від нього. **Функція хешування (хэш-функция)** – перетворення, які відображають рядки змінної довжини в безліч рядків фіксованої довжини, як правило, меншою, ніж у початкових рядків.

Тобто у основі електронний-цифрового підпису (ЕЦП) лежить поняття хэш-функції виду

$$S = H(k, T), \tag{2.3}$$

де: S – ЕЦП,
k – ключ,
T – текст.

Функція задовольняє наступним умовам:

- початковий текст може бути безмежний;
- функція S повинна мати фіксований розмір;
- за наявності відомого значення k ця функція легко відновлюється $T = H–1(S, k)$ при невідомому k це неможливо.

Якщо є необхідність в забезпеченні і цілісності, і секретності повідомлення, то доцільно спочатку підписати повідомлення, а потім вже зашифрувати його разом з підписом.

## 2.7 Комп'ютерні віруси та антивірусні програми

### 2.7.1 Віруси. Види й класифікації

Історично виникнення вірусів пов'язане з ідеєю створення програм, що самовідтворюються, – концепції, що йде своїм корінням в п'ятдесяті роки. Ідея механізмів, що самовідтворюються, досліджувалася ще Джоном фон Нейманом, який в 1951 р. запропонував метод створення таких механізмів.

Таким чином, попередниками вірусів були різного роду програми (деякі з них у вигляді ігор), принцип роботи яких полягав в здатності самовідтворюватися. Вже на початку 60-х з'явилася "Гра для опівнічників", в якій декілька асемблерних програм, названих "організмами", завантажувалися в пам'ять комп'ютера. Організми, створені одним гравцем (тобто що належать до одного вигляду), повинні були знищувати представників іншого вигляду і захоплювати життєвий простір. Переможцем вважався той гравець, чиї організми захоплювали всю пам'ять або набирали найбільшу кількість очок. І вже тоді в описі гри термін "вірус" застосовувався до одного з видів організмів.

Приблизно у 1970 р. була створена програма, що самовідтворюється, для однієї з перших комп'ютерних мереж – APRAnet. Програма Creeper подорожувала по мережі, виявляючи свою появу повідомленням: "I'M THE CREEPER ... CATCH ME IF YOU CAN".

Потім була написана програма Rabbit, яка розмножувалася на трьох сполучених між собою машинах IBM, причому появу нових підзадач викликало уповільнення роботи, а потім і повне зависання машин.

Другим прикладом вірусоподібних програм була досить відома гра Animal, розроблена для Univac 1108. Суть цієї гри полягала в тому, що людина замислювала деяку тварину, і програма, ставлячи питання, намагалася визначити, яку тварину загадав чоловік. Автор передбачив в ній можливість самовіютворення. Коли програма вгадувала неправильно, вона просила користувача запропонувати питання, яке дозволило б поліпшити її здібності до відгадування даної тварини. Запам'ятавши це питання, програма не тільки модифікувала себе, але і намагалася переписати свою оновлену (покращену) копію в інший каталог. Якщо там вже була програма Animal, то вона стиралася. Інакше створювалася нова копія.

Найзнаменитішим вірусом того часу став мережевий вірус Морриса. Аспірант факультету інформатики Корнельського університету інфікував за допомогою написаної ним програми велику кількість комп'ютерів (близько 6000), підключених до американської національної мережі Internet. "Черв'як Моріса" вражав тільки комп'ютери типу SUN 3 і VAX, які використовували варіанти ОС UNIX версії 4 BSD. Збитки від нього були оцінені в 96 мільйонів доларів.

Події 1985-86 рр. за часом співпали з швидким зростанням виробництва і різким зниженням цін на ПЕВМ серії IBM PC. Тому другий етап в розвитку вірусів пов'язаний з досягненням "критичної маси" виробництва цього наймасовішого комп'ютера в історії розвитку обчислювальної техніки. В 1987 р., одночасно в декількох країнах відбулися спалахи зараження комп'ютерів вірусами. Віруси почали бути загрозою для всіх користувачів персональних ЕОМ. На відміну від першого етапу, коли розробки вірусоподібних програм носили дослідницький характер, і автори, заручившись згодою користувачів, займалися чистим експериментатором на благо системного програмування, другий етап носить характер протистояння користувачів безвідповідальним або навіть кримінальним "елементам".До числа "першопроходців" на даному етапі відносять бутовий вірус "Brain", розроблений братами Алві. За неперевіреними (і, ймовірно, завищеним) даними, приведеним McAfee, він заразив тільки в США більше 18 тисяч комп'ютерів.

В кінці восьмидесятих паніку в США і західноєвропейських країнах викликала історія з так званою AIDS Information Trojan – троянською програмою, що розповсюджувалася у складі пакету з базою даних про захворювання синдромом придбаного імунодефіциту (СНІД). Як програма, так і база даних були записані на дискеті, яка була розіслана 20 тисячам замовників, включаючи ряд медичних і суспільних організацій США, Франції, Великобританії, ФРН, Данії, Норвегії, Швеції і багатьох інших країн.

На сьогоднішній день ніхто в світі не може вказати точну кількість таких програм, оскільки вони з'являються кождного дня все нові і нові. Але проблема боротьби з такими програмами вимагає щонайменше класифікації вказаних програм.

**Визначення терміну "комп'ютерний вірус"**

Перші дослідження штучних конструкцій, що саморазмножающихся, проводилися в середині нинішнього століття відомими ученими-математиками Джоном фон Нейманом, Норбертом Винером, академіком АН СРСР і АН УРСР Віктором Михайловичем Глушковым. У їх працях дано визначення і проведено математичний аналіз кінцевих автоматів, у тому числі і самовідтворюваних.

Термін "комп'ютерний вірус" з'явився пізніше – офіційно вважається, що його уперше спожив студент Каліфорнійського університету (США) Фред Коуэн в серпне 1984 р. на 7-ій конференції, яка відбулася в США. Відтоді минуло немало часу, але чіткого визначення терміну і досі немає.

Дати формальне визначення поняття "Комп'ютерний вірус" дуже складно. Традиційне визначення Фреда Коуэна – "комп'ютерний вірус – програма, яка може заражати інші програми, модифікуючи їх дода-

ванням своїй, можливо зміненою, копії" – при детальному розборі ставить в "безвихідь": незрозуміле, що це за "копія", якщо вона не співпадає з оригіналом.

Комп'ютерний вірус – одно з найцікавіших явищ, яке можна спостерігати в результаті розвитку комп'ютерної і інформаційної техніки. Суть його зводиться до того, що програми набувають властивостей, властивих живим організмам: вони народжуються, розмножуються, помирають.

Головна умова існування вірусів – універсальна інтерпретація інформації в обчислювальних системах. Вірус в процесі зараження програми може інтерпретувати її як дані, а в процесі виконання – як старанний код. Цей принцип був покладений в основу усіх сучасних комп'ютерних систем, які використовують архітектуру фон Неймана.

Але зовсім очевидно, що ключовим поняттям у визначенні вірусу є його можливість до саморазмножению, оскільки це – єдиний критерій, який дозволяє відрізнити віруси від інших програм. При цьому "копії" вірусу можуть структурно і функціонально відрізнятися один від одного. Можна дати вірусу визначення типу : "вірус – це програма, яка робить інші віруси", але воно рекурсивне. Отже, ми зіткнулися з майже філософською проблемою: як визначити об'єкти, ключовими властивостями яких є породження собі подібних, причому подібність визначається за допомогою тієї самої властивості – робити собі подібних і так далі. Вихід полягає у встановленні відношення подібності вірусу і його копії за допомогою набору стосунків еквівалентності, які належать до структури, змісту, алгоритму вірусу і які мають сенс для заданої операційної системи комп'ютера.

З поняттям "Комп'ютерний вірус" тісно пов'язано таке поняття, як сигнатуру. Сигнатура – це фрагмент коду, який є присутнім в усіх копіях вірусу і тільки в них. Іншими словами, сигнатура – це підпис вірусу, який однозначно визначає його присутність або відсутність в програмі.

**Комп'ютерний вірус** – це невелика, складна, ретельно складена і небезпечна програма, яка може самостійно розмножуватися, переноситися на диски і дискети, прикріплятися до програм, передаватися мережею, порушуючи роботу комп'ютера. При цьому вона намагається виконати приховане самокопирование в різні області старанних кодів інших програм, максимально захищається від виявлення і після закінчення "інкубаційного" періоду заявляє про себе непрогнозованими руйнівними діями.

Сформулювавши визначення комп'ютерного вірусу, таким чином, можна зробити узагальнювальні висновки. Дійсно, можна вважати, що комп'ютерний вірус – це звичайна програма для ЕОМ, тобто результат творчої розумової праці людини, і нічого спільного з біологічним походженням вірусів не має.

Програма, в якій знаходиться вірус, називається "зараженою". Коли така програма починає свою роботу, то спочатку управління отримує вірус.

Вірус знаходить і "заражає" інші програми, а також виконує які-небудь шкідливі дії, наприклад, нівечить файли або таблицю розміщення файлів на диску, "засмічує" оперативну пам'ять. Для маскування вірусу дії відносно зараження інших програм і завдання шкоди можуть виконуватися не завжди, а тільки за певних умов. Після того, як вірус виконає необхідні йому дії, він передає управління тій програмі, в якій знаходиться, і вона працює так, як і звичайна, тобто зовнішня робота зараженої програми виглядає точно так, як і незараженою.

Багато різновидів вірусів побудовано таким чином, що при запуску зараженої програми вірус залишається резидентним, тобто до перезавантаження, в пам'яті комп'ютера і тільки іноді заражає програми і виконує шкідливі дії.

У багатьох користувачів персональних ЕОМ в результаті незнання механізму роботи комп'ютерних вірусів, а також під впливом різних чуток і некомпетентних публікацій в газетах, науково-популярних журналах, а також коментарів по телебаченню створився своєрідний комплекс залякування вірусами ("вирусофобия"). Цей комплекс проявляється по-різному.

По-перше, це – схильність приписувати будь-яке ушкодження даних або незвичайні явища в комп'ютері вірусам.

Наприклад, якщо у "вирусофоба" не форматується дискета, то він пояснює це не дефектами дискети або дисковода, а дією вірусів. Якщо на жорсткому диску з'являється збійний блок, то в цьому теж винні віруси, хоча найчастіше незвичайні явища в комп'ютері викликані помилками користувача, програмами або дефектами устаткування, а не дією вірусів.

По-друге, існують перебільшені уявлення про можливості вірусів. Деякі користувачі думають, наприклад, що досить вставити в дисковод заражену дискету, щоб комп'ютер заразився вірусом. Поширена також думка, що для комп'ютерів, які об'єднані в мережу або навіть стоять в одному приміщенні, зараження одного комп'ютера обов'язково приведе до зараження інших комп'ютерів. Кращими ліками від "вирусофобии" є знання того, як діють віруси, що вони можуть і чого вони не можуть. Віруси є звичайними програмами, і нездібні ні до яких надприродних дій.

**Історія виникнення комп'ютерних вірусів**

Попередниками вірусів були програми, принцип роботи яких полягав в здатності саморазмножаться. Ці програми клонували себе, займали системні ресурси і таким чином знижували продуктивність системи. Найчастіше вони не передавалися від системи до системи і були суто місцевими явищами – помилками або витівками системних програмістів, що обслуговували комп'ютер. Перший же інцидент, який сміливо можна назвати епідемією "комп'ютерного вірусу", стався на системі Univax 1108. Вірус, що дістав назву "Pervading Animal", дописував себе до виконуваних

файлів – робив практично те ж саме, що тисячі сучасних комп'ютерних вірусів.

Уперше під операційну систему Тепех був створений вірус "The Creeper", що використав для свого поширення глобальні комп'ютерні мережі. Програма Creeper подорожувала по мережі, виявляючи свою появу повідомленням: "I' M Till. CREEPER .. CATCH МЕг IF YOU CAN". Вірус був в змозі самостійно увійти до мережі через модем і передати свою копію видаленій системі. Для боротьби з цим вірусом була створена програма "The Reeper" – перша відома антивірусна програма.

Епідемія першого IBM – PC вірусу "Brain", що заражає 360Кб дискети практично миттєво захопила весь світ. Вірус "Brain" являвся також і першим стелс-вирусом. При спробі читання зараженого сектора він "підставляв" його незаражений оригінал.

Незалежно один від одного з'являється, ще декілька вірусів для IBM – PC. Це знаменитий у минулому "Lehigh", що заражає тільки COMMAND. COM, "Suriv-1" (інша назва – "Aprilt st"). COM-файлы, що заражає, "Suriv-2", що заражає (уперше) ЕХД-файли. і "Suriv-3", що заражає як COM-, так і EXE-файли. З'являються також декілька завантажувальних вірусів ("Yale" в США, "Stoned" в Новій Зеландії і "PingPong" в Італії) і перший файловий вірус "Cascade", що самошифрующийся.

Не залишилися осторонь і ні IBM-компьютеры: було виявлено декілька вірусів для Apple Macintosh, Commodore Amiga і Atari ST. У грудні 1987 року сталася перша відома поголовна епідемія мережевого вірусу "Cristmas Tree", написаного на мові REXX і поширюючого себе в операційному середовищі VM/CMS. 9-го грудня цей вірус був запущений в мережу Bitnet в одному з університетів Західної Німеччини і проник через шлюз в European Academic Research Network (EARN) і потім – в мережу IBM VNet. Через чотири дні (13 грудня) вірус паралізував мережу – вона була забита його копіями. При запуску вірус виводив на екран зображення різдвяної ялиночки і розсилав свої копії усім користувачам мережі, чиї адреси були присутніми у відповідних системних файлах NAMES і NETLOG.

У 1988 році почалася поголовна епідемія мережевого вірусу Моріса (інша назва – Internet Worm). Вірус заразив більше 6000 комп'ютерних систем в США (включаючи NASA Research Institute) і практично паралізував їх роботу. Унаслідок помилки в коді вірусу він. як і вірус-черв'як "Cristmas Tree", необмежено розсилав свої копки по інших комп'ютерах мережі і. таким чином, повністю забрав під себе її ресурси. Загальні збитки від вірусу Моріса були оцінені в 96 мільйонів доларів.

Вірус використовував для свого розмноження помилки в операційній системі Unix для VAX і Sun Microsystems. Окрім помилок в Unix вірус використовував декілька інших оригінальних ідей, наприклад, підбір паролів

користувачів. Через рік стався інцидент з "троянським конем" "Aids". Було розіслано 20.000 його копій на дискетах, помічених як "AIDS Information Diskette Version 2.0". Після 90 завантажень системи "Троя" шифрувала імена усіх файлів на диску, робила їх невидимими (атрибут "hidden") і залишав на диску тільки один читаний файл – рахунок на 189 доларів, який слід було послати за адресою PO Box 7, Panama. Автор "Трої" був спійманий і засуджений до тюремного укладення.

На початку дев'яностих років з'явився перший поліморф-генератор MtE, на базі якого через деякий час з'являється відразу декілька полиморф-вирусов. MtE став також прообразом декількох наступних полиморф-генераторов. Епідемія вірусу "Michelangelo" ("March6") і пов'язана з цим істерія – це перший відомий випадок, коли антивірусні компанії роздували галас навколо вірусу не для того, щоб захистити користувачів від якої-небудь небезпеки, а для того, щоб притягнути увагу до свого продукту, тобто в цілях отримання комерційної вигоди.

Усього більшого значення набуває проблема вірусів па компакт-дисках. Швидко ставши популярними, ці диски виявилися одним з основних шляхів поширення вірусів. Зафіксовано відразу декілька інцидентів, коли вірус потрапляв на майстер-диск при підготовці партії компакт-дисків. В результаті на комп'ютерний ринок були випущені досить великі наклади (десятки тисяч) заражених дисків. Природно, що про їх лікування говорити не доводиться – їх доведеться просто знищити.

Один з поворотних моментів в історії вірусів і антивірусів : в "живому виді" виявлений перший вірус для Microsoft Word ("Concept"). Буквально за місяць вірус "блетів" усю земну кулю, заполонив комп'ютери користувачів MS – Word і міцно зайняв перше місце – в статистичних дослідженнях, що проводяться різними комп'ютерними, – виданнями.

У січні 1996 року відбувається два досить помітних події – з'явився перший вірус для Windows 95 ("Win 95.Boza") і епідемія украй складного полиморф-вируса "Zhcngxi" в Санкт-Петербурзі. З березня місяця перша епідемія вірусу для Windows 3.x. Його ім'я – "Win.Tentacle". Цей вірус заразив комп'ютерну мережу в госпіталі і декількох інших установах у Франції. Цікавість цієї події полягала в тому. що це був ПЕРШИЙ Windows – вирус, що вирвався на свободу. До того часу усе Windows – вирусы жили тільки в колекціях і електронних журналах вирусоносителей. А в "живому виді" зустрічалися тільки завантажувальні, DOS – і Macro-вирусы.

"OS 2.AL – P" – перший вірус для OS/2, що коректно заражає, EXL – файлы цієї операційної системи. До цього в OS/2 зустрічалися тільки віруси, які записувалися замість файлу, знищуючи його або діючи методом "компаньйон".

"Laroux" – перший вірус для Microsoft Excel, до того ж спійманий в "живому виді" (практично одночасно в двох нафтовидобувних компаніях

на Алясці і в ЮАР). Як і у MS – Word – вирусов, принцип дії "Laroux" грунтується па наявності у файлах так званих макросів – програм на мові Basic. Такі програми можуть бути включені в електронні таблиці Excel так само, як і в документи MS – Word. Як виявилось, вбудований в Excel мова Basic також дозволяє створювати віруси. Цей же вірус в квітні 1997 став причиною епідемії в комп'ютерних фірмах Москви.

"Win 95.Punch" – перший "резидентний" вірус для Win95. Завантажується в систему як VxD-драйвер, перехоплює звернення до файлів і заражає їх.

З 1996 року почався широкомасштабний наступ комп'ютерних паразитів на операційну систему Windows32 (Windows95 і Windows MT) – и на додатки 1 Microsoft Office. За цей і наступний рік з'явилося декілька десятків вірусів для Windows 95/NT і декілька сотень макро-вірусів. У багатьох з них вирусописатели застосовували абсолютно нові прийоми і методи зараження, додавали стеле- і полиморф-механизмы і тому подібне. Таким чином, комп'ютерні віруси вийшли на новий виток свого розвитку – на рівень 32-бітових операційних систем. За два роки віруси для Windows32 повторили приблизно все ті ж стадії, що рівно 10 років до того пройшли DOS-вирусы, проте на абсолютно новому технологічному рівні.

"Linux.Bliss" – перший вірус для Linux (різновид Юнікса). Так віруси зайняли ще одну "біологічну" нішу. З часом макро-віруси перебралися в Office97/2000/XP.

"Homer" – перший мережевий вірус-черв'як, що використовує для свого розмноження File Transfer Protocol (ftp).

З'являється нова форма вірусу – черв'яки mlRC. Виявилось, що найбільш популярна утиліта Windows IRC (Internet Relay Chat), відома як rnlRC, містила "діру" що дозволяє вірусним скриптам передавати себе по IRC-каналам. У черговій версії IRC w діра була закрита і mIRC-черви канули в лету.

З'являються віруси, що використовують усе більш складні прийоми зараження комп'ютерів і нові методи проникнення через комп'ютерні мережі. Окрім вірусів на арену виходять також численні троянські програми, що крадуть паролі доступу в Інтернет, і декілька утиліт прихованого адміністрування. Разом з цим почалася епідемія цілого сімейства вірусів "Win 32.HLLP.DeTroie", що не лише заражають виконувані файли Windows32, але і здатні передати своєму "хазяїну" інформацію про заражений комп'ютер. Унаслідок використання специфічних бібліотек, присутніх тільки у французькій версії Windows, епідемія торкнулася тільки франкомовних країн. Також був виявлений ще один тип вірусу, що заражає таблиці Excel, – "Excel^Paix" (чи "Formula.Paix"). Цей тип макро-вірусу для свого впровадження в таблиці Excel використовує не звичайну для вірусів

область макросів, а формули, які, як, виявилось, також можуть містити код, що саморазмножающийся.

Зростає епідемія вірусу "Win 95.CIH", що стала спочатку масовою, потім глобальною, а потім поголовною – повідомлення про зараження комп'ютерних мереж і домашніх персональних комп'ютерів обчислювалися сотнями тисяч. Початок епідемії зареєстрований на Тайвані, де невідомий хакер заслав заражені файли в місцеві Інтернет конференції. Звідти вірус пробрався в США, де через недогляд зараженими виявилися відразу декілька популярних Web-серверов – вони поширювали заражені вірусом ігрові програми. Швидше за все, саме ці заражені файли на ігрових серверах і послужили причиною поголовної епідемії вірусу, що не слабшала впродовж усього року. Слід звернути увагу також на небезпечний прояв вірусу: залежно від поточної дати вірус стирав Flash BIOS, що в деяких випадках могло привести до необхідності заміни материнської плати.

Поява гучного "BackOrifice"" ("Backdoor.BO") – утиліти прихованого (хакерського) адміністрування видалених комп'ютерів і мереж. Слідом за "BackOrifice" з'явилися декілька інших аналогічних програм: "NetBus", "Phase" і інші. Також з'явився перший вірус, що заражає виконувані модулі Java – "Java.StangeBrew". Цей вірус не представляв якої-небудь небезпеки для користувачів Інтернет, оскільки на видаленому комп'ютері неможливо використовувати необхідні для розмноження функції. Проте він проілюстрував той факт, що атаковані вірусами, також можуть бути і додатки, активно використовувані при перегляді Web-серверов.

Вірус "VBScript.Rabbit" – Інтернет-експансія комп'ютерних паразитів продовжилася трьома вірусами, що заражають скрипти VisualBasic (VBS-файлы), які активно застосовуються при написанні Web-страниц. Як логічний наслідок VBScript-вирусов стало поява повноцінного HTML-вируса ("HTML.Internal"). Стає досить очевидним, що зусилля вірусописателей починають концентруватися навколо мережевих застосувань, і справа йде до появи повноцінного мережевого вірусу-черв'яка, що використовує можливості MS Windows, Office і що заражає видалені комп'ютери, Web-сервера і що активно поширюється по електронній пошті.

Широке і повсюдне використання глобальної мережі Інтернет привело до того, що в 2000-2001 роках найбільше поширення отримали t віруси класів I – Worm (Internet Worm) і VBS – Worm. Обидва класи вірусів були черв'яками і використовували для свого розмноження електронну пошту, маскуючись під документи, фотографії знаменитостей, корисні програми.

**Творці комп'ютерних вірусів і стан правової бази у сфері антивірусної безпеки.**

На наш погляд, основну масу вірусів створюють люди, які тільки що вивчили мову асемблера і хочуть спробувати свої сили, але не можуть знайти для них кращого застосування. З-під пера подібних "умільців" часто

виходять або модифікації "класичних" вірусів, або віруси украй примітивні з великою кількістю помилок ("студентські").

Найнебезпечнішу групу складають "професійні" віруси. Ці ретельно продумані і відлагоджені програми створюються професійними, у багатьох випадках дуже талановитими програмістами. Такі віруси часто використовують оригінальні і досить складні алгоритми, які недокументированны, а також не відомі способи проникнення в системні області операційних систем. Наприклад, таблиця розміщення файлів FAT є уразливим елементом файлової структури, тому багато програм-вірусів нівечать саме її.

Часто "професійні" віруси виконуються за технологією "стелс". Для автора і його колег залишаються загадковими ті причини, які можуть змусити нормальну людину направити свої неабиякі здібності на виконання такої безглуздої роботи. Не виключено, що деякі з таких людей – це невдахи, які не зуміли ефективно використовувати свої знання і досвід (рис.2.25).

Дамо коротку характеристику групам розробників комп'ютерних вірусів.

***Технопаты.*** Це той самий різновид вандалів, які "розмальовують" кабіни ліфтів, стіни під'їздів, огорож і так далі. Опис цього виду психічних відхилень можна знайти у будь-якому навчальному посібнику по психіатрії.

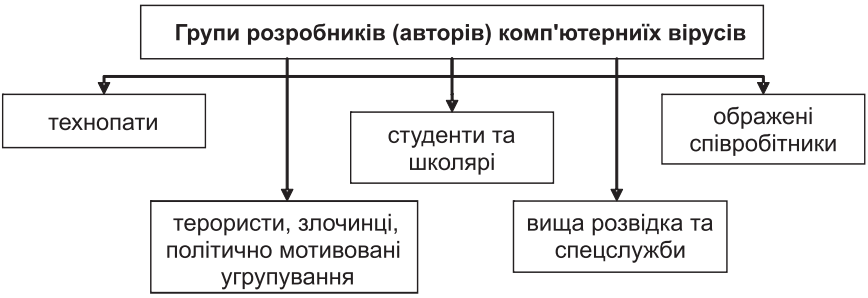



Рисунок 2.25. Класифікація групи розробників комп'ютерних вірусів

***Студенти і школярі.*** У більшості внз, технікумів і шкіл доступ до комп'ютерів досить не контролюється, що дає можливість студентам, учням, технікумів і школярам старших класів займатися чим-небудь, включаючи написання вірусних програм. Написання вірусної програми не потребує особливої підготовки, окрім знайомства з операційною системою і мовою асемблера. На конференції ANTIVIR/як юморной метод профілактики було запропоновано включити в учбовий план лабораторні роботи з написання вірусів і приймати у студентів диференційований залік. Автор цієї пропозиції керувався тим. що у внз це одно з надійних засобів назавжди відбити бажання займатися улюбленим предметом.

***Ображені співробітники.*** Деякі троянські програми створювалися спеціально як засіб мести "поганому начальникові". Слід зазначити швидкий ріст комп'ютерних вірусів "вітчизняного" виробництва. Це природно: адже коли платять лише за присутність на роботі, тобто можливість займатися будь-якою справою, знічев'я" можна спробувати запрограмувати вірус. Тим більше що програмування вірусів все більше і більше перетворюється з мистецтва в ремесло, а серед програмістів все рідше зустрічаються "інтелектуали" і все частіше ремісники – "таперы", які не можуть не вплинути на "індекси порядності" серед представників цієї професії.

В результаті цього комп'ютерне хуліганство росте. Необхідно відмітити, що готовність програмістів помститися за образи за допомогою логічних бомб була вже виявлена у багатьох випадках. Тому така можливість повинна враховуватися керівниками різного рівня при звільненні співробітників, особливо тих, хто є, неврастенічною особою.

***Терористи, злочинці, політично мотивовані угрупування.*** Комп'ютерні віруси можуть бути використані як засіб шантажу або пропаганди. До цієї групи можна віднести розробників вірусу, який потрапляє у базу даних по Спиду (Aids Information Trojans), бутового вірусу Stoned з гаслом легалізувати марихуану та ін. Сюди необхідно віднести також розробників засобів захисту від несанкціонованого копіювань, які передбачають різного роду "каральні заходи" при запуску програми на комп'ютері, що відрізняється від того, на якому виконувалася інсталяція. Віруси можуть використовуватися як зброя шантажу або засіб дискредитації. В зв'язку з цим необхідно нагадати про такий небезпечний злочин, як "вірусний рекет". Один з різновидів останнього – це свідоме зараження програмного забезпечення, яке поставляється разом з комп'ютером, з метою добування додаткового прибутку у вигляді плати за наступну дезинфекцію.

На заході відзначалися випадки вірусного шантажу, коли невідома особа по телефону повідомляє про небезпеку "вибуху" встановленої вірусної "міни", яка здатна знищити цінну інформацію. Так, як і у випадку з шантажистами, які повідомляють про встановлення міни в літаку, навіть тоді, коли небезпека є невірною, персонал витрачає багато часу на пошуки і перевірку усього програмного забезпечення.

***Військова розвідка і спецслужби.*** За деякими даними, комп'ютерні віруси розглядаються різними спецслужбами як одно з можливих заходів боротьби з супротивником. Дійсно, неважко уявити собі катастрофічні наслідки попадання руйнівного вірусу в комп'ютерну систему, яка має стратегічне і тактичне значення. В той же час результати відповідних досліджень, якщо вони проводяться, будуть цілком таємними і вірогідність попадання розроблених для цієї мети вірусів на цивільні комп'ютерні мережі, залишатиметься дуже малою.

Отже, якщо на одному кінці спектру знаходиться дрібний шкідник, який створює вірус з метою продемонструвати "всьому світу" "карко-

ломний", на його думку, ефект, то на іншому кінці спектру знаходиться злочинець, який використовує вірус для шантажу користувачів. У будь-якому випадку виявлення і спроба притягнути до судової відповідальності розробників комп'ютерних вірусів належить до важливої форми боротьби з поширенням комп'ютерних вірусів.

**Правовий статус**

Сьогодні Україна стоїть перед проблемою подальшого розвитку сучасної біотехнології, інформаційних технологій, телекомунікаційних систем і інформатизації суспільства. Головним завданням інформатизації суспільства є створення правових, економічних, технологічних, соціальних і освітніх першооснов для забезпечення будь-якому потенційному користувачеві у будь-який час, у будь-якому місці країни доступу до інформації, яка потрібна для вирішення відповідних господарських, технічних, наукових, соціальних і особистих проблем.

Впровадження електронної обробки інформації в економіці, управлінні кредитній фінансовій сфері зумовило виникнення комп'ютерної злочинності в Україні. Звідси і виникають правові проблеми, особливість яких полягає в тому, що вони починають регулювати громадські стосунки лише тоді, коли вони є досить розвиненими і проявляють в собі протиріччя, які необхідно усунути. Негативним наслідком комп'ютеризації суспільства є поява так званих комп'ютерних порушень. Поширення комп'ютерних систем, об'єднання їх в комунікаційні мережі збільшило можливості їх електронного втручання. Діапазон таких дій дуже широкий: від досить традиційних злочинів до дій, які вимагають високої математичної і технічної підготовки.

Об'єкти атак, які належать комп'ютерним злочинам, можна розділити по категоріях:

- ✧ самі комп'ютери;
- ✧ об'єкти, які можуть бути атаковані за допомогою комп'ютера як інструменту;
- ✧ об'єкти для яких комп'ютер є довкіллям.

Коли комп'ютер сам є об'єктом атаки, то необхідно розглядати його як систему, оскільки він ніколи не використовується самостійно, а тільки в комбінації з периферійними пристроями. Для будь-якої частини цієї загальної системи є реальна загроза стати об'єктом атаки.

Операційну систему і прикладні програми можна розглядати як частина комп'ютерної системи, і як самостійний об'єкт, для якого комп'ютер є довкіллям. З розвитком комп'ютерної техніки стає все складніше провести чітку межу між програмним забезпеченням і апаратною реалізацією. Погрози програмному забезпеченню можуть викликати різні помилки, модифікація програми, її копіювання та ін. Але комп'ютер є і сховищем

машинної інформації – даних і програм. І те і інше є предметом посягань різного роду – від копіювання до знищення або до крадіжки.

Комп'ютерні атаки, на відміну від вбивств або крадійства, поки що не скрізь признаються як злочини. Закони по боротьбі з ними існують, як правило, тільки в індустріально розвинених країнах, які сильно залежать від комп'ютерних технологій. Наприклад, у випадку з вірусом "I Love You" філіппінській владі припало на декілька днів відстрочити перевірки і обшук підозрюваних через те, що вони не могли знайти законну підставу для цього.

Однією з найбільших проблем може бути встановлення самого факту здійснення злочину. Це викликано тим, що зовнішні прояви комп'ютерного злочину у багатьох випадках мало помітні. Так, незаконне копіювання інформації залишається не виявленим, введення в комп'ютер вірусу у більшості випадків списується на не передбачену помилку користувача, який не зміг його "відловити" при спілкуванні із зовнішнім комп'ютерним світом.

Із-за складнощів математичного і апаратного забезпечення рівень розкриття комп'ютерних злочинів нині дуже низький. Часто при очевидно корисливих злочинах потерпілі не поспішають заявити в правоохоронні органи. Іноді винні просто звільняються, переводяться в інші структурні підрозділи. Відмова від кримінального переслідування веде до відсутності загального попередження, він ніби запрошує інших спробувати свої можливості.

У розкритті факту здійснення злочину дуже часто не зацікавлені посадовці, в обов'язки яких входить забезпечення комп'ютерної безпеки. Визнання факту ставить під сумнів їх професійну кваліфікацію, а неадекватність заходів по комп'ютерній безпеці, може викликати серйозні внутрішні проблеми.

Потерпілий може відмовитися від розслідування, боячись, що його фінансові або службові таємниці можуть бути оголошені під час суду.

Прокурори і слідчі органи відмічають, що документи, які надають їм потерпілі від комп'ютерних злочинів, часто бувають недостатніми для пред'явлення обгрунтованих звинувачень.

Цікавим аспектом комп'ютерних злочинів є складність визначення загальних збитків. Часто розслідування комп'ютерних злочинів є дуже дорогим. Тому організації не бажають збільшувати свої втрати додаванням до збитків витрат на розслідування. Через обмежені матеріальні ресурси потерпілі відмовляються від розкриття злочину.

Відомо що, будучи виявленими, комп'ютерні злочинці у багатьох країнах відбувають легкі покарання.

На початку 80-х комп'ютери уряду США піддалися цілому ряду атак. Хакери направляли їх через Нідерланди, тому що в цій країні не існувало

законів, що передбачають покарання за подібні злочини. І уряд Нідерландів ухвалив такий закон.

Обговорюване зараз європейськими лідерами угода зажадає від урядів ухвалення законів проти хакерства, комп'ютерного шахрайства і дитячої порнографії в Internet, а також встановлення заходів покарань і міжнародної співпраці в розслідуванні таких злочинів. Індія, Таїланд і декілька інших країн також обговорюють необхідність ухвалення законів про комп'ютерні злочини.

Нині США мають в розпорядженні найсильніше законодавство для боротьби з хакерством, створенням вірусів, неавторизованим доступом і комп'ютерним шахрайством.

З технічної точки зору різні види комп'ютерних вірусів відрізняються за принципом їх дії. Але в правовому аспекті ці відмінності несуттєві, оскільки усі види вірусів змінюють, обробляють або руйнують дані. А це означає, що в правовому відношенні комп'ютерні віруси рівнозначні. Якщо такі віруси впроваджуються в чужу систему і заподіюють їй збиток, виникає природне питання: хто повинен нести за це відповідальність.

Така відповідальність може мати, передусім, правову основу.

Труднощі в правовому відношенні виникають тоді, коли треба розрізнити посереднє виконання або співвиконавицю, а також у випадках співучасті в чужому діянні, тобто випадках підбурювання або пособництва. Для багатьох це області неусвідомленої правової відповідальності, оскільки не усі усвідомлюють те, що відповідальність може нести не лише прямий виконавець.

До правових засобів захисту належать чинні в країні закони, укази і інші нормативні акти, які регламентують правила користування інформацією і відповідальність за їх порушення. Вони є стримуючим чинником для потенційних порушників.

У Статті 361 Карного кодексу України "Про незаконне втручання в роботу електронно-обчислювальних машин (комп'ютерів), систем і комп'ютерних мереж" вказано, що "незаконне втручання в роботу автоматизованих електронно-обчислювальних машин, їх систем або комп'ютерних мереж, що привело до збитку або знищення комп'ютерної інформації, або носіїв такої інформації, а також поширенню комп'ютерного вірусу шляхом застосування програмних і технічних засобів, призначених для незаконного проникнення в ці машини, системи або комп'ютерні мережі і здатних завдати збитку, або знищення комп'ютерної інформації або носіїв такої інформації, – караються штрафом до сімдесяти не обкладеними податками, мінімумів прибутків громадян або виправними роботами на строк до двох років, або обмеженням волі на один і той же термін".

Одні і ті ж дії, якщо вони завдали важливої шкоди або скоєні повторно або за попередньою змовою групою осіб. – караються обмеженням волі на

строк до п'яти років або позбавленням волі на термін від трьох до п'яти років.

До морально-етичних засобів протидії належать усілякі норми поведінки, які традиційно склалися з поширенням ЕОМ в країні або в суспільстві. Ці норми переважно не є обов'язковими і законодавчо не затверджені, але їх невиконання у більшості випадків веде до падіння авторитету, престижу людини, групи осіб або організації.

***Адміністративні засоби захисту*** – це засоби організаційного характеру, які регламентують процеси функціонування системи обробки інформації, використання її ресурсів, діяльність персоналу, а також порядок взаємодії користувачів з системою так, щоб найбільшою мірою ускладнити або вилучити можливість реалізації загрози безпеки комп'ютерної системи.

До фізичних засобів захисту належать різного роду механічні, електрочи електронно-механічні пристрої і споруди, спеціально призначені для створення фізичних перешкод на можливих шляхах проникнення і доступу потенційних порушників до компонент комп'ютерних систем і інформації, яка вимагає захисту.

***Технічними засобами захисту*** є різні електронні пристрої і спеціальні програми, які виконуються самостійно або в комплексі з іншими засобами, функції захисту, тобто ідентифікація, і аутентифікація користувачів, розмежування доступу до ресурсів, реєстрація подій, криптографічний захист інформації.

Як свідчить практика промислово розвинених країн, зростання комп'ютерної злочинності є однією з характерних і закономірних ознак періоду економічного спаду. Ця проблема залишається актуальною як в нашій країні, так і за її межами, а особливість її полягає в тому, що комп'ютерна злочинність має тенденцію до зростання і придбаває транснаціонального характеру.

Поки в Україні комп'ютерна злочинність не придбала поширення, але необхідно бути готовими до ситуації, коли комп'ютерні правопорушення і злочини можуть стати звичайним явищем.

Чим швидше здійснюватиметься комп'ютеризація наших установ, тим гостріше виникатимуть питання про охорону інформації, попередження зловживань, комп'ютерну безпеку.

Віруси можна розділити на класи за наступними основними ознаками (рис. 2.26):

- ✧ місце існування;
- ✧ операційна система (ОС);
- ✧ особливості алгоритму роботи;
- ✧ деструктивні можливості.

Залежно від місця існування віруси можна розділити на:

- ✧ файлові;
- ✧ завантажувальні;

- ✧ макровіруси;
- ✧ мережеві.

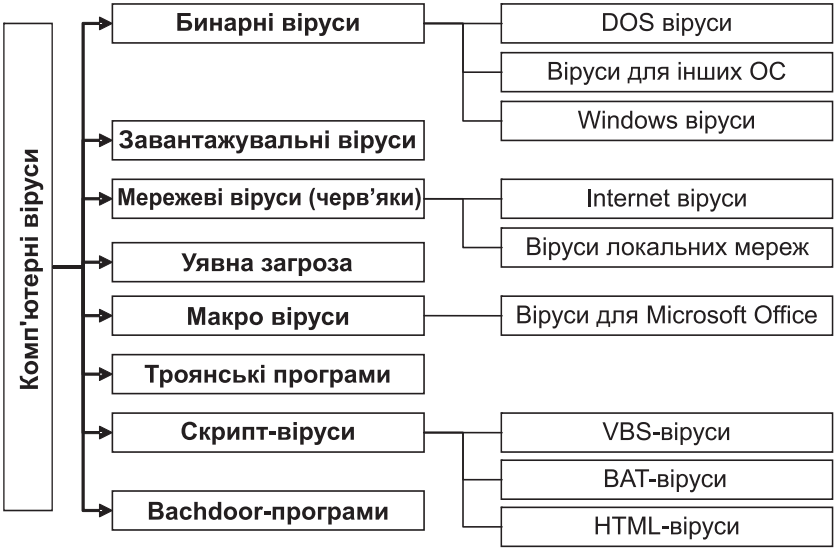



Рисунок 2.26. Класифікація комп'ютерних вірусів

**Бінарні (файлові) віруси** (найбільш поширений тип вірусів) – або різними способами впроваджуються у виконувані файли, або створюють файли-двійники (віруси-компаньйони), або використовують особливості організації файлової системи (link-вирусы).

**Завантажувальні віруси** записують себе або в завантажувальний сектор диска (boot-сектор), або в сектор, системний завантажувач вінчестера (Master Boot Record), що містить, або міняють покажчик на активний boot-сектор.

**Макровіруси** заражають файли-документи і електронні таблиці декількох популярних редакторів.

**Мережеві віруси** використовують для свого поширення протоколи або команди комп'ютерних мереж і електронної пошти.

Існує велика кількість поєднань, наприклад файлово-завантажувальні віруси, що заражають як файли, так і завантажувальні сектори дисків. Такі віруси, як правило, мають досить складний алгоритм роботи, часто застосовують оригінальні методи проникнення в систему, використовують "стеле-" і полиморфик-технологии. Інший приклад такого поєднання – мережевий макровірус, який не лише заражає редаговані документи, але і розсилає свої копії по електронній пошті.

Операційна система (вірніше, до ОС об'єкти якої схильні зараженню), що заражається, є другим рівнем ділення вірусів на класи. Кожен файловий або мережевий вірус заражає файли якою-небудь однієї або декількох ОС – DOS, Windows. Win95/NT, OS/2 і т. д. Макровіруси заражають файли форматів Word, Excel Office 97. Завантажувальні віруси також орієнтовані на конкретні формати розташування системних даних в завантажувальних секторах дисків.

Серед особливостей алгоритму роботи вірусів виділяються наступні:

- ✧ резидентность;
- ✧ використання "стелс"-алгоритмов;
- ✧ самошифрование і полиморфичность;
- ✧ використання нестандартних прийомів.

Резидентний вірус при інфікуванні комп'ютера залишає в оперативній пам'яті свою резидентну частину, яка потім перехоплює звернення ОС до об'єктів зараження і впроваджується в них. Резидентні віруси знаходяться в пам'яті і є активними аж до виключення комп'ютера або перезавантаження ОС. Нерезидентні віруси не заражають пам'ять комп'ютера і зберігають активність обмежений час. Деякі віруси залишають в оперативній пам'яті невеликі резидентні програми, які не поширюють вірус, Такі віруси вважаються нерезидентними.

Резидентними можна рахувати макровіруси, оскільки вони також є присутніми в пам'яті комп'ютера впродовж усього часу роботи зараженого редактора. При цьому роль ОС бере на себе редактор, а поняття "Перезавантаження операційної системи" трактується як вихід з редактора.

У багатозадачних ОС час "життя" резидентного DOS-вируса також може бути обмежене моментом закриття зараженого DOS-окна, а активність завантажувальних вірусів в деяких операційних системах обмежується моментом інсталяції дискових драйверів ОС.

Використання "стелс"-алгоритмов дозволяє вірусам повністю або частково приховати себе в системі. Найбільш поширеним "стелс"-алгоритмом є перехоплення запитів ОС на читання-запис заражених об'єктів і потім "стелс"-вирусы або тимчасово лікують їх, або підставляють замість себе незаражені ділянки інформації. У разі макровірусів найбільш популярний спосіб – заборона викликів меню перегляду макросів. Один з перших файлових "стелс"-вирусов – вірус Frodo, перший завантажувальний "стелс"-вирус – Brain.

Самошифрование і полиморфичность використовуються практично усіма тванями вірусів для того, щоб максимально ускладнити процедуру виявлення вірусу. Полиморфик-віруси (polymorphic) досить важко піддаються виявленню; вони не мають сигнатур, тобто не містять жодної постійної ділянки коду. У більшості випадків два зразки одного і того ж полиморфик-вируса не матимуть жодного збігу. Це досягається

шифруванням основного тіла вірусу і модифікаціями того, що програми-розшифровує.

Різні нестандартні прийоми часто використовуються у вірусах для того, щоб як можна глибше заховати себе в ядрі ОС (як це робить вірус "ЗАРАЗА"), захистити від виявлення свою резидентну копію (віруси TPVO, Trout2), утруднити лікування від вірусу (наприклад, поміщають свою копію в Flash – BIOS) і т. д.

По деструктивних можливостях віруси можна розділити на:

- ✧ **нешкідливі**, тобто що ніяк не впливають на роботу комп'ютера (окрім зменшення вільної пам'яті на диску в результаті свого поширення);
- ✧ **безпечні**, вплив яких обмежується зменшенням вільної пам'яті на диску і графічними, звуковими і іншими ефектами;
- ✧ **небезпечні віруси**, які можуть привести до серйозних збоїв в роботі комп'ютера;
- ✧ **дуже небезпечні** – в алгоритм їх роботи свідомо закладені процедури, які можуть викликати втрату програм, знищити дані, стерти необхідну для роботи комп'ютера інформацію, записану в системних областях пам'яті, і навіть, як свідчить одна з неперевірених комп'ютерних легенд, сприяти швидкому зносу частин механізмів, що рухаються, – вводити в резонанс і руйнувати голівки деяких типів вінчестерів.

Але навіть якщо в алгоритмі вірусу не знайдено гілок, що завдають збитку системі, цей вірус не можна з повною упевненістю назвати нешкідливим, оскільки проникнення його в комп'ютер може викликати непередбачувані і іноді катастрофічні наслідки. Адже вірус, як і всяка програма, має помилки, в результаті яких можуть бути зіпсовані як файли, так і сектори дисків (наприклад, цілком нешкідливий на перший погляд вірус DenZuk досить коректно працює з 360-кілобайтними дискетами, але може знищити інформацію на дискетах більшого об'єму). Досі попадаються віруси, визначальні COM або EXE не по внутрішньому формату файлу, а по його розширенню. Природно, що при неспівпаданні формату і розширення імені файл після зараження виявляється непрацездатним. Можливо також "заклинювання" резидентного вірусу і системи при використанні нових версій DOS, при роботі в Windows або з іншими потужними програмними системами. І так далі.

По особливостях алгоритму можна виділити наступні групи вірусів:

**Віруси-"супутники" (companion)** – це віруси, що не змінюють файли. Алгоритм роботи цих вірусів полягає в тому, що вони створюють для EXE-файлів файли-супутники, що мають те ж саме ім'я, але з розширенням COM або ВАТИ. наприклад, для файлу XCOPY.EXE створюється файл XCOPY. COM. Вірус записується в Сом-файл і ніяк не змінює EXE-файл. При за-

пуску такого файлу DOS першим виявить і виконає Сом-файл, тобто вірус, який запустить і EXE-файл;

2) **Віруси-"черв'яки" (worm)** – віруси, які поширюються в комп'ютерній мережі і так само як і вирусы-"спутники", не змінюють файли або сектори на дисках. Вони проникають в пам'ять комп'ютера з комп'ютерної мережі, обчислюють мережеві адреси інших комп'ютерів і розсилають по них свої копії. Такі віруси іноді створюють робочі файли на дисках системи, але можуть взагалі не звертатися до ресурсів комп'ютера.

3) **"Паразитичні"** – усі віруси, які при поширенні своїх копій обов'язково змінюють вміст дискових секторів або файлів. У цю групу відносяться усі віруси, які не є "черв'яками" або "супутниками".

4) **"Студентські"** – украй примітивні віруси, що часто нерезидентні і містять велике число помилок;

5) **"Стелс"-віруси (віруси-невидимки, Stealth),** що є дуже досконалими програмами, які перехоплюють звернення операційної системи до уражених файлів або секторів дисків і "підставляють" замість себе незаражені ділянки інформації. Окрім цього такі віруси при зверненні до файлів використовують досить оригінальні алгоритми, що дозволяють "обманювати" резидентні антивірусні монітори.

До особливостей алгоритму вірусу можна віднести і швидкість його поширення. Швидкість поширення файлових вірусів, що заражають файли, тільки при запуску на виконання буде нижче, ніж у вірусів, що заражають файли також при їх відкритті, перейменуванні, зміні атрибутів файлу і так далі.

Існують також віруси, які є комбінаціями перелічених вище типів вірусів.

### 2.7.2 Проблеми антивірусного захисту інформації

**Загальнодоступність Internet**

Internet і інформаційна безпека несумісні по самій природі. Internet народився як чисто корпоративна мережа, проте, нині, за допомогою єдиного стека протоколів TCP/IP і єдиного адресного простору, об'єднує не лише корпоративні і відомчі мережі, але і рядових користувачів, які мають можливість отримати прямий доступ в Internet зі своїх домашніх комп'ютерів за допомогою модемів і телефонної мережі загального Користування.

Чим простіше доступ в Мережу, тим нижче рівень інформаційної безпеки, тому первинна простота доступу в Internet – гірше за крадійство, оскільки користувач може навіть і не дізнатися, що у нього були скопійовані – файли, і програми, не кажучи вже про можливість їх псування і коригування (наприклад коригування суми в контракті, або зміни номера рахунку і тому подібне).

Бурхливий ріст Internet характеризується щорічним подвоєнням числа користувачів – унаслідок легкості і дешевизни доступу в Internet до усіх світових інформаційних ресурсів. Платою за користування Internet є загальне зниження інформаційної безпеки, тому для запобігання несанкціонованому доступу до своїх комп'ютерів усі корпоративні і відомчі мережі, а також підприємства, що використовують технологію Intrenet, ставлять фільтри (fire – wall) між внутрішньою мережею і Internet, що фактично означає вихід з єдиного адресного простору.

З'являються усі нові і нові страшні історії про те, як комп'ютерні зломщики, що використовують усе більш витончені прийоми, проникають в чужі бази даних. Фахівці стверджують, що подібні побоювання безпідставні, оскільки у компаній, що мають доступ і до відкритих приватних мереж, практично рівні шанси стати жертвами комп'ютерного терору. Кожна організація, що має справу з якими б то не було цінностями, рано чи пізно стикається з посяганням на них. Передбачливі починають планувати захист заздалегідь, непередбачливі після першого великого "проколу. Так або інакше, встає питання про те що, як і від кого захищати. У області інформації дилема безпеки формулюється таким чином: слід вибирати між захищеністю системи і її відкритістю.

**Інфіковані зовнішні носії**

За родом своєї діяльності мені доводилося стикатися з різними мережами і різними політиками безпеки в останніх. Як показує практика, зовнішні носії несуть загрозу зараження трохи меншу, в порівнянні з Інтернет небезпеками і як правило ПК або навмисно заражають, або по незнанню. Цей шлях Зараження дуже поширений на локальних робочих станціях, які працюють без мережі. Перевіряти файли які заносяться на ПК із зовнішніх носіїв.

**Інфіковані поштові повідомлення**

Минулі роки ознаменувалися поштовими епідеміями, і, не дивлячись на те, що адміністратори, та і кінцеві користувачі вже уміло, захищаються від подібних погроз, все одно дуже велика частина інцидентів зараження ПК відбувається через поштові повідомлення.

Захист на рівні сервера і на ПК користувача.

4. Сайти з інфікованими файлами (неліцензійне ПЗ, кряки, паролі, ключі, порно реферати, скрипти і тому подібне)

5. Безвідповідальність користувачів при роботі

6. Низький рівень обізнане про погрози.

7. Відсутність контролю над діями користувачів

Цим можна пояснити такі дії користувача як, умисну зупинку роботи антивірусного монітора, не регулярне проведення сканування системи не дотримання правил комп'ютерної гігієни. І як правила люди згадують про

АВПЗ тільки після того, як бачить наслідки роботи вірусу або якісь помітні пригальмовування системи або збої. Правильна політика безпеки дозволить не далечінь користувачеві можливості самостійно керувати АВПЗ на його робочій станції.

8. Повільна реакція провайдерів (адміністраторів ЛВМ) на нові погрози.

**Правила комп'ютерної гігієни**

***Правило перше***. Украй обережно відноситеся до програм і документів MS Office, які отримуєте з глобальних мереж. Перш ніж запустити файл на виконання або відкрити документ/таблицю, обов'язково перевірте їх на наявність вірусів. Використовуйте спеціалізовані антивіруси для автоматичної перевірки усіх файлів, що надходять електронною поштою.

***Правило друге***. Краще купувати дистрибутивні копії програмного, забезпечення у офіційних продавців, чим безкоштовно або майже безкоштовно, копіювати їх з інших джерел або купувати піратські копії. При цьому значно знижується вірогідність зараження, хоча відомі випадки купівлі інфікованих дистрибутивів. Як наслідок з цього правила витікає необхідність зберігання дистрибутивних копій ПО (у тому числі копій операційної системи), причому копії бажано зберігати на захищених від запису дискетах. Користуйтеся тільки джерелами програм і інших файлів, що добре зарекомендували себе, хоча це не завжди рятує (наприклад, на WWW-сервере Microsoft досить довгий час знаходився документ, заражений макровірусом Wazzu). Мабуть, єдиними надійними з точки зору захисту ют вірусів являються ftp/WWW антивірусних фірм-розробників.

***Правило третє***. Намагайтеся не запускати неперевірені файли, у тому числі отримані з комп'ютерної мережі. Бажано використовувати програми тільки з надійних джерел. Перед запуском нових програм обов'язково перевірте їх одним або декількома антивірусами. Якщо навіть жоден антивірус не зреагував на файл, який був переписаний з мережі Internet, – не квапитеся його запускати. Почекайте тиждень, якщо цей файл раптом виявиться, заражений новим невідомим вірусом те, швидше за все хто-небудь "настане на граблі" раніше вас і своєчасно повідомить про це. Бажано також, щоб при роботі з новим ПО в пам'яті резидентний знаходився який-небудь антивірусний монітор. Якщо програма, що запускається, заражена вірусом, то такий монітор допоможе виявити вірус і зупинити його поширення. Усе це призводить до необхідності обмеження кола осіб, допущених до роботи на конкретному комп'ютері. Як правило, найчастіше схильні до зараження розраховані на багато користувачів ПК.

***Правило четверте***. Користуйтеся утилітами перевірки цілісності інформації. Такі утиліти зберігають в спеціальних базах даних інформацію про системні області дисків (чи цілком системні області) і інформацію про файли (контрольні суми, розміри, атрибути, дати останньої модифікації

файлів і так далі). Періодично порівнюйте інформацію, що зберігається в подібній базі даних, з реальним вмістом вінчестера, оскільки практично будь-яка невідповідність може служити сигналом про появу вірусу або "троянської" програми.

***Правило п'яте***. Періодично зберігайте файли, з якими ведеться робота на зовнішньому носії, тобто робіть резервні копії (backup – копии). Витрати на копіювання файлів, що містять початкові тексти програм, бази даних, документацію, значно менше витрат на відновлення цих файлів при прояві вірусом агресивних властивостей або при збої комп'ютера.

За наявності стримера або якого-небудь іншого зовнішнього носія великого об'єму має сенс копіювати увесь вміст вінчестера. Але оскільки часу на створення подібної копії потрібно значно більше, чим на збереження тільки робочих файлів, має сенс робити такі копії рідше.

***Інші правила***. Якщо немає нужди щодня вантажити систему з дискети, поставте в BIOS Setup порядок завантаження "спочатку – C:, потім – A:". Це надійно захистить комп'ютер від завантажувальних вірусів. Не спокушайтеся вбудованою в BIOS захистом від вірусів, вона досить просто обходиться багатьма вірусами. Те ж вірне для систем антивірусного захисту, вбудованих в MS Office : вони можуть бути відключені вірусом (чи самим користувачем, оскільки ці системи можуть сильно заважати в роботі).

### 2.7.3 Загальна характеристика антивірусних програм для захисту робочої станції

Найбільш ефективні у боротьбі з комп'ютерними вірусами антивірусні програми. Проте відразу хотілося б відмітити, що не існує антивірусів, що гарантують стовідсотковий захист від вірусів, і заяви про існування таких систем можна розцінити як недобросовісну рекламу, або як непрофесіоналізм. Таких систем не існує, оскільки на будь-який алгоритм антивіруса завжди можна запропонувати контралгоритм вірусу, невидимого для цього антивіруса (зворотне, на щастя, теж вірно: на будь-який алгоритм вірусу завжди можна створити антивірус). Більше того, неможливість існування абсолютного антивіруса була доведена математично на основі теорії кінцевих автоматів, автор доказу – Фред Коэн. Слід також звернути увагу на декілька термінів, вживаних при обговоренні антивірусних програм.

**Помилкове спрацьовування (False positive)** – детектування вірусу в незараженому об'єкті (файлі, секторі або системній пам'яті). Зворотний термін – False negative, тобто недетектування вірусу в зараженому об'єкті.

**Сканування за запитом (on – demand)** – пошук вірусів по запиту користувача. У цьому режимі антивірусна програма неактивна до тих пір, поки не буде викликана користувачем з командного рядка, командного файлу

або програми-розкладу (system scheduler). Сканування на льоту (real – time, on – access) – постійна перевірка на віруси об'єктів, до яких відбувається звернення (запуск, відкриття, створення і тому подібне). У цьому режимі антивірус постійно активний, він є присутнім в пам'яті "резидентний" і перевіряє об'єкти без запиту користувача.

**Типи антивірусних програм**

Найпопулярнішими і ефективнішими антивірусними програмами є антивірусні сканери (інші назви: фаги, полифаги). Слідом за ними по ефективності і популярності слідують CRC-сканеры (також: ревізор, checksumer, integrity checker). Часто обидва приведені методи об'єднуються в одну універсальну антивірусну програму, що значно підвищує її потужність. Застосовуються також різного типу монітори, фільтри.

**Сканери**

Принцип роботи антивірусних сканерів заснований на перевірці файлів, секторів і системної пам'яті і пошуку в них відомих і нових (невідомих сканеру) вірусів. Для пошуку відомих вірусів використовуються так звані маски. Маскою вірусу є деяка постійна послідовність кода, специфічна для цього конкретного вірусу. Якщо вірус не містить постійної маски або довжина цієї маски недостатньо велика, то використовуються інші методи. Прикладом такого методу є алгоритмічна мова, що описує усі можливі варіанти коду, які можуть зустрітися при зараженні подібного типу вірусом. Такий підхід використовується деякими антивірусами для детектування полиморф-вирусов. У багатьох сканерах використовуються також алгоритми евристичного сканування, тобто аналіз послідовності команд в об'єкті, що перевіряється, набір деякої статистики і ухвалення рішення ("можливо, заражений" або "не заражений") для кожного об'єкту, що перевіряється. Оскільки евристичне сканування є багато в чому імовірнісним методом пошуку вірусів, то на нього поширюються багато законів теорії вірогідності. Наприклад, чим вище відсоток вірусів, що виявляються, тим більше кількість помилкових спрацьовувань. До достоїнств сканерів відноситься їх універсальність, до недоліків – розміри антивірусних баз, які сканерам доводиться "тягати за собою", і відносно невелика швидкість пошуку вірусів.

**Ревізори**

Принцип роботи ревізорів заснований на підрахунку CRC – сумм (контрольних сум) для присутніх на диску файлів/системних секторів. Ці CRC – суммы потім зберігаються у базі даних антивіруса, як, втім, і деяка інша інформація: довжини файлів, дати їх останньої модифікації і так далі. При наступному запуску ревізори звіряють дані, що містяться у базі даних,

з реально підрахованими значеннями. Якщо інформація про файл, записана у базі даних, не співпадає з реальними значеннями, то ревізор сигналізують про те, що файл був змінений або заражений вірусом. Ревізори, використовуючі "антистелс" – алгоритми, є досить сильною зброєю проти вірусів: практично 100% вірусів виявляються виявленими майже відразу після їх появи на комп'ютері. Проте у цього типу антивірусів є природжений недолік, який помітно знижує їх ефективність. Цей недолік полягає в тому, що вони не здатні упіймати вірус у момент його появи в системі, а роблять це лише через деякий час, вже після того, як вірус розійшовся по комп'ютеру. Ревізори не можуть детектувати вірус в нових файлах (у електронній пошті, на дискетах, у файлах, відновлюваних з backup або при розпаковуванні файлів з архіву), оскільки в їх базах даних відсутня інформація про ці файли. Більше того, періодично з'являються віруси, які використовують цю "слабкість" ревізорів, заражають тільки новостворювані файли і залишаються, таким чином, невидимими для ревізорів.

**Резидентні монітори**

Резидентні монітори – це резидентні програми, перехоплюючі вирусоопасные ситуації і що повідомляють про це користувачеві. До вирусоопасным відносяться виклики на відкриття для запису у виконувані файли, запис в завантажувальні сектори дисків або MBR вінчестера, спроби програм залишитися резидентний і так далі, тобто виклики, які характерні для вірусів в моменти їх розмноження, а так само відкриття заражених файлів до запуску/копіювання. До достоїнств моніторів відноситься їх здатність виявляти і блокувати вірус на самій ранній стадії його розмноження, що, до речі, буває дуже корисно у випадках, коли давно відомий вірус постійно "виповзає невідомо звідки". До недоліків відносяться існування шляхів обходу захисту монітора і велика кількість помилкових спрацьовувань, і уповільнення роботи системи, що і послужило причиною для практично повної відмови користувачів від подібного роду антивірусних програм. Необхідно також відмітити такий напрям антивірусних засобів, як антивірусні монітори, виконані у вигляді апаратних компонентів комп'ютера ("залоза"). Найбільш поширеною є вбудована в BIOS захист від запису в MBR вінчестера. Проте, як і у випадку з програмними моніторами, такий захист легко обійти прямим записом в порти контроллера диска, а запуск DOS -утилиты FDISK негайно викликає помилкове спрацьовування захисту.

Існує дещо більше універсальних апаратних моніторів, але до перелічених вище недоліків додаються також проблеми сумісності із стандартними конфігураціями комп'ютерів і складності при їх установці і налаштуванні. Усе це робить апаратні монітори украй непопулярними на тлі інших типів антивірусного захисту.

**Імунізатори**

Імунізатори діляться на два типи: імунізатори, що повідомляють про зараження, і імунізатори, блокуючі зараження яким-небудь типом вірусу. Перші зазвичай записуються в кінець файлів (за принципом файлового вірусу) і при запуску файлу кожного разу перевіряють його на зміну. Недолік у таких иммунизаторов всього один, але він легальний: абсолютна нездатність повідомити про зараження стелс-вирусом. Тому такі иммунизаторы, як і блокувальники, практично не використовуються нині. Другий тип імунізації захищає систему від поразки вірусом якогось певного виду. Файли на дисках модифікуються таким чином, що вірус приймає їх за вже заражені (приклад – сумнозвісний рядок "MsDos", що оберігає від викопного вірусу "Jerusalem"). Для захисту від резидентного вірусу в пам'ять комп'ютера заноситься програма, що імітує копію вірусу. При запуску вірус натикається на неї і вважає, що система вже заражена. Такий тип імунізації не може бути універсальним, оскільки не можна імунізувати файли від усіх відомих вірусів: одні віруси вважають вже зараженими файли, якщо час створення файлу містить мітку 62 секунди, а інші – 60 секунд. Проте, незважаючи на це, подібні иммунизаторы в якості півзаходу можуть цілком надійно захистити комп'ютер від нового невідомого вірусу аж до того моменту, коли він визначатиметься антивірусними сканерами.

**Планувальник завдань**

Його можна назвати додатковою утилітою для сканера. Варто вам одного разу його настроїти, і він робитиме за вас те, що люди робити забувають – запускати сканер на тестування дисків і тек. В указаний час Планувальник завдань запускає сканер і примушує його тестувати вказані вами об'єкти. Дуже зручно налаштовувати такі профілактичні тестування на обідній час, або час технічної перерви, коли тестування комп'ютера не створює перешкод для роботи.

Вибираючи антивірусну програму для свого ПК, зверніть увагу на основні критерії якості. Такими є: вірусна база, швидкість тестування і якість евристичного аналізатора. Іншими словами, наскільки велика кількість вірусів, які програма здатна знайти і знешкодити, як швидко вона це робить і наскільки добре вона уміє розпізнавати (і перемагати!) нові віруси. Безумовно, для такого святого справи, як боротьба з вірусами, потрібна ліцензійна програма. Це якраз той випадок, коли профілактика на порядок дешевша за лікування. Ліцензійні антивірусні програми імпортного виробництва стоять від $90. Значно дешевше обзавестися антивірусом вітчизняної розробки УНА – Українським Національним Антивірусом, тим більше що ця програма ні в чому не поступається популярним зарубіжним аналогам.

Основні критерії якості антівирусного (АВ) програмного забезпечення (АВПЗ):

– ***розмір вірусної бази***. Антивірусна база (АВБ) це набір сигнатур (унікальний код вірусу) і модулі, що лікують, згідно з якими відбувається детектування шкідливих файлів в системі і їх лікування. Кількість "записів" в АВ базі показує кількість типів вірусів, які однозначно визначаються АВПЗ і коректно лікуються або віддаляються. Таким чином, чим більше розмір АВ бази тим більше вірусів знаходить і лікує АВПЗ.

На 19.10.2005:

УНА-140777

Касперский – 145611 Але розмір АВБ це не головний критерій вибору, оскільки різні АВ компанії по різному їх визначають (модифікації і тому подібне).

– ***швидкість тестування***. Цей параметр показує, який час витрачає АВПЗ на перевірку файлів. При проведенні тестів на швидкість роботи вам необхідно враховувати не лише швидкість перевірки детектування і лікування/видалення інфікованих файлів, але і час роботи з чистими файлами, оскільки в 99% випадках АВПЗ перевіряє "чисті" файли.

– ***ресурсоемкость АВПЗ***. Дуже важливий параметр, оскільки може стати критичним при роботі на "слабких" робочих станціях. Чи призводити до неможливості роботи під час перевірок (простий робочого часу і як наслідок втрати організації).

– ***якість лікування/видалення***. Мало детектувати вірус в системі, його необхідно коректно вилікувати і видалити. Є в АВ практиці такі випадки, коли АВПЗ при лікуванні залишає якісь частини вірусного коду або тільки блокує його завантажувальну частину і тому подібне. Такі випадки підвищують можливість повторного зараження або використання таких "дір" зловмисником.

– ***якість евристичного аналізатора***. Евристичний аналізатор це набір алгоритмів, які дозволяють визначати нові віруси (що не входять в АВ базу) методами евристики (за поведінковими ознаками) Якісний евристичний аналізатор дозволить вам з деякою часткою ймовірності запобігати епідеміям, тому випадку, якщо цієї сигнатури не опиниться у вірусній базі. Вірус написаний для вас, вам не повезло і вірус дійшов до вас раніше чим в АВ лабораторію.

– ***ліцензійне ПЗ***. Використовуючи Ліцензійне АВПЗ ви придбаваєте ряд бонусів. А саме гарантії розробників на функціональність ПЗ, регулярні оновлення АВБ, тих. підтримку, розбір висланого вами файлу на наявність вірусу.

– ***частота виходу оновлень***. Частота виходу оновлень це швидкість реакції на нові епідемії. Чим швидше АВ компанія реагує на нові віруси тим більша вірогідність своєчасно захистити вашу ЛОМ і ПК зокрема від нових вірусних епідемій. УНА 1-2 рази в добу, в моменти епідемій частіше.

– ***технічна підтримка***. При виборі антивірусного ПЗ необхідно враховувати не лише факт наявності технічної підтримки, але і її розташування (можливість виїзду до клієнта, витрати на телефонні розмови), мови спілкування і тому подібне.

– ***помилкові спрацьовування***. Також варто обернути уваги на помилкові спрацьовування АВПЗ, у кожного з АВ вони є і цей факт приховувати не варто, але при більшій кількості подібних інцидентів може постраждати робота компанії. Наприклад, якщо АВ поштовий фільтр заблокує довгожданий лист для керівника, а насправді кореспонденція виявиться чистою. Тому варто звернути увагу і на цей чинник.

**Постулати побудови комплексної системи антівирусної защіти (КСАЗ) в ЛОМ**

– Повнота охоплення усієї локальної мережі організації;

САЗ повинна в обов'язковому порядку охоплювати усе сервера і робочі станції ЛОМ. Інакше не захищені елементи мережі нестимуть постійну загрозу зараження усій мережі.

– Багаторівнева системи захисту;

Коли ми говоримо про багаторівневу САЗ, ми повинні розуміти, що йдеться про захист не лише робочих станцій і серверів, але і про перевірку потоків (таких як пошта або веб трафік), що входять в мережу.

Так, наприклад, поштове повідомлення може перевірятися на поштовому сервері АВ фільтром і на робочій станції користувача АВ монітором – це двох рівнева перевірка пошти. Можна поставити два поштові фільтри – відповідно три рівні.

– Використання АВПЗ від різних розробників;

Використання АВПЗ від різних розробників пояснюється специфікою роботи різних АВ компаній і їх реакцією на епідемії, так, наприклад, реакція на нові віруси АВ компанії залежить від регіону і часового поясу, де розташована основна АВ лабораторія. І немає нічого дивного в тому, що вірус, який почав своє поширення в Штатах швидше додадуть місцеві АВ компанії, така ж ситуація з додаванням вірусів і в Україні. До того ж різні алгоритми пошуку, різні АВ бази.

При використанні двох різних АВПП на одній робочій станції потрібно враховувати, що два АВ монітора не можуть спільно працювати. А ось проведення по черзі сканування системи сканерами від різних виробників значно підвищує вірогідність знаходження вірусу в системі.

Фахівці нашої компанії рекомендують використовувати АВПЗ від різних розробників, при побудові багаторівневої системи зашиті.

Пам'ятайте, що ми придбаваємо не лише достоїнства кожної системи, але і недоліки.

– Наблюдаемость і контроль роботи КСА1;

При побудові САЗ, та і власне будь-який АС, дуже важливою є організація контролю роботи системи. У нашому випадку це, системи сигналізації адміністраторові про події в системі (інцидентах виявлення вірусу), можливість перегляду поточної інформації по роботі системи і тому подібне.

Загальна система ведення і перегляду журналу можливість формування звітів по роботі САЗ. Інформація такого характеру має бути повною для можливості відстежити вузькі місця мережі і подальшим усуненням останніх.

– Можливість локалізації епідемій;

Структура мережі має бути продумана так, щоб на випадок проникнення вірусу в мережу і зараження ним якогось елементу, останній можна було б локалізувати на карантин. Цю процедуру вам демонстрував у своїй доповіді мій колега Олександр.

– Єдиний центр управління КСАЗ.

У САЗ має бути реалізована система централізованого управління, тобто можливість конфігурації, моніторингу, установки клієнтських модулів, управління розкладами роботи, оновлення системи, управління груповими політиками і тому подібне з одного робочого місця адміністратора мережевої безпеки.

**Правила антивірусного захисту ЛОМ**

– захист робочих станцій;
– захист точок обміну графіку;
– захист поштового графіку;

На практиці використовуються антивірусні поштові фільтри, мета яких не пропустити поштові повідомлення що несуть шкідливий код до кінцевого користувача.

Рекомендується використовувати антивірусні поштові фільтри не менше двох різних виробників, така необхідність викликана тим, що, як правило, реакція на поштові епідемії залежить від ряду чинників, серед яких хочу відмітити територіальне розташування антивірусних компаній їх профіль і спеціалізацію.

Для підвищення стабільності роботи серверів і гнучкості реалізації схеми проходження внутрішнього периметра рекомендуємо виносити поштові фільтри на індивідуальні сервера з реалізацією можливості виключення одного або декількох фільтрів з поштового ланцюжка.

З досвіду роботи в області технічного захисту інформації на великих об'єктах (від 100 робочих станцій і більше) найчастіше, виникає проблема з локалізацією епідемії для успішної ліквідації наслідків. Нами апробована і впроваджена на декількох об'єктах, що добре зарекомендувала себе, технологія сегментації ЛВМ на невеликі сегменти, або по роду занять (відділи, підрозділи) або за іншими виставленими критеріями.

Не для кого не секрет, що нові шкідливі програми (Інтернет черв'яки, троянські програми) перш ніж потрапити у будь-яку антивірусну лабораторію вражають деяку кількість комп'ютерних систем, і збиток від первинного поширення залежить в першу чергу від того на скільки для них буде надана свобода в тій або іншій мережі, в другу чергу безумовно залежить від того наскільки швидко інформація про зараження потрапить до антивірусної компанії, продукти якої встановлені в цій мережі. Ви можете понизити первинний збиток, якщо ваша мережа потрапить під первинне поширення, розділивши мережу або за допомогою окремих комутуючих пристроїв, для незалежних сегментів мережі, з єдиним захищеним центром обміну інформації, або за допомогою технології віртуальних мереж VLAN. Безумовно, технологія VLAN є гнучкішою і масштабованою для застосування як в локальних лак і в розподілених мережах.

– Робота користувачів мережі з устоями користувача;
– Опечатання ПК;
– Установка пароля на вхід в BIOS;
– Завантаження ОС тільки з системного диска.

Для зменшення ризику заразити файл на сервері адміністраторам мереж слід активно використовувати стандартні можливості захисту мережі : обмеження прав користувачів, установку атрибутів "тільки на читання" або навіть "тільки на запуск" для усіх виконуваних файлів (на жаль, це не завжди виявляється можливим) і так далі. Використовуйте спеціалізовані антивіруси, перевіряючі на льоту файли, до яких йде звернення. Якщо це з якої-небудь причини неможливо, регулярно перевіряйте сервер звичайними антивірусними програмами. Значно зменшується ризик зараження комп'ютерної мережі при використанні бездисковых робочих станцій. Бажано так само перш ніж запустити нове ПЗ, спробувати його на тестовому комп'ютері, не підключеному до загальної мережі.

**Типи АВПЗ вживані для побудови КСАЗ**

– Антивірусні поштові фільтри

**Поштовий антивірусний фільтр** – це програма, призначена для перевірки поштових повідомлень що проходять через поштовий сервер на наявність в них вірусних вкладень і блокування заражених повідомлень.

Поштовий фільтр дозволяє попередити попадання вірусів, що поширюються через електронну пошту, на комп'ютер користувача (одержувача).

– Антивірусні фільтри web графіку

Антивірусний фільтр web трафіку перевіряє трафік, що входить і витікаючий, через протоколи FTP, HTTP, SMTP на рубежі шлюзу доступу в Інтернет. І блокує потоки з інфікованими даними.

– АВПЗ дли захисту точок обміну даними

АВПЗ такого типу призначене для захисту файлових серверів, серверів баз даних і тому подібне. Окрім стандартних функцій захисту, вбудована

функція перевірки ресурсу сервера при видаленому зверненні до останнього.

– АВПЗ для захисту робочих станцій у складі мережі

Комплексне рішення централізованого антивірусного захисту робочих станцій у складі комп'ютерних мереж з єдиним вузлом управління клієнтами.

Локалізація "вірусної інфекції" в комп'ютерних системах і запобігання епідемії.

**Локалізація "вірусної інфекції" в комп'ютерних системах**

– Виявлення типу інфекції

Звернеться за консультацією в антивірусний центр.

– За наявності ЛОМ, розстикування вузлів, для локалізації інфекції відключити мережеві з'єднання на робочій станції або сервері. І локально лікувати антивірусним ПЗ. Закрити усю інформацію, що зберігається на сервері. (PPS presentation)

– Запобігання епідемії (аудит систем і ПЗ).

**Методики протидії вірусам**

Народна мудрість свідчить: "якщо хочеш бути здоровий – загартовуйся". Говорячи іншими словами: лікуйся не новомодними ліками, а просто не доводь себе до хвороби взагалі, завчасно підвищуючи свій імунітет. На жаль, у світі комп'ютерів цю приказку практично ніколи не згадують, ну а керуються їй – взагалі лічені одиниці. Дещо перебільшуючи, можна сказати, що ми маємо справу з синдромом природженого імунодефіциту. Адже віруси – не сплять! І день від дня стають все підступніше, изощреннее і сильніше! Ріст пропускної спроможності каналів Інтернет дозволяє зловмисникам вражати воістину астрономічне число комп'ютерів за рекордно короткий час. Єдиний спосіб протистояти антивірусам – збільшити захищеність свого вузла. Про те, що саме слід для цього зробити, розказано нижче.

**Резервне копіювання**

Перш ніж говорити про резервне копіювання, слід зауважити, що цілісності ваших даних погрожують не стільки віруси, скільки збої устаткування, помилки програмного забезпечення, невірні дії операторів, нарешті, ті самі труби опалювання, які у будь-який момент може прорвати. На загальну думку, фахівців з інформаційної безпеки, віруси складають лише малу частину від усіх причин руйнування даних взагалі. Висока надійність сучасної обчислювальної техніки створює оманливу ілюзію її повної благонадійності і про необхідність своєчасного резервного копіювання переважна більшість користувачів просто забувають. І це-то в наше століття переможного ходу резервних накопичувачів усіх видів і місткостей! Гаразд,

стримери – это довго, Iomega Zip – занадто дорого, але що заважає придбати CR – RW? Мізерна вартість самого приводу, практично безкоштовні "болванки" і божевільна швидкість запису дозволяють резервувати вміст свого жорсткого диска хоч щодня, витрачаючи на це не більше десятка хвилин (взагалі-то, резервувати увесь диск цілком немає ніякої нужди, – досить записати лише змінені за день файли). Наявність резервної копії дозволяє швидко "піднятися" після будь-якого збою, до мінімуму зводячи збитки від втраченої інформації.

Причому, при перезаписі інформації неприпустимо затирати колишню копію, – адже немає ніяких гарантій, що свіжі версії файлів вже не заражені (і спотворені) вірусом, а раз так: резервна копія втратить свій сенс, адже вона міститиме ті ж самі дані, що є на жорсткому диску! Щонайменше, слід мати дві – три резервні копії, зроблених в різні моменти часу. Наприклад, на один диск записувати інформацію щодня, на іншій – раз в одну два тижні, ну а на третю – раз в місяць. Чим більше ви маєте копій, тим безболезненней виявиться "відкат" у разі вірусної атаки.

Так само слід врахувати можливість появи вірусів, що підтримують перезаписувані диски і що нищівні містяться на них дані (або ж, що складніше, але технічно все-таки здійсненно, файли, що заражають, безпосередньо на CD – RW). Отже, при відновленні інформації з резервної копії ніколи не вставляйте диск в привід, що пише, – краще скористайтеся звичайним CD – ROM'ом.

**Перехід на захищені операційні системи**

Масштаби раз у раз виникаючих вірусних епідемій не в останню чергу пояснюються тим. що в офісах переважної більшості компаній і корпорацій встановлена спочатку незахищена Windows 98, що є ласою мішенню для вірусів і хакерів. Так, звичайно, Windows 2000/XP коштує набагато дорожче, а її переваги (далеко не очевидні на перший погляд) зводяться якраз до захищеності. Міра захищеності ж – це і є те властивість, на якій багато хто вважає за краще економити. Дійсно, поки грім не гряне і жаренный півень не клюне, керівники фірм частенько взагалі не замислюються про безпеку. Зате потім, коли вірусна (хакерська) атака здійсниться, вони приводять абсолютно фантастичні цифри про вартість загублених даних. Помилуйте! Якщо хоч би 1% від заявленої суми був витрачений на грамотний захист ваших комп'ютерів, 99%, що залишилися, зараз були б з вами! Про необхідність резервного копіювання вже було сказано вище (власне, цю необхідність ніхто і не оспорює), а ось на рахунок доцільності оснащення робочих станцій Windows 2000/XP єдиної думки навіть серед фахівців з інформаційної безпеки немає, і досить багато хто з них вважає таку міру невиправданим марнотратством. Отже остаточне вирішення цього питання – залишається за вами.

**Зменшення привілеїв користувачів до мінімуму**

Операційні системи сімейства NT вигідно відрізняються від Windows 98/Me тим, що дозволяють присікати потенційно небезпечні дії користувачів, і серйозно обмежують повноваження запущених цими користувачами додатків. Зокрема, заборону на модифікацію виконуваних файлів робить поширення більшості вірусів просто неможливою! Правда, мережеві черв'яки, що взагалі не впроваджуються ні в які файли і задовольняються лише зараженням оперативної пам'яті, виживуть і в NT, але! – правильна політика розмежування доступу до файлів документів зведе наслідки деструктивних дій вірусу до мінімуму. На жаль, багато користувачів, що безладно відносяться до власної безпеки, постійно сидять в системі під "Адміністратором", а потім дивуються: чому ця хвалена NT їх не рятує?

І навіть ті користувачі, які входять в систему з правами "простих користувачів" і без нужди ніколи її не підвищують, практично ніколи не замислюються про доцільність заборони модифікації тих файлів, які в цей період часу ними не використовуються. Причому, слід зазначити уразливість служби Run As. що дозволяє запускати програми від імені інших користувачів. На перший погляд, ніякої загрози нібито немає, – усі програми, що запускаються таким чином, вимагають явного введення пароля і віруси "дістатися" до них неначе не можуть. На жаль, – мені припаде вас жорстоко розчарувати. По-перше, навіть початкуючому програмістові нічого не варто написати програму, що відстежує появу діалогового вікна "запуск програми від імені іншого користувача" і що викрадає що вводиться цим самим користувачем пароль. По-друге, якщо та або інша програма запущена, вірус зможе захопити над нею управління, просто емулюючи введення з миші і клавіатури! Незважаючи на те. що доки таких вірусів все ще немає, ніколи не дозволяйте собі запускати більше привілейовані програми з менш привілейованих. Краще, завершивши поточний сеанс роботи, увійдіть до системи під відповідним користувачем знову. І – внимание! – працюючи під адміністратором, ні в якому разі не запускайте ті програми, які можуть бути модифіковані з привілеями звичайного користувача!

**Скорочення надмірної функціональності програм**

"Монструозність" сучасного програмного забезпечення надзвичайно утрудняє його тестування і практично не залишає ніяких надій на виявлення усіх "дір" в розумні терміни. Причому, велика частина функціональних можливостей того ж Outlook'а – явно надмірна і реально вимагається лише невеликій частині користувачів.

Ось, наприклад, перегляд HTML-формата листів. Безперечно – це зручно. Але ось чи необхідно? Без HTML-писем в переважній більшості випадків можна і обійтися, а вже підтримка скриптів в листах і поготів не потрібна (хіба що спамерам -банеры крутити). Створення власного міні-

мально функціонального поштового клієнта – далеко не така складна завдання, який здається на початку. У мене, зокрема, на це пішли всього два вечори і, звертаючись до корпоративних користувачів, я хочу сказати: повірте, вартість розробки персонального поштового клієнта зневажливо мала в порівнянні із загальними витратами, що виділяються на забезпечення безпеки. Мінімально функціональний поштовий клієнт насправді настільки простий, що є цілком реальна можливість реалізувати його без "дір". А відсутність "дір" автоматично відсікає усі віруси, що неявно запускаються! Єдина серйозна проблема полягає в обробці HTML-писем. Реалізація власного браузеру – затея практично не реальна, а використання готових компонентів від Internet Explorer або Netscape Navigator з невідворотною неминучістю наслідує усі їх "діри" і ваш поштовий клієнт виявиться захищений анітрохи не краще того ж Outlook Express! Проте, хто сказав, що поштовий клієнт повинен містити повноцінний браузер? Підтримка десятка основних тегів для комфортного читання листів – цілком достатньо.

**Моніторинг зміни файлів**

Впровадившись в систему, вірус починає в ній розмножуватися, і ось тут-то його можна і "підсікти", – досить лише періодично контролювати цілісність існуючих файлів і відстежувати появу нових. Існує велика кількість – утиліт, що виконують таку операцію, проте в Windows 98 і Windows 2000/XP для вирішення поставленого завдання можна обійтися і штатними засобами, – досить запустити утиліту SFC.EXE. Не дивлячись на те, що хитрі віруси можуть цю програму і обдурити, міра достовірності результатів перевірки дуже висока, і на цьому ловляться переважна більшість вірусів. Варто відмітити, що операційні системи класу NT (до яких належать і Windows 2000/XP) забороняють звичайним користувачам модифікувати системні файли, так само як і вносити потенційно небезпечні зміни в конфігурацію системи, а тому вижити в такому середовищі вірусу виявиться дуже і дуже важко. З деякою натяжкою можна навіть сказати, що взагалі неможливе. Як вже відзначалося вище, не варто економити на безпеці, ставлячи на робочі місця не NT, а дешевшу Windows 98/Me.

**Контроль за зверненням до файлів**

Хорошим засобом виявлення вторгнень (як вірусних, так і хакерських) служить щедро розкидана приманка, – файли, до яких легальні користувачі через свої потреби не звертаються, але які з високою мірою вірогідності буде заражені вірусом або зажадалися хакером. У першому випадку в ролі наживки можуть виступити будь-які виконувані файли взагалі, а в другому – будь-які файли з інтригуючими іменами (наприклад, "номери кредитних карток менеджерів фірми.doc").

Як виявити сам факт звернення до файлів? Ну, по-перше, операційна система сама відстежує дату останнього звернення до документу (не

плутати з датою його створення), упізнати яку можна через пункт "Властивості" контекстного меню вибраного файлу. Досвідчені користувачі можуть отримати ту ж саму інформацію через популярну оболонку FAR Manager – просто натисніть <Ctrl – 5> для відображення повної інформації про диск. І, якщо дата останнього відкриття файлу раптом несподівано змінилася, – знайте, до вас прийшла біда і потрібно влаштовувати серйозні розбирання із залученням фахівців, якщо, звичайно, ви не можете впоратися із заразою власними силами. Звичайно, досвідчені хакери і ретельно продумані віруси таку примітивну міру можуть обійти, але практика показує, що в переважній більшості випадків вони про це забувають.

**Контроль за станом системи**

Не усі віруси, проте, впроваджують свій код в чужі виконувані файли. Спрягавшись в далекому куточку жорсткого диска (як правило, це каталог Windows\System, вміст якого ніхто з користувачів не знає і не перевіряє), вони змінюють конфігурацію системи так, щоб отримувати управління при кожному завантаженні (чи, на худий кінець – эпизодически).

Найпримітивніші віруси прописують себе в "Автозавантаження" (див. Пуск ⇨ Програми ⇨Автозавантаження), а переважна більшість інших – в наступну гілку системного реєстру:

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Widows\CurfentVersion\Run]

Звичайно, існують і більше екзотичні способи (на зразок файлів config.sys/autoexec.bat в Windows 98/Me і config.nt/autoexec.nt в Windows 2000/XP), але вони практично не знаходять собі застосування.

**Ненормальна мережева активність**

Часто віруси (і впроваджена хакерами "Троя") видають себе тим, що проявляють ненормальну мережеву активність (звернення по нетипових для цього користувача мережевих адресах і портах, різко збільшений трафік або підозрілий час). Почнемо з того, що практично жоден вірус не аналізує облікові записи поширених поштових клієнтів і, отже, самостійно визначити адресу вашого поштового сервера не може. Як же тоді черв'яки ухитряються поширюватися? Та дуже просто: частина з них використовує жорстко прошиті усередині себе адреси безкоштовних SMTP-серверов, інші ж зв'язуються з одержувачами листів безпосередньо, тобто звертаються безпосередньо до їх поштового сервера. І те, і інше – демаскує вірус, виключаючи, звичайно, той випадок, коли за іронією долі і вірус, і заражена ним жертва використовують один і той же сервер витікаючої пошти. Враховуючи, що переважна більшість корпоративних користувачів використовують свій власний сервер для розсилки пошти, вірогідністю випадкового збігу адрес можна нехтувати.

Для протоколювання мережевого трафіку існує величезна безліч різноманітних програм, найдоступніша з яких (хоча і не найкраща) – netstat, що входить в штатний пакет постачання операційних систем Windows 98/Me і Windows 2000/XP.

Так само має сенс придбати будь-який сканер портів (шукайте його на хакерських сайтах або, на худий кінець, скористайтеся все тій же netstat) і періодично оглядати усі порти свого комп'ютера, – чи не відкрилися серед них нові. Деякі мережеві черв'яки встановлюють на комп'ютери, що заражаються, компоненти для видаленого адміністрування, які найчастіше працюють через TCP і рідше, – через UDP протоколи. Утиліта netstat дозволяє контролювати на предмет відкритих портів і ті, і інші.

**Аналіз отриманих з мережі файлів**

Переважна більшість поштових черв'яків поширюють своє тіло через вкладення (також звані аттачментами) і, хоча існує принципова можливість створення вірусів, що цілком уміщаються в заголовку або основному телі листа, на практиці з такими доводиться стикатися украй рідко, та і живуть вони все більше в лабораторних умовах. Таким чином, завдання виявлення поштових черв'яків зводиться до аналізу кореспонденції, що містить підозрілі вкладення.

**Методики видалення**

Видалення вірусів з системи, ймовірно, найпростіша операція з усіх, розглянутих вище. Дуже добре зарекомендувала себе наступна методика: після установки операційної системи і усіх супутніх їй застосувань, просто скопіюйте вміст диска на резервний носій і потім, – коли буде зафіксований факт вторгнення в систему (вірусного або хакерського – не важливо), – просто відновите усі файли з резервної копії назад. Головна перевага цього способу – його швидкість і не вередливість до кваліфікації адміністратора (чи, в загальному випадку, особи, що здійснює видалення вірусів). Звичайно, слід пам'ятати об те, що :

а) досить багато програм, той же Outlook Express, наприклад, зберігають свої дані в одному з підкаталогів теки Windows і, якщо поштову базу не зарезервувати окремо, то після відновлення системи вона просто зникне, що не є добре;

б) деякі віруси впроваджуються не лише у файли, але так само і в boot – і MBR-сектора, і для видалення їх звідти одного лише перезапису файлів недостатньо;

в) вірус може перехопити системні виклики і відстежувати спроби заміщення виконуваних файлів, імітуючи свій перезапис, але не виконуючи його насправді.

Тому, надійніший наступний шлях: завантажившись зі свідомо стерильного CD-диска (чи на худий кінець дискети) ви видаляєте теки

Windows (WINNT) і Program Files, а потім начисто встановлюєте заново операційну систему разом з усіма застосуваннями. Звичайно, це повільно, але нічого більше кращого запропонувати, мабуть, просто неможливо.

Природно, у разі зараження відомими вірусами, можна вдатися і до допомоги антивірусів, проте, існує дуже висока вірогідність зіткнутися з некоректним видаленням вірусу з файлів, внаслідок чого система або повністю втрачає свою працездатність, або вірус залишається не долікованим і "виживає", і частенько після цього вже не детектується антивірусом. Вже краще просто видалити заражений файл, відновивши його з резервної копії! Звичайно, це не гарантує того, що в системі не залишилося компонентів, приховано впроваджених вірусом, проте, такі компоненти (якщо вони взагалі є) можуть бути виявлені по одній з описаних вище методик. Правда, це вимагає певного часу і не факт, що виявиться меншим, ніж вимагається для повного переустановлення операційної системи.

**Порівняння антивірусних програм**

Компанія "Троянер-інфо" протестувала 20 антивірусних і антитроянських програм. Фахівці німецького сайту Троянер-інфо провели чергове тестування антивірусного програмного забезпечення.

Дослідження проводилося 9 листопада на базі AMD Athlon 700, 256 Мб Ram, Windows 2000 Professional. У дослідженні брало участь сто Троянів, як поширених, так і достатньо рідкісних. Деякі з них були частково модифіковані, або архівуються EXE-архіваторами. В якості критерію перевірки виступала здібність до розпізнавання, але не видалення вірусів. Час сканування одержаний з усередненої швидкості движків, сканерів:

| Місце | Програма | % | Час сканування (сек) |
|---|---|---|---|
| 1 | Kaspersky Anti-Virus | 99 | 9 |
| 1 | AVK | 99 | 23 |
| 2 | McAfee | 97 | 10 |
| 3 | PC Door Guard | 88 | 18 |
| 3 | PC-Cillin | 88 | 28 |
| 4 | TDS | 87 | 15 |
| 5 | Dr. Web | 86 | 6 |
| 5 | AntiVir PE | 86 | |
| 6 | ANTS | 85 | 3 |
| 7 | Norman | 84 | 58 |
| 8 | NOD32 | 82 | 1 |
| 9 | FP-Win | 80 | 3 |
| 10 | Anti-Trojan | 79 | 2 |

На жаль, багато антивірусних програм дотепер пропускають навіть найпоширеніші трояни, не говорячи вже про новітні віруси.

При вирішенні багатьох задач із прикладної області діяльності людини майбутній спеціаліст стикається з проблемою захисту інформації, програмних продуктів, технічного забезпечення АРМ фахівців, комп'ютерних мереж, інформаційних систем, для спеціалістів істотного значення набуло вміння не тільки застосовувати сучасні комп'ютерні програми, інформаційні технології, але й уміти захищати інформацію, програмні пристрої, комп'ютерні мережі та їх складові від впливу комп'ютерних вірусів.

## 2.8 Захист від збоїв і несанкціонованого доступу в ОС

### 2.8.1 Базові технології безпеки ОС

**Основні засоби захисту, убудовані в ОС**

Засоби захисту, вбудовані в ОС, займають особливе місце в системі безпеки. Їхнім основним завданням є захист інформації, що визначає конфігурацію системи, і потім – користувальницьких даних. Такий підхід є природним, оскільки можливість змінювати конфігурацію робить механізми захисту безглуздими.

Проблема захисту інформації в комп'ютерних системах прямо пов'язана з вирішенням двох головних питань:

- ✧ забезпечення схоронності інформації,
- ✧ контроль доступу до інформації (забезпечення конфіденційності).

Ці питання тісно взаємозалежні й не можуть вирішуватися окремо. Схоронність інформації означає захист її від руйнування й збереження структури збережених даних. Система контролю доступу до інформації повинна забезпечувати надійну ідентифікацію користувачів і блокувати будь-які спроби несанкціонованого читання й запису даних. У той же час система контролю не повинна знижувати продуктивність роботи інформаційних систем і звужувати коло розв'язуваних завдань.

***Системні засоби аутентифікації користувачів***. Перше, що повинна перевірити операційна система в тому випадку, якщо вона має хоча б мінімальні засоби захисту, – це чи треба їй взаємодіяти із суб'єктом, що намагається одержати доступ до яких-небудь інформаційних ресурсів. Для цього існує список поіменованих користувачів, відповідно до якого може бути побудована система розмежування доступу.

Під **ідентифікацією** розуміють визначення тотожності користувача або користувальницького процесу, необхідне для керування доступом. Після ідентифікації звичайно виробляється аутентифікація. Під **аутентифікацією** користувача (суб'єкта) розуміють встановлення його дійсності. При вході в систему користувач повинен пред'явити ідентифікуючу інформацію, що визначає законність входу й права на доступ. Ця інформація перевіряється, визначаються повноваження користува-

ча (аутентифікація), і користувачеві дозволяється доступ до різних об'єктів системи (авторизація). Під **авторизацією** (санкціонуванням) мається на увазі надання дозволу доступу до ресурсу системи.

У даній ситуації істотної виявляється політика керування користувальницькими паролями, що визначає правила їхнього призначення, зберігання, зміни й інші пов'язані із цим питання. Чим більші можливості для проведення подібної політики надає адміністраторові операційна система, тим більше шансів на те, що парольна аутентифікація буде діючим інструментом захисту.

Паролі з часом стають відомими. Це змушує періодично проводити їхню заміну. Вважається, що в інформаційних системах з низькими вимогами до забезпечення безпеки пароль повинен мінятися кожні три місяці, а в міру збільшення значущості питань, пов'язаних з несанкціонованим доступом, зазначений термін скорочується до шести тижнів.

Не менш важливим є і мінімально припустимий час між двома послідовними змінами паролів, оскільки така зміна – типова дія у випадку одержання ким-небудь несанкціонованого доступу до системи або ресурсів користувача.

Однією з розповсюджених загроз безпеки інформаційної системи є термінал, залишений користувачем без догляду під час роботи. Як контрзахід можна автоматично блокувати доступ або переривати сеанс роботи в системі через деякий час після припинення активності користувача.

Існують утиліти, що дозволяють проводити закриття екрана автоматично, однак застосовувати їх не рекомендується, оскільки при цьому виникають умови для установки програми, емулючої закриття екрана і зчитування користувальницького пароля.

Особливу небезпеку представляє віддалений вхід у систему через телефонну мережу. Оскільки контролювати цю мережу неможливо, то необхідна установка додаткових паролів на послідовні порти.

***Розмежування доступу користувачів до ресурсів***. Керування доступом може бути досягнуте при використанні дискреційного або мандатного керування доступом.

*Дискреційне керування доступом* – найбільш загальний тип керування доступом. Основний принцип цього виду захисту полягає в тому, що індивідуальний користувач або програма, що працює від імені користувача, має можливість явно визначити типи доступу, які можуть здійснити інші користувачі (або програми, виконувані від їхнього імені) до інформації, що перебуває у веденні даного користувача. Дискреційне керування доступом відрізняється від мандатного захисту тим, що воно реалізує рішення з керування доступом, прийняті користувачем.

*Мандатне керування доступом* реалізується на основі результатів порівняння рівня допуску користувача і ступеня конфіденційності інформації.

Існують механізми керування доступом, що підтримують ступінь деталізації керування доступом на рівні наступних категорій:

- ✧ власник інформації;
- ✧ задана група користувачів;
- ✧ всі інші авторизовані користувачі.

Це дозволяє власникові файлу (або каталогу) мати права доступу, що відрізняються від прав усіх інших користувачів і визначати особливі права доступу для зазначеної групи людей або всіх інших користувачів.

У загальному випадку існують наступні права доступу:

- ✧ доступ для читання;
- ✧ доступ для запису;
- ✧ додаткові права доступу (тільки модифікація або тільки додавання);
- ✧ доступ для виконання всіх операцій.

Керування доступом користувача може здійснюватися на рівні каталогів або на рівні файлів. Керування доступом на рівні каталогу приводить до того, що права доступу для всіх файлів у каталозі стають однаковими. Наприклад, користувач, що має доступ для читання до каталогу, може читати (і, можливо, копіювати) будь-який файл у цьому каталозі. Права доступу до каталогу можуть також забезпечити явна заборона доступу, що запобігає будь-який доступ користувача до файлів у каталозі.

У деяких ОС можна управляти типами звертань до файлу крім контролю за тим, хто може мати доступ до файлу. Реалізації можуть надавати опцію керування доступом, що дозволяє власникові позначати файл як поділюваний або заблокований (монопольно використовуваний).

*Поділювані* файли дозволяють здійснювати паралельний доступ до файлу декільком користувачам одночасно. *Блокований* файл буде дозволяти доступ до себе тільки одному користувачеві в цей момент. Якщо файл доступний тільки для читання, призначення його поділюваним дозволяє групі користувачів паралельно читати його.

*Механізми привілеїв* дозволяють авторизованим користувачам ігнорувати обмеження на доступ або, інакше кажучи, легально обходити керування доступом, щоб виконати яку-небудь функцію, одержати доступ до файлу тощо. Механізм привілеїв повинен включати концепцію мінімальних привілеїв (принцип, відповідно до якого кожному суб'єктові в системі надається найбільш обмежена безліч привілеїв, необхідних для виконання завдання).

*Принцип мінімальних привілеїв* повинен застосовуватися, наприклад, при виконанні функції резервного копіювання. Користувач, який авторизований виконувати функцію резервного копіювання, повинен мати доступ для читання до всіх файлів, щоб копіювати їх на резервні носії інформації. Однак користувачеві не можна надавати доступ для читання до всіх файлів через механізм керування доступом.

Наявність декількох шляхів одержання підвищених привілеїв є потенційно уразливим місцем у захисті операційної системи. Особливо небезпечно, коли переустановка ідентифікатора користувача здійснюється не бінарним файлом, а програмою командного інтерпретатора, що пояснюється легкістю її модифікації.

Зазначена обставина змушує адміністратора системи контролювати штатні користувальницькі командні інтерпретатори. Оскільки більшість користувачів обходиться обмеженим набором додатків, у ряді випадків можна зафіксувати коло доступних програм, що особливо актуально при проведенні нормативної політики безпеки. Воля користувача обмежується межами його каталогу й можливістю використовувати програми тільки з дозволених каталогів.

Іноді користувач взагалі не має потреби в безпосередній взаємодії з операційною системою, працюючи постійно з яким-небудь додатком, наприклад клієнтом бази даних. У цьому випадку доцільно використовувати можливості розмежування доступу, які надаються СУБД.

***Засіб перевірки коректності конфігурації ОС***. Операційна система має велику кількість настроювань і конфігураційних файлів, що дозволяє адаптувати ОС для потреб конкретних користувачів інформаційної системи. Однак це створює небезпеку появи слабких місць, тому для перевірки цілісності і коректності поточної конфігурації в ОС повинна бути передбачена спеціальна утиліта.

При запуску утиліта спочатку проводить верифікацію прав доступу до системних файлів. Потім перевіряє системні файли й порівнює їх з описом у майстер-файлі, що містить установки, що відповідають вибраному рівню безпеки. У ході виконання завдання для системних файлів перевіряються власник і група, права доступу, розмір і контрольна сума, кількість посилань і час останньої модифікації.

Результати виконання програми записуються в текстовому виді в спеціальних файлах. Всі коректування, проведені програмою, протоколюються, і систему можна в будь-який момент повернути до колишнього стану, що страхує адміністратора від незворотних дій.

***Інструмент системного аудиту***. Питання інформаційної безпеки не можуть успішно вирішуватися, якщо немає засобів контролю за подіями, що відбуваються, оскільки тільки маючи хронологічний запис усіх здійснених користувачами дій, можна оперативно виявляти випадки порушення режиму інформаційної безпеки, визначати причини порушення, а також знаходити й усувати потенційно слабкі місця в системі безпеки. Крім того, наявність аудиту в системі відіграє роль стримуючого фактора: знаючи, що дії фіксуються, багато зловмисників не ризикують робити свідомо карних дій.

Програмні засоби, що здійснюють такий контроль, називаються засобами аудиту. Оскільки в інформаційній системі підприємства є кілька

функціональних рівнів, на кожному з них бажані засоби моніторингу подій. Сьогодні наявність механізмів аудиту є обов'язковою вимогою до великих програмних продуктів, що працюють на кожному з рівнів.

Аудит неможливий без ідентифікації й аутентифікація користувачів. З цією метою при вході в систему програмою аудиту користувачеві присвоюється унікальний ідентифікатор. Реєстраційні дії виконуються спеціалізованим аудит-демоном, що проводить запис подій у реєстраційний журнал відповідно до поточної конфігурації. Аудит-демон стартує в процесі завантаження системи.

Кожна подія належить якому-небудь класу аудиту. Такий розподіл спрощує аналіз великої кількості подій. Приналежність подій до класів і набір класів можуть бути сконфігуровані системним адміністратором.

Існує біля двадцяти класів подій, що відслідковуються. Кожний клас має два імені – повне і скорочене. Для будь-якого класу встановлюється один із трьох прапорів аудита: аудит у випадку успішного виконання дії, аудит невдалих спроб, безумовний аудит.

***Мережеві засоби захисту***. Захист інформації на мережевому рівні має певну специфіку. Якщо на системному рівні проникнути в систему можна було лише в результаті розкриття користувальницького пароля, то у випадку розподіленої конфігурації стає можливим перехоплення користувальницьких імен і пароля технічними засобами. Наприклад, стандартний мережний сервіс telnet пересилає користувальницьке ім'я й пароль у відкритому виді. Це змушує знову розглядати завдання аутентифікації користувачів, але вже в розподіленому випадку, включаючи й аутентифікацію машин-клієнтів. Високий ступінь захисту досягається шляхом заміни стандартних відкритих сервісів на сервіси, що шифрують параметри користувача/ машини-клієнта, щоб навіть перехоплення пакетів не дозволяло розкрити ці дані. Нарешті, немаловажне значення має аудит подій, що відбуваються в розподіленому інформаційному середовищі, оскільки в цих умовах зловмисник не настільки помітний і має досить часу й ресурсів для виконання своїх завдань.

Стандартні засоби захисту, що існують в ОС, не є настільки ж всеохопними, що й на системному рівні. Справа в тому, що якщо на системному рівні однорідність гарантована і будь-які зміни можуть вводитися досить ефективно, те в умовах локальної мережі застосовується, як правило, набір різнорідного устаткування, що функціонує під керуванням різних ОС, виробники яких апріорно не зацікавлені у відповідності засобів цих систем концепції безпеки.

### 2.8.2 Ядро безпеки ОС

**Ядро безпеки (ЯБ) ОС** – *набір програм, що управляють частинами системи, відповідальними за безпеку*. ЯБ реалізує політику забезпечення

безпеки системи. Ця політика складається з безлічі правил нагляду і охорони взаємодій між *суб'єктами* (процеси) і *об'єктами* (файли, пристрої, ресурси міжпроцесорної взаємодії).

Дія в ОС *підзвітна,* якщо її можна простежити для конкретного користувача. ЯБ підвищує підзвітність шляхом встановлення відповідності між всіма входами в систему і реальними користувачами.

*Повноваження ядра* асоціюються із процесами. Вони дозволяють процесу виконати певні дії, якщо процес має необхідний привілей.

*Повноваження підсистем* асоціюються з користувачами. Вони дозволяють користувачеві виконувати певну дію за допомогою команд, віднесених до підсистеми.

*Підсистема* – набір файлів, пристроїв і команд, які слугують певній меті. Повноваження ядра заносяться в "безліч повноважень", асоційованих з кожним процесом. Повноваження встановлюються за замовчуванням, користувач може їх і переустановити.

Коли користувач входить у захищену ОС, має місце обмежена ідентифікація й перевірка дійсності. Система за вхідним іменем перевіряє пароль у базі даних паролів. Якщо ім'я знайдене, система пізнає користувача шляхом зашифрованого пароля з вмістом відповідного поля в базі даних паролів.

ЯБ розширює стандартний механізм. Існують певні правила, що обмежують припустимі паролі, нові процедури для генерації й зміни паролів. Розташування й захист певних частин бази даних паролів змінені. Адміністратор має більший контроль над процесом входу. Цей аспект системи підтримує окремий користувач – адміністратор упізнавання.

Крім того, ЯБ надає повний "слід" дій – *журнал обліку*. Журнал містить записи про кожну спробу доступу суб'єкта до суб'єкта (успішні й невдалі), про кожну зміну суб'єкта, об'єкта, характеристик системи. Підсистема обліку управляється спеціальним адміністратором обліку. Адміністратор обліку управляє зібраною інформацією, що допомагає адміністраторові з'ясувати, що трапилося із системою, коли й хто в цьому брав участь.

Одна з важливих функцій ЯБ – локалізація потенційних проблем, пов'язаних з безпекою. Обмежувальний механізм складається з:

- ✧ парольних обмежень;
- ✧ обмежень на використання терміналів;
- ✧ вхідних обмежень.

Адміністратор упізнавання може дозволяти користувачам самостійно вводити паролі або використовувати зґенеровані паролі. Пароль може піддаватися перевірці на очевидність.

Визначаються наступні стани паролів:

- ✧ пароль коректний;

- ✧ пароль прострочений (користувач може ввійти в систему й змінити пароль, якщо в нього є на це повноваження);
- ✧ пароль закритий (користувач заблокований, необхідна допомога адміністратора).

Користувачі часто не підкоряються примусовій періодичній зміні паролів, відновлюючи попереднє значення. Щоб перешкоджати цьому, крім максимального, установлюється ще й мінімальний час дії паролів.

Підтримується можливість генерувати звіти про різні аспекти функціонування системи: паролі, термінали, входи.

Жодна ОС не є абсолютно безпечною. Можливі наступні шляхи вторгнення:

- ✧ хтось може довідатися пароль іншого користувача або одержати доступ до термінала, з якого в систему ввійшов інший користувач;
- ✧ користувач із повноваженнями зловживає своїми привілеями;
- ✧ добре обізнаний користувач одержав неконтрольований доступ безпосередньо до комп'ютера.

Небезпечно надавати обладнання для відкритого доступу користувачам. Будь-які засоби захисту системи будуть марні, якщо обладнання, носії збережених версій і дистрибутиви не захищені.

Підсистема обліку ОС реєструє події, що відбуваються в системі, важливі з погляду безпеки, у журналі обліку, що згодом можна аналізувати. Облік дозволяє аналізувати зібрану інформацію, виявляючи способи доступу до об'єктів і дії конкретних користувачів. Підсистема обліку з великим ступенем надійності гарантує, що спроби обійти механізм захисту й контролю повноважень будуть враховані.

Мережеві сервери – це ворота, через які зовнішній світ одержує доступ до інформації на комп'ютері. Тому ЯБ повинне:

- ✧ визначити, яку інформацію/дія запитує клієнт;
- ✧ вирішити, чи має клієнт право на інформацію, що запитує сервіс;
- ✧ передати необхідну інформацію/виконати дії.

Помилки в сервері й "чорні ходи" можуть наразити на небезпеку захист усього комп'ютера, відкриваючи систему будь-якому користувачеві в мережі, обізнаному про ваду. Навіть відносно безпечна програма може привести до руйнування всієї системи.

### 2.8.3 Механізм ідентифікації й аутентифікації в ОС Windows NT

Механізм ідентифікації й аутентифікаціїї користувача в ОС **Windows NT** реалізується спеціальним процесом Winlogon, що активізується на початковому етапі завантаження ОС і залишається активним протягом усього періоду її функціонування. Ядро операційної системи регулярно перевіряє стан даного процесу і у випадку його аварійного завершення відбувається аварійне завершення роботи всієї операційної системи.

При вході в систему користувач передає в системну функцію **LogonUser** своє ім'я, пароль і ім'я робочої станції або домену, в якому даний користувач зареєстрований. Якщо користувач успішно ідентифікований, функція **LogonUser** повертає покажчик на маркер доступу користувача, що надалі використовується при будь-якому його звертанні до захищених об'єктів системи.

У кожний момент часу **Winlogon** може перебувати в одному із станів, показаних на рис. 2.27.

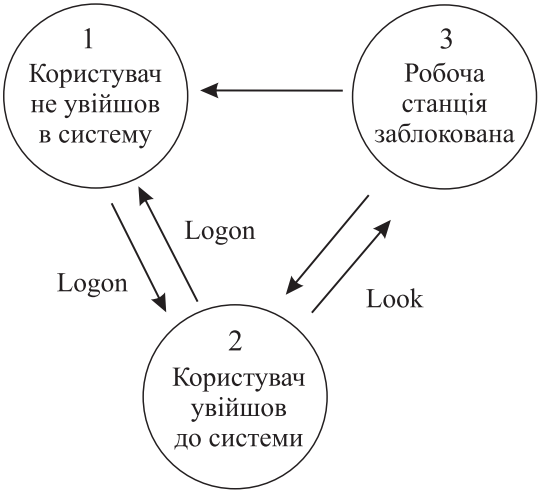



Рисунок 2.27. Тимчасова діаграма процесів аутентифікації

Коли користувач ще не ввійшов у систему, **Winlogon** перебуває в стані 1, користувачеві пропонується ідентифікувати себе й надати підтверджувальну інформацію (у стандартній конфігурації – пароль). Якщо інформація, уведена користувачем, дає йому право входу в систему, активізується оболонка системи (як правило, **Program Manager**) і **Winlogon** перемикається в стан 2.

Коли користувач увійшов у систему, **Winlogon** перебуває в стані 2. У цьому стані користувач може припинити роботу, вийшовши із системи, або заблокувати робочу станцію. У першому випадку завершує всі процеси, пов'язані з сеансом, що завершується, і перемикається в стан 1. У другому випадку **Winlogon** виводить на екран повідомлення про те, що робоча станція заблокована, і перемикається в стан 3.

У стані 3 Winlogon виводить на екран запрошення користувачеві ідентифікувати себе й розблокувати робочу станцію. Це може зробити або її користувач, що заблокував, або адміністратор. У першому випадку система

повертається в той стан, у якому перебувала безпосередньо перед блокуванням, і перемикається в стан 2. У другому випадку всі процеси, пов'язані з поточним сеансом, завершуються, і **Winlogon** перемикається в стан 1.

Коли робоча станція заблокована, фонові процеси, запущені користувачем до блокування, продовжують виконуватися.

### 2.8.4 Механізми захисту в ОС UNIX

***Ідентифікація й аутентифікація***. Розглянемо стандартну процедуру ідентифікації й аутентифікації користувача. Система шукає ім'я користувача у файлі **/etc/passwd** і, якщо користувач ідентифікується (тобто його ім'я знайдене), аутентифікація полягає в порівнянні образу аутентифіикації від введеного пароля з еталоном. При цьому передбачені деякі правила щодо характеристик пароля і можливості його зміни. Але, як показала практика, цих правил недостатньо для реалізації надійного захисту.

У надійній системі стандартна процедура ідентифікації й аутентифікації розширена. У ній передбачено більше правил, що стосуються типів використовуваних паролів. Введено процедури генерації й зміни паролів. Змінено місце розташування й механізм захисту деяких частин бази даних паролів. Адміністраторові аутентифікації надані додаткові можливості для контролю за діями користувачів.

Для посилення якісних і кількісних характеристик процедур ідентифікації й аутентифікації в UNIX існують наступні засоби.

a) задання адміністратором обліку користувачів і терміналів певних вимог на паролі:
   1) обмеження мінімальної довжини пароля, що вводиться користувачем;
   2) вимога наявності в паролі обов'язкової мінімальної кількості букв нижнього регістру, букв верхнього регістру, цифр і спеціальних символів;
   3) заборона користувачеві введення власних паролів; дозвіл вводити тільки паролі, створені системою.
b) задання адміністратором тимчасових обмежень на частоту змінюваності і часу життя паролів. При цьому для зручності користувача можливе задання інтервалу між початком вимоги зміни пароля і закінченням терміну його дії.
c) автоматичне блокування користувача на вході в систему за давністю пароля, за числом неуспішних спроб входу.
d) задання кількості й номерів терміналів для входу в систему для кожного користувача.
e) перевірка системою паролів користувачів при введенні на їхні семантичні і контекстуальні особливості (входження

ідентифікатора, ім'я користувача, повторюваність символів тощо).

f) зберігання зашифрованих паролів не в /etc/passwd, як у старих версіях, оскільки цей файл відкритий для читання всім користувачам, а в закритому від доступу окремому файлі.

g) одержання статистичної інформації з часу роботи користувача в системі, його блокуванню, номеру термінала тощо.

Крім цього існує можливість блокування за числом неуспішних спроб входу не тільки користувача, але і терміналу. При цьому можна задати інтервал часу, який повинен пройти між спробами реєстрації. Також передбачене ведення записів про успішні і неуспішні спроби входу в систему.

Добре себе зарекомендувало використання командного інтерпретатора rsh. Користувач, по-перше, не може перейти нікуди із свого домашнього довідника. По-друге, він може використовувати тільки команди з тих довідників, які визначені в змінній оточення PATH. При цьому змінити значення змінної оточення PATH користувач не може. По-третє, користувач не може задавати повні імена програмних файлів і перенаправляти потоки введення-виведення.

***Захист файлової системи***. Очевидно, що найбільша увага в питаннях захисту операційної системи повинна бути приділена захисту файлової системи.

Кожний файл у системі має унікальний індекс. Індекс – це керуючий блок. У літературі він також називається індексним дескриптором, i-node, або і-вузлом. Індекс містить інформацію, необхідну будь-якому процесу для того, щоб звернутися до файлу, наприклад, права власності на файл, права доступу до файлу, розмір файлу й розташування даних файлу у файловій системі. Процеси звертаються до файлів, використовуючи чіткий набір системних викликів і ідентифікуючи файл рядком символів, які виступають як складене ім'я файлу. Кожне складене ім'я однозначно визначає файл, завдяки чому ядро системи перетворить це ім'я в індекс файлу. Індекси існують на диску в статичній формі, і ядро зчитує їх у пам'ять перш, ніж почати з ними працювати.

Традиційно у файлових системах ОС UNIX за доступ до всіх типів файлів (файли, каталоги й спеціальні файли) відповідають 9 біт, які зберігаються в і-вузлі.

Перша група з 3 біт визначає права доступу до файлу для його власника, друга – для членів групи власника, третя – для всіх інших користувачів.

Наприклад, права доступу "**rwxr-xr-**" до файлу означають, що власник файлу має повний доступ, члени групи мають можливість читання й виконання, всі інші мають можливість тільки читати даний файл. Для каталогу установка біта виконання "**x**" означає можливість пошуку (добування) файлів із цього каталогу.

Така система захисту файлів існує досить давно й не викликає істотних нарікань. Дійсно, для того щоб вручну, тобто не використовуючи системні виклики й команди, змінити права доступу до файлу, необхідно мати доступ до області і-вузлів. Для того щоб мати доступ до області і-вузлів, необхідно змінити права доступу спеціального файлу (наприклад, /dev/root), біти доступу якого також зберігаються в області і-вузлів.

Іншими словами, якщо випадково або навмисне не зіпсувати права доступу до всіх файлів системи, установлені за замовчуванням (звичайно правильно) при інсталяції, то можна з великим ступенем імовірності гарантувати безпеку роботи системи.

Деякі UNIX-системи (наприклад, Solaris) надають додаткові можливості з керування правами доступу до файлів шляхом використання списків керування доступом (Access Control List). Даний механізм дозволяє для кожного користувача або для окремої групи встановити індивідуальні права доступу до заданого файлу. При цьому списки доступу зберігаються всіма системними копіюваннями способами й архівування. Не можна сказати, що введення цього механізму принципово поліпшує захист файлів, але він вносить деяку гнучкість у процедуру формування прав доступу до файлів.

***Контроль цілісності системи***. Цілісність системи повинна контролюватися ОС. В ОС **UNIX** контроль цілісності системи здійснюється рядом спеціальних команд.

Стандартна послідовність дій після виникнення збоїв у системі або яких-небудь інших відхиленнях наступна:

1) виконання перевірки файлової системи;
2) складання контрольного звіту;
3) перевірка бази даних аутентифікації;
4) перевірка дозволів для системних файлів.

***Засоби аудиту***. Будемо вважати, що дія контролюється, якщо можна знайти реального користувача, що її здійснив. Система контролю UNIX реєструє події в системі, пов'язані із захистом інформації, записуючи їх у контрольний журнал. У контрольних журналах можлива фіксація проникнення в систему й неправильне використання ресурсів. Контроль дозволяє переглядати зібрані дані для вивчення видів доступу до об'єктів і спостереження за діями окремих користувачів і їхніх процесів. Спроби порушення захисту й механізмів авторизації контролюються. Використання системи контролю дає високий ступінь гарантії виявлення спроб обійти механізми забезпечення безпеки. Оскільки події, пов'язані із захистом інформації, контролюються і враховуються аж до виявлення конкретного користувача, система контролю служить стримуючим засобом для користувачів, що намагаються некоректно використовувати систему.

Відповідно до вимог до надійних систем ОС повинна створювати, підтримувати й захищати журнал реєстраційної інформації, що відноситься

до доступу до об'єктів, контрольованих ОС. При цьому повинна бути можливість реєстрації наступних подій:

1) використання механізму ідентифікації й аутентифікації;
2) внесення об'єктів в адресний простір користувача (наприклад, відкриття файлу);
3) видалення об'єктів;
4) дії адміністраторів;
5) інші події, що зачіпають інформаційну безпеку.

Кожний реєстраційний запис повинна включати наступні поля:

1) дата й час події;
2) ідентифікатор користувача;
3) тип події;
4) результат дії.

Система контролю UNIX використовує системні виклики й утиліти для класифікації дій користувачів, підрозділяючи їх на події різного типу. Наприклад, при виникненні події типу "DAC Denials" (відмова доступу при реалізації механізму вибірного розмежування доступу) реєструються спроби такого використання об'єкта, які не допускаються дозволами для цього об'єкта. Іншими словами, якщо користувальницький процес намагається писати у файл із доступом "тільки для читання", те виникає подія типу "DAC Denials". Якщо переглянути контрольний журнал, то легко можна побачити повторювані спроби доступу до файлів, на які не отримані дозволи.

Істотно підвищує ефективність контролю наявність реєстраційного ідентифікатора користувача (LUID). Після проходження користувачем процедур ідентифікації й аутентифікації, тобто безпосереднього входу в систему, кожному процесу, створюваному користувачем, присвоюється реєстраційний ідентифікатор користувача. Даний ідентифікатор зберігається на весь сеанс роботи. Кожний контрольний запис, згенерований системою контролю, містить для кожного процесу реєстраційний ідентифікатор поряд з ефективним і реальним ідентифікаторами користувача й групи. Таким чином, виявляється можливим облік дій користувача.

Окремо варто розглянути реалізацію механізму контролю для ядра. Даний механізм генерує контрольні записи, виходячи з діяльності користувальницьких процесів, за допомогою системних викликів ядра. Кожний системний виклик ядра містить рядок у таблиці, де вказується його зв'язок з питаннями захисту інформації і якому типу події він відповідає. Крім того, використовується таблиця кодів помилок, що дозволяє класифікувати системні виклики на конкретні події, пов'язані із захистом інформації.

Наприклад, системний виклик **open** класифікується як подія "Зробити об'єкт доступним". Якщо користувач виконує системний виклик **open** для файлу **/unix** і даний системний виклик завершується успішно,

то генерується контрольний запис про цю подію. Однак якщо системний виклик **open** закінчується невдало внаслідок того, що користувач запросив у системному виклику доступ на запис файлу **/unix**, не маючи дозволу, та ця дія класифікується як подія "Відмова доступу" для даного користувача і об'єкту **/unix**. Отже, системний виклик можна відобразити в кількох типах подій залежно від об'єкта, до якого здійснюється доступ, і (або) результату виклику.

Однак варто мати на увазі, що при включенні всіх подій контролю і при активній роботі користувачів, обсяг записуваної інформації може досягати декількох мегабайт на одного користувача в годину.

## КОНТРОЛЬНІ ПИТАННЯ

1. Дайте визначення інформаційній безпеці.
2. Чим визначається інформаційна безпека?
3. Перерахуєте види погроз в інформаційній сфері.
4. Що відноситься до внутрішніх і зовнішніх джерел погроз інформаційної безпеки?
5. Дайте визначення організаційно-правовому забезпеченню інформаційної безпеки.
6. Які види збитку може нанести порушення інформаційної безпеки?
7. Які ви знаєте погрози інформаційним системам?
8. У чому полягає порушення конфіденційності? Наведіть приклад.
9. У чому полягає порушення достовірності? Наведіть приклад.
10. У чому полягає порушення цілісності? Наведіть приклад.
11. У чому полягає порушення доступності? Наведіть приклад.
12. Яким чином можна класифікувати погрози інформації?
13. Які чинники впливають на вимоги до захисту інформації в конкретній інформаційній системі?
14. Які способи захисту інформації вам відомі?
15. За допомогою яких засобів реалізується захист інформації?
16. Що таке "система захисту інформації"?
17. За якими ознаками можна класифікувати комп'ютерні мережі?
18. Чим відрізняється мережа з комутацією каналів від мережі з комутацією пакетів?
19. Перерахуєте відомі вам види дротяних з'єднань.
20. Чи Вірне вираження "глобальна комп'ютерна мережа Інтернет"?
21. Що таке "однорангова мережа"?
22. Що таке "топологія мережі"?
23. Чим зіркоподібна топологія відрізняється від кільцевої?
24. На якій основі вибирається топологія комп'ютерної мережі?
25. Що таке "вита пара"?
26. Порівняєте між собою кабелі типу "вита пара" і "тонкий коаксиал". У чому переваги і недоліки того і іншого рішення?
27. Яке призначення мережевих адаптерів?
28. Що таке "хаб" і для чого він використовується?
29. Що таке "мережевий протокол" і навіщо він потрібний?
30. У чому різниця між мережевою моделлю і стеком протоколів?
31. Перерахуєте відомі вам протоколи транспортного рівня стека TCP\IP.
32. За що відповідають протоколи прикладного рівня?
33. Перерахуєте рівні мережевої моделі OSI.
34. Які програми для роботи з сервісами Інтернету вам відомі?
35. Що включає сучасне трактування терміну "мультимедіа"?
36. Що таке "мультимедійний документ"?

## ТЕСТИ ДЛЯ САМОКОНТРОЛЮ

1. Комп'ютерна мережа – це

✧ система розподіленої обробки інформації, яка є комплексним об'єднанням комп'ютерів засобами передачі даних

✧ інформаційна мережа, до складу якої входить обчислювальне устаткування

✧ комунікаційна мережа, у якій продуктом генерування, переробки, зберігання й використання є інформація

2. Комп'ютерні мережі можуть бути класифікованими за порядком доступу до інформаційних ресурсів на

✧ відкриті
✧ закриті
✧ державні
✧ комутовані
✧ однорідні
✧ загальні

3. Комп'ютерні мережі можуть бути класифікованими за масштабом використання на

✧ корпоративні
✧ регіональні
✧ державні
✧ глобальні
✧ однорангові
✧ загальні

4. Топологія – це

✧ геометричне відображення з'єднання робочих станцій у мережі
✧ взаємозалежна сукупність багатьох обчислювальних мереж
✧ об'єднання транснаціональних комп'ютерних мереж
✧ інформаційна мережа, до складу якої входить обчислювальне встаткування

5. Установлення з'єднання між відправником й одержувачем з можливістю обміну повідомленнями без помітних тимчасових затримок характеризує режим роботи

✧ on-line ("на лінії")
✧ off-line ("поза лінією")

6. Протокол – це

✧ сукупність угод щодо способу подання даних, що забезпечує їхню передачу в потрібних напрямках і правильній інтерпретації даних всіма учасниками процесу інформаційного обміну

✧ частина інформації, передана незалежно від інших частин того самого повідомлення в обчислювальних мережах з комутацією пакетів
✧ зв'язок із установленням з'єднання

7. Фізичний рівень семирівневої моделі протоколів взаємодії локальної обчислювальної мережі (моделі OSI) представлений на:
✧ 1-му рівні
✧ 7-му рівні
✧ 5-му рівні
✧ 4-му рівні

8. Робоча станція – це
✧ персональний комп'ютер, що працює під управлінням власної операційної системи
✧ центральний персональний комп'ютер
✧ сервер для зв'язку з модемами

9. Пристрої, що встановлюють з'єднання на 4-м (транспортному) рівні моделі OSI, при цьому верхні рівні мережі (5,6 й 7) повинні бути однакові, називається:
✧ маршрутизаторами
✧ мостами
✧ концентраторами
✧ шлюзами

10. Архітектура мережі відображає
✧ топологію
✧ протоколи
✧ інтерфейси
✧ технічні засоби
✧ програмні засоби
✧ правильних відповідей не має

11. Система уніфікованих апаратних, конструктивних та програмних засобів, необхідних для організації надійної взаємодії елементів мережі – це:
✧ інтерфейс
✧ протоколи
✧ сервер

12. Протокол TCP – це
✧ протокол транспортного рівня
✧ міжмережевий протокол Інтернет

- ✧ одиниця вимірювання інформації
- ✧ словник термінів та понять
- ✧ правильної відповіді немає

13. Метод зв'язку імен і IP-номерів, називається
- ✧ сервер DNS
- ✧ сервер для зв'язку з модемами
- ✧ сервер печатки

14. Яке ім'я домена верхнього рівня означає Україна?
- ✧ ua
- ✧ au
- ✧ uk
- ✧ su
- ✧ by

15. Що не відносять до основних сервісів телекомунікаційних технологій?
- ✧ проектування СУБД
- ✧ електронну пошту
- ✧ передачу файлів
- ✧ телеконференції
- ✧ довідкові служби (дошки оголошень)
- ✧ відеоконференції
- ✧ доступ до інформаційних ресурсів (інформаційним базам) мережевих серверів

16. Організація документів, при якій один документ або текст може включати різноспрямовані посилання або покажчики (адреси) на інші документи і посилання називається
- ✧ гіпертекст (Hypertext)
- ✧ гіперпосилання (Hypertext links, hyperlinks)
- ✧ World Wide Web
- ✧ WAIS
- ✧ гіпертекстова система Gopher
- ✧ електронна пошта (electronic mail, email)

17. До програм-броузеров для Internet відносятся
- ✧ MS Internet Explorer
- ✧ Gopher
- ✧ Opera
- ✧ Outlook Express

18. Що теке безпека інформації?

✧ підтримка на заданому рівні тих параметрів інформації, що перебуває в автоматизованій системі, які забезпечують установлений статус її зберігання, обробки й використання

✧ гарантія того, що авторизовані користувачі завжди одержать доступ до даних

✧ ступінь (міра) захищеності інформації, збереженої й оброблюваної в автоматизованій системі, від негативного впливу на неї з погляду порушення її фізичної й логічної цілісності (знищення, перекручування) або несанкціонованого використання

19. Що таке ідентифікація?

✧ присвоєння суб'єктам і об'єктам доступу унікального ідентифікатора у вигляді номера, шифру, коду й т. інш. з метою одержання доступу до інформації

✧ гарантія того, що авторизовані користувачі завжди одержать доступ до даних

# РОЗДІЛ 3.
# ПРОГРАМНІ ЗАСОБИ РОБОТИ ЗІ СТРУКТУРОВАНИМИ ДОКУМЕНТАМИ. ОСНОВИ ОФІСНОГО ПРОГРАМУВАННЯ ТА WEB-ДИЗАЙНУ

## 3.1 Основи web-дизайну

**Остання п'ятниця липня**
**День системного адміністратора (System Administrator Day/Sysadmin Day). Відзначається щорічно. Запропонований сисадмином з Чикаго Тедом Кекатосом (Ted Kekatos). Уперше відзначався 28 липня 2000 р.**

### 3.1.1. Основні питання web-дизайну

3.1.1.1 Визначення web-дизайну

Більшість дискусій щодо поняття web-дизайну, як правило, зводяться до розгляду інших питань. Адже, хоча, кожен має деяке уявлення про те, що таке Web-дизайн, лише небагато здатні дати йому точне визначення. Певні компоненти, такі як графічний дизайн або програмування задіюються в кожній дискусії, проте їх важливість в процесі створення сайтів варіює від людини до людини і від сайту до сайту. Деякі вважають за найбільш важливою стороною web-дизайну є створення і структуризація вмісту, або, виражаючись формальніше, інформаційної архітектури. Інші чинники, серед безлічі інших: простота використання, цінність і призначення сайту в рамках загальної діяльності організації, достаака – чітко розміщуються у сфері web-дизайну. Отримавши вплив з бібліотечної справи, графічного дизайну, програмування, організації мереж, проектування інтерфейсів призначених для користувача, зручності використання і безлічі інших джерел, web-дизайн дійсно є мультидисциплінарной областю діяльності.

Основні аспекти web-дизайну охоплюють п'ять областей:

*Вміст.* Сюди входять форма і організація вмісту сайту. Можливий діапазон – від того, як написаний текст до того, як він організований, представлений і структурований за допомогою технології розмітки, такий як HTML.

*Зорові образи.* Відноситься до компонування екранного простору на сайті. Це компонування зазвичай створюється за допомогою HTML, CSS або навіть Flash і може включати графічні елементи,

що виконують функції прикраси або навігації. Візуальна сторона сайту – це найбільш очевидний аспект web-дизайну, але не єдина, і не найважливіша, сторона дисципліни.

*Технологія.* Хоча застосування різноманітних базових web-технологий ніби HTML або CSS потрапляє до даної категорії, під технологією в цьому контексті частіше маються на увазі різні інтерактивні елементи сайту, особливо створені з використанням програмних методів. Це можуть бути елементи в діапазоні від мов сценаріїв, що працюють на стороні клієнта, на зразок JavaScript, до серверних застосувань, таких як Java-сервлети.

*Доставка.* Швидкість і безвідмовність доставки сайту в мережі Internet або внутрішній корпоративній мережі пов'язані з вживаним апаратним та програмним забезпеченням і задіяною мережевою архітектурою.

*Призначення.* Причина, у зв'язку з якою сайт існує, часто пов'язана з економічними питаннями та, ймовірно, є найбільш важливою частью web-дизайну. Цей елемент слід враховувати при ухваленні будь-яких рішень, що зачіпають інші області.

**Піраміда web-дизайну**

Всі компоненти web-дизайну можна представити через метафору web-пірамиди, що представлена на рис. 3.1. Вміст – це цегла, складові піраміду, а фундамент міцно розташований на зорових образах і технологіях та в значній мірі грунтуються на економіці.




Рисунок 3.1. Web-піраміда: грані web-дизайну

Піраміда дозволяє дизайнерам з легкістю представити всі сторони web-дизайну в їх взаємодії, але не може запропонувати майже нічого, щоб забезпечити глибше розуміння мережевого середовища.

**Мережеве середовище**

Більш практичним варіантом роздуму про web-дизайне представляється обговорення різноманітних складових частин мережевого середовища, показаних на рис. 3.2.

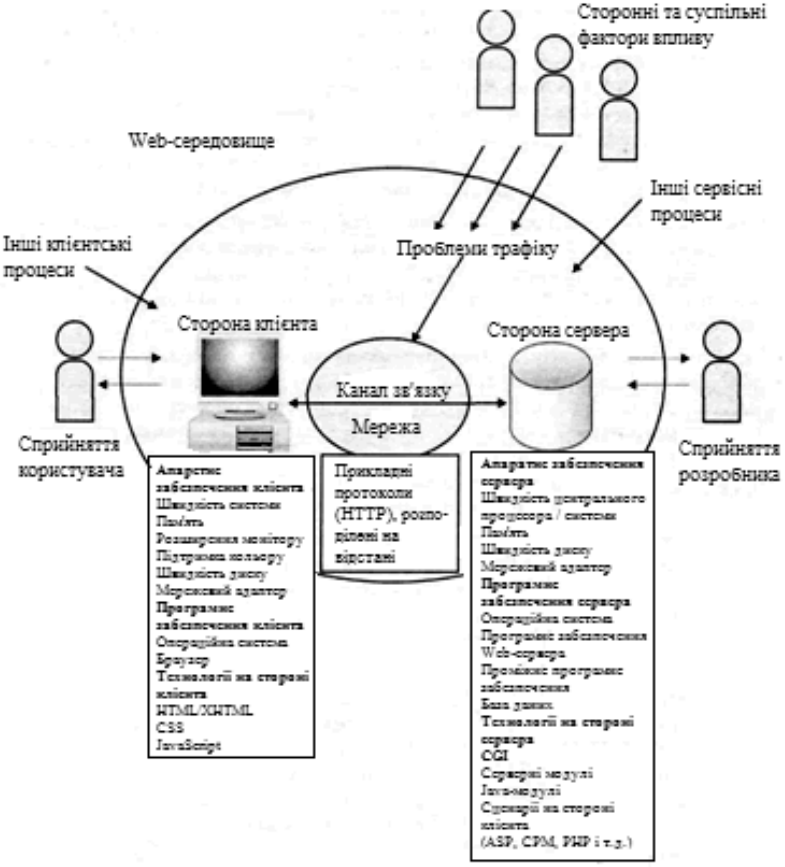



Рисунок 3.2. Компоненти мережевого середовища

На сьогодні, web-сайти, в основному, відносяться до базової моделі мережевого клієнт-серверного програмування з трьома загальними елементами.

*Сторона сервера.* Сюди входять апаратне і програмне забезпечення web-сервера, а також програмні елементи і вбудовані технології. Діапазон цих технологій тягнеться від простих програм CGI, написаних на PERL, до комплексних багатоланкових застосувань на основі Java; тут же враховуються, прикладні технології, наприклад – сервери баз даних, які можуть забезпечувати підтримку web-сайту.

*Сторона клієнта.* Сторона клієнта пов'язана з web-браузером і підтримуваними ним технологіями, такими як мови HTML, CSS. і JavaScript, елементи управління ACTIVEX і змінні модулі Netscape, які використовуються для створення представлення сторінки або забезпечення інтерактивних функцій.

*Мережа.* Мережа характеризує різні елементи зв'язності, призначені для доставки web-сайта користувачеві. Такими елементами можуть бути різні мережі в загальнодоступній частині мережі Internet або приватні з'єднання усередині корпорації, які часто називаються внутрішньою мережею.

Детальне розуміння технічних аспектів мережевого середовища, включаючи мережевий компонент, має величезне значення для кваліфікації web-дизайнера.

Схема web-пірамиди знов нагадує нам про важливість призначеного для користувача компоненту, оскільки насправді Web-дизайн є родом діяльності, пов'язаний з мережевим програмуванням і певними питаннями призначених для користувача спрямованості.

Web-сайти використовуються як механізм спілкування між власниками сайту і його користувачами, а іноді – між самими користувачами.

Канал зв'язку між власником сайту і його відвідувачем може змінюватися. Користувачі мають можливість публікувати інформацію на розгляд власників сайту або навіть інших користувачів, тим самим створюючи більш багатобічний канал зв'язку (рис. 3.3).

Як правило, в ході будь-якого процесу передачі інформації більшості користувачів, якщо все працює нормально, нічого не відомо про середовище. Хоча користувачі знаходяться під впливом середовища, вони часто не проводять відмінності між окремими компонентами, такими як мережа.

Так що ж, після всієї цієї дискусії, ми можемо сказати про те, що таке web-дизайн? Очевидно, це сильно орієнтована на користувача мультидисциплінарна дизайнерська область діяльності на яку впливає мистецтво, технології, вміст і бізнес.

**Web-дизайн** – це мультідисциплінарна область діяльності, що відноситься до планування і виробництва web-сайтів, яке включає (але не вичерпується) технічну розробку, структуризацію інформації, візуальний дизайн і доставку по мережі.

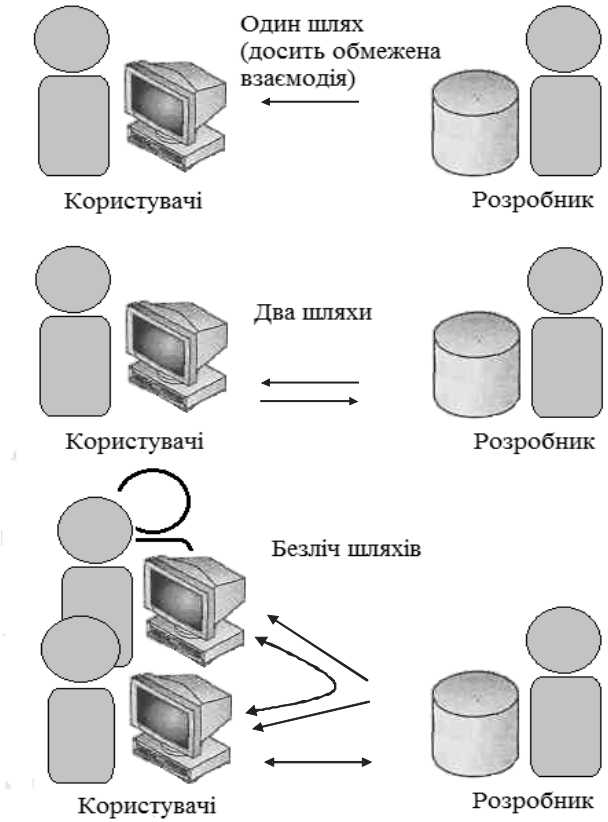



Рисунок 3.3. Багатобічний канал зв'язку

3.1.1.2 Введення в web-стандарти

Головне в web-розробці – це відповідність стандартам. Після багатьох років конкуренції між браузерами, "знущань" з HTML і використовуваних методів, породжених відсутністю загальних принципів, web-дизайнери нарешті усвідомили ті переваги постійності, які надають введення певних правил – використання зрозумілої і логічної побудови кодів HTML або XHTML для розмітки вмісту, каскадних таблиць стилів для всіх форм уявлення і єдиної об'єктної моделі документа для скриптів.

Розробники інструментів, які ми використовуємо для перегляду і створення web-сторінок, рухаються у бік повної підтримки стандартів, що надасться змогу здійснювати розробку більш ефективно, дешеве і надати сумісність від низу до верху.

**Організація з стандартизації**

Організація World Wide Web Consortium (W3C) здійснює формування і нагляд за розробкою web-технологій, включаючи XML, HTML, і їх численних сфер застосування. Також він не випускає з уваги проблеми вищого рівня, такі, як забезпечення доступу до матеріалів як можна більшого числа пристроїв і користувачів, а також закладають фундамент для майбутньої розробки, роблячи Web-матеріали сумісними від низу до верху.

W3C не є офіційною організацією з стандартизації, а швидше плодом спільних зусиль експертів в web-областях, робота яких направлена на впорядкування розробки web-технологій. Консорціум W3C публікує свої ідеї з приводу того, як повинна здійснюватися обробка (наприклад, HTML-розмітки) в документах, які називаються рекомендаціями (Recommendations). Велика частина таких рекомендацій стає фактично стандартами для web-розробки. Існують і інші організації зі стандартизації, які також впливають на Web і Internet в цілому, зокрема:

*ISO (International Organization for Standardization)* – це організація зі стандартизації, яка управляє більш ніж 10 000 стандартів у всіх областях, від інформаційних систем і наборів символів до розмірів 220-мм плівки і розміру гранул клею. Підтвердження відповідності стандартам, яке надає організація, допомагає забезпечувати сумісність комерційних і інформаційних технологій у всьому світу.

*IETF (Internet Engineering Task Force)* – це міжнародне співтовариство мережевих дизайнерів, операторів, провайдерів, дослідників, яких турбує розвиток Інтернету загалом. Ця організація публікує запити на коментарі (Request for Comments, RFCs), які визначають, як працює Інтернет, включаючи FTP, TCP/IP, HTTP і електронну пошту.

*Ecma International* – організація, раніше відома як ECMA (European Computer Manufacturers As sociation), є Європейською асоціацією зі стандартизації інформаційних і комунікаційних систем. Ecma International управляє стандартам в області інформаційних технологій, зокрема ECMAScript, стандартизованою версією JavaScript.

*Unicode Consortium* - організація управляє стандартом Unicode для багатомовних наборів символів.

*ANSI (American National Standards Institute)* – охоплює широкий діапазон дійсних стандартів, зокрема ASC (American Standard Code for Information Interchange).

Internet був побудований на основі стандартів. Оскільки Internet не знаходиться у власності і не управляється якоюсь однією компанією або людиною, рішення про те, яким чином слід вирішувати різні завдання, за традицією приймаються колективно, шляхом пропозиції, обговорення і впровадженням способу рішення кожної конкретної задачі.

Ще до появи web-стандартів, протоколи, системні імена та інші мережеві технології, знаходилися під управлінням IETF. Процес стандартизації починається, коли з'являється потреба в певній функції (наприклад, потрібні вкладення в електронну пошту) і одна людина або група пропонує систему для задоволення цієї потреби. Після фази обговорення пропозиція надається широкій публіці через RFC. Як тільки новинка розроблена і узгоджена, технологія стає стандартом. Звичайно, таке пояснення вельми спрощене. Для отримання інформації про процес узгодження стандартів або про те, які нові технології зараз знаходяться в розробці, звертайтеся на сайт IETF (www.ietf.org).

Web був предметом того ж процесу розробки, як будь-який інший інтернет-протокол. Проблема полягала в тому, що азарт і авантюризм, які породила рання web, привели до того, що розробка HTML і інших технологій велася з швидкістю, що перевершувала традиційні темпи затвердження стандартів. Отже, коли W3C почав в 1994 році працювати над стандартами HTML, виробники браузеров їх чекати не почали.

Щоб оволодіти ринком браузеров, компанія Netscape випустила браузер зі своїм власним набором HTML-тегів, що значно поліпшив зовнішній вигляд web-сторінок. Конкуруючи з Netscape, Microsoft відповіла своїм власним набором тегів і можливостей, і так почалася війни браузеров. Обидві компанії винні у використанні підходу, що закликає "дати людям те, що вони хочуть", не звертаючи увагу на те, як це вплине на середовище в цілому в довгостроковій перспективі. Проблема лише посилилася, коли web-дизайн переріс мову HTML і почав включати каскадні таблиці стилів, JavaScript і DHTML.

В результаті ми отримали в спадок масу тегів і технологій, які працюють тільки в тому або іншому браузері, а також безліч елементів, які ніяк не описують структуру документа (найбільш відомий – тег <font>). Це здвинуло початкове призначення HTML – описувати структуру змісту документа, але не його візуальне уявлення. Хоча зараз web-стандарти більш організовані, W3C все ще вимушений враховувати поправку на безліч використовуваних невірних кодів.

Трохи часу було потрібно, щоб співтовариство розробників почали дотримуватися рекомендацій, створюваних W3C. Чемпіоном в цьому став проект Web Standards Project (WASP www.webstandards.org) колектив web-розробників, заснований в 1998 році, під тиском яких розробники браузерів та інструментів, співтовариство дизайнерів примушені були дотримуватися загальних принципів.

**Переваги стандартів**

Ми все ще знаходимося в очікуванні того щасливого дня, коли всі браузери чітко слідуватимуть рекомендаціям W3C, але це не причина,

щоб відкладати на потім створення своїх власних відповідних стандартам матеріалів. Стандарти дають безліч переваг, якими ви можете починати користуватися вже зараз.

**Доступність**

Звичайно, web-матеріали будуть видимими за допомогою найрізноманітніших браузеров і пристроїв. Окрім графічних браузерів, з якими всі сьогодні знайомі, краще всього вони можуть відображатися альтернативними пристроями, такими, як мобільні телефони, кишенькові комп'ютери, або допоміжними пристроями – засобами читання з екрану для людей з порушеннями зору. Створюючи структуровані, з логічною розміткою документи, відповідно до вимог доступності, ви забезпечуєте їх краще сприйняття значним числом користувачів.

**Сумісність від низу до верху**

Наступні стандарти створюватимуться на основі сучасних. Тому матеріали, які чітко слідують стандартам сьогодні, матимуть довге життя, до того дня, коли застарілі елементи більше не підтримуватимуться.

**Простіша і швидша розробка**

Протягом багатьох років web-розробникам намагалися компенсувати відмінності браузеров, іноді, вдаючись до створення декількох різних версій одного сайту. Правильна розмітка структури документа і стратегічне використання таблиць стилів дозволяють створити одну версію змісту, яка обслуговуватиме всіх відвідувачів. А оскільки документ, керівник візуальним стилем, відокремлений від змісту, розробку дизайну і редакторську правку можна вести одночасно, що може прискорити процес виробництва. Така економія часу та відповідність стандартам набуває комерційний сенс.

**Швидше завантаження і відображення**

Документи, в яких для управління використовується нестандартний HTML (таблиці, теги font і прозорі зображення), як правило, збільшуються в розмірах. Видалення цих елементів і використання для управління представленням таблиць стилів дозволяє створювати файли набагато менших розмірів, які швидше завантажуються і можуть забезпечити додаткову економію трафіку. Крім того, сучасні браузери відображають сторінки в стандартному режимі – швидше, ніж в режимі зворотньої сумісності.

**Сучасні Web-стандарт**

Розглянемо сучасні стандарти, які відносяться до структурного, представницького і поведінкового аспектів дизайну.

Web-дизайн і розробку часто розглядають, використовуючи термін "рівні" (а іноді навіть називаючи його «листковим пирогом», що звучить

краще), запозичуючи багаторівневу модель, яку часто застосовують для опису мережевих протоколів:

- ✧ розмічений документ утворює собою *рівень структури,* який є фундаментом, на основу якого можуть наноситися інші рівні;
- ✧ далі йде *рівень уявлення,* визначуваний каскадними таблицями стилів (Cascading Style Sheets), в яких міститися інструкції про те, як документ повинен виглядати на екрані, як повинен звучати, коли він зачитується вголос, і як має форматуватися при друку;
- ✧ поверх цих рівнів розташовується *рівень поведінки* – скрипти і програми, які формують інтерактивні і динамічні ефекти на сайті.

Ви можете вивчити подальший розвиток web-технологій (так W3C називає свої готові опубліковані стандарти) на сайті W3C *(www.w3.org).*

**Рівень структури**

Початкове призначення HTML як мови розмітки, полягає в описі структури документа, а не інструкцій, що визначають його зовнішній вигляд. Структурна розмітка документа формує основу, на яку можна накладати рівень уявлення і рівень поведінки.

Нижче приведено сучасні стандартні мови для структурної розмітки:

*XHTML 1.0 (Extensible Hypertext Markup Language) і XHTML 1.1* – це просто HTML 4.01, переписаний для чіткішого дотримання синтаксичних правил XML. XHTML 1.1 втратив застарілі елементи і атрибути. Він має модульну структуру для забезпечення можливості майбутнього розширення. В даний час в процесі розробки знаходиться версія XHTML 2.0. Останньою версією HTML була HTML 4.01, яка як і раніше підтримується сучасними браузерами, але не є сумісною від низу до верху. За адресою *www.w3.org/MarkUp/*можна знайти посилання на специфікації XHTML 1.0, XHTML 1.1 і HTML 4.01.

*XML 1.0 (Extensible Markup Language* – є набором правив для створення нових мов розмітки. Він дозволяє розробникам створювати власні набори тегів для специфічних сфер застосування. За додатковою інформацією звертайтеся за адресою www.w3.org/XML/.

**Рівень уявлення**

Ці дані стали винятковою перевагою каскадних таблиць стилів (Cascading Style Sheets, CSS). Стандарти таблиць стилів розробляються поетапно:

*Cascading Style Sheets (CSS) Level 1* – даний стандарт таблиць стилів згадується в рекомендації з 1996 р. і зараз повністю підтримується сучасними версіями браузерів. Версія стандарту Level 1 містить правила управління відображенням тексту, полями і межами.

*CSS Level 2.1* – дана рекомендація більше всього відома введенням абсолютного позиціонування елементів web-сторінок. У 1998 році версія

Level 2 досягла стадії рекомендації, а доповнення 2.1 на даний момент має статус Candidate Recommendation. Підтримка CSS 2.1 в сучасних версіях браузерів ще неповна.

*CSS Level 3* – версія Level 3 створена на основі Level 2, але має модульну будову для спрощення розширення в майбутньому і для того, щоб пристрої могли підтримувати логічні піднабори. Ця версія продовжує знаходиться у стадії розробки.

Ви можете знайти посилання на все ці три специфікації CSS за наступною адресою: www.w3.org/Style/CSS.

**Рівень поведінки**

Скрипти і програми рівня поведінки вводять в сайт інтерактивні можливості і динамічні ефекти.

**Об'єктні моделі**

*Об'єктна модель документа* (Document Object Model, DOM) дозволяє скриптам і додаткам здійснювати доступ до документа і змінювати його зміст, структуру і стиль, звертаючись за допомогою формалізованої системи імен до його частин, атрибутів і способів маніпулювання. Спочатку кожен значний браузер мав свою власну DOM, що утруднювало створення інтерактивних ефектів, що працюють у всіх браузерах.

*Document Object Model (DOM) Level 1 (Ядро)* охоплює ядро документів HTML і XML, а також навігацію і маніпуляцію документами. Специфікацію DOM Level 1 можна знайти за адресою *w3c. org/TR/REC-DOM-Level-1/*

*DOM Level 2* – включає об'єктну модель таблиць стилів, яка робить можливим маніпулювання інформацією про стилі. Посилання на базовий і інші модулі специфікації DOM Level 2 можна знайти за адресою *www. w3.org/DOM/DOMTR.*

**Скрипти**

Компанія Netscape в своєму браузере Navigator 2.0 запропонувала мову web-скриптів JavaScript. Спочатку він називався "Livescript", але пізніше компанія Sun змінила бренд і в назву було додано слово "Java". Компанія Microsoft відповіла своєю власною мовою JScript, але до певного ступеня підтримувала у версії 3.0 свого браузера і JavaScript. Необхідність в стандарті, що охоплює різні браузери, була очевидною.

*JavaScript 1.5/ECMAScript 262 – к*онсорціум W3C розробив стандартизованную версію JavaScript в співпраці з Ecma International – міжнародною асоціацією, стандартизацією інформаційних і комунікаційних систем. Згідно інформації на сайті Mozilla, мова JavaScript компанії Netscape є частиною стандартної мови скриптів ECMAScript, що має лише невеликі відмінності від опублікованого стандарту. У звичайній практиці розробники говорять просто "JavaScript", маючи на увазі дану реалізацію стандарту.

Повну специфікацію можна знайти за наступною адресою: *www.ecma-international. org/publications/standards/Ecma-262.htm.*

**Інші технології, засновані на XML**

XML – це метамова, використовувана для створення інших мов розмітки і додатків. Цей могутній інструмент дозволив розробити декілька спеціалізованих стандартів. Розглянемо тільки декілька. Використовуючи модульну структуру XHTML і інших XML-специфікацій, можна змішувати і зіставляти XML-модули в одному документі, наприклад: XHTML, SVG і MATHML.

*SVG 1.1 (Scalable Vector Graphics)* – мова, заснована на XML і призначена для визначення двомірної і змішаної векторно-растрової графіки. За докладною інформацією звертайтеся до специфікації за адресою www.w3.org/Math/.

*MATHML 2.0 (Mathematical Markup Language)* – це XML-мова для визначення елементів математичної нотації, як в структурі, так і в змісті для передачі та обробки в web математичних символів. За додатковою інформацією з даної специфікації звертайтеся за адресоюwww.w3.org/Math/.

*SMIL 1.0 (Synchronized Media Integration Language) і SMIL 2.0* – XML-мова для створення мультимедійного уявлення, в якому поєднуються зображення, текст, аудіо і відео. За додатковою інформацією і специфікаціями звертайтеся за адресою www.w3.org/AudioVideo/.

3.1.1.3 Характеристика сайтів

Важливо уміти характеризувати сайти за їх призначенням чи за зовнішнім виглядом.

Сайти можна згрупувати за декількома загальними принципами. Розглянемо відвернуті класифікації та зорові.

**Відвернуті класифікації**

*Документо-ориентовані,* або *інформаційні* сайти надають користувачам інформацію, але при цьому забезпечують дуже обмежений рівень інтерактивності (окрім надання користувача можливостями перегляду, пошуку або сортування представленої інформації).

*Прикладні*, або *проблемно-орієнтовані* сайти дозволяють користувачеві взаємодіяти з інформацією або виконувати яке-небудь завдання, наприклад, переказ грошових коштів з банківського рахунку або придбання товарів.

*Гібридні* сайти об'єднує функцій двох попередніх; у міру того як розділова межа між інформацією і прикладним завданням розмивається, цей тип сайтів стає все більш поширеним.

На рис. 3.4 зображена схема переходу від простого, статичного документо-ориентированного сайту (такі сайти часто називають "буклетним забезпеченням") до повнофункціональних програмних застосувань.

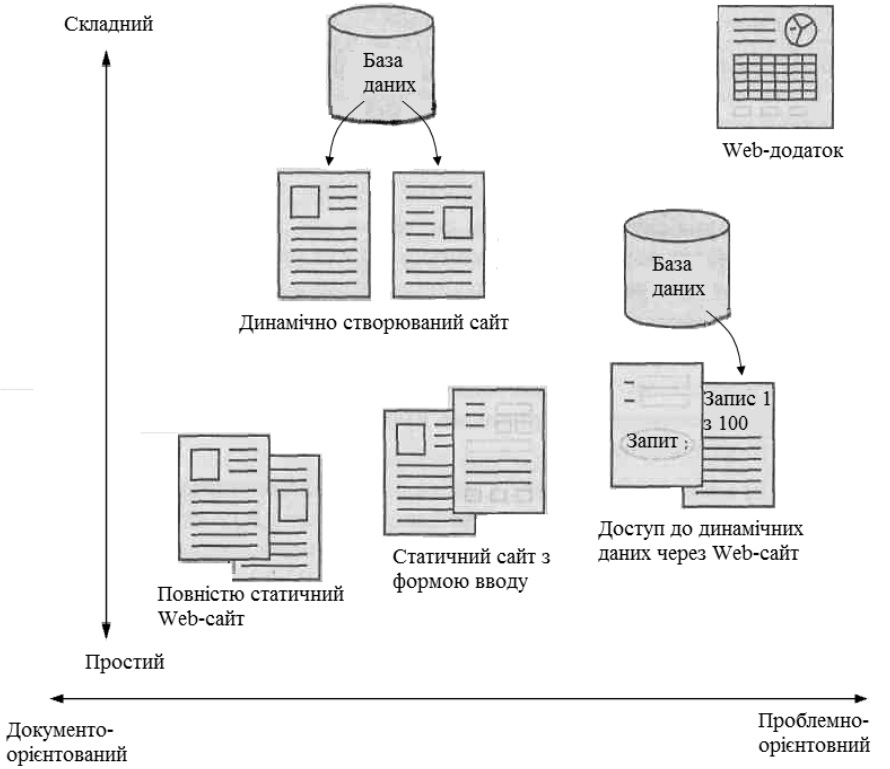



Рисунок 3.4. Схема переходу від простого, статичного документо-орієнтованого сайту до повнофункціональних програмних застосувань

Така відвернута класифікація припускає, що існує перехід від більш документо- або друкарсько-орієнтованих web-сайтів до більш інтерактивно-програмних.

Інший спосіб, яким ми можемо скористатися для класифікації сайтів – розсортувати їх за більш широкими категоріями.

*Інформаційні сайти.* На таких сайтах представлена інформація за конкретною тематикою або про певну організацію ("буклетне забезпечення"). Це найпоширеніші в мережі Internet web-сайти; з часом вони іноді переймають деякі риси інших категорій сайтів.

*Операційні сайти.* Сайтом такого типу можна скористатися з метою виконання якої-небудь операції або завдання. У дану категорію входять сайти, зайняті в електронній комерції.

*Сайти співтовариств.* На цих сайтах представлена інформація або засоби, пов'язані із здійсненням операцій, але наголос робиться на взаємодію

між відвідувачами. Сайти, засновані на співтовариствах, мають тенденцію до фокусування на конкретній темі або людині; вони заохочують взаємодію між схожими особами.

*Розважальні сайти.* Ці сайти створюються для ігор або якоїсь цікавої взаємодії, для якої можуть об'єднуватися елементи операційного, інформаційного типів і сайтів співтовариств.

*Інші сайти.* У цю категорію входять художні або експериментальні сайти, особисті web-простори, такі як web-журнали (також звані blogs), а також сайти, які можуть не слідувати загальноприйнятим web-погодженням або не мати чітко економічного призначення.

Окрім цього, можна згрупувати сайти на основі організацій, які підтримують або оплачують ведення сайту. В рамках цього типу класифікації убачаємо п'ять основних груп.

*Комерційні сайти.* Сайт з цієї групи створюється і підтримується організацією або індивідуумом для отримання комерційної вигоди –безпосередньо через електронну комерцію, або побічно – стимулювання придбання товарів або послуг зовні Internet.

*Урядові сайти.* Найвищим органом у відношенні до такого сайту є урядова організація, а призначенням сайту є задоволення якої-небудь суспільної або правової потреби.

*Освітні сайти.* Сайтом такого типу займається якась освітня установа (можливо, що має відношення до урядових органів); він використовується для забезпечення освітніх або дослідницьких завдань.

*Філантропічні сайти.* Філантропічний сайт існує з метою просування цілей некомерційної організації або добродійної діяльності приватної особи чи організації.

*Персональні сайти.* Такий сайт існує виключно з розсуду якоїсь людини або групи людей з будь-яких причин, зазвичай будучи плодом виплеску творчої енергії або формою самовираження особи.

Класифікація може виявитися складним завданням. Наприклад, освітні сайти насправді можуть потрапляти в категорію урядових. Деякі сайти з категорії персональних можуть, ймовірно, належати до групи філантропічних або комерційних – залежно від причини, з якої людина береться за створення сайту. Тепер ми перейдемо до зорових характеристик сайтів і розглянемо декілька прикладів категорій сайтів, що нерідко зустрічаються в Internet.

**Візуальні класифікації**

Групуючи сайти за візуальною ознакою, ми стикаємося з діапазоном, на одній стороні якого – сайти, які в значнішому ступені грунтуються на текстовому вмісті, а на іншій – ті, що більше фокусуються на графічному

уявленні або зображеннях. У Internet є чотири найбільш поширених дизайнерських школи.

*Тексто-орієнтировані сайти.* Такі сайти проектуються з акцентуванням на текстовий вміст. Вони, відносно "легкі", без зусиль завантажуються, а їх дизайн досить мінімалістичний.

*Сайти в стилі GUI.* На цих сайтах дотримуються деякі угоди, що пов'язано з графічним призначеним для користувачів інтерфейсом (GUI – graphical user interface), узяті з проектування програмного забезпечення, наприклад: вирівняні догори рядки меню, піктограми і спливаючі вікна. Серед GUI-орієнтованих сайтів є і прості GUI-компоненти, що додаються здебільшого в тексто-орієнтовані сайти, і повнофункціональні web-додатки із спеціальними елементами інтерфейсу. На рис. 3.5 показано декілька прикладів GUI-стилізованих web-сайтів.

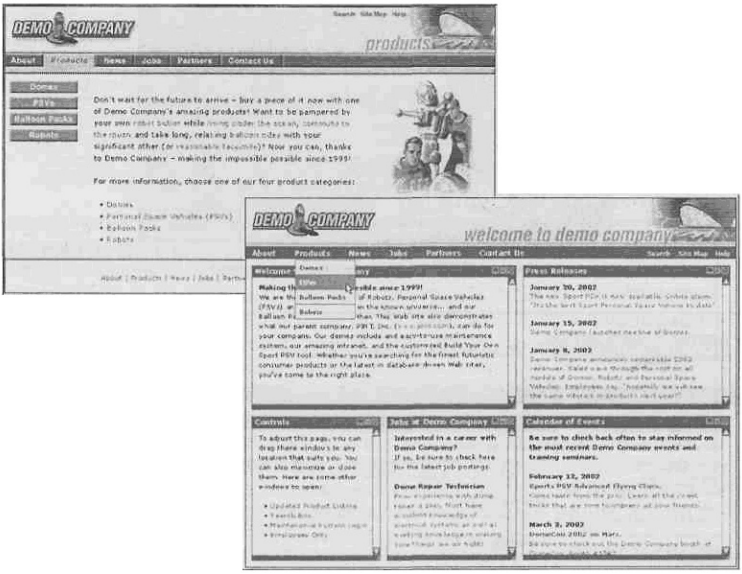



Рисунок 3.5. Приклад Web-сайту з невеликою кількістю GUI

*Метафоричні сайти.* Метафоричні сайти черпають ідеї з "реального життя". Наприклад, в дизайні і навігації сайту, присвяченого автомобілям, можуть бути задіяна панель приладів та кермо. Метафорично оформлений сайт, представлений на рис. 3.6, як правило, надзвичайно яскравий візуально або інтерактивний. Це може вводити в оману одних користувачів і привертати інших.

*Експериментальні сайти.* Творці таких сайтів намагаються робити речі, трохи відступаючи від норм. Як показано на рис. 3.7, на сайтах, екс-

периментального стилю оформлення, нерідко задіяно творчість, непередбачуваність, новаторство і навіть хаотичність.

Безумовно, в Internet ми знаходимо змішення форм, а отже і потенційно нову класифікацію сайтів.

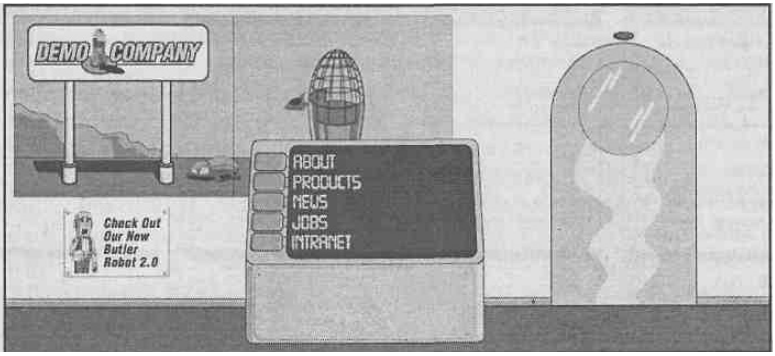



Рисунок 3.6. Метаморфічний дизайн




Рисунок 3.7. Експерементальний дизайн

Спостерігається приховане підвищення впливу інших категорій дизайну – для цього досить поглянути на такі жанри web-дизайна, як сайти електронної комерції, сайти типу “каталог і корзина”, а також сайти персональних мережевих журналів. Так, на рис. 3.8. представлено приклад сайту-порталу.

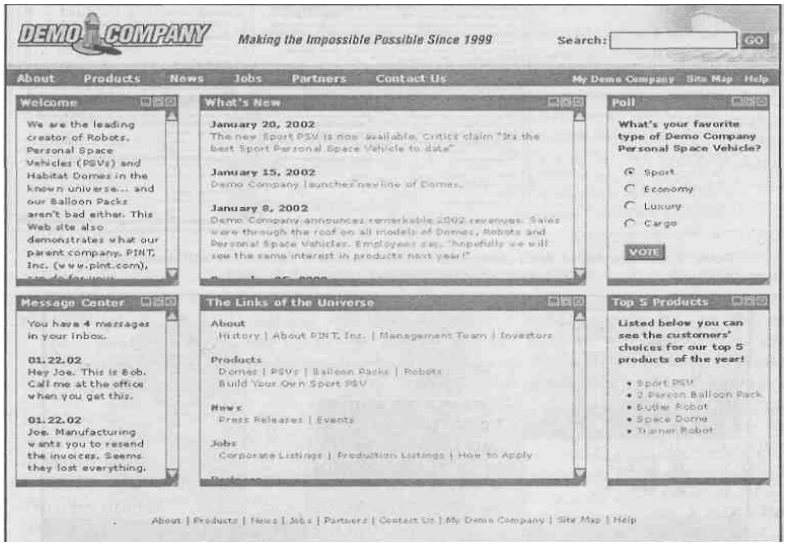



Рисунок 3.8. Дизайн у стилі порталу

### 3.1.2 Типи й архітектури сайтів

Як правило, до числа основних проблем сучасного web-дизайну відносяться наступні:

- ✧ потреби дизайнера і сподівання користувачів;
- ✧ рівновага форми і функції;
- ✧ якість виконання;
- ✧ співвідношення традиції і нововведення.

Почнемо з найважливішої проблеми: чи варто робити дизайн сайту орієнтованим на користувача, або на дизайнера?

**Дизайн, орієнтований на користувача**

Поширена проблема web-дизайну полягає в орієнтуванні web-сайту на користувачів.

**Правило**. Ви не користувач.

Ви повинні прийняти той факт, що безліч користувачів не обов'язково будуть детально знайомі з сайтом, який ви так ретельно вибудовували.

**Правило.** Користувачі не дизайнери.

Далеко не кожен користувач фактично є або має бути web-дизайнером.

З урахуванням всього сказаного можна з упевненістю стверджувати, що ключем до створення успішного і практичного web-сайту буде постійне прагнення дизайнера поставити себе на місце майбутнього його користу-

вача і розглянути проблему розробки сайту з погляду цього користувача. ***Дизайн, сконцентрований на користувачі*** (User-centered design), – це термін, використовуваний для опису методу web-дизайну, який припускає облік насамперед інтересів користувача.

Загальні положення науки про мислення (когнитології), які допомагають нам краще зрозуміти основні можливості користувачів.

**Правило**. Виконуйте розробку для середньостатистичного користувача, але враховуйте відмінності.

Більшість сайтів дозволяють користувачеві робити вибір і дають йому можливість контролю над тим, що з ним відбувається, але це все відбувається під впливом вимог дизайнера.

**Баланс форми і функції**

Ключовою проблемою Web-дизайну є проблема дотримання балансу форми і функції сайту. Під впливом модернізму багато дизайнерів почали дотримуватися уявлення про те, що форма в що б то не було повинна відповідати його функції. Вважатимемо, що форма – це одна підстава нашої метафоричної піраміди web-дизайну, тоді як функція – це інше її підстава. Між формою і функцією необхідно підтримувати постійної і чіткої взаємодії. Простіше кажучи, форма сайту повинна безпосередньо витікати з його призначення.

**Правило**. Забезпечте зв'язок зовнішнього вигляду сайту і його призначення.

**Виконання**

Web-сайт слід вважати за відмінний, тільки якщо він корисний, зручний у використанні, безпомилковий і привабливий. Щоб сайт міг вважатися за добре розроблений, його виконання має бути відмінним. Це означає:

- ✧ сайт не дає ніяких збоїв де б то не було;
- ✧ HTML-код має бути абсолютно точно вивірений;
- ✧ зображення правильно збережені;
- ✧ сторінка відображалася браузером саме так, як спочатку замислював дизайнер;
- ✧ будь-які інтерактивні елементи (будь то сценарії клієнтської частини, реалізовані за допомогою JavaScript, або CGI-сценарии, що виконуються сервером) зобов'язані функціонувати відповідним чином і не приводити до повідомлень про помилки;
- ✧ система навігації сайту повинна чітко працювати у будь-який момент часу.

Поява порушених зв'язків (broken link), що супроводиться повідомленнями про помилку 404: "Not Found", ніяк не можна віднести до ознак якісно виконаного сайту.

**Правило**. Виконання сайту має бути бездоганним або практично бездоганним.

Більшість фахівців з Internet часто не мали підготовки в областях обчислювальної техніки, організації мереж, теорії гіпертексту, когнитології і у всіх інших дисциплінах, які можуть впливати на якість створюваного сайту. Майстри web-дизайну повинні знати і поважати простір, в який входить все від браузери і пропускна спроможності до програмування і протоколів.

**Ортодоксальність і новаторство**

Дизайнерам потрібно поважати угоди, що стосуються навігаційних альтернатив, розміщення елементів навігації, кольорів і тому подібне Ці ідеї не обмежують дизайн; вони просто укладають сайти у впізнанну форму, щоб користувачі не виявляли, що сайти, які вони відвідують, повністю відрізняються один від одного.

**Правило**. Не існує ніякого "правильного" web-дизайну для відповідного для будь-якого сайту.

Прешим кроком для розробки сайтів є оволодіння основними принципи web-дизайну, а також процедури розробки і оцінки, що допоможуть зконструювати web-сайт.

Як зазначалося раніше, досить складно точно визначити, що мається на увазі під терміном "Web-дизайн". В кращому разі ми бачимо, що **Web-дизайн** – це мультидисциплинарная область діяльності, яка складається з п'яти основних компонентів: вмісту, зорових образів, технології, доставки і призначення. Угода про тонку рівновагу між формою і функцією, користувачем і дизайнером, вмістом і завданням, традицією і інновацією – ось висока мета web-дизайнера.

### 3.1.2.1 Типи сайтів

Як і програмне забезпечення, в якому можна виділити безліч різновидів призначених для користувача застосувань, від ігор до офісних програмних продуктів, Web-сайти підрозділяються на декілька типів. В основному сайти можуть бути класифіковані і віднесені до категорій, таким як Intranet – і extranet-сайти, або таким специфічним видам, як комерційні і домашні сторіночки. Кожен тип сайту матиме свій дизайн, визначуваний його призначенням. Відповідна організація сайту допомагає досягти поставленої мети. Існує безліч структур сайтів – від простих лінійних організацій до складних змішаних ієрархій. Наука про мислення (когнитология) і принципи побудови графічного інтерфейсу користувача, що склалися, можуть надати деякі ключі до визначення того, які структури працюють добре. Проте, структура добре розробленого сайту не завжди очевидна для користувача, і не зобов'язана бути такою.

Існує безліч способів класифікації Web-сайтів. Серед можливих – угрупувань по аудиторії, рівню інтерактивності, частоті змін, розміру,

типу вживаної технології, візуальному стилю, і, поза сумнівом, призначенню сайтів.

Наступні три загальні категорії Web-сайтов прийняті всюди:

- ✧ загальнодоступні Web-сайти (public);
- ✧ екстрамережеві (extranet);
- ✧ внутрішньомережеві (intranet).

**Публічним** Web-сайтом, Web-сайтом Internet, зовнішнім Web-сайтом або просто Web-сайтом являється той сайт, відвідувачі якого не обмежені явно певним класом користувачів**.**

Зовнішнім Web-сайтом являється у відомому сенсі публічне місце, доступне для вільного відвідування будь-якому з Internet. Не кожен користувач у світі захоче зайти на сайт – він необов'язково має бути розроблений для такого широкого круга користувачів, але для потенційних відвідувачів такого сайту в принципі немає ніяких обмежень.

Протилежністю може вважатися **внутрішньомережевий** Web-сайт, зазвичай званий просто як **intranet**. Внутрішньомережевий сайт зазвичай є дуже закритим і доступний, як правило, тільки для користувачів певної приватної мережі. **Внутрішньомережевий Web-сайт** – це сайт, який належить певній організації, і у більшості випадків функціонує в межах приватної мережі, а не в Internet.

У проміжку між цими крайнощами – зовнішнім Web-сайтом і внутрішньою мережею – знаходитиметься напівприватний сайт. **Extranet** – це найпоширеніший зразок **напівприватного** сайту. Прикладом extranet може служити сайт, обслуговуючий партнерські компанії або торгових посередників. **Extranet** – це Web-сайт, доступний обмеженому класу користувачів, але цей доступ здійснюється через Internet.

Основна різниця між трьома основними формами сайтів полягає в аудиторії. Публічні Web-сайти абсолютно відкриті, тоді як intranet і extranet є більше привілейованими.

Залежно від типу Web-сайта початкові дані для його розробки можуть значно розрізнятися, як показано в таблиці 3.1.

Таблиця 3.1

**Порівняння сайтів різних типів**

| | **Intranet-сайти** | **Extranet-сайти** | **Публічні сайти** |
|---|---|---|---|
| **Інформація про користувачів** | Повна | Неповна | Досить незначна |
| **Можливість планування** | Можливо | Зазвичай можливо | Від важко до неможливо |
| **Пропускна спроможність** | Висока | Варіюється | Значно варіюється |
| **Можливість задати технологію** | Так | Іноді | Рідко |

Ця класифікація сайтів є найбільш загальним розподілом за аудиторією.

**Класифікація за інтерактивністю**

Ще один спосіб класифікації сайтів – збудувати їх за мірою інтерактивності. Багато сайтів не відрізняються сильною інтерактивністю, а складаються головним чином із статичного вмісту, який користувач може переглядати і по якому він може проводити пошук. Такі сайти частенько іменуються статичними, користувач не має можливості безпосередньо вносити в них зміни.

Сайт є **статичним**, якщо його вміст порівняно незмінно і користувачі не мають можливості впливати на відображення або діапазон даних, що переглядаються. Говорячи коротко, відвідувач сайту має мінімальні можливості взаємодії з вмістом сайту, як правило тільки вибір порядку його перегляду.

Перегляд статичного сайту нагадує читання газети.

З іншого боку, багато сайтів, що особливо мають відношення до співтовариств, дозволяють користувачам до певної міри вносити або змінювати вміст – такі сайти можна назвати інтерактивними. Рівень інтерактивності, який може мати в розпорядженні користувач по відношенню до вмісту сайту, може варіювати від простої можливості коментувати створення вмісту з наступним редагуванням власником сайту або іншими користувачами, або ж без такого.

Сайт є **інтерактивним**, якщо його користувачі мають можливість безпосередньої взаємодії з вмістом сайту або іншими його користувачами.

**Класифікація за частотою змін**

Інший вимір класифікації сайту – по частоті зміни вмісту. Сайти, які ніколи не змінюються, можна назвати статичними; ті ж, що піддаються змінам, можна розглядати як **динамічні**. Термін *динамічний* в цьому контексті вживається стосовно вмісту сторінки, а не до її генерації, яка є окремою проблемою, їх ми коротко розберемо. Більшість сайтів не є абсолютно статичними; мало-помалу, з часом, сторінки зазвичай піддаються змінам. Чим частіше сайт змінюється, тим більше динамічним його можна рахувати. Вміст може модифікуватися на регулярній основі, наприклад, щодня, кожного тижня, і так далі, або ж зміни можуть робитися менш запланованим способом.

Крім того, сайти можуть змінюватися безперервно. Приміром, сайт, що **персоналізується**, змінюється з кожним новим відвідувачем, часто – у вигляді реакції або на поточні, або на минулі дії користувача. Характерний приклад – сайт електронної комерції, на якому можуть висуватися "спеціальні пропозиції", ради відвідувачам на основі їх попередньої

купівельної поведінки або навіть поведінки інших людей, що відвідали цю сторінку. Інші приклади сторінок, що персоналізуються – портали (my.yahoo.com), які передбачають так звані сторінки "мого власного" стилю, що настроюються користувачами так, щоб вони відповідали їх конкретним інтересам.

Сайт є таким, що **персоналізується**, якщо його зміст безпосередньо пристосований для певного користувача, і користувач, як правило, явно може визначити зміст, образ або технології сторінки.

На динамічних сайтах дуже важливо показати користувачам, коли їх вміст оновлювався.

**Класифікація за часом створення сторінок**

При обліку часу в дизайні сайтів важливо чітко визначати, коли сторінки насправді проектуються для відвідувачів. У багатьох випадках сторінки статичні з тієї причини, що вони створюються заздалегідь, до появи користувача, а змінюються дуже трохи. У інших випадках сторінки можуть споруджуватися в запланований час, коли їх вміст створюється або змінюється. Нарешті, сторінки можуть генеруватися якраз тоді, коли їх просить користувач; такі сторінки часто іменуються динамічно генерованими.

**Динамічно генерована сторінка** створюється для користувача за запитом або під час перегляду.

Динамічно генеровані сторінки володіють безліччю переваг.

По-перше, вміст можна настроїти так, щоб воно відповідало очікуванням користувача. Поширеним різновидом динамічних сторінок є сторінки результатів пошуку. Крім того, можливе створення динамічних сторінок з урахуванням умов перегляду або технологічних обмежень.

Недолік полягає в тому, що динамічно генеровані сайти набагато складніше створювати, і вони задіюють значні потужності сервера, т. до. кожна сторінка повинна створюватися наново при кожному візиті користувача.

Ще однією перевагою є те, що динамічно генеровані сторінки частенько простіше обслуговувати. Приміром, на динамічних сайтах "Зовнішній вигляд сторінки" можна містити в загальних шаблонах, нижні колонтитули додавати відразу на усі сторінки, навігацію зберігати в загальних файлах і т. д.

Для зберігання вмісту динамічно генерованих сайтів часто використовується база даних.

Порівняльна характеристика статичних і динамічно генерованих сайтів представлено на рис. 3.9.

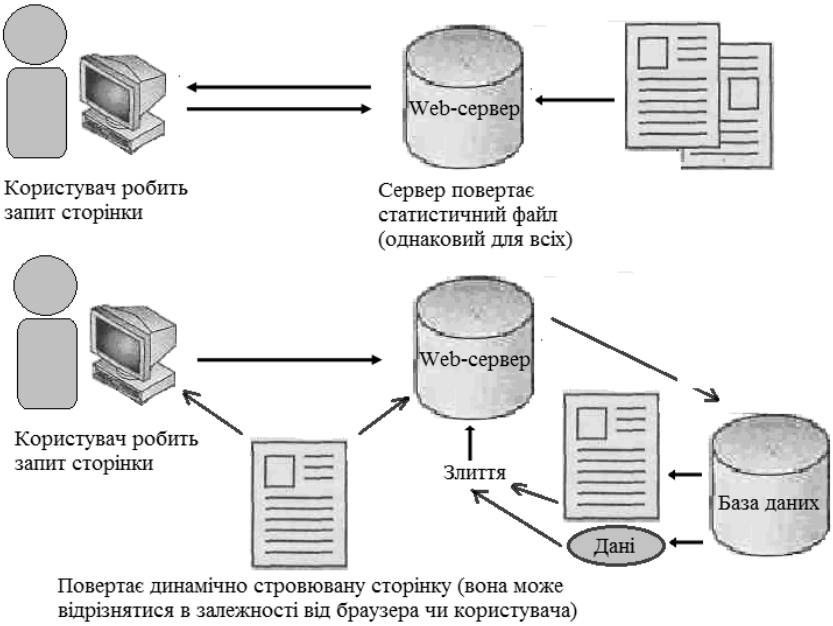



Рисунок 3.9. Порівняння статичних і динамічно генерованих сайтів

**Класифікація за розміром**

Ще одно можливе міркування при класифікації Web-сайтов торкається обліку їх розміру.

Хоча точних категорій, визначальних те, що великий, а що є дрібний сайт, не існує, наступні групи представляються корисними:

– < 10 сторінок – дуже маленький сайт;
– 10-100 сторінок – невеликий сайт;
– 100-1000 сторінок – сайт середнього розміру;
– 1000-10 000 сторінок – великий сайт;
– > 10 000 сторінок – дуже великий сайт.

Значення в цих групах показують зусилля задіяного персоналу при розробці сайту.

**Класифікація за використанням технологій**

Класифікація сайтів за їх розміром або мірою інтерактивності часто безпосередньо перетинається з технічними міркуваннями.

Безліч сайтів, згрупованих за домінуючим типом використання на них технології, від простих документів до повнофункціональних програмних застосувань на базі Internet.

**Класифікація за призначенням**

Як ми могли помітити, існує безліч способів класифікації сайту, включаючи угрупування по їх аудиторії, частоті їх модифікації, технології або зовнішньому вигляду. Проте ці характеристики часом можуть виглядати занадто абстрактними.

Коротке обговорення типів сайтів буде корисним, якщо класифікувати і розглядати відмінності між різними сайтами.

Один дуже загальний метод, що дозволяє дати сайту характеристику, полягає у віднесенні сайту до однієї з категорій: комерційний, розважальний, інформаційний, навігаційний, художній або персональний.

**Класифікація на вигляд**

За цією ознакою сайти підрозділяються на чотири візуальні групи:

- *текcто-орієнтовані.* Зосередження на текстовому вмісті; обмежений дизайн і графіка;
- *стилізовані під GUI.* Наслідування угод, що стосуються графічних інтерфейсів;
- *метафоричні.* Надання інтерфейсу з широкими можливостями, чиє оформлення частенько засноване на метафорі, узятій з реального світу;
- *експериментальні.* Недотримання угод, представлення вмісту і навігації сайту по-новому або в несподіваному варіанті.

Приклади кожного стилю представлені на рис. 3.10.

**Комерційні сайти**

Комерційними сайтами є ті сайти, що створені переважно для підтримки бізнесу якої-небудь організації.

З урахуванням змішаної аудиторії основні цілі комерційних сайтів включають:

- **поширення основної інформації –** сайт використовується для поширення інформації про продукцію і послуги, що надаються організацією. Інша інформація, що надається, в основному включає дані про те, як зв'язатися з фірмою способами, відмінними від Web;
- **підтримку –** частина сайту може бути призначена для надання клієнтам інформаційної допомоги відносно ефективного користування продуктами або послугами, що надаються організацією;
- **зв'язки з інвесторами –** відкриті акціонерні товариства або компанії, що шукають зовнішніх інвесторів, можуть створити сайт або розділ сайту для публікації інформації про поточну економічну ситуацію в компанії, а також про майбутні можливості для вкладення інвестицій;

- **зв'язок з громадськістю –** багато фірм використовують Web-сайти для надання інформації різним організаціям, що займаються збором даних, а також добровільного обнародування основної інформації про фірму;
- **пошук службовців –** Web-сайти часто використовуються для розміщення оголошень про прийом на роботу і переваги роботи в компанії;

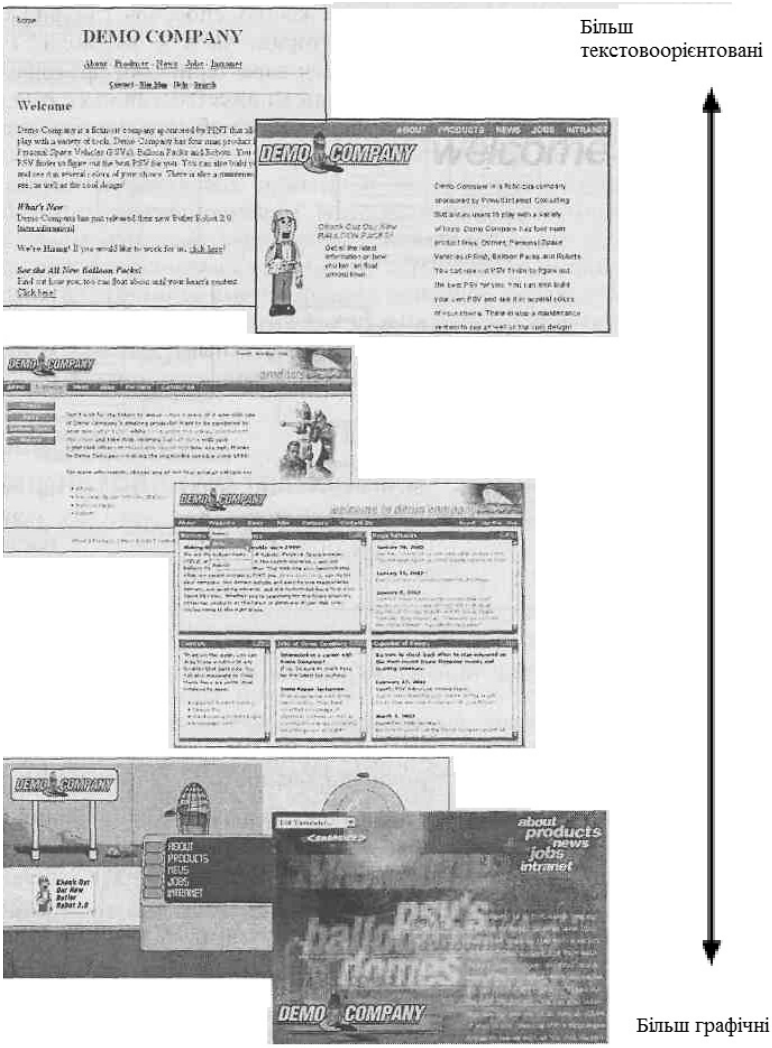



Рисунок 3.10. Візуальний діапазон Web-сайтів

**електронну комерцію –** зростаюче число комерційних Web-сайтів дозволяє відвідувачеві, як кінцевому споживачеві, так і бізнес-партнерові, подібно до торгового посередника, вести бізнес безпосередньо за допомогою Web-сайта. Звичайні можливості, підтримувані сайтами електронної комерції, включають транзакції, такі як оформлення замовлення, отримання відомостей про стан замовлення і залишок на рахунку. Таким чином, ми можемо виділити комерційні сайти, що спеціалізуються на електронній комерції. Такі сайти зазвичай називають *операційними*.

***Основна мета*** будь-якого комерційного сайту полягає в обслуговуванні користувачів способом, який потрібно може принести компанії вигоду безпосередньо чи опосередковано.

**Інформаційні сайти**

Інформаційні сайти відрізняються від комерційних головною метою поширення інформації. До них відносяться:

- ✧ урядові сайти;
- ✧ освітні сайти;
- ✧ сайти новин;
- ✧ сайти некомерційних організацій;
- ✧ релігійних груп;
- ✧ різні громадські сайти,
- ✧ часто відносяться до категорії інформаційних. У основі управління таким сайтом можуть лежати деякі комерційні чинники, але *основна мета* сайту, що полягає в наданні інформації, далека від сфери стимулювання ділових операцій. Визначити неоднорідну аудиторію інформаційного сайту досить складно, т. к. вона великою мірою залежить від типу інформації, що надається. Аудиторія сайту складається з тих, хто має інтерес або необхідність проглянути інформацію, що міститься на сайті.

Цілі інформаційних сайтів можуть значно різнитися. Сайт університету може намагатися допомогти відвідувачам отримати знання по певній темі, наприклад, історії Америки. Головна мета інформаційних сайтів деяких певних релігійних, громадських або політичних груп може полягати в спробі підштовхнути людей приєднатися до групи або зробити пожертвування до її фонду. Мета нових сайтів може полягати в наданні інформації про поточні події в зручному для користувачу виді, в результаті люди стануть покладатися на цей ресурс досить для того, щоб нові сайти могли продавати увагу своїх відвідувачів. Урядовий сайт може мати на меті інформування громадян про зміни в законах, спроби переконати їх поступити на громадянську або військову службу, або навіть змусити го-

лосувати на виборах певним чином. Області зіткнення між комерційними і інформаційними сайтами можуть бути значними, але завжди пам'ятаєте, що *основна відмінність* полягає в тому, що комерційні сайти мають під собою набагато більший економічний стимул, чим інформаційні. Інформаційні сайти можуть бути побудовані з дотриманням критеріїв дизайну, які не мають економічного сенсу. Комерційні сайти завжди мають на меті ріст прибутку фірми, і ця мета часто більше передбачувана.

**Розважальні сайти**

Розважальні сайти в основному є комерційними, але вони заслуговують на окрему увагу. ***Мета*** розважальних сайтів полягає в тому, щоб просто розважити своїх відвідувачів. В деякому розумінні вони продають розваги або, іншими словами, намагаються продати приємні переживання. Не дивлячись на те, що комерційні сайти, такі як сайти електронної комерції, хочуть, щоб їх відвідувачі отримали позитивні емоції або навіть розважилися, само розвага насправді носить другорядний характер. Хоча сайт для продажу одягу теж може бути оформлений в стилі джунглів і бавити відвідувача небилицями про подорожі в далекі країни, але що лежить в основі цього ідея полягає в тому, що позитивні емоції допомагають продавати одяг. Якщо одяг не розкуповується, то сайт припиняє свою роботу. У разі розважального сайту товаром, що продається, є само розвага.

Створення приємних емоцій, будь то відвідування Web-сайта, відеогра або перегляд фільму, це не те, що можна легко спроектувати. Утримувати глядача, цілком займаючи його увагу і роблячи задоволеним, може бути важким, і стереотипний підхід до цього працює не завжди. Приміром, Голлівуд не залишає спроби зрозуміти, чому одні очікувані блокбастери провалюються в прокаті, тоді як невідомі незалежні фільми стають успішними. Новизна – це єдина річ, яка, здається, завжди розкуповуватиметься. Якщо історія занадто нагадує щось, що людина вже випробовувала раніше, то часто це виглядає нудним і трафаретним. Web-сайтам, створеним для розваг, зазвичай необхідно порушити угоди, щоб бути успішними.

Для розважальних сайтів новизна або несподіванка можуть виявитися кориснішими, ніж структура або послідовність.

**Навігаційні сайти**

***Навігаційним сайтом*** є той сайт, чиє первинне завдання полягає в наданні людям допомоги в пошуку шляху через Internet. Такі сайти називаються **порталами** відколи вони стали грати роль основних центрів, що розміщують покажчики на інші місця в Internet.

**Портал –** це сайт, який є основною відправною точкою в подорожі користувача по Internet і служить для надання користувачам допомоги в пошуку інформації. Портали часто намагаються надати настільки багато інформації і послуг для вирішення завдань, що стоять перед користувачами,

наскільки це тільки можливо, і тим самим заохочують користувачів залишатися на сайті або, принаймні, постійно на нього повертатися.

Навігаційні сайти можуть також включати пошукові машини або каталоги сайтів, які, в тій же година, часто є основою багатьох сайтів-порталів.

**Громадські сайти**

*До громадських (соціальних)* відносяться ті сайти, чия мета полягає в створенні для членів певного суспільства місця дислокації, де вони можуть спілкуватися між собою. Відвідувачі приходять на сайт, який по характеру часто є дуже інформативним, не лише для того, щоб знайти певну інформацію, що цікавить їх, але і щоб вступити в контакт з іншими людьми, що мають схожі з ними переконання або інтереси. Зміст громадських сайтів так само сильно варіюється, як і інформаційних. Деякі суспільства можуть бути дуже *відкритими*, і членство в них може бути засноване на демографічному принципі, наприклад, жіноче товариство. Інші ж можуть бути дуже *вузьконаправленими* і націленими на певну групу індивідуалів, таких як студенти коледжу Asian American в Південній Каліфорнії. Основна відмінність між чисто інформаційними або комерційними сайтами і сайтами суспільств полягає в можливості їх відвідувачів спілкуватися один з одним. Якщо з часом можливість спілкування відвідувачів сайту між собою стане звичайним явищем, то специфічна відмінність громадських сайтів буде втрачена.

**Сайт співтовариства –** це будь-який сайт, на якому надається можливість простої взаємодії між відвідувачами сайту, і який служить для його відвідувачів місцем зустрічей, а не тільки перегляду фіксованого вмісту.

**Художні сайти**

Художній сайт є самовираженням якого-небудь автора. *Мета* такого сайту може полягати в тому, щоб надихнути, прояснити або розважити його відвідувачів. В деяких випадках сайт може являтися просто результатом спроби художника виразити свої переживання. Його може не цікавити, що відвідувачі подумають про сайт. До тих пір, поки сайт робить художника щасливим, він успішний. Вплив користувача на художній сайт зводиться до того, що він або може його подумки схвалити, або просто пройти мимо.

Дизайн художніх сайтів може навмисно ігнорувати загальнопоширені Web-соглашения.

**Персональні сайти**

Схожий на художній сайт, персональний сайт, часто званий персональною домашньою сторінкою або просто домашньою сторінкою, є самовираженням його творця. Персональні сторінки можуть створюватися для того, щоб бути показаними друзям або родичам, або просто для того,

щоб спробувати освоїти новий предмет, такий як HTML. Деякі сторінки виглядають як вівтарі поклоніння своїм творцям, виконані в пихатій спробі стати відомими через Web. Інші персональні сторінки є сайтами-резюме, призначеними для вказівки працедавцям на потенційних співробітників, що знаходяться в процесі пошуку роботи. Крім того, отримала популярність нова форма персональної сторінки, виступаюча в ролі мережевого щоденника або журналу, її називають "блог" (blog).

Перш ніж ми продовжимо, слід ненадовго відволіктися на прояснення соціального підтексту персональних Web-сайтов. В деякому розумінні мета персональної сторінки полягає в тому, щоб персоніфікувати окрему особу в Web. До нещастя, це може бути досить небезпечною ідеєю. Незважаючи на уявну очевидність того, що не слід розміщувати на персональній сторінці таку інформацію, як номер кредитної карти, номер соціального страхування, номер банківського рахунку і т. інш., міра подробиць, опублікованих на багатьох персональних сторінках, лякає. Багато людей розміщують на Web-страницах інтимні подробиці свого життя, від фотографій своїх друзів і родичів до буквально своїх ежедневников. Такий мережевий ексгібіціонізм може здаватися нешкідливим, проте слід враховувати можливість того, що хто-небудь захоче увійти в довіру за допомогою цієї інформації. Користувачі повинні розуміти, повідомляючи в мережу про те, що вони люблять, і до чого випробовують неприязнь, вони приводять в життя мрію фахівців з маркетингу. На основі такої інформації легко побудувати профіль користувача і згодом використовувати його для розсилки націленої і настирної пошти. Пам'ятайте, що публікація персональної Web-сторінки не занадто відрізняється від розміщення інформації на дошці оголошень в міському парку. Ніколи не можна передбачити, хто прочитає інформацію і як він з нею поступить.

### 3.1.2.2 Структура сайту

Знаючи тип сайту та іншу інформацію, ми можемо починати накладати на нього структуру. Існують дві структурні сторони кожного Web-сайту:

- ✧ логічна;
- ✧ фізична структура.

Логічна структура описує документи, які пов'язані з іншими документами. Вона визначає зв'язки між ними. Однак логічне розташування документів у межах сайту може не мати відношення до реального фізичного їх розміщення.

Фізична структура описує, де документ знаходиться в дійсності, відображаючи, наприклад, шлях до каталогу документа на Web-сервері або його розташування в базі даних.

Для користувача більш важлива логічна, ніж фізична структура Web-сайту.

Як особа, що займається підтримкою сайту, ви повинні мати вичерпні знання про його фізичну структуру, а користувачам це знати необов'язково. Чи не розкривайте фізичну файлову структуру сайту, коли це можливо.

Всі сучасні Web-сервери підтримують можливість відображення для створення віртуальних шляхів, тому пряме перенесення логічної структури на фізичну файлову систему не потрібно. Логічна структура документів сайту не повинна повністю відповідати фізичній структурі.

З точки зору програміста, URL сайту можна розглядати як загальнодоступний інтерфейс. Кожен відкритий URL є потенційной адресою для доступу до сайту. Якщо є можливість, то уникайте розкриття всіх URL, використовуючи будь-якої із прийомів, починаючи від фреймів і закінчуючи динамічними сторінками, це дозволить змінювати внутрішню реалізацію сайту, не турбуючись про те, що користувачі помітять це.

**Моделі організації сайту**

Існують чотири основні логічні організаційні форми, використовувані Web-сайтами:

- ✧ лінійна,
- ✧ решітка,
- ✧ ієрархія,
- ✧ павутина.

Варіації на тему деяких з схем також є звичайними, як і їх поєднання, що лежать в основі більш великих сайтів. Вибір правильної організації сайту є важливим кроком на шляху створення зручного для використання сайту. Наприклад, для сайту, орієнтованого на online-продажу, більш практичною буде лінійна форма, при якій слайд 2 слідує за слайдом 1. У певному сенсі користувач майже змушений переглядати вміст сайту в тому порядку, в якому цього хоче дизайнер. Якщо подання матеріалу було організовано в іншій манері, такий як дерево, то це може підбурювати користувачів здійснювати доступ до слайдів не по порядку і можливо зменшить вплив, що підштовхує відвідувачів сайту до здійснення покупки. У той же час послідовний доступ мало підходить для подання іншої інформації, такої, наприклад, як питання технічної підтримки, тому примушуючи користувача пробиратися через сторінки з марною інформацією, ми можемо викликати його крайнє роздратування. Мета вибору найбільш придатною для вмісту організаційної форми полягає в тому, щоб зробити складне вміст більш зрозумілим.

**Лінійна організація**

Лінійна форма є найбільш популярною з усіх структур сайтів з причини того, що традиційні друковані інформаційні засоби слідують цьому стилю організації. Наприклад, книги звичайно пишуться таким чином, що одна сторінка слідує за іншою в лінійному порядку. Представлення інформації в

лінійній формі часто буває зручним при обговоренні покрокових процесів, але існують випадки, Коли може вимагатися додаткова інформація. Для забезпечення більшої гнучкості лінійні форми можуть злегка модифікуватися, але при дуже великому розширенні вони перетворюються в грати, ієрархію або павутину.

***Строго лінійна організація***

Строго лінійна організація сприяє впорядкованого просуванню по основній частині інформації, як показано на рис. 3.11.

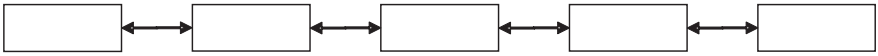


Рисунок 3.11. Строго лінійна організація

У Web така форма може добре підходити для презентації, що нагадує "слайд-шоу" і дає можливість новим відвідувачам ознайомитися з оглядом пропонованої компанією продукції та самою компанією. Використовуючи контрольовану послідовну організацію, подібну лінійній формі, дизайнер може бути впевнений, що користувач отримує інформацію в наміченому порядку.

Лінійний стиль організації забезпечує значну передбачуваність за рахунок того, що дизайнер точно знає, куди користувач потрапить на наступному кроці. Внаслідок цього знання може виявитися можливим здійснити попередню завантаження (preload), або попередню вибірку (prefetch), наступного блоку інформації, що допоможе поліпшити сприйняту продуктивність сайту. Наприклад, поки користувач переглядає інформацію одного екрана зображення для наступного екрану можуть завантажуватися в кеш браузера. Коли користувач просувається до наступного екрану, сторінка завантажується з кеша, створюючи у користувача ілюзію дуже швидкої роботи сайту. Попереднє завантаження не має сенсу, якщо не можна передбачити наступний крок користувача, як це можна зробити у випадку лінійної організації.

Унаслідок того, що насправді у користувача немає вибору за винятком руху вперед або назад, він може порахувати лінійну форму занадто обмежує його свободу. Тому часто буває дуже важливим дозволити користувачеві знати, наскільки далеко в лінійній структурі він знаходиться і що розміщено на попередній і наступній сторінці після поточної. Повідомити користувачеві, на якій сторінці в послідовності сторінок він знаходиться, дуже просто, достатньо розмістити на сторінці мітку, подібну "Сторінка X з Y", де X – це номер поточної сторінки, а Y – загальна кількість сторінок. Це корисно, т. к. концепція сторінок є невластивою для Web, і цей тип позначення не вказує на попередню і наступну сторінки в послідовній структурі, подібно позначенню, наведеному на рис. 3.12.

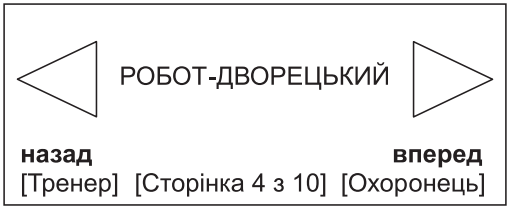



Рисунок 3.12. Приклад вказування місцезнаходження в лінійній структурі сайту

Реалізувати сувору прямолінійну форму на Web-сайті може виявитися проблематичним через функції повернення в браузері, так що зазвичай передбачається, що всі лінійні форми двонаправлено, якщо тільки сайт не запрограмований на іншу дію.

***Лінійна форма з альтернативами***

Враховуючи, що лінійна організація є підходящою для подання інформації в зумовленому порядку, не слід забувати, що вона дає занадто мало можливостей для взаємодії користувача з вмістом сайту. Лінійна організація з альтернативами імітує інтерактивність, надаючи два або більше варіантів переходу зі сторінки, які, в кінцевому рахунку, закінчуються зазначенням користувачеві повернутися на іншу сторінку в послідовності, як показано на рис. 3.13.

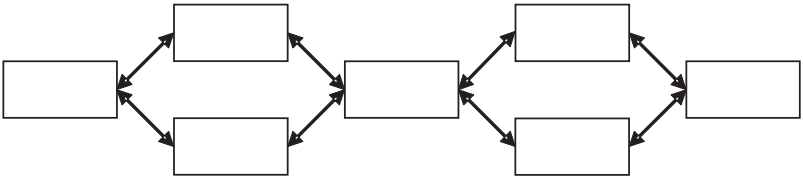


Рисунок 3.13. Лінійна форма з альтернативами

Для цієї форми організації існує безліч застосувань. Постанемо собі сайт-запитальник, який на кожній сторінці задає користувачеві питання, яке потребує відповіді "Так" або "Ні" і просуває користувача на наступну сторінку, грунтуючись на даному відповіді. Хоча для користувача це може виглядати як робота деякого внутрішнього механізму, насправді існують два наперед заданих шляху, і у користувача просто створюється ілюзія інтерактивності. Сайт, присвячений питанням здоров'я, може задавати користувачам загальні питання щодо їх здоров'я, щоб викликати їх інтерес. Опитування може починатися з питання, такого як "Ви курите?" Користувачі, які відповіли "Так" на це питання, просуваються на сторінку, що описує шкоду куріння, в той час як ті, хто відповів "Ні", бачать

повідомлення, схвалює їх рішення відмовитися від сигарет. Незалежно від відповіді на перше питання обидві категорії користувачів потраплять на сторінку з другим питанням. У цьому випадку сторінки є статичними, і динамічної генерації сторінок не відбувається, однак у користувача створюється враження, що має місце якась інтерактивність. Незалежно від наявності вибору лінійна структура з альтернативами зберігає загальне лінійне напрямок через набір документів. На жаль, наявність декількох шляхів робить попередню завантаження сторінок для цієї форми сайта більш складною.

***Лінійна форма зі свободою вибору***

Лінійна структура зі свободою вибору добре працює, коли необхідно зберегти загальний напрямок, але при цьому потрібно додати легкі варіації, такі як пропуск певних сторінок. Такий тип гіпертекстової організації може бути корисний для online-оглядів, де деякі користувачі мають можливість пропустити певні неактуальні питання. Беручи до уваги те, що така структура в основному тільки дозволяє перестрибувати вперед, її часто називають лінійною структурою з переходами вперед. Прикладом такої структури в дії може бути презентація велосипедів. У той час як деякі основні сторінки є загальними для всіх велосипедів, визначені сторінки можуть бути пропущені, якщо користувача цікавлять виключно гірські або шосейні велосипеди. В паперової документації огляд, який вимагає пропуску певних моментів, грунтуючись на деякому критерії, відповідає лінійній формі зі свободою вибору. Основна ідея такої структури сайту показана на рис. 3.14.

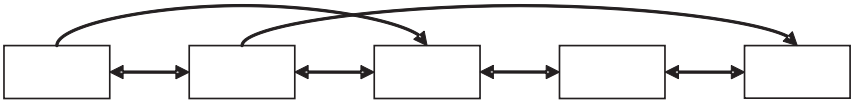


Рисунок 3.14. Лінійна форма з вільним вибором

І знову така організація нагадує інтелектуальну систему навіть незважаючи на те, що часто є не більше, ніж статичними файлами в добре продуманої гіпертекстової структурі.

***Лінійна форма з бічними відгалуженнями***

Лінійна організація з бічними відгалуженнями (рис. 3.15) дозволяє контролювати відхилення від основного напрямку. При цьому користувач має можливість трохи відхилитися в бік, проте структура все ж примушує його повернутися до основного шляху, зберігаючи початкове напрям руху. Припустимо, що стаття про жаб представлена в лінійній формі. Гіперпосилання на певне слово, таке як лілія, може вести на сторінку, яка не

має відношення до основної теми, де наводиться визначення слова і, можливо, посилання на кілька сторінок з обговоренням зв'язку між жабами і лілеями. Бічне відгалуження може закінчуватися тупиком або поверненням назад до основного напряму. Бічне відгалуження в лінійній послідовності подібно бічній врізки в журнальній статті. Воно дозволяє не відводити користувача далеко від основної теми, і при цьому невеликий обсяг додатково поданої інформації розширює кругозір. Роблячи побічну замітку частиною основної лінійної послідовності, можна злегка порушити безперервність вихідного викладу. Однак коли в лінійну послідовність додається занадто багато бічних відгалужень, структура починає виглядати як звичайне дерево або ієрархія.

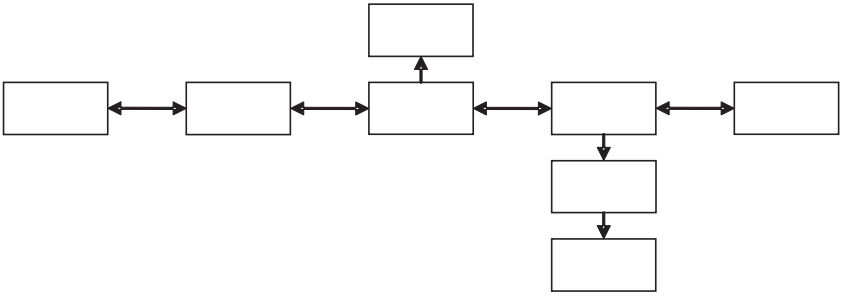


Рисунок 3.15. Лінійна форма з боковими відвітленнями

**Решітка**

Решітка – це односпрямована лінійна структура, в якій присутні як горизонтальні, так і вертикальні зв'язки між елементами. Так як решітка має просторову організацію, вона добре підходить для сукупності споріднених елементів, однак структура суворої решітки є (поки що) рідкісною для Web. Правильно розроблена решітка має горизонтальні і вертикальні орієнтири, тому користувач не відчуває себе заблукали усередині сайту. Наприклад, предмети в каталозі одягу можуть бути зібрані в категорії, такі як сорочки, брюки і куртки. Інший спосіб організації інформації може полягати в розбитті по цінових категоріях. Структура в стилі решітки (рис. 3.16) дозволить користувачеві легко переглядати як ціни, так і певну лінію одягу.

Очевидно, що ґратчаста структура дуже регулярна, і користувач може легко орієнтуватися, однак існує не так багато видів однорідної інформації, досить добре пристосованою для такого роду організації. Єдине помітне виключення – це каталог виробів.

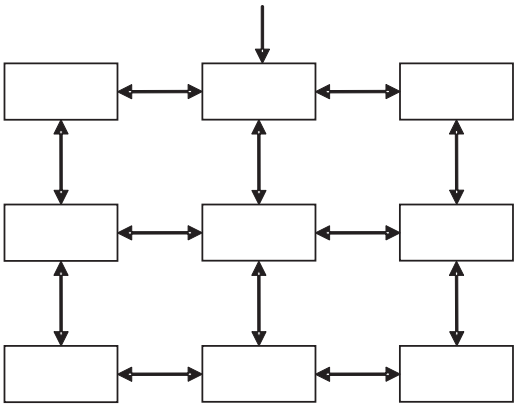


Рисунок 3.16. Решітка

**Ієрархія**

Найбільш часто зустрічається в Web гіпертекстової структурою є форма дерева або ієрархії. Хоча ієрархія може не забезпечувати просторової структури, притаманної решітці, або передбачуваності та керованості лінійної структури, вона надзвичайно важлива, т. к. може бути модифікована для того, щоб приховати або показати стільки інформації, скільки необхідно. Ієрархії починаються з кореневої сторінки, яка часто є домашньою сторінкою сайту або розділу. Домашнє або коренева сторінка дерева сайту служить як сторінки-орієнтира, т. к. часто сильно відрізняється за зовнішнім виглядом від інших сторінок сайту. Орієнтири сайту, такі як домашні сторінки, є ключем до створення успішної навігації. Для переходу з домашньої сторінки представлені різні варіанти вибору. У міру просування користувача в глиб сайту варіанти вибору стають все більш специфічними, поки не приведуть до пункту призначення або кінцевої сторінці в дереві досліджуваного сайту. З цієї причини існує тенденція описувати дерева з позицій їх глибини і ширини.

**Вузькі дерева**

Вузьке дерево (рис. 3.17) пропонує тільки кілька варіантів вибору, але може вимагати великої кількості клацань кнопкою миші, щоб досягти кінцевого місця призначення. При такій організації глибина переважає над шириною.

Для досягнення кінцевої сторінки вузьке дерево може вимагати від користувача зробити свій вибір велику кількість разів, але для деяких сайтів це дуже ефективний спосіб швидко направити користувачів в правильний розділ. Візьмемо, наприклад, Web-сайт служби з підбору персоналу. Він має два основних види аудиторії: шукають роботу і роботодавців, які мають

вакансіями. Явне розділення цих двох аудиторій ще на домашній сторінці і вибір користувачем потрібної категорії сприяють швидкому і легкому доступу до відповідних розділів сайту. Розширення діапазону варіантів вибору верхнього рівня для включення специфічних можливостей, призначених для шукачів роботи і роботодавців, може збивати з пантелику відвідувачів сайту. Використання вузької ієрархії в якості засобу послідовного просування по сайту може допомогти утримувати користувача на правильному напрямку. Однак при цьому зазвичай збільшується число клацань кнопкою миші, необхідних для досягнення кінцевої мети. Дуже важливо знайти баланс між цими двома факторами і уникнути установки зайвих бар'єрів між користувачем і необхідною йому інформацією. Пам'ятайте, ієрархія сайту може виявитися занадто вузькою, якщо існує багато сторінок, які в структурі явно повинні бути розташовані відразу після домашньої сторінки. Пам'ятайте, що користувачі хочуть отримати "нагороду", а нескінченне клацання мишею по сторінках, містить дуже мало корисної інформації, є для них більш ніж дратівливим.

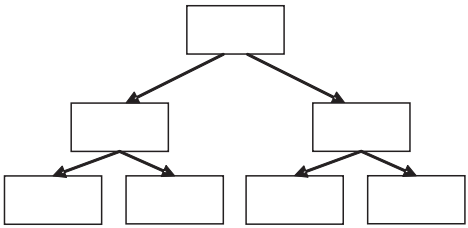


Рисунок 3.17. Вузька ієрархія

**Широкі дерева**

Широке дерево або широка ієрархія (рис. 3.18) ґрунтується на великій кількості варіантів вибору. Її основний недолік полягає в тому, що вона може пропонувати надто багато варіантів у вигляді сторінок, що мають велику кількість виходять з них посилань.

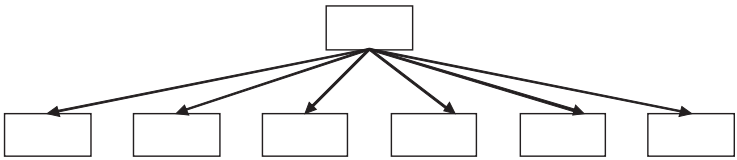


Рисунок 3.18. Широкая ієрархія

Незважаючи на те, що користувач повинен тільки один або два рази клацнути кнопкою миші, щоб дістатися до необхідного йому вмісту, час, витрачений на вивчення всіх первинних варіантів вибору, може бути

надмірним. Багато людей думають, що все важливе має бути винесено на домашню сторінку. Однак, якщо на домашній сторінці розмістити всі посилання, то ієрархія буде втрачена і інформація теж може втратити свою ефективність – у певному сенсі "загубиться в натовпі". Вибір розумного балансу між глибиною сайту і його шириною – це окреме питання.

**Заплутані дерева**

Дійсність Web полягає в тому, що структура стандартного дерева використовується дуже рідко. У стандартному дереві немає перехресних посилань, і для досягнення інших частин дерева часто потрібно повернутися назад. Розглянемо ситуацію, коли користувач знаходиться на сторінці A в структурі, наведеною на рис. 3.19. Щоб досягти сторінки B, він змушений повернутися на два рівні назад, а потім знову рушити вперед.

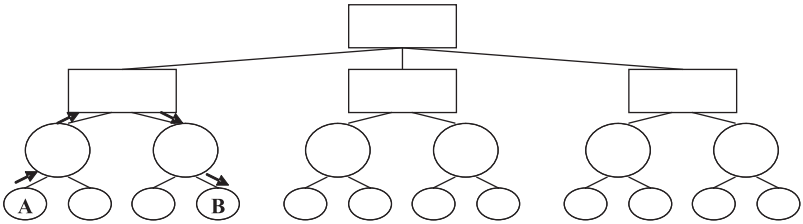



Рисунок 3.19. Дерева можуть вимагати повернення назад

Хоча в Web для повернення можна задіяти кнопку Назад браузера, на сторінки часто додаються посилання, щоб ті користувачі, хто потрапив на сторінки не через основний шлях, могли переміщатися по сайту. У багатьох випадках сторінки пов'язані перехресними посиланнями за допомогою панелі навігації або явних зворотних посилань, які допомагають користувачам швидко переміщатися по структурі сайту. Розглянемо схему сайту, показану на рис. 3.20.

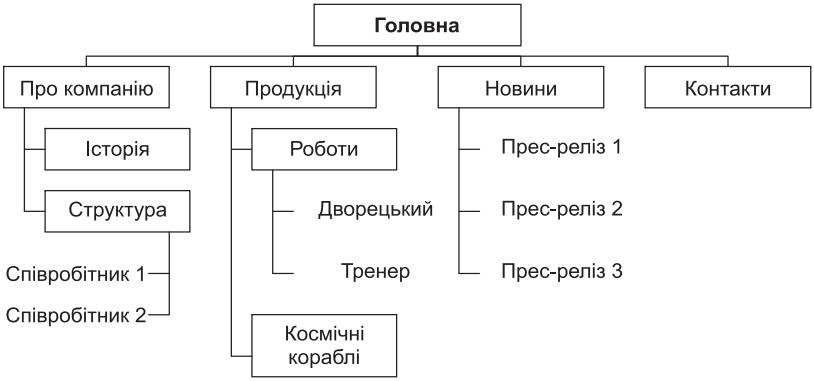



Рисунок 3.20. Простая ієрархія сайту

Поширеним прийомом є створення для сайту панелі навігації, яка містить основні розділи сайту, такі як Home, About, Products, News і Contact (рис. 3.21).




Рисунок 3.21. Приклад панелі навігації сайту

З такою панеллю навігації набагато простіше перестрибнути з розділу в розділ без необхідності здійснювати для цього переходи тому. Однак схема сайту може бути набагато більш складною і виглядати приблизно так, як та, що показана на рис. 3.22.

Зворотні і перехресні посилання в структурі сайту значно збільшують її складність. У цьому випадку вважайте, що тільки головні сторінки розділів пов'язані перехресними посиланнями. Уявіть, що буде, якщо всі сторінки сайту будуть пов'язані подібним чином.

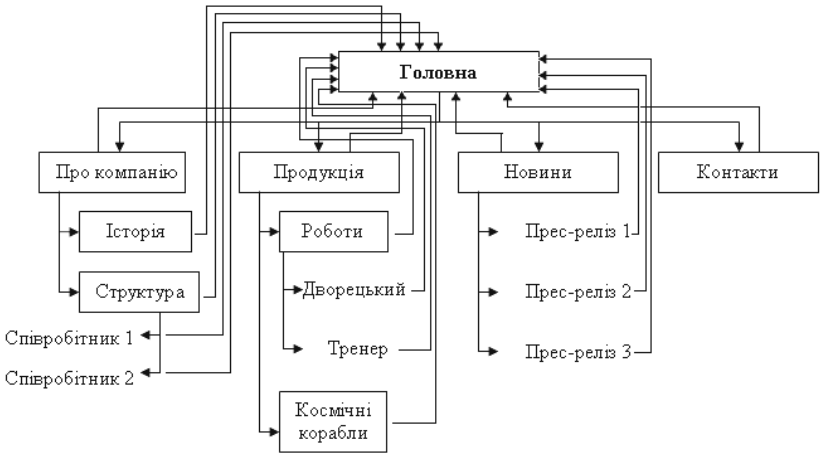



Рисунок 3.22. Ієрархія сайту з відображенням зворотних посилань

**Повне зв'язування**

Сайт, кожна сторінка якого пов'язана посиланням з кожної іншої сторінкою цього сайту, може розглядатися в якості структури, званої повне зв'язування (full mesh). Ілюстрація, наведена на рис. 3.23, показує повне зв'язування для сайту з п'ятьма сторінками.

При повному зв'язуванні кількість посилань дорівнює числу сторінок, помноженому на число сторінок мінус один. Це означає, що для сайту з п'яти сторінок кількість посилань буде дорівнює 20. Для сайту з 20 сторінок кількість посилань складе 90. Для сайту з 100 сторінок – число посилань

складе 9900 (100x99) і для 1000-сторінкового сайту їх число буде в районі одного мільйони! В дійсності з точки зору практичності повне зв'язування не є гарним вибором. Насправді, більшість сайтів схильні використовувати часткове зв'язування з перехресними посиланнями тільки на найбільш важливі сторінки.

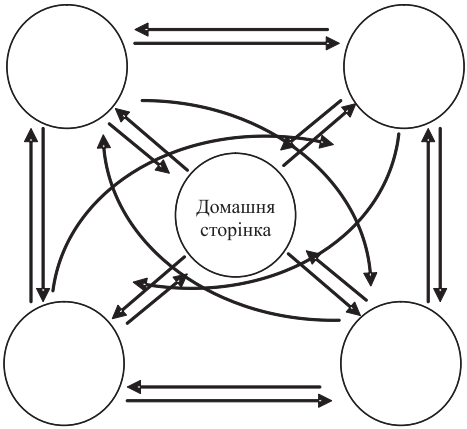



Рисунок 3.23. Домашня сторінка

**Змішані форми**

У той час як широке дерево може показувати занадто багато, надмірно вузька ієрархія приховує занадто багато інформації. Лінійний підхід може давати користувачеві занадто малий контроль, в той час як павутина надає користувачеві занадто багато свободи.

У деяких випадках необхідно розширити ієрархію, щоб дозволити варіантам вибору вказувати на вершину дерева. Така структура називається змішаною формою або змішаної ієрархією, і дерево є домінуючою формою структури. Змішана форма (рис. 3.24), ймовірно, є найбільш часто зустрічається в Web видом організації сайту. Усередині змішаної форми можуть міститися лінійні ділянки, пропуски і навіть решітки. Розглянемо сайт, який містить кнопку Завантажити або подібну їй кнопку, провідну в глиб структури сайту. Це нагадує лінійну структуру з переходами. Інші сайти можуть містити лінійні ділянки, вхід на які доступний тільки з певних сторінок сайту. Хоча в цій формі просторова організація виражена не так явно, як в інших структурах сайтів, ієрархія все ж є найбільш очевидною частиною в більшості змішаних сайтів.

Один часто зустрічається тип змішаної структури полягає у використанні лінійної структури для входу на сайт і дерева з моменту досягнення справжньої домашньої сторінки. Для цього використовують-

ся виставлені на показ сторінки з інформацією про сайт, які ведуть до центральної сторінці, звідки користувач може почати перегляд сайту. Структурна схема такої організації показана на рис. 3.25.

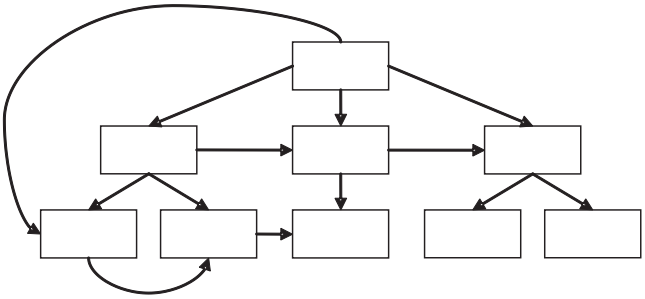


Рисунок 3.24. Змішана ієрархія

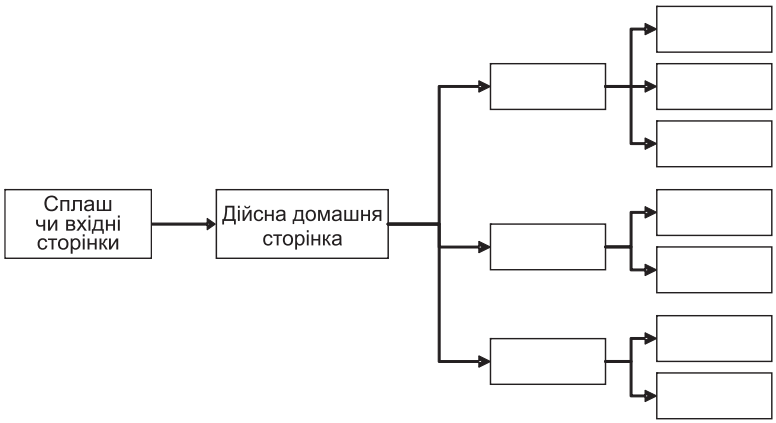



Рисунок 3.25. Дерево з лінійним входом

Інший тип, який насправді не є унікальним, позначається терміном структура "маточина і спиці" (hub and spoke). Багато сайтів складаються з головних сторінок, званих маточинами, і підлеглих сторінок, доступ до яких здійснюється через спиці. Щоб відвідати інші сторінки сайту, користувач змушений повернутися до головної сторінки.

Багато портали використовують цей стиль для заохочення повторного відвідування сторінок. Проте насправді між моделлю маточини зі спицями і стандартним деревом немає різниці, як це показано на рис. 3.26.

Одна з переваг центру і спиці полягає в тому, що ця модель може забезпечити простий спосіб осмислення сайту: центральні розділи вмісту (центр) зі спицями спорідненого вмісту, які користувач коротенько переглядає

перед поверненням в центр. Інша, пов'язана з цим причина, по якій дизайнери можуть віддати перевагу мислити в категоріях центру і спиці, а не простих дерев, полягає в тому, що ця модель здатна надати вдалий спосіб візуалізації вмісту сайту. Приміром, деякі засоби відображення карти сайту демонструють схеми сайтів саме в такому вигляді, т. К. Він легше для складання, ніж структура дерева. На рис. 3.27 показаний приклад візуалізації сайту в стилі "маточина і спиці".

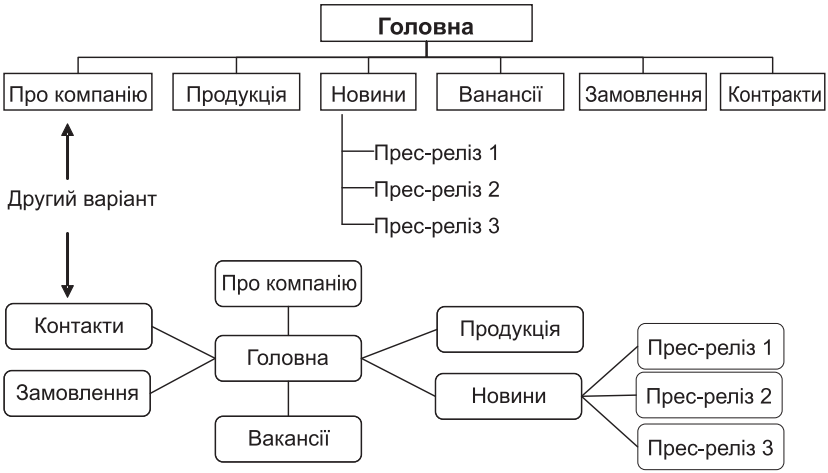



Рисунок 3.26. Модель "маточина і спиці" і деревоподібна структура дуже схожі

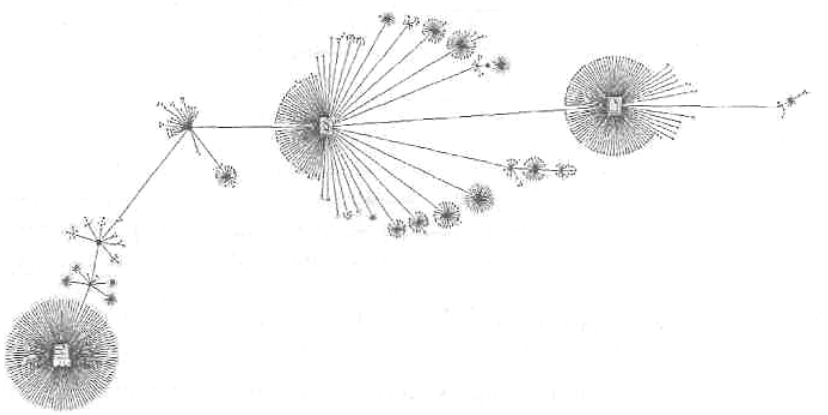


Рисунок 3.27. Модель "маточина і спиці" добре підходить для візуального представлення структури сайту

**Стиль павутини**

Коли в структуру набору документів внесено дуже багато перехресних посилань, переходів вперед і інших додатків, для користувача вона стає незрозумілою. Якщо сукупність документів виглядає так, ніби не має різличимой структури, як показано на рис. 3.28, то вона називається павутиною (pure web).

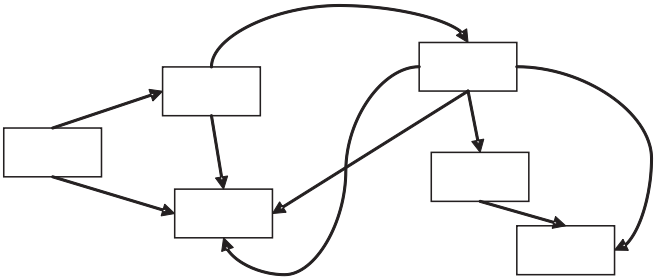


Рисунок 3.28. Павутина

Павутина може бути складна для використання, тому, що вона позбавлена виразною просторової орієнтації. Незважаючи на те, що в разі правильного вибору можна отримати швидкий доступ до інформації, на сайті з такою неочевидній структурою складно орієнтуватися. Якщо структура сайту незрозуміла або неочевидна для користувача, він може вдатися до навігації, заснованої на використанні домашньої сторінки, і при вирішенні нового завдання завжди буде повертатися на верхній рівень структури.

Крім цього, гідністю менш структурованої форми є те, що вона забезпечує великі виразні можливості. Приміром, технічна стаття може надавати посилання на мають до неї відношення схеми, підтверджуючі документи і навіть витяги з зовнішніх джерел. Організація сайту може не вписуватися в жодну з більш структурованих форм. Незважаючи на аргументи, які стверджують, що збиває з пантелику структура у формі павутини може призвести користувача до втрати уваги і робить складним формування уявної карти сайту, насправді це не є проблемою, якщо інформація або стоять завдання правильно опрацьовані.

**Інші стилі сайтів**

Вхід і вихід є для користувача ключовими віхами. Тому інший спосіб розподілити Web-сайти за категоріями може ґрунтуватися на кількості вхідних точок сайту. Використання точок виходу не є практичним, т. К. Кожна сторінка сайту може розглядатися як вихід, якщо користувач просто вирішив піти з сайту. Коли всі документи сайту розташовують загальнодоступними URL, це є прикладом "пористої" структури. Сайт з пористою

структурою не примушує користувача входити через звичайні точки входу, такі як домашня сторінка, головні сторінки розділів і т. П. Більшість користувачів, ймовірно, потрапляють на сайт через ці сторінки, але теоретично будь URL, як би глибоко в структурі сайту він не мав охоти, може бути точкою входу. На противагу пористим, сайтом з "монолітної" структурою буде вважатися той, що строго обмежує точки входу на сайт декількома або навіть одним URL.

У табл. 3.2 зібрані основні переваги і недоліки двох типів сайтів.

Таблиця 3.2

**Порівняння пористої і монолітної форм сайтів**

| Тип сайту | Переваги | Недоліки |
|---|---|---|
| Пориста форма | Користувач має самостійність | Знижує можливість змінювати внутрішні сторінки без зміни вхідних зовнішніх посилань |
| | Дозволяє користувачеві входити, задавши будь URL, або за допомогою закладки | Не так просто використовувати загальноприйняту точку входу для розміщення оголошень, підготовки користувача або орієнтує інформації |
| Монолітна форма | Не розкриває внутрішню структуру сайту, дозволяючи більш просто здійснювати зміни та обслуговування | Віднімає у користувачів самостійність |
| | Змушує користувача входити на сайт через відомі точки | Може обмежити ефективність роботи зовнішніх пошукових машин |
| | Робить напрямок руху користувачів більш передбачуваним | |

Можливо, деякі читачі бажають знати, чому сайти слід робити монолітними або наполовину пористими, і як цього можна досягти. Перш за все, зрозумійте, що протягом довгого часу розділи сайтів робилися саме такими. Розглянемо, наприклад, процедуру підрахунку вартості товарів у кошику. Надавати користувачам можливість в ході цієї процедури встановлювати закладки на глибокі сторінки не має сенсу. Хоча користувач може встановити закладку на певний розташування, на сайті все одно буде застосований який-небудь варіант управління сеансом для організації закінчення терміну дії сторінок або позбавлення користувачів права входу

посередині процесу. Ми також могли б перевірити запити на сторінку, щоб визначити, з якою іншою сторінки відбувається перехід на неї, і обмежити доступ. Серед інших прикладів менш пористих структур сайтів – деякі види динамічного вмісту і захищені розділи сайтів, що вимагають реєстрації. З плином часу від застосування заданих точок доступу до вмісту сайтів доведеться відмовитися, щоб добитися їх зручною змінності.

Інший спосіб охарактеризувати сайти може полягати в підрахунку кількості натискань кнопки миші, необхідних для досягнення необхідної сторінки – глибокі і неглибокі сайти. Обговоримо вибір між структурами у вигляді вузького і широкого дерева. Вузьке дерево може для досягнення сторінок, розташованих глибоко в структурі сайту, вимагати від користувача натиснути на кнопку миші безліч разів. Широке дерево потребують меншої кількості натискань, але при цьому користувачеві доведеться переглянути велику кількість посилань в пошуках тієї, що його цікавить. Очевидно, що найкращим вибором є рівновага між широтою діапазону посилань і глибиною сайту. Різні Web-дослідження говорять про те, що користувачі воліють сайти, що вимагають меншої кількості клацань кнопкою миші, і вітають широкий вибір варіантів. Гарне правило полягає в тому, щоб вважати допустимою глибиною для досягнення необхідного користувачеві вмісту три клацання кнопкою миші.

Досить часто специфіка створюваного сайту диктує те, яка структура буде використана.

**Вибір структури і типу сайту**

Вибір правильної структури для Web-сайту зводиться до подання інформації у вигляді сукупності сторінок, часто званої інформаційної архітектурою. Вибір правильної структури для сайту є комплексним завданням, і на нього може впливати безліч факторів. Наприклад, самі дані можуть припускати певний спосіб організації. Цей спосіб можна розглядати як підхід знизу вгору. Наприклад, структура слайд-шоу дійсно повинна бути лінійною, т. к. логічна послідовність викладу може бути втрачена, якщо інформація буде представлена в іншій формі, такий як дерево.

Інший спосіб, що дозволяє взяти до уваги організацію інформації, більшою мірою є підходом зверху вниз і заснований на тому, як дані будуть використовуватися. Цей підхід надав би пріоритет тим, хто користується сайтом, і тому, як дані забезпечуються і споживаються. Наприклад, лінійні структури увазі менший контроль з боку користувача і обмежену виразність, але при цьому є більш передбачуваними. Починаючі користувачі віддають перевагу простим структури, такі як лінійні структури і глибокі дерева, т. к. зробити вибір на такій структурі відносно легко.

Кожен вид структури сайту має свої недоліки і переваги. На рис. 29 показано зв'язок між виразністю і передбачуваністю різних структур сайтів. У той час як лінійні структури є більш передбачуваними, вони надають

відносно обмежений огляд. Форма павутини є дуже виразною, але при цьому може бути також дуже заплутаною. Ієрархічні і змішані структури знаходяться десь посередині. При створенні сайтів, які не є динамічними, кращим вибором буде націленість на золоту середину. Не дивно, що більшість сайтів мають тенденцію демонструвати якусь форму ієрархії.

Правильний інформаційний дизайн є запорукою створення успішного Web-сайту. Якщо сайт має чудове зміст і чудовий інтерфейс, але кепську інформаційну архітектуру, він може бути порівняно безуспішним. Сайт втрачає свою ефективність, якщо користувач не може легко знайти інформацію. Зараз більшість сайтів використовують змішано-ієрархічний підхід, який є звичним для багатьох користувачів Web. Проте залежно від цілей сайту кілька типів структур можуть комбінуватися. Наприклад, незважаючи на те, що основна структура сайту може бути ієрархічною, для надання інформації про компанію доцільно використовувати чисто лінійну структуру, а обмежену ієрархію – для розділу технічної підтримки.




Рисунок 3.29. Структура сайту: виразність проти передбачуваності

Ключовий момент у структурі сайту – зробити орієнтацію на сайті більш простий для користувача. Завжди пам'ятайте, що користувачі не збираються глибоко вникати в лежачу в основі сайту структуру, тому не змушуйте їх робити це. Враховуйте точку зору користувача, який хоче увійти на сайт, досліджувати його, намагаючись вирішити свої завдання, і в підсумку покинути його. Користувачі не цікавляться структурою сайту до тих пір, поки вони можуть отримати те, що хочуть, простим шляхом. Тому

будь-яка вибрана для сайту структура повинна допомагати користувачам орієнтуватися і збільшувати ймовірність досягнення успіху у вирішенні поставлених перед ними завдань.

Один із способів класифікації сайтів – це розгляд їх аудиторії. Публічні Web-сайти схильні мати широку аудиторію, в той час як аудиторія закритих intranet-сайтів може бути визначена з граничною точністю. Розгляд аудиторії може зробити значний вплив на міркування щодо дизайну сайту. Крім того, сайти можна класифікувати за розміром, технології і візуального оформлення, однак найбільш важлива класифікація стосується призначення сайту. Очевидно, не всі сайти переслідують одну і ту ж мету, а тому не обов'язково повинні керуватися однаковими дизайнерськими міркуваннями. Комерційні сторінки мають більш різноманітні передумови, ніж розважальні. Дизайнерам слід бути завжди обережними і не застосовувати один і той же критерій дизайну до сайту без урахування його аудиторії і призначення. Однак незалежно від аудиторії і мети більшість сайтів поділяють схожу організацію. Деякі сайти мають прості архітектури, подібні лінійної послідовності сторінок, в той час як інші демонструють складну ієрархію або змішані форми архітектури. При розробці структури завжди слід враховувати відкриття когнітології (науки про мислення) і намагатися досягти балансу між глибиною сайту і широтою діапазону посилань. Дизайнери повинні розуміти, що логічна організація сайту і фізична організація, це не одне і те ж. Насправді структура сайту набагато більш часто корисна дизайнерам, ніж користувачам. Добре пророблена структура здатна поліпшити організацію сайту, при цьому користувачі можуть і не усвідомлювати її, т. к. вони просто просуваються у напрямку до шуканого вмісту або намагаються вирішити певну задачу.

### 3.1.3 Базові Web-технології

#### 3.1.3.1 Web-браузери

Хоча людський чинник може бути найбільш важливою стороною Web-коммуникаций, ефективний Web-дизайн також надзвичайно залежний від правильного технічного виконання. Якщо сайт невдало сконструйований або повний помилок, відвідувачі можуть випустити з уваги його ідею або призначення. Щоб досягти успіху в Web-дизайне, професіонали повинні повністю розуміти елементи мережевого середовища.

Всесвітня павутина представлена як клієнт-серверна система в масштабах загальнодоступної мережі під назвою Internet. Трьома компонентами будь-якої клієнт-серверної системи є сторона клієнта, сторона сервера і мережа. Основні компоненти, що становлять Всесвітню павутину, наочно продемонстровані раніше.

Зараз ми проведемо дослідження кожного з цих найважливіших компонентів по черзі, і почнемо з боку клієнта, яка, в основному, визначається браузером.

Web-браузер є інтерпретатором наших Web-сайтів. Надзвичайно важливо мати на увазі, який браузер підтримується, і які можливості він має в розпорядженні. На даний момент двома найбільш поширеними браузерами є Microsoft Internet Explorer (вживаний більшістю користувачів браузерів) і Netscape Communicator (Navigator). Хоча більшість користувачів, що звертаються до загальнодоступним Web-сайтам, застосовують саме їх, існує безліч інших версій споживаних браузерів.

**Історія браузерів**

Історія браузерів дозволяє описати той контекст, в якому нині розробляються і проектуються Web-сайти. У цьому короткому і спрощеному огляді позначені деякі найбільш важливі події в розробці основних браузерів, які привели до створення сучасного середовища Web-дизайна.

Якщо вас цікавить історія браузерів і Web, звертайтеся до детальнішої хронології і до емуляторів старих браузерів, які можна знайти на сайті Deja Vu (www.dejavu.org).

***1991-1993: народження World Wide Web***

Тім Бернерс-Ли почав роботу над управлінням гіпертекстовою інформацією у фізичній дослідницькій лабораторії CERN. Сторінки, що містять тільки текст, можна було переглядати за допомогою простого відрядкового браузеру.

***1993: поява NCSA Mosaic***

Браузер Mosaic був створений Марком Андрессеном, студентом в National Center for Supercomputing Applications (NCSA). Хоча це був не перший браузер, що дозволяє розміщувати графіку на сторінці, він був, поза сумнівом, найбільш популярним, оскільки був доступний для різних платформ. Можливість додавати зображення на сторінки стала однією з причин швидкого росту популярності Web. Браузер Mosaic також підтримував звуки, відео, закладки і форми. Усе Web-сторінки цього часу відображалися у вигляді чорного тексту на сірому фоні.

***1994: вихід Netscape 0.9***

Марк Андрессен створив компанію Mosaic Communications Corp. (яка пізніше перетворилася на Netscape Communications) і випустив браузер Netscape 0.9. Ранні браузери не були безкоштовними (виключення робилося для студентів і викладачів). Щоб забезпечити кращий зовнішній вигляд порівняно з безкоштовними браузерами типу Mosaic і, отже, притягнути покупців, компанія Netscape створила свої власні HTML-теги, не оглядаючись на традиційний процес стандартизації. Наприклад, в Netscape 1.1 були включені теги для зміни кольору фону Web-сторінки і форматування тексту з таблицями.

***1996: вихід Microsoft Internet Explorer 3.0***

Нарешті, Microsoft включилася в Web-ігри, випустивши першу конкурентоздатну версію браузеру, оснащеного своїм власним набором тегів і можливостей. Це також був перший браузер, що підтримує таблиці стилів, які у той час були погано зрозумілою методикою розробки.

***1996-1999: початок воєн браузерів***

Впродовж декількох років Web-спільнота спостерігало за тим, як Netscape і Microsoft билися за домінуюче положення на ринку браузерів. Результатом став набір HTML-тегів від різних компаній і несумісні реалізації нових технологій, таких, як JavaScript, каскадні таблиці стилів і динамічний HTML. В якості позитивного ефекту конкуренція між Netscape і Microsoft сприяла швидкому розвитку середовища в цілому.

***1998: Netscape випускає код браузеру Communicator у рамках ліцензії з відкритим початковим кодом***

Цей сміливий крок дозволив тисячам розробників взяти участь у вдосконаленні браузеру Communicator. Врешті-решт вирішено було почати все з початку. Група Mozilla, до якої увійшла частина співробітників Netscape, очолила розробку браузеру з відкритим початковим кодом.

***2000: випущений Internet Explorer 5 для Mac***

Це важливий етап, оскільки це перший браузер, що повністю підтримує Рекомендації HTML 4.01 і CSS 1 і високо встановлює планку відповідності стандартам для інших браузерів. Це також перший браузер, що повністю підтримує формат PNG з альфа-прозрачностью.

***2000: компанію Netscape купила AOL***

Цей етап розглядався як офіційне визнання поразки Netscape у війні браузерів. Вбудований в операційну систему кожного ПК, працюючого під управлінням Windows, Internet Explorer був страшним супротивником. Netscape втратила грунт під ногами, створюючи громіздкі застосування, що включають усе відразу, і дарма витратила декілька років на переписування з нуля версії 6 свого браузеру. На момент написання цієї книги Netscape лише точка на графіках використання браузерів, і усі її версії разом – це лише 1% усіх браузерів проти, приблизно, 90% для усіх версій Internet Explorer.

***2003: заснована компанія Mozilla Foundation***

Відкритий код браузеру Mozilla продовжував розвиватися компанією Mozilla Foundation (яка була заснована, частково, компанією AOL), що тільки що сформувалася.

***2005: вихід браузеру Mozilla Firefox***

Firefox 1.0 зірвав оплески співтовариства розробників завдяки потужній підтримці Web-стандартов і поліпшеною безпекою в порівнянні з Internet Explorer. Firefox має велике значення, оскільки це перший браузер, що відвоював у Microsoft істотну частку ринку.

3.1.3.2 Комп'ютерні методи пошуку

Методи пошуку в тексті, використовувані людиною, представлені наступними формами:

- ✧ пошук "згори" (по змісту з анотаціями глав і, можливо, менш великих розділів);
- ✧ пошук "знизу" (за допомогою різних покажчиків);
- ✧ пошук за допомогою гіпертекстових зв'язків (перехресних посилань);
- ✧ повнотекстовий пошук шляхом перегляду усього тексту.

Комп'ютерні методи пошуку реалізуються в ІПС, БД, БЗ і пошукових машинах Internet.

У інформаційно-пошукових системах застосовуються наступні методи пошуку:

a) індексування текстів і пошук за ключовими словами (по індексу);
b) пошук, що включає морфологічний розбір і ототожнення різних граматичних форм слів;
c) пошук з ранжируванням документів по мірі релевантности запиту;
d) використання формальних пошукових мов;
e) комплексних методів.

У технологіях БД і БЗ разом з перерахованими застосовуються наступні методи пошуку:

- ✧ використання формальних мов запитів, що дозволяють описувати умови спільного входження ключових слів в документ (цей напрям представляють SQL-подібні мови);
- ✧ методи семантичного аналізу тексту.

Засоби автоматичного витягання знань з текстових ресурсів Internet реалізуються в пошукових машинах. При цьому розрізняють:

a) методи ітеративного пошуку;
b) методи пошуку по вибірці;
c) методи, що використовують каталоги (рубрикатори і класифікатори, організуючі безліч документів в дерева або ліс);
d) семантичні методи пошуку, використовуючі підходи ИИ.

Створення засобів побудови і виконання запитів на ЕЯ є серйозною проблемою. До подібних пошукових механізмів, наприклад, відносяться:

- ✧ засоби, що реалізовують пошук типу "де";
- ✧ засоби пошуку особливих контекстних явищ;
- ✧ засоби виконання фактографічних запитів та ін.

Серед інтелектуального пошукового інструментарію для WWW слід згадати утиліту Echo Search фірми Iconovex Соф. Вона написана на мові Java і працює на базі платформ Windows і Macintosh. Ця утиліта позбавляє

користувача від необхідності розбиратися в результатах пошуку : вона аналізує вміст web-серверов, знайдене декількома засобами пошуку (використовуються шість популярних пошукових машин), і потім формує описи досліджених вузлів. Утиліта Echo Search створює текстові копії сторінок, що відповідають ключу пошуку, і записує їх на жорсткий диск. Проводячи лінгвістичний аналіз вмісту цих сторінок, Echo Search автоматично генерує покажчик, що коротко представляє зміст кожної сторінки.

Internet росте вибухоподібно, тому вірогідність наявності в нім необхідної інформації постійно збільшується. Для пошуку інформації в Internet служать різні класи пошукових засобів:

- ✧ каталоги (directories);
- ✧ підбірки посилань (bookmarks);
- ✧ пошукові машини (search engines);
- ✧ БД адрес електронної пошти (email addresses databases);
- ✧ засоби пошуку в архівах Gopher (Gopher archives);
- ✧ системи пошуку файлів (FTP search);
- ✧ системи пошуку новин (usenet news).

**Каталог ресурсів Internet**

Каталог ресурсів Internet – постійно оновлювана і поповнювана система посилань на ресурси, розподілені по ієрархічній структурі категорій. На верхньому рівні каталогу представлені найзагальніші категорії (рубрики), наприклад, “наука”, “бізнес”, “розваги” і т. д. На рівнях, що пролягають нижче, ці рубрики декомпозируются на підпорядковані рубрики, що мають більше приватний характер. Наприклад, верхній рівень каталогу mail.ru містить рубрики: “Автомобілі”, “Бізнес і фінанси”, “Навколо світу”, “Держава російська”, “домівка”, “Інтернет”, “Комп’ютери”, “Культура/Мистецтво”, “Медицина і здоров’я”, “непізнане”, “Освіта/Наука”, “Відпочинок”, “Робота і заробіток”, “ЗМІ”, “Спорт”, “Довідки”, “Товари і послуги”, “Гумор”.

На нижньому рівні каталогу вказуються посилання на конкретні ресурси Internet (сайти і Web-сторінки), забезпечені короткими описами їх вмісту.

Каталоги ресурсів Internet незамінні, коли людина має недостатньо точне уявлення про мету пошуку. Деякі з них дозволяють виконувати пошук за ключовими словами в короткому описі вмісту ресурсів.

Каталоги полегшують пошук за рахунок впорядкованості посилань на ресурси. Усі інтелектуальні функції залишаються за людиною. Те ж можна сказати про підбірки посилань на інформаційні ресурси Internet. Такі підбірки є відсортованими по темах адресами ресурсів.

Формування і актуалізація каталогів і підбірок посилань виконуються вручну персоналом тих, що відповідають ІС. Подібна робота вимагає високої кваліфікації і досить трудомістка.

Нижче перераховані деякі універсальні каталоги ресурсів Intemet:

- Yahoo! (http://www.yahoo.com);
- MSN (http://search.msn.com);
- AOL (http://search.aol.com);
- About (http://www.about.com);
- Search (http://www.search.com);
- Яndex (http://www.yandex.ru);
- Rambler (http://www.rambler.ru);
- Апорт (http://www.aport.ru);
- Город-ОК! (http://link.cid.ru);
- Пингвин (http://www.able.ru) и др.

Разом з універсальними існують і спеціалізовані каталоги, що систематизують відомості про ресурси Internet, що мають певну тематичну спрямованість. Наприклад:

- каталог громадських ресурсів Інтернет (некомерційних і громадських організацій, засобів масової інформації, електронних бібліотек і видань, БД, ІС, Intemet-проектов і т. інш.; http://www.ngo.ru);
- каталог внз Росії (http://www. 5ballov.ru/universities);
- каталог освітніх сайтів (http://www.allbest.ru).

**Пошукові системи**

Пошукові машини (чи пошукові системи) дозволяють знаходити ресурси Internet безпосередньо по їх текстовому вмісту. Функціонування пошукової машини включає два базові процеси:

- індексування ресурсів Internet (автоматична побудова і оновлення індексу);
- пошук по індексу по запитах користувачів.

У Міжнародному каталозі пошукових машин (Search Engine Colossus – http://www.searchenginecolossus.com) зареєстровано понад 2300 систем з 232 країн. За даними цього каталогу більше 80 % користувачів Internet знаходять інформаційні ресурси за допомогою пошукових машин, 57 % користувачів щодня застосовують пошукові машини, щодня виконується до 450 млн пошукових запитів, пошукові машини служать джерелом відомостей для 55 % усіх покупок в on-line.

До найбільш відомих пошукових машин відносяться:

- AltaVista (http://www.altavista.com);
- Excite (http://www.excite.com);
- HotBot (http://www.hotbot.com);
- Lycos (http://www.lycos.com);
- Yahoo! (http://www.yahoo.com);
- AOL (http://search.aol.com);
- MSN (http://search.msn.com);

- Infoseek (http://infoseek.go.com);
- Google (http://www.google.com.ru);
- About (http://www.about.com);
- Search (http://www.search.com);
- Dogpile (http://www.dogpile.com);
- Япёех (http://www.yandex.ru);
- Rambler (http://www.rambler.ru);
- Апорт (http://www.aport.ru);
- Rundex (http://www.rundex.ru).

Слід зазначити, що багато пошукових машин включають і каталоги ресурсів Internet.

Застосування пошукових машин для пошуку в Internet ефективно, якщо користувач представляє, які ключові слова характеризують потрібні ресурси.

Додаткові можливості надає режим розширеного пошуку, в якому можна задавати правила пошуку. Часто це значно збільшує вірогідність знаходження необхідної інформації.

Агент – самий інтелектуальний з компонентів пошукової машини. Він має автономність, має блоки навігація, що управляє "переміщенням" по мережі, і механізми індексації, засновані на деякій базі правил. Агенти реалізуються як прості програмні системи, що просять інформацію з вузлів Internet. Фізично по мережі агенти не переміщаються. Вони індексують отримані сторінки і заносять результати у БД.

Пошукові механізми відрізняються різноманітністю. Деякі агенти слідують по кожному посиланню на кожній знайденій сторінці і потім, у свою чергу, досліджують кожне посилання на новій сторінці і т. інш. Як правило, агенти ігнорують посилання до графічних і мультимедійних файлів, файлів з даними (наприклад, архівам), БД та ін. Ряд агентів переглядає сторінки з урахуванням їх популярності.

Різновидами агентів є кроулери, "роботи" і "павуки". Кроулери (crawlers) переглядають заголовки сторінок і повертають пошуковій машині тільки перше знайдене посилання. "Роботи" проходять по посиланнях різної глибини і вкладеності. "Павуки" (spiders) повідомляють про зміст знайденого документу, індексують його і пересилають витягнуту інформацію у БД пошукової машини.

Системою правил для усього цього співтовариства автономних програм управляють адміністратори пошукових машин. Вони ж встановлюють параметри алгоритмів визначення міри релевантности документу і запиту. Зазвичай в цих алгоритмах враховуються:

- кількість слів запиту в текстовому вмісті документу (т. е. у HTML-коді);
- теги, в яких ці слова трапляються;

– місце розташування шуканих слів в документі;
– питома вага слів, відносно яких визначається релевантность, в загальній кількості слів документу;
– час існування web-сайтів;
– індекс цитованості web-сайту та ін.

Засобом повнотекстового пошуку, орієнтованим на локальні інформаційні масиви і корпоративні мережі, служить інтелектуальна система "Слідопит" (розробник – компанія "Медиалингва"; http://www.sledopyt.ru). Вона дозволяє сформувати дерево зон пошуку у вигляді ієрархії тек з документами. Для кожної області може бути побудований окремий індекс.

Система обробляє документи у форматах DOC, DOT, RTF, XLS, PPT, TXT, HTML і PDF, документи цих форматів в ZIP-архивах, повідомлення електронної пошти Microsoft Outlook, а також вкладення в ці повідомлення в перерахованих форматах (у тому числі в ZIP-архивах). При індексуванні враховується морфологія російської і англійської мов. Передбачена можливість автоматичного оновлення індексів.

"Слідопит" здійснює пошук як за вмістом, так і за атрибутами документів. Запит представляється у вигляді фрази на ЕЯ або виражається на формальній мові з використанням логічних операторів. У запит можуть одночасно входити терміни російської і англійської мов.

### 3.1.3.3 Мови розмітки

Основою будь-якої web-сторінки є розмітка. Технології розмітки, такі як HTML, XHTML і XML, визначають структуру і можливе значення вмісту сторінки. Незважаючи на поширену думку про те, що мови розмітки визначають зовнішній вигляд web-сторінки, і не менш поширене застосування HTML в цьому стилі, зовнішній вигляд насправді повинен досягатися за допомогою інших технологій, зокрема, таблиць стилів.

### 3.1.3.4 Технології програмування

Зрозуміти основну ідею додавання програмування на сайт нескладно, проте можна з легкістю збитися з пантелику від числа технологій, з яких треба робити вибір, особливо якщо враховувати, що кожна з них індивідуальна. Насправді, технології web-програмування можна розділити на дві базисні групи: працюючі на стороні клієнта і на стороні сервера. Технології сторони клієнта – це ті, що запускаються клієнтом, в основному в контексті браузеру, хоча деякі технології на зразок аплетів Java або елементів управління ActiveX можуть насправді справляти враження роботи або дійсно працювати, за межами браузеру, а допоміжні застосування діють повністю таким чином. Поза сумнівом, замість цього програми можуть працювати і працюють на сервері і називаються відповідно – програмування на стороні сервера.

Знання різних аспектів мережевого середовища обов'язкове для web-дизайнерів, які прагнуть підвищити свій професійний рівень. Навіть якщо наголос робиться лише на створення зовнішнього інтерфейсу, дизайнерам слід мати хоч би поверхневе уявлення про різноманітні компоненти web-сайтів, починаючи від систем адресації і закінчуючи web-службами на базі XML. Деякі з цих компонетів, такі як web-браузери, HTML/XHTML, CSS, JavaScript і аудіовізуальні формати, мають бути вже дуже звичні; в той же час користь інших, ніби XML і мережевих протоколів, дизайнерам, працюючим з візуальними засобами, може здатися дуже сумнівною. Проте, з переходом від простого, орієнтованого на друк web-дизайна до більше інтерактивних, заснованих на програмних засобах web-сайтам, дизайнерам настійно рекомендується підвищувати свою кваліфікацію в області програмування і мережевих технологій.

**13 вересня**
**День програміста**
Неофіційне свято програмістів, що відмічається на 256-й день року. Число 256 (2 у восьмій мірі) вибрано тому, що ця кількість чисел, які можна виразити за допомогою одного байта. У високосні роки це свято випадає на 12 вересня, в звичайні – на 13 вересня. Відзначається за пропозицією Валентина Балта, співробітника однієї з веб-студій (він збирав підписи на підтримку визнання цього дня офіційним святом). Примітка: якщо вважати з 0 (що відповідає традиціям програмістів), то треба відмічати свято в 255-й день року.

## 3.2 Основи офісного програмування

### 3.2.1 Основні поняття алгоритмізації та програмування

#### 3.2.1.1 Поняття та основні властивості алгоритмів

Проектування алгоритмів і програм – найбільш відповідальний етап життєвого циклу програмного продукту, що визначає наскільки створювана програма відповідає специфікаціям і вимогам кінцевих користувачів.

У 820 році нашої ери в Бухарі був написаний підручник "Аль-Джабр Ва-аль-Мукабала" ("Наука виключення скорочення"), в якому були описані правила виконання чотирьох арифметичних дій над числами в десятковій системі числення. Автором підручника був арабський математик Мухаммед Бен Муса аль-Хорезмі. Від слова "альджебр", яке використовується у назві підручника, пішло слово "алгебра", а від імені аль-Хорезмі – слово "алгоризм", що пізніше перетворилося в слово "алгоритм".

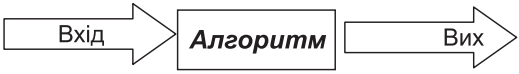
Точне математичне визначення алгоритму і вивчення цього поняття є предметом спеціальної математичної дисципліни – теорії алгоритмів, що у свою чергу, спирається на апарат математичної логіки.

Алгоритм – це деяке правило перетворення інформації, застосування якого до заданої інформації приводить до результату – нової інформації.

Алгоритм – це чітко визначена, для конкретного виконавця послідовність дій, які спрямовані на досягнення поставленої мети або розв'язання задачі певного типу.

Алгоритм – це скінчений набір інструкцій з перетворення інформації (команд), виконання яких приводить до результату. Кожна інструкція алгоритму містить точний опис деякої елементарної дії з перетворення інформації, а також (в явному або неявному вигляді) вказівку на інструкцію, яку необхідно виконувати наступною.

Інформацію, вихідну для алгоритму, прийнято називати його входом, а результат виконання – виходом.




***Приклад***. Алгоритм Евкліда: обчислення найбільшого спільного дільника цілих позитивних чисел *A* і *B: НСД(A, B)*.

**Алгоритм Евкліда;**
**Вхід A, B;**
**Вихід:** НСД**;**
**Початок**
  **Поки A ≠ B виконувати**
    **Якщо A < B**
    **то Обчислити B = B - A**
    **інакше Обчислити A = A - B;**
НСД **присвоїти значення A**
**Кінець.**

Таким чином, алгоритм описує деталізований покроковий процес перетворення інформації. Виконавець алгоритму не тільки виконує дії, але і запам'ятовує їхній результати. Для відображення цього факту в записі алгоритму ми використовуємо літерні позначення даних. Ці позначення називають іменами, а самі дані – величинами. Для позначення величин в алгоритмах використовуються імена.

Послідовний порядок виконання команд необхідний тільки для того, щоб результати виконання попередніх команд можна було використати як вихідні дані у команді, що виконується.

Поняттю алгоритму притаманні такі властивості:

***зрозумілість***. В алгоритмі повинні бути лише операції, які знайомі виконавцеві. При цьому виконавцем алгоритму може бути: людина, комп'ютер, робот тощо.

***масовість***. За допомогою складеного алгоритму повинен розв'язуватися цілий клас задач.

***однозначність***. Будь-який алгоритм повинен бути описаний так, щоб при його виконанні у виконавця не виникало двозначних вказівок. Тобто різні виконавці згідно з алгоритмом повинні діяти однаково та прийти до одного й того ж результату.

***правильність***. Виконання алгоритму повинно давати правильні результати.

***скінченність***. Завершення роботи алгоритму повинно здійснюється в цілому за скінчену кількість кроків.

***дискретність***. Алгоритм повинен складатися з окремих завершених операцій, які виконуються послідовно.

***ефективність***. Алгоритм повинен забезпечувати розв'язання задачі за мінімальний час з мінімальними витратами оперативної пам'яті.

Відзначимо ще одну важливу властивість алгоритму – він розрахований на виконавця, який розуміє і уміє виконувати команди алгоритму.

На практиці найбільш поширеними є наступні форми запису алгоритмів:

- ✧ у вигляді таблиць;
- ✧ графічний запис у формі блок-схем (графічний або геометричний спосіб запису алгоритму);
- ✧ словесний запис або псевдокоди (лексикографічний або словесно-крокoвий спосіб запису алгоритму);
- ✧ мова програмування, представлені алгоритмічною мовою у вигляді послідовності операторів (операторний спосіб).

*Словесна форма запису алгоритму* є описом на природній мові послідовних етапів обробки даних. Даний спосіб не має широкого поширення, оскільки такі описи строго не формалізуються, допускаються неоднозначність тлумачення окремих приписів. Алгоритм, записаний за допомогою псевдокоду, є напівформалізованим описом на умовній алгоритмічній мові, що включає як основні елементи мови програмування, так і фрази природної мови, загальноприйняті математичні позначення і інші.

*Графічна форма запису*, яку також називають схемою алгоритму, структурною схемою або блок-схемою алгоритму, є зображенням алгоритму у вигляді послідовності пов'язаних між собою функціональних блоків, кожен з яких відповідає виконанню одного або декількох дій. Всередині блоків записується їх короткий зміст. Графічний запис є компактнішим і наочнішим в порівнянні із словесною.

3.2.1.2 Блок-схеми та основні алгоритмічні конструкції

*Блок-схема* – це спосіб представлення алгоритму в графічній формі, у вигляді геометричних фігур, сполучених між собою лініями (стрілками). Форма блока визначає тип дії, а текст всередині блоку дає детальне пояснення конкретної дії. Стрілки на лініях, що сполучають блоки схеми, вказують послідовність виконання команд, передбачених алгоритмом. Блок-схеми, за рахунок наочності спрощують створення ефективних алгоритмів, розуміння роботи вже створених, а як наслідок і їх оптимізацію. Існуючі стандарти на типи блоків дозволяють легко адаптувати алгоритми, створені у вигляді блок-схем до будь-яких існуючих на сьогоднішній день мов програмування.

Зображення блоків у алгоритмі, їх розміри, товщина ліній, кут нахилу ліній тощо, регламентуються Державним стандартом "Схеми алгоритмів, програм, даних і систем", а саме : 19.701-90 (ISO 5807-85).

Блоки у блок-схемах з'єднуються лініями потоків. У кожен блок може входити не менше однієї лінії, з блоку ж (окрім логічного) може виходити лише одна лінія потоку. З логічного блоку завжди виходять дві лінії потоку: одна у випадку виконання умови, інша – при її невиконанні. Бажано, щоб лінії потоку не перетинались.

Алгоритм може бути детальним, або спрощеним (деякі зрозумілі блоки можуть не записуватись, інакше алгоритм збільшується в розмірі).

Якщо рішення задачі складне і досить довге, то алгоритм утворюється достатньо великим. Уникнути цього можна, замінивши деяку закінчену послідовність кроків алгоритму блоками, які будуть допоміжними алгоритмами. Блок зазвичай не елементарний, його розміри вибираються залежно від необхідності, проте якщо він правильно складений, то має усі необхідні ознаки алгоритмічного кроку: має точку входу (чіткий виділений початок) і може бути умовним або безумовним. Різні блоки алгоритму пов'язані один з одним тільки через точки входу і виходу, тому якщо блок вірно вирішує свою задачу, то його внутрішня структура несуттєва для іншої частини алгоритму. Таке блокове представлення особливо зручно на перших етапах рішення складних завдань, коли деталізація блоків робиться пізніше і, можливо, іншими розробниками.

У блок-схемах завжди є початок і кінець, що означають еліпсами, між ними – послідовність кроків алгоритму, сполучених стрілками. На рис. 3.30 представлено функціональні елементи блоксхем.

Кроки бувають *безумовними* (зображаються прямокутниками, паралелограмами) і *умовними* (зображаються ромбами). З ромба завжди виходять дві стрілки – одна означає подальший шлях, у разі виконання умови (позначається зазвичай словом "та" або "+"), інша – невиконання (словом

“ні” або “-”). Введення з клавіатури або вивід на екран значення вираження зображається паралелограмом. Команда, що виконує обробку дій (команда привласнення), зображається в прямокутнику.

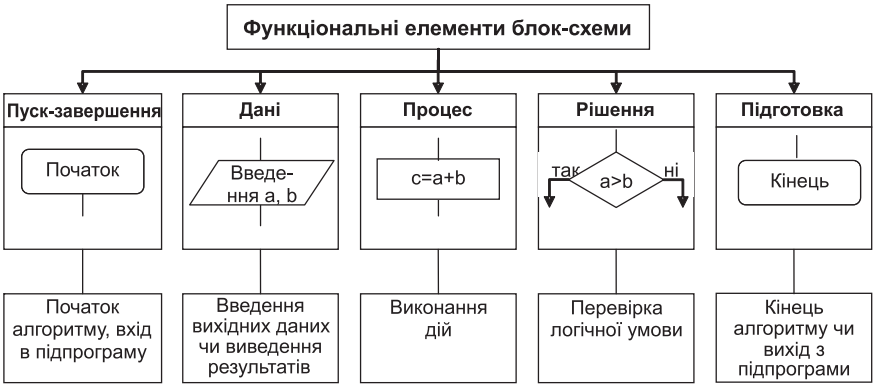



Рисунок 3.30. Функціональні елементи блок-схем

Основні елементи блок-схем представлені на рис. 3.31.

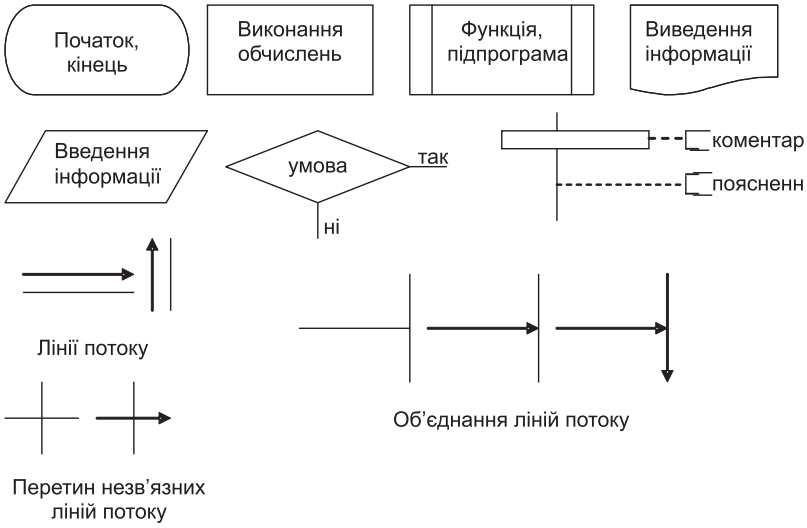
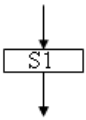



Рисунок 3.31. Основні елементи блок-схем

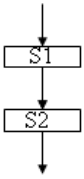
**Основні алгоритмічні конструкції:**

Структурною елементарною одиницею алгоритму є проста команда, що означає один елементарний крок переробки або відображення

інформації. Проста команда на мові схем зображається у вигляді функціонального блоку.

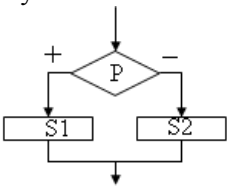



Цей блок має один вхід і один вихід. З простих команд і перевірки умов утворюються складені команди, що мають складнішу структуру і теж один вхід і один вихід.

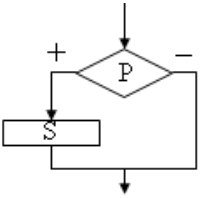



Структурний підхід до розробки алгоритмів визначає використання тільки базових алгоритмічних структур (конструкцій): дотримання, галуження, повторення, які мають бути оформлені стандартним чином.

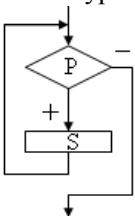
Розглянемо основні структури алгоритму.



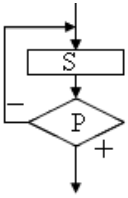

Команда ***дотримання*** складається тільки з простих команд. На малюнку прості команди мають умовне позначення S1 і S2. З команд дотримання утворюються лінійні алгоритми. Прикладом лінійного алгоритму буде знаходження суми двох чисел, введених з клавіатури.




Команда ***галуження*** – це складена команда алгоритму, в якій залежно від умови P виконується або одне S1, або інше S2 дія. З команд дотримання і команд галуження складаються алгоритми (алгоритми галуження), що розгалужуються. Прикладом такого алгоритму, буде знаходження більшого з двох чисел, введених з клавіатури.




Команда ***галуження*** може бути ***повної*** і ***неповної*** форми. *Неповна форма команди галуження* використовується тоді, коли необхідно виконувати дію S тільки у разі дотримання умови P. Якщо умова P не дотримується, то команда галуження завершує свою роботу без виконання дії. Прикладом команди галуження неповної форми буде зменшення в два рази тільки парного числа.




Команда ***повторення*** – це складена команда алгоритму, в якій залежно від умови P можливе багатократне виконання дії S. З команд дотримання і команд повторення складаються циклічні алгоритми (алгоритми повторення). На малюнку представлена команда повторення з передумовою. Називається вона так тому, що спочатку перевіряється умова, а вже потім виконується дія. Причому дія виконується, поки умова дотримується. Приклад циклічного алгоритму може бути наступний: поки з клавіатури вводяться позитивні числа, алгоритм виконує знаходження їх суми.

Команда ***повторення з передумовою*** не є єдино можливою. Різновидом команди повторення з передумовою є команда ***повторення з параметром***. Вона використовується тоді, коли відома кількість повторень

дії. У блок-схемі команди повторення з параметром умова записується не в ромбі, а в шестикутнику. Прикладом циклічного алгоритму з параметром буде знаходження суми перших 20 натуральних чисел.

У команді повторення з постумовою спочатку виконується дія S і лише потім, перевіряється умова P. Причому дія повторюється до тих пір, поки умова не дотримується. Прикладом команди повторення з постумовою буде зменшення позитивного числа до тих пір, поки воно ненегативне. Як тільки число стає негативним, команда повторення закінчує свою роботу.

За допомогою з'єднання тільки цих елементарних конструкцій (послідовно або вкладенням) можна "зібрати" алгоритм будь-якої складності (рис. 3.32).

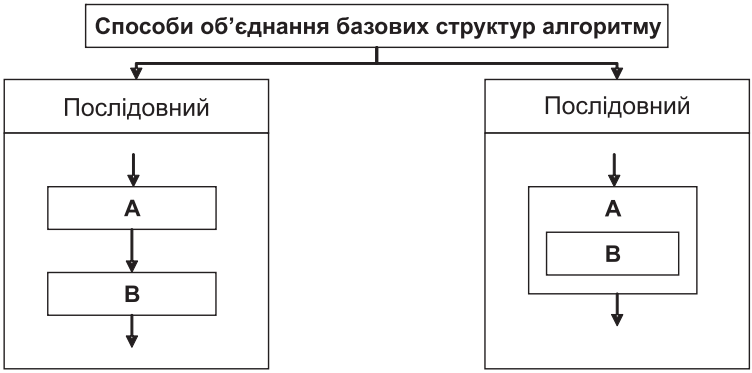



Рисунок 3.32. Способи обєднання структур алгоритму

Кожна вказана конструкція може бути без внесення змін в структуру реалізована на будь-якій мові програмування, наприклад, Паскалі чи Бейсику. Тому необхідно грамотно скласти алгоритм за допомогою блок-схеми, а вже потім, знаючи, як записуються команди на конкретній мові програмування, набрати програму на комп'ютері і отримати результат, запустивши її на виконання. Приклад запису алгоритму у вигляді блок-схеми, псевдокодів і на мові Паскаль (рис 3.33).

**Алгоритмічні конструкції** – це способи управління процесами обробки даних. Виділяють три базові алгоритмічні конструкції:

1) лінійні алгоритми;
2) алгоритми розгалуженої структури;
3) алгоритми циклічної структури.

***Лінійні алгоритми***. Алгоритм називається *лінійним*, якщо блоки алгоритму виконуються один за одним. Алгоритми лінійної структури не містять умовних і безумовних переходів, циклів (рис. 3.34).

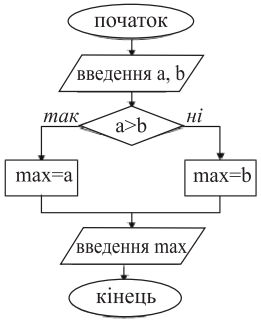
| Блок – схема | Псевдокоди | Pascal |
|---|---|---|
|  | **алг** більше з чисел<br>**вел** a, b, max<br>**поч**<br>**введення** a, b<br>**якщо** a>b<br>**то** max:=a<br>**інакше** max:=b<br>**все**<br>**виведення** max<br>**кін** | **program** ostatok;<br>**var** a, b, max: real;<br>**begin**<br>**readln** (a, b);<br>**if** a>b<br>**then** max:=a<br>**else** max:=b;<br>**writeln** (max)<br>**end**. |



Рисунок 3.33. Приклад запису алгоритму у вигляді блок-схеми, псевдокодів і на мові Паскаль

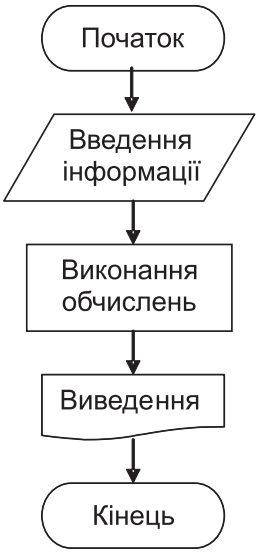



Рисунок 3.34. Приклад запису лінійного алгоритму

***Алгоритми розгалуженої структури***. Якщо обраний метод розв'язання задачі передбачає виконання різних дій в залежності від значень будь-яких змінних, але при цьому кожна гілка алгоритму в процесі розв'язання задачі виконується не більше одного разу, алгоритм називається *розгалуженим*. Приклади блок-схем наведено на рис. 3.35.

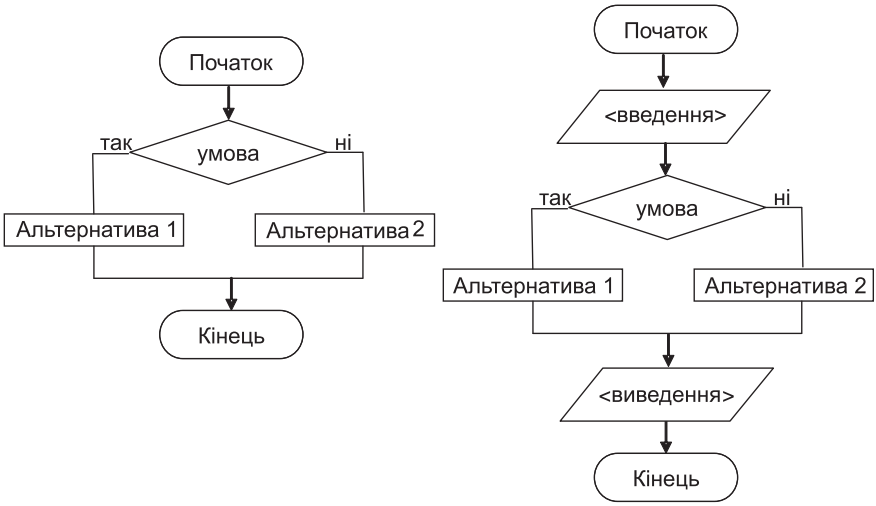



Рисунок 3.35. Приклад запису алгоритму, що розгалужується, для знаходження найбільшого з двох чисел

***Алгоритми циклічної структури***.

**Цикл** – це команда виконавцеві (компілятору) багаторазово повторити послідовність певних команд.

При багатократному проходженні деяких ділянок алгоритму в процесі виконання алгоритм називається *циклічним*. Кількість проходжень циклу повинна бути повністю визначена алгоритмом розв'язання задачі, інакше виникає "зациклювання", при якому процес розв'язання задачі не може завершитися.

Алгоритми розв'язку задач циклічної структури можуть бути такими, що при однократному проході циклу деякі ділянки алгоритму виконуються неодноразово, тобто всередині циклу існують інші цикли. Алгоритми такої структури називаються алгоритмами *з вкладеними циклами* (рис. 3.36).

Блок-схема складається з наступних базових структур: дві складені команди (команда дотримання і команда повторення з передумовою), далі проста команда. Усі команди сполучені послідовно. Конструкції оформлені стандартним чином, тому їх легко розпізнати і перекласти мовою програмування. Кожна конструкція має один вхід і один вихід.

Пунктирні стрілки в таблиці відбивають послідовність виконання технологічного ланцюжка. Після запису алгоритму у вигляді блок-схеми кожна команда перекладається шкільною алгоритмічною мовою, а вже потім на мову програмування.

Запишемо алгоритм обчислення суми перших n натуральних чисел. Для цього скористаємося циклом з параметром, оскільки заздалегідь відомо

скільки разів виконуватиметься команда знаходження суми. В усіх ланках ланцюжка змінемо цикл "доки" на цикл "для" і наведемо приклад перекладу алгоритму з мови блок-схем на шкільну алгоритмічну мову і на мову програмування Pascal (рис. 3.37).

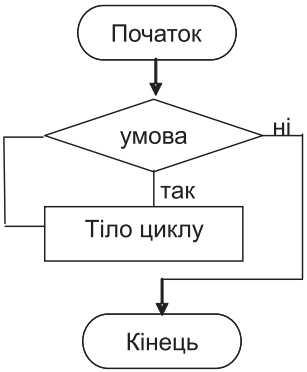



Рисунок 3.36. Приклад запису циклічного алгоритму

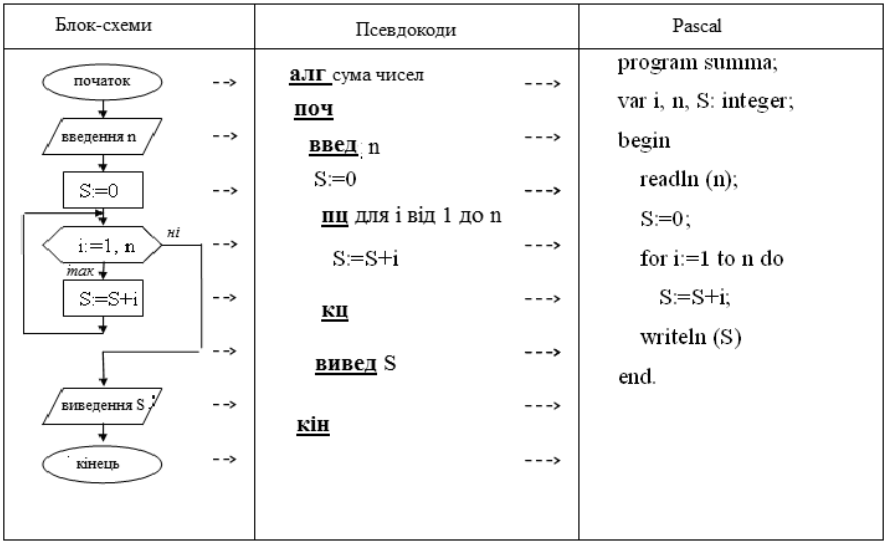



Рисунок 3.37. Технологія трансляції программ суми перших n натуральних чисел

Розглянемо знаходження кількості натуральних чисел, сума яких не більша за задану. Для цього скористаємося командою повторення з по-

стумовою. Приклад перекладу алгоритму з мови блок-схем на шкільну алгоритмічну мову і на мову програмування Pascal представлена на рис. 3.38.

Створений алгоритм вирішення задачі слід перекласти на мову зрозумілу ЕОМ. Існує значна кількість таких мов – C/C++, JAVA, PASCAL, VB та інші. Вони називаються *мовами програмування*.

**Мова програмування** – мова, яка використовується для формального запису алгоритмів. Більшість мов програмування відносяться до алгоритмічних мов. Запис алгоритму на алгоритмічній мові називають програмою, а процес перекладу на дану мову – ***програмуванням***.

Мова, для формального запису алгоритмів, називається ***алгоритмічною мовою***. При описі будь-якої мови (у тому числі природньої, наприклад, російського, англійського і так далі) використовуються наступні поняття: *алфавіт*, *синтаксис* і *семантика*.

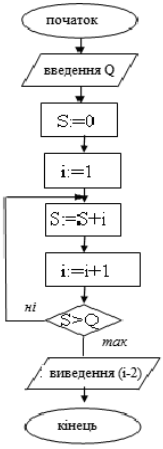
| Блок-схеми | Псевдокоди | Pascal |
|---|---|---|
|  | **алг** кількість значень<br>**вел ціл** i, S, Q<br>**введ** Q<br>S:=0<br>i:=1<br>**викон**<br>S:=S+i<br>i:=i+1<br>**до** S>Q<br>**вивед** (i-2)<br>**кін** | **program** kolitchestvo;<br>**var** i, S, Q: integer;<br>**begin**<br>**readln** (Q);<br>S:=0;<br>i:=1;<br>**repeat**<br>S:=S+i;<br>i:=i+1<br>**until** S>Q;<br>**writeln** (i-2)<br>**end**. |

Рисунок 3.38. Технологія трансляції програми обчислення кількості натуральних чисел, сума яких не більша за задану

**Алфавіт мови** – це безліч простих знаків, які можуть бути використані в текстах цієї мови. Послідовність символів алфавіту називають ***словом***. Правила, згідно з якими утворюються слова з алфавіту, називаються ***граматикою***. Сама ж мова – це безліч усіх слів, записуваних в цьому алфавіті згідно з цією граматикою.

**Синтаксис** – це набір правив, що визначають можливі поєднання (конструкції) з букв алфавіту. Для опису синтаксису мови, як правило, використовують іншу мову (метамову) або синтаксичні діаграми.

**Семантика** – це набір правил, визначальних значення (сенс) окремих конструкцій мови.

Однією з найпоширеніших алгоритмічних мов є мова Pascal, яка корисна як для початківців, так і для досвідчених програмістів. Навчання програмуванню найчастіше грунтується на цій мові.

Взагалі ЕОМ не розрахована на те, щоб розуміти мови програмування. Апаратура розпізнає і виконує тільки машинну мову, програма на якій являє собою не більш ніж послідовність двійкових чисел.

Поява мов програмування була пов'язана з осмисленням того факту, що перекладання алгоритму, написаного "майже" природною мовою, на машинну мову може бути автоматизований і, отже, покладений на плечі машини. Тут важливо розрізняти мову і її реалізацію. Сама ***мова*** – це система запису, яка регламентується набором правил, що визначають його лексику і синтаксис. **Реалізація мови** – це програма, яка перетворює цей запис у послідовність машинних команд у відповідності з прагматичними і семантичними правилами, визначеними у мові.

**Система команд виконавця** – точно описана обстановка, що включає формулювання вирішуваної задачі, перелік об'єктів, що залучаються до умови завдання і в її рішення, і можливості виконавця: властивості об'єктів, які він може упізнати і дії, які він може зробити. Формальне виконання алгоритму створюють засоби реалізації: компілятор або інтерпретатор, перевіряючи семантику.

**Компілятори** транслюють весь текст програми, написаний мовою програмування, у машинний код у ході одного безперервного процесу. При цьому створюється повна програма у машинних кодах, яку потім можна виконувати без участі компілятора.

**Інтерпретатор** у кожний момент часу розпізнає і виконує по одному реченню (оператору) програми, за ходом справи перетворюючи речення, яке обробляється у машинну програму. Різниця між компілятором і інтерпретатором подібна різниці між синхронним перекладом усної мови і письмовим перекладом тексту.

У принципі, будь-яка мова може бути і компілюємою, і інтерпретуємою, однак у більшості випадків кожна мова має засіб реалізації, якому віддають перевагу. Кожний з цих засобів має як свої переваги, так і недоліки:

**Основні переваги компіляції:**

- ✧ швидкість виконання готової програми;
- ✧ незалежність програми від системи реалізації.

**Основні недоліки компіляції:**

- ✧ деякі незручності, які перевіряються програмістом при написанні і налагодженні великих програм;
- ✧ порівняно великий об'єм пам'яті, який займає компілятор в ОЗП.

Для роботи в мові програмування використовуються спеціальні пакети програм, які називають ***системами програмування***. У їх склад входять:

- ✧ оболонка;
- ✧ текстовий редактор;
- ✧ компілятор;
- ✧ редактор зв'язків;
- ✧ завантажувач;
- ✧ налагоджував;
- ✧ бібліотеки стандартних процедур і функцій;
- ✧ сервісні програми.

3.2.1.3 Основні етапи проектування програми

Процес проектування програми подібний процесам проектування складних систем. До його основних етапів входить:

- ✧ постановка задачі і вибір її математичної моделі;
- ✧ розробка алгоритму розв'язування задачі;
- ✧ вибір апаратного обладнання і мови програмування;
- ✧ написання програми;
- ✧ налагодження і редагування;
- ✧ контрольні випробування;
- ✧ експлуатація.

Насправді проектування програми – не лінійний, а циклічний процес, на кожному кроці якого можливе повернення до будь-якого з попередніх кроків.

### 3.2.2 Розвиток мов програмування

3.2.2.1 Розвиток мов програмування

***Покоління мов програмування***

Мови програмування прийняті ділити на п'ять поколінь. У перше покоління входять мови, створені на початку 50-х років, коли тільки з'явилися перші комп'ютери. Це була перша мова асемблера, створена за принципом "одна інструкція – один рядок".

Розквіт другого покоління мов програмування припав на кінець 50-х – початок 60-х років. У той час було розроблено символічний асемблер, у якому з'явилося поняття змінної. Він став першою повноцінною мовою програмування. Завдяки його виникненню помітно зросли швидкість розробки й надійність програм.

Поява третього покоління мов програмування прийнято відносити до 60-х років. У цей час виникли універсальні мови високого рівня, з їхньою допомогою вдається вирішувати завдання з будь-яких областей. Такі якості нових мов, як відносна простота, незалежність від конкретного комп'ютера й можливість використання потужних синтаксичних конструкцій, до-

зволили різко підвищити продуктивність праці програмістів. Зрозуміла більшості користувачів структура цих мов залучила до написання невеликих програм (як правило, інженерного або економічного характеру) значне число фахівців з некомп'ютерних областей. Переважна більшість мов цього покоління успішно застосовується й сьогодні.

З початку 70-х років по теперішній час триває період мов четвертого покоління. Ці мови призначені для реалізації великих проектів, підвищення їхньої надійності й швидкості створення. Вони звичайно орієнтовані на спеціалізовані області застосування, де отримати результати можливо домогтися лише використовуючи не універсальні, а проблемно-орієнтовані мови, що оперують конкретними поняттями вузької предметної області. Як правило, у ці мови вбудовуються потужні оператори, що дозволяють одним рядком описати таку функціональність, для реалізації якої на мовах молодших поколінь потрібні були б тисячі рядків вихідного коду.

Народження мов п'ятого покоління відбулося в середині 90-х років. До них ставляться також системи автоматичного створення прикладних програм за допомогою візуальних засобів розробки, без знання програмування. Головна ідея, що заставляється в ці мови – можливість автоматичного формування результуючого тексту на універсальних мовах програмування (який потім потрібно відкомпілювати). Інструкції вводяться в комп'ютер у максимально наочному виді за допомогою методів, найбільш зручних для людини, не знайомого із програмуванням.

***Огляд мов програмування високого рівня***

**Fortran (Фортран)**. Це перша компільована мова, створена Джимом Бэкусом в 50-ті роки. Програмісти, які розробляли програми виключно на асемблері, виражали серйозний сумнів в можливості появи високопродуктивної мови високого рівня, тому основним критерієм при розробці компіляторів Фортрану була ефективність коду, який виконується. Хоча у Фортрані уперше були реализовано ряд найважливіших зрозуміти програмування, зручність створення програм була принесена в жертву можливості отримання ефективного машинного коду. Проте для цієї мови була створена величезна кількість бібліотек, починаючи від статистичних комплексів і кінчаючи пакетами управління супутниками, тому Фортран продовжує активно використовуватися у багатьох організаціях. Є стандартна версія Фортрану HPF (High Performance Fortran) для паралельних суперкомп'ютерів з безліччю процесорів.

**Cobol (Кобол)**. Це компільована мова для застосування в економічній області і рішення бізнес-завдань, розроблений на початку 60-х років. Він відрізняється великою "багатослівністю" – його оператори іноді виглядають як звичайні англійські фрази. У Коболі були реалізовані дуже потужні засоби роботи з великими об'ємами даних, які зберігаються на різних зовнішніх носіях. На цій мові створено дуже багато доповнень, які активно

експлуатуються і сьогодні. Досить сказати, що найбільшу зарплату в США отримують програмісти на Коболі.

**Algol (Алгол)**. Компільована мова, створена в 1960 р. Він був покликаний замінити Фортран, але через більше складну структуру не отримав широкого поширення. У 1968 р. була створена версія Алгол 68, за своїми можливостями вона і сьогодні випереджає багато мов програмування, проте через відсутність досить ефективних комп' ютерів для неї не вдалося вчасно створити ефективні компілятори.

**Pascal (Паскаль)**. Мова Паскаль, створена наприкінці 70-х років основоположником безлічі ідей сучасного програмування Никлаусом Виртом, багато в чому нагадує Алгол, але в ньому посилений ряд вимог до структури програми і є можливості, що дозволяють успішно застосовувати його при створенні великих проектів.

**Basic (Бейсик)**. Для цієї мови є й компілятори, і інтерпретатори, а по популярності він посідає перше місце у світі. Він створювався в 60-х роках як навчальна мова й дуже проста у вивченні.

**C (Сі)**. Дана мова була створена в лабораторії Bell і спочатку не розглядалася як масова. Вона планувалася для заміни асемблера, щоб мати можливість створювати настільки ж ефективні й компактні програми, і в той же час не залежати від конкретного типу процесора. Сі багато в чому схожа на Паскаль і має додаткові засоби для прямої роботи з пам'яттю (покажчики). На цій мові в 70-ті роки написаний безліч прикладних і системних програм і ряд відомих операційних систем (Unix).

**C++ (Сі++)**. Сі++ представляє собою об'єктноорієнтоване розширення мови Сі, створене Бьярном Страуструпом в 1980 р. Значана кількість нових потужних можливостей, які дозволили різко підвищити продуктивність програмістів, наклалася на успадковану від мови Сі певну низкорівність, внаслідок чого створення складних і надійних програм зажадало від розроблювачів високого рівня професійної підготовки.

**Java (Ява)**. Ця мова була створена компанією Sun на початку 90-х років на основі Сі++. Він покликаний спростити розробку доповнень на основі Сі++ шляхом виключення з нього усіх низькорівневих можливостей. Але, головна особливість цієї мови – компіляція не в машинний код, а в платформно-незалежний байт-код (кожна команда займає один байт). Цей байт-код може виконуватися за допомогою інтерпретатора – віртуальною Java-Машини JVM (Java Virtual Machine), версії якої створені сьогодні для будь-яких платформ. Завдяки наявності величезної кількості Java-машин програми на Java можна переносити не лише на рівні початкових текстів, але і на рівні двійковий байт-коду, тому за популярністю мову Ява сьогодні займає друге місце у світі після Бейсика.

Особлива увага в розвитку цієї мови приділяється двом напрямам: підтримці всіляких мобільних пристроїв і мікрокомп'ютерів, які вбудовуються в побутову техніку (технологія Jini) і створенню платформно-

незалежних програмних модулів, здатних працювати на серверах в глобальних і локальних мережах з різними операційними системами (технологія Java Beans). Назазі основнимй недоліком мови є невисока швидкодія, тому що мова Ява що інтерпретується.

***Мови програмування баз даних***

Ця група мов відрізняється від алгоритмічних мов, насамперед, розв'язуваними завданнями. База даних – це файл (або група файлів), що представляє собою впорядкований набір записів, що мають однакову структуру й організованих за єдиним шаблоном (як правило, у табличному виді). База даних може складатися з декількох таблиць. Зручно зберігати в базах даних різні відомості з довідників, картотек, журналів бухгалтерського обліку й т.д.

При роботі з базами даних найчастіше потрібно виконувати наступні операції:

- ✧ створення/модифікація властивостей/видалення таблиць у базі даних;
- ✧ пошук, відбір, сортування інформації із запитів користувачів;
- ✧ додавання нових записів;
- ✧ модифікація існуючих записів;
- ✧ видалення існуючих записів.

Перші бази даних з'явилися дуже давно, як тільки з'явилася потреба в обробці більших масивів інформації й вибірки груп записів за певними ознаками. Для цього була створена структурована мова запитів SQL (Structured Query Language). Він заснований на потужній математичній теорії й дозволяє виконувати ефективну обробку баз даних, маніпулюючи не окремими записами, а групами записів.

Для керування більшими базами даних і їхньої ефективної обробки розроблені СУБД (Системи Управління Базами Даних). Практично в кожної СУБД крім підтримки мови SQL є також своя унікальна мова, орієнтована на особливості цієї СУБД, без можливості поширення на інші системи. Сьогодні у світі налічується п'ять ведучих виробників СУБД: Microsoft (SQL Server), IBM (DB2), Oracle, Software AG (Adabas), Informix і Sybase. Їх продукти націлені на підтримку одночасної роботи тисяч користувачів у мережі, а бази даних можуть зберігатися в розподіленому виді на декількох серверах. В Oracle є убудована мова PL/SQL, в Informix – INFORMI 4GL, в Adabas – Natural і т.д.

З появою персональних комп'ютерів були створені так звані настільні СУБД. Родоначальником сучасних баз даних для ПК прийнято вважати СУБД dBase II, мова якої була такою, що інтерпретується. Потім для нього були створені компілятори, з'явилися СУБД FoxPro і Clipper, що підтримують діалекти цієї мови. Сьогодні схожі, але несумісні версії сімейства dBase реалізовані в продуктах Visual FoxPro фірми Microsoft і Visual dBase фірми Inprise.

***Мови програмування для Інтернету***

З активним розвитком глобальної мережі було створено чимало популярних мов програмування, адаптованих спеціально для Інтернету. Всі вони відрізняються характерними рисами: мови є такими, які не інтерпритуються, інтерпретатори для них поширюються безкоштовно, а самі програми – у вихідних текстах. Такі мови називають скрипт-мовами.

**HTML**. Загальновідома мова для оформлення документів. Вона є простою і містить елементарні команди форматування тексту, додавання малюнків, завдання шрифтів і кольорів, організації посилань і таблиць. Всі web-сторінки написані мовою HTML або використовують його розширення.

**Perl**. У 80-х роках Ларри Уолл розробив мову Perl. Вона задумувалась як засіб ефективної обробки великих текстових файлів, генерації текстових звітів і управління завданнями. За потужністю Perl значно перевершує мови типу Сі. В ній введено багато функцій роботи з рядками, масивами, усілякі засоби перетворення даних, управління процесами, роботи з системною інформацією та ін.

**Tcl/Tk**. Наприкінці 80-х років Джон Аустираут розробив популярну скрипт-мову Tcl і бібліотеку Tk. У Tcl він спробував втілити бачення ідеальної скрипт-мови. Tcl орієнтований на автоматизацію рутинних процесів і складається з потужних команд, призначених для роботи з абстрактними нетипізованими об'єктами. Він незалежний від типу системи й при цьому дозволяє створювати програми із графічним інтерфейсом.

**VRML**. В 1994р. була створена мова VRML для організації віртуальних тривимірних інтерфейсів в Інтернеті. Вона дозволяє описувати в текстовому виді різні тривимірні сцени, висвітлення й тіні, текстури (покриття об'єктів), створювати свої міри, подорожувати по них, "облітати" з усіх боків, обертати в будь-яких напрямках, масштабувати, регулювати освітленість і т.д.

***Мови моделювання***

При створенні програм і формуванні структур баз даних нерідко застосовуються формальні способи їх представлення – формальні нотації, за допомогою яких можна візуально представити (зображувати за допомогою миші) таблиці баз даних, об'єкти програми і взаємозв'язку між ними в системі, що має спеціалізований редактор і генератор початкових текстів програм на основі створеної моделі. Такі системи називаються CASE-системами. У них активно застосовуються нотації IDEF, а останнім часом все більшу популярність завойовує мова графічного моделювання UML.

***Інші мови програмування***

**PL/1 (ПЛ/1)**. В середині 60-х років компанія IBM вирішила обєднати досягнення мов Фортран, Кобол і Алгол. У результаті в 1964 р. на світ з'явилася нова компільована мова програмування, яка дістала назву

Programming Language One. У цій мові була реалізована величезна кількість унікальних рішень, корисність яких вдалося оцінити тільки через десятки років, в епоху великих програмних систем. За своїми можливостями ПЛ/1 значно перевагає більшість інших мов (С, Паскаль). Наприклад, в ПЛ/1 є присутньою унікальна можливість вказівки точності обчислень, що відсутнє у Сі++ і Яви. Ця мова і сьогодні продовжує підтримуватися компанією IBM.

**Smalltalk (Смолток)**. Робота над даною мовою почалася в 1970 р. у дослідницькій лабораторії корпорації XEROX, а закінчилися через 10 років, втілившись в остаточному варіанті інтерпретатора SMALLTALK-80. Дана мова оригінальна тим, що її синтаксис компактний і базується винятково на понятті об'єкта. У цій мові відсутні оператори або дані. Усе, що входить у Смолток, є об'єктами, а самі об'єкти спілкуються один з одним винятково за допомогою повідомлень (наприклад, поява вираження Й+1 викликає посилку об'єкту Й повідомлення "+", тобто "додати", з параметром 1, що вважається не числом – константою, а теж об'єктом). Більше ніяких керуючих структур, за винятком "оператора" розгалуження (насправді функції, що належить стандартному об'єкту), у мові відсутня, хоча їх можна дуже просто змоделювати. Сьогодні версія VisualAge for Smalltalk активно розвивається компанією IBM.

**LISP (Ліспі)**. Мова програмування, що інтерпретується, створена в 1960г. Джоном Маккарти. Орієнтована на структуру даних у формі списку і дозволяє організовувати ефективну обробку великого обсягу текстової інформації.

**Prolog (Пролог)**. Створений на початку 70-х років Аланом Колмерое. Програма на даній мові, в основу якої покладена *математична модель теорії вирахування предикатів*, будується з послідовності фактів і правил, а потім формулюється твердження, що Пролог буде намагатися довести за допомогою уведених правил. Людина тільки описує структуру завдання, а внутрішній "мотор" Прологу сам шукає рішення за допомогою методів пошуку й зіставлення.

**Ada (Ада)**. Названий на ім'я леді Августи Ади Байрон, дочки англійського поета Байрона і його віддаленої родички Анабелли Милбэнк. У 1980 р. сотня експертів Міністерства оборони США відібрали з 17 варіантів саме цю мову, розроблену невеликою групою під керівництвом Жана Ішбіа. Вона задовольнилав на той час усі вимоги Пентагону, а до сьогоднішнього дня в його розвиток вкладені десятки мільярдів доларів. Структура самої мови схожа на Паскаль. Вона містить засоби строгого розмежування доступу до різних рівнів специфікацій, доведена до межі потужність керуючих конструкцій.

**Forth (Форт)**. Результат спроби Чарльза Мура в 70-х роках створити мову, яка володіє потужними засобами програмування, яка могла би бути

ефективно реалізованим на комп'ютерах з невеликими об'ємами пам'яті, а компілятор міг би видавати дуже швидкий і компактний код, тобто служити заміною асемблеру. Проте, складнощі сприйняття програмного тексту, записаного в незвичайній формі, досить ускладнює пошук помилок і з появою Сі мова Форт виявився забутою.

3.2.2.2 Сучасні системи програмування

***Основи візуального програмування інтерфейсу***

Скільки існує програмування, стільки існує в ньому і безвихідь, в яку воно постійне впадає і з яких зрештою виходить. Одним з таких безвиходів, або криз, не дуже давно був пов'язаний з розробкою графічного інтерфейсу користувача. Програмування вручну, тобто без використання звичних користувачеві вікон, кнопок, меню, обробки подій, миші і клавіатури, включення в програми зображень і звуку вимагав усе більш і більше часу програміста. У ряді випадків увесь цей сервіс починав займати до 80-90% об'єму програмних кодів. Причому, в деяких випадках, уся ця праця нерідко зводилася нанівець, оскільки через рік – інший змінювався загальноприйнятий стиль графічного інтерфейсу і усе доводилося починати знову.

Вихід з цієї ситуації позначився завдяки двом підходам. Перший – стандартизація багатьох функцій інтерфейсу, завдяки чому з'явилася можливість використовувати бібліотеки, наявні, наприклад, в Windows. У результаті при зміні стилю графічного інтерфейсу (наприклад, при переході від Windows 3.x до Windows 95) доповнення змогли автоматично пристосовуватися до нової системи без якого-небудь перепрограмування. На цьому шляху створилися прекрасні умови для вирішення одного з найважливіших завдань вдосконалення техніки програмування – повторного використання кодів. Один раз розроблені форми, компоненти, функції могли бути згодом неодноразово використані для вирішення різних завдань. Кожен програміст отримав доступ до розработкам інших програмістів і до величезних бібліотек, створених різними фірмами. Причому була забезпечена сумісність програмного забезпечення, розробленого на різних алгоритмічних мовах.

Другим революційним кроком, що кардинально полегшив життя програмістів, з'явилася поява візуального програмування, що виникло в Visual Basic і нашли блискуче втілення в Delphi й C++Builder фірми Borland.

Візуальне програмування дозволило звести проектування користувальницького інтерфейсу до простих і наочних процедур, які дають можливість за мінути або годинники зробити те, на що раніше йшли місяці роботи. У сучасному виді в Delphi це виглядає так.

Ви працюєте в Інтегрованому Середовищі Розробки (ICP або Integrated development environment – IDE) Delphi. Середовище надає вам форми (у

додатку їх можуть бути трохи), на яких розміщаються компоненти. Звичайно це віконна форма, хоча можуть бути й невидимі форми. На форму за допомогою миші переносяться й розміщаються піктограми компонентів, наявних у бібліотеках Delphi. За допомогою простих маніпуляцій ви можете змінювати розміри й розташування цих компонентів. При цьому ви увесь час у процесі проектування бачите результат – зображення форми й розташованих на ній компонентів. Вам не треба багаторазово запускаючи додаток і вибираючи найбільш удалі розміри вікна й компонентів. Результати проектування ви бачите, навіть не компілюючи програму, негайно після виконання якоїсь операції за допомогою миші.

Але переваги візуального програмування не зводяться до цього. Саме головне полягає в тім, що під час проектування форми й розміщення на ній компонентів Delphi автоматично формує коди програми, включаючи в неї відповідні фрагменти, що описують даний компонент. А потім, у відповідних діалогових вікнах, користувач може змінити задані за замовчуванням значення якихось властивостей цих компонентів і, при необхідності, написати оброблювачі якихось подій. Фактично, проектування зводиться до розміщення компонентів на формі, завданню деяких їхніх властивостей і написанню, при необхідності, оброблювачів подій.

Компоненти можуть бути візуальні та не візуальні. Візуальні компоненти відразу відображуються на екрані в процесі проектування в такому ж виді, у якому їх побачить користувач під час виконання додатка. Це дозволяє дуже легко вибрати місце їхнього розташування і їхній дизайн – форму, розмір, оформлення, текст, колір і т.д. Невізуальні компоненти не відображені на формі в процесі проектування у вигляді піктограм, але користувачеві під час виконання вони не видні, хоча й виконують для нього за кадром досить корисну роботу.

У бібліотеки візуальних компонентів Delphi включена безліч типів компонентів і їхня номенклатура дуже швидко розширюється від версії до версії. Наявного вже зараз цілком достатньо, щоб побудувати практично будь-який складний додаток, не прибігаючи до створення нових компонентів. При цьому навіть недосвідчений програміст, може створювати додатки, які виглядають зовсім професійно.

Компоненти досить легко створювати самостійно, тому у світі сьогодні поширюються тисячі безкоштовних і платних компонентів для найбільш відомих RAD-середовищ, з яких формуються бібліотеки компонентів – об'єктні репозиторії. Компоненти виступають у ролі "будівельних цеглинок", що дозволяють збирати готовий додаток з багатими можливостями, написавши всього десяток рядків вихідного коду, і такий компонентний підхід до створення програм уважається дуже перспективним, тому що без зайвих зусиль і на законних підставах допускає повторне використання чужої праці.

***Системи швидкої розробки додатків***

Візуальне обєктно-орієнтоване програмування сприяло створенню технології, що одержала назву "швидка розробка додатків" – RAD. Ця технологія характерна для нового покоління систем програмування, до якого ставиться й Delphi.

Першої в цьому новому світі більше простого й наочного інтерфейсу було середовище Visual Basic. Вона сформувала новий стиль взаємодії розроблювача програми з комп'ютером, що дозволяє наочно конструювати користувальницький інтерфейс за допомогою миші, а не звичайним для колишніх часів шляхом: написанням кодів, їхньою трансляцією й виконанням програми, після чого тільки й можна переглянути, як це виглядає на екрані.

Хоча Visual Basic мав широкий попит і допоміг відкрити світ програмуванню для людей, які володіють лише певними навиками, він не розв'язує багато проблем. Головні з них – низька продуктивність розроблювальних додатків при їхньому виконанні, недостатня строгість і об'єктна орієнтованість мови, що сприяє швидкій розробці виробів, а не створенню потужних ефективних додатків, а також ряд інших недоліків.

Delphi – це наступний крок у розвитку середовища швидкої розробки додатків. Вона виправляє багато дефектів, виявлені в Visual Basic. Розроблювачі Delphi створили інструмент, що на перший погляд схожий на середовище Visual Basic, хоча в дійсності він помітно кращий.

Delphi базується мовою Object Pascal. Компілятори з мов сімейства Паскаль фірми Borland (починаючи з Turbo Pascal 1.0) були одними з найшвидших компіляторів. У цей час Object Pascal – це объектно-орієнтована мова із твердою опорою у вигляді компілятора.

Треба мати на увазі, що орієнтація додатків Delphi на Object Pascal не звужує можливостей розробника. Додатка Delphi можуть бути використані розробки на інших мовах, включаючи C++ і навіть асемблер. Можна використовувати бібліотеки, створені іншими фірмами, зокрема, Microsoft або незалежними розроблювачами. Можна реалізовувати свої розробки у вигляді самостійно виконуваних файлів або у вигляді пакетів, що підтримують виконання ряду додатків.

Окремо треба сказати про одну з головних завдань Delphi – розробка додатків для роботи з базами даних. У цій області Delphi займає одну з передових позиції, працюючи з будь-якими системами керування базами даних.

***Основні системи програмування***

З універсальних мов програмування сьогодні найбільш популярні наступні:

Бейсик (Basic) – для освоєння вимагає початкової підготовки (загальноосвітня школа);

Паскаль (Pascal) – вимагає спеціальної підготовки (школи з поглибленим вивчанням предмета й загальнотехнічні ВУЗ);

Си++ (С++), Ява (Java) – вимагають професійної підготовки (спеціалізовані середні й вищі навчальні заклади).

Для кожної із цих мов програмування сьогодні є чимало систем програмування, що випускаються різними фірмами й орієнтованих на різні моделі ПК і операційні системи. Найбільш популярні наступні візуальні середовища швидкого проектування програм для Windows:

- Basic: Microsoft Visual Basic;
- Pascal: Borland Delphi;
- C++: Borland C++Bulider;
- Java: Symantec Cafe.

Для розробки серверних і розподілених додатків можна використовувати систему програмування Microsoft Visual Basic, C++, продукти фірми Inprise під маркою Borland, практично будь-які засоби програмування на Java.

### 3.2.2.3 Архітектура програмних систем

Тоді як більшість автономних доповнень – офісні програми, середовища розробки, системи підготовки текстів і зображень, виконуються на одному комп'ютері, великі інформаційні комплекси (наприклад, система автоматизації підприємства) складаються з десятків і сотень окремих програм, які взаємодіють один з одним за допомогою мережі, виконуючись на різних комп'ютерах. У таких випадках говорять, що вони працюють в різній програмній архітектурі. Вона ділиться на наступні групи:

- ✧ автономні додатки, які працюють на одному комп'ютері.
- ✧ доповнення у файл – серверній архітектурі. Комп'ютери користувачів системи об'єднані в мережу, при цьому на кожному з них (на клієнтському місці) запущені копії однієї і тієї ж програми, які звертаються за даними до сервера – спеціального комп'ютера, який зберігає файли, одночасно доступні усім користувачам (як правило, це бази даних). Сервер має підвищену надійність, високу швидкодію, великий об'єм пам'яті, на нім встановлена спеціальна серверна версія операційної системи.

При одночасному обігу декількох програм до одного файлу, наприклад, з метою його відновлення, можуть виникнути проблеми, пов'язані з неоднозначністю визначення його вмісту. Тому кожна зміна загальнодоступного файлу виділяється в транзакцію – елемнтарну операцію з обробки даних, що має фіксований початок, кінець (успішне або неуспішне завершення) і ряд інших характеристик.

Особливість цієї архітектури в тім, що всі обчислення виконуються на клієнтських місцях, що вимагає наявності на них досить продуктивних

ПК (це так звані системи з товстим клієнтом – програмою, що виконує всю обробку одержуваної від сервера інформації).

– Додатка в клієнт – серверна архітектура. Ця архітектура схожа на попередню, тільки сервер крім простого забезпечення одночасного доступу до даних здатний ще виконувати програми (звичайно виконуються СУБД – тоді сервер називається сервером баз даних), які беруть на себе певний обсяг обчислень (у файл – серверній архітектурі він реалізується повністю на клієнтських місцях). Завдяки цьому вдається підвищити загальну надійність системи, тому що сервер працює значно більш стійко, ніж ПК, і зняти зайве навантаження із клієнтських місць, на яких вдається використовувати дешеві комп'ютери. Додаткі, що запускаються на них, реально здійснюють невеликі обсяги обчислень, а іноді займаються тільки відображенням одержуваної від сервера інформації, тому вони називаються тонкими клієнтами.

– Додатка в багатоланковій архітектурі. Недолік попередньої архітектури в тім, що різко зростає навантаження на сервер, а якщо він виходить із ладу, то робота всієї системи зупиняється. Тому в деяких випадках у систему додається так званий сервер додатків, на якому виконується вся обчислювальна робота. Інший сервер баз даних обробляє запити користувачів, на третьому може бути встановлена спеціальна програма – монітор транзакцій, що оптимізує обробку транзакцій і балансує навантаження на сервери. У більшості практичних випадків всі сервери з'єднані послідовно – ланцюгово, і вихід з ладу однієї ланки якщо й не зупиняє всю роботу, то принаймні різко знижує продуктивність системи.

– Додатка в розподіленій архітектурі. Щоб уникнути недоліків розглянутих архітектур, були розроблені спеціальні технології, що дозволяють створювати програму у вигляді набору компонентів, які можна запускати на будь-яких серверах, зв'язаних у мережу (компоненти як би розподілені по мережі). Основна перевага подібного підходу в тім, що при виході з ладу будь-якого комп'ютера спеціальні програми – монітори, які стежать за коректністю роботи компонентів і дозволяють їм "переговорюватися" між собою, відразу перезапускають тимчасово зниклий компонент на іншому комп'ютері. При цьому загальна надійність всієї системи стає дуже високої, а обчислювальне завантаження розподіляється між серверами оптимальним чином. Доступ до можливостей будь-якого компонента, призначеного для спілкування з користувачем, здійснюється з довільного клієнтського місця. При цьому, тому що всі обчислення відбуваються на серверах, з'являється можливість створювати надтонкі клієнти – програми, що тільки відображають одержувану з мережі інформацію й потребуючих мінімальних комп'ютерних ресурсів. Завдяки цьому доступ до компонентної системи можливий не тільки із ПК, але й з невеликих мобільних пристроїв.

Окремий випадок компонентного підходу – доступ до серверних додатків із браузерів через Інтернет.

Сьогодні найбільш популярні три компонентні технології – CORBA консорціуму OMG, Java Beans компанії Sun і COM+ корпорації Microsoft. Ці технології будуть визначати розвиток інформаційної індустрії в найближчі десятиліття.

### 3.2.3 Автоматизація завдань в інтегрованих середовищах

3.2.3.1 Мова макросів – засіб автоматизації роботи з додатками

Основним завданням програмного забезпечення інтегрованих середовищ комп'ютера є спрощення, автоматизація й прискорення виконання завдань. Пакет Microsoft Office надає користувачам два засоби автоматизації роботи з додатками: мова макросів і мова Visual Basic for Applications (VBA). Оскільки можливості макросів обмежені, для створення складних додатків варто використовувати мову VBA, яка доступна у всіх додатках Microsoft Office.

**Макроси**

Якщо певна дія часто повторюється, її виконання можна автоматизувати за допомогою макросу. ***Макрос*** – набір з однієї або декількох макрокоманд, що виконують певні операції й використовуються при автоматизації часто виконуваних завдань. ***Макрокоманда*** – основний компонент макросу, замкнута інструкція, самостійно або з іншими макрокомандами визначальна виконувані в макросі дії. ***Макрос*** – це серія команд і функцій, що зберігаються в модулі Visual Basic. Їх можна виконувати щораз, коли необхідно виконати дане завдання.

Перед тим як записати або написати макрос, необхідно спланувати кроки й команди, які він буде виконувати. Вираження, що перевіряє виконання деякої умови для здійснення (нездійснення) переходу до зазначеного у вираженні макрокоманді, зветься умовне вираження в макросі. Якщо при записі макросу була допущена помилка, її виправлення буде також записано. Щораз при записі макросу він зберігається в новому модулі, приєднаному до додатка.

Редактор Visual Basic дозволяє змінювати макроси, а також копіювати їх або з одного модуля в інший, або між різними додатками. Крім того, можна перейменовувати модулі, у яких зберігаються макроси, або перейменовувати самі макроси.

Іншими словами, макрос є командою, що створюється самим користувачем і служить для автоматизації шаблонних операцій. Макроси позбавлять користувача від нудної механічної роботи, надавши більше часу для творчості й міркувань. Варто відзначити, що перевага макросів багато в чому пояснюється тим, що мова макрокоманди, називаної Visual Basic, тісно пов'язаний з функціями додатків, і, по суті, макроси є аналогами

команд. Це означає, що, вивчаючи процеси створення й запуску макросів, користувач працює безпосередньо зі структурою функцій. Завдяки тісному взаємозв'язку функцій і макросів користувач може закріплювати команди або макроси за будь-якими меню, комбінаціями клавіш і панелями інструментів, у тому числі й створених самим користувачем.

Виражаючись більш точно, макроси дозволяють перепрограмувати додатка: модифікувати його функції, змінюючи назви пов'язаних з ними команд і їхнє розташування в меню, перетворюючи виконувані ними дії й заміняючи клавіші, якою вони привласнені. Так само вільно можна звертатися й із кнопками панелей інструментів і операціями, які вони викликають.

Для створення макросів і роботи з ними, треба, щонайменше, знать, як вибираються команди, переміщається курсор уведення й виділяється текст.

**Створення макросів**

Щоб скористатися макросом, його треба спочатку створити, а потім запустити.

Існує два способи створення макросів:

- ✧ записом дій користувача;
- ✧ за допомогою команди "Макрос" з меню "Сервіс" (MS Office).

При створенні макросу першим способом він складається з операторів так, ніби програма писалася на якій-небудь мові програмування. Користувач повинен бути знаком з мовою, на якому складаються макроси.

Другий спосіб – запис команд і натискання клавіш – більше простій і найбільш прийнятний для користувача. Досить активізувати убудовану в програму функцію запису макрокоманд і виконати потрібні дії. Наприклад, у програмі Excel для цього треба (див. ілюстрації в Додатку 5 Файлу матеріалів):

У меню "Сервіс– вибрати підменю "Макрос" і вибрати команду "Почати запис".

Увести ім'я макросу у відповідне поле. Першим символом ім'я макросу повинна бути буква. Інші символи можуть бути буквами, цифрами або знаками підкреслення. В ім'ї макросу не допускаються пробіли; як роздільники слів варто використовувати знаки підкреслення.

Щоб виконати макрос із клавіатури за допомогою сполучення клавіш, увести відповідну букву в поле Сполучення клавіш. Для малих літер використовується сполучення "CTRL+ букв", а для заголовних – "CTRL+SHIFT+ буква", де "буква" – будь-яка клавіша на клавіатурі. Буква, використовувана в сполученні клавіш, не може бути цифрою або спеціальним символом. Задане сполучення клавіш буде заміняти будь-яке встановлене за замовчуванням в Microsoft Excel, поки книга, що містить даний макрос, відкрита.

У поле "Зберегти в книзі" вибрати книгу, у якій повинен бути збережений макрос.

Щоб створити короткий опис макросу, увести необхідний текст у поле "Опис".

Нажати кнопку "ОК".

За замовчуванням при записі макросу в Microsoft Excel використовуються абсолютні посилання. Макрос, записаний з абсолютними посиланнями, при виконанні завжди обробляє ті ж осередки, які оброблялися при його записі. Для того, щоб за допомогою макросу обробляти довільні осередки, варто записати його з відносними посиланнями. Для цього варто нажати кнопку "Відносне посилання" на панелі інструментів "Зупинка запису". Відносні посилання будуть використовуватися до кінця поточного сеансу роботи в Microsoft Excel або до повторного натискання кнопки "Відносне посилання".

Виконати макрокоманди, які потрібно записати.

Нажати кнопку "Зупинити запис" на відповідній панелі інструментів.

Для того щоб макрос спочатку вибирав певний осередок, виконував макрокоманду, а потім вибирав інший осередок, пов'язану з активної, потрібно використовувати й абсолютну, і відносну посилання при записі макросу. При записі відносних посилань кнопка "Відносне посилання" повинна бути натиснута. При записі абсолютних посилань кнопка "Відносне посилання" не повинна бути натиснута.

Для автоматизації роботи в додатках можна створювати групи макросів. Група макросів – це набір зв'язаних макросів, що зберігаються під загальним ім'ям.

Налагодження й виконання макросів

Для налагодження й зміни записаного макросу необхідне знайомство з редактором Visual Basic.

Щоб одержати перші навички в налагодженні й редагуванні макросів в Microsoft Excel, треба:

Відкрити книгу, що містить макрос.

У меню "Сервіс" установити покажчик на пункт "Макрос" і вибрати команду "Макроси".

У поле "Ім'я макросу" увести ім'я того макросу, якому потрібно виконати.

Натиснути кнопку "Змінити".

У результаті цих операцій на екрані з вікно Visual Basic зі спеціальним вікном макросу (модуль VB), у якому буде виведений текст макрокоманд (рис. 3.39). Вікно макросу – вікно на екрані, у якому створюється й змінюється макрос.

У цьому вікні можливо:

- ✧ змінити макрокоманди відповідно до необхідності;
- ✧ скористатися кнопками на панелі інструментів для запуску й перевірки результатів редагування.

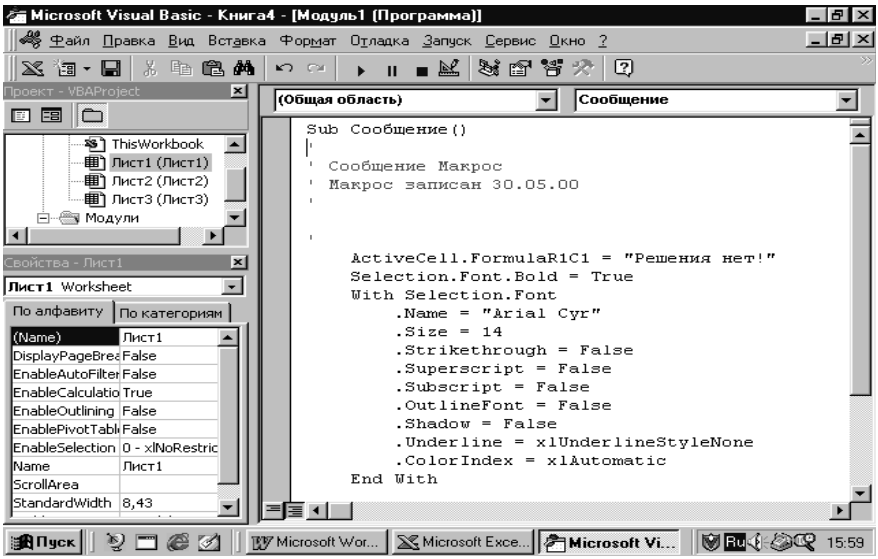



Рисунок 3.39. Спеціальне вікно макросу (модуль VB)

Однак, якщо користувач не має достатньо навиків при роботі в Visual Basic, то варто протестувати макрокоманди. При тестуванні макрокоманди вікно документа повинне бути активно. Якщо при тестуванні макросу виявиться помилка, виконання макросу зупиниться, невірний оператор підсвітиться інверсним кольором, і на екрані з'явиться повідомлення про помилку. Характер повідомлення залежить від типу помилки, однак у кожному разі більшість із них зрозуміло тільки програмістам. Закінчивши редагування, варто закрити вікно макросу й закрити вікно редактора, зберігши уведені зміни.

Готовий макрос можна запустити на виконання одним з наступних способів:

- ✧ вибравши з меню "Сервіс" команду "Макрос". У поле "Ім'я" варто вказати потрібний макрос і вибрати кнопку "Виконати". Макрос виконається.
- ✧ призначивши макросу кнопку на панелі інструментів і клацнувши цю кнопку. Для присвоєння макросу кнопки на панелі інструментів або команди меню варто вибрати команду меню "Настроювання", вибрати записуваний макрос у списку "Команди" і перетягнути його на панель інструментів або в меню. Потім треба нажати кнопку "Закрити".
- ✧ закріпивши макрос за комбінацією клавіш і натиснувши дану комбінацію.

3.2.3.2 Мова Visual Basic for Applications (VBA)

**Visual Basic (VB)**

Visual Basic (VB) для додатків – це середовище програмування, засноване мовою програмування Visual Basic для Windows. VBA приходить на зміну елементарним мовам макропрограмування, які звичайно включалися в додатки. VBA є високоефективним засобом для розробки додатків, оскільки він належить до об'єктно-орієнтованих мов програмування й має простоту макромов. Користувачі можуть записувати свої дії й створювати макроси без вивчення складностей мови. Запис дій, а потім перегляд записаного коду є найпростішим способом для самостійного вивчення VBA.

При створенні програми більша частина роботи в Visual Basic виробляється на етапі візуального програмування. Це сприяє тому, що при створенні програми вже видно, як вона буде виглядати в дії. Привабливою рисою Visual Basic є здатність використання передових технологій програмування.

Одне з найважливіших переваг Visual Basic полягає в тому, що можна розробляти програму невеликими кроками, а після кожного кроку запускати її й спостерігати в дії. Це створює зворотний зв'язок і дозволяє оцінювати й коректувати програму на всіх етапах її розробки. Однак треба пам'ятати, що необхідно зберегти створену частину програми перед запуском, тому що можлива помилка в коді може привести до зависання й втрати інформації.

**Об'єкти**

В Visual Basic для додатків застосовується модель об'єктно-орієнтованого програмування. Об'єктами можуть бути діаграми, малюнки, фігурний текст, таблиці й т.п., що вставляються в документ. Об'єктам відомо, як їм "маніпулювати самими собою й відтворювати себе", якщо вони впроваджені в документ іншого додатка. В Visual Basic для додатків використовуються переваги, надавані цією можливістю для керування зазначеними об'єктами.

Об'єкти VBA являють собою зручний спосіб для зберігання й приховання даних і коду в програмі. Замість того щоб написати програму для маніпулювання деякими даними, можна об'єднати дані й код, що маніпулює цими даними, в один об'єкт. Із цього моменту доступ до даних і їхній обробка здійснюються тільки засобами об'єкта.

Це використовується при впровадженні об'єктів одного додатка в об'єкти іншого. Впроваджений об'єкт сам піклується про себе, від об'єкта – контейнера потрібно тільки надати впровадженому об'єкту відповідні ресурси. Наприклад, при підключенні кнопки Visual Basic до робочого аркуша робочий аркуш не знає, яке дія необхідно виконати при клацанні мишею по ній, ця інформація вміщується у викликуваної об'єктом процедурі.

До об'єктів VBA в Microsoft Excel відносяться кнопки, пункти меню, діапазони осередків робочого аркуша й робочі аркуші цілком. Майже все, що можна побачити на екрані в процесі виконання додатка, є об'єктом. Об'єкти в Visual Basic для додатків можна представити у вигляді декількох контейнерів. Найбільший контейнер – це об'єкт-додаток, що представляє собою поточну програму в процесі виконання, наприклад, Excel. Об'єкт Application в Excel містить об'єкти меню (Menu), об'єкти-елементи керування (Control), об'єкти-робочі книги (Workbook) і т.д. Усередині об'єкта Робоча книга перебувають об'єкти-аркуші (робочі аркуші, аркуші макросів, аркуші модулів, аркуші діалогу й т.д.), усередині об'єктів-аркушів розташовані об'єкти діапазону (Range). Для інших додатків передбачені подібні об'єкти для виконання їхніх специфічних завдань. Перелік об'єктів приводиться в кожному додатку в документації й у довідковій системі для різних додатків Microsoft Office.

Для доступу до певного об'єкта в Visual Basic для додатків потрібно вказати ім'я зовнішнього об'єкта контейнера, потім ім'я вкладеного в нього об'єкта контейнера й т.д., доти, поки не буде досягнутий необхідний об'єкт. Наприклад, в Excel для доступу до осередку B5 на робочому аркуші 3 у робочій книзі "Книга 1" використовується наступне посилання (імена об'єктів контейнерів розділяються крапкою):

Application. Workbooks ("Книга 1").Worksheet ("Аркуш3"). Range ("B5")

Клас об'єктів – це посилання на загальний тип або класифікацію об'єктів. Наприклад, в Visual Basic для додатків кожний осередок або діапазон осередків на робочому аркуші Excel є об'єктом Range, що являє собою реалізацію класу Range. Об'єднання всіх об'єктів певного класу в групу створює сімейство. Так, всі робочі аркуші в робочій книзі Excel перебувають у сімействі Worksheet. Крім того, робітники аркуші належать також і сімейству Sheets, що включає всі типи аркушів (робітник аркуш, діаграма, модуль, діалог) у робочій книзі.

Дані об'єкта називаються властивістю об'єкта. Властивість об'єкта визначає, як об'єкт виглядає або як він поводиться. Значеннями властивостей можуть бути текстові рядки, числа, логічні значення або нумеровані списки. Нумерований список – це пронумерований перелік значень, у якому номер використовується для вибору певного значення. Наприклад, властивість Color більшості об'єктів – це нумерований список, у якому значенню 0 відповідає відсутність кольору, значенню 1 – чорний колір, 2 – білий, 3 – червоний, 4 – зелений, 5 – синій і т.д. VBA і інші сумісні додатки в Microsoft Office містять переліки визначених констант для нумерованих списків, перерахованих в описі властивості в довідковій системі; ці константи можна використовувати замість чисел.

Найпростіший спосіб для написання програми, що змінює властивості об'єктів, полягає в тому, щоб створити макрос із необхідними установками й потім скопіювати процедуру зміни властивості в програму. Для доступу до властивостей об'єктів використовується наступний синтаксис:

***object.property***

де: object – об'єкт, властивості якого потрібно змінити або переглянути;

property – ім'я властивості.

Наприклад, щоб установити в Excel значення властивості Formula (уміст осередку) для осередку B5 рівним ABS(B4), якщо осередок перебуває на аркуші "Аркуш1" у робочій книзі "Книга2", можна використовувати наступний оператор:

***Workbook("Книга2").Worksheet("Аркуш1").Range("B5").Formula = "ABS(B4) "***

Для одержання значення цієї властивості із цього ж осередку й для збереження його в змінної My Formula можна використовувати наступний оператор:

***My Formula =***
***Workbook("Книга2").Worksheet("Аркуш1").Range("B5").Formula***

При створенні операторів застосовуються розглянуті вище правила, що ставляться до пропуску об'єктів контейнера.

**Методи**

Методи в Visual Basic для додатків – це блоки коду, що зберігаються в об'єкті, які дозволяють маніпулювати даними об'єкта. Наприклад, для об'єкта Range метод Calculate обчислює заново значення формул у виділених осередках, а метод Clear – видаляє вміст осередків. Методи роблять дії з об'єктами й даними, що втримувалися в них, на відміну від властивостей, які тільки встановлюють значення. Довідкова система допоможе визначити, які методи варто використовувати для того або іншого об'єкта й отримати додаткову інформацію про особливості окремих методів.

Доступ до методів об'єкта і їхнє виконання здійснюється майже так само, як доступ до властивостей об'єкта. Основне розходження полягає в тім, що, наприклад, доступ до властивості завжди є частиною формули, а метод повинен бути частиною формули тільки тоді, коли він повертає значення.

Наприклад, в Excel метод Rows повертає сімейство, що із всі рядки в діапазоні. Щоб установити за допомогою цього методу значення властивості RowHeight всіх рядків поточного обраного діапазону рівним 20, використовується формула:

***Selection.Rows.RowHeight = 20***

Для одержання числа рядків у поточному виділенні можна використовувати метод Rows для повернення сімейства й властивість Count для повернення числа елементів у сімействі:

***NumRows = Selection.Rows.Count***

Деякі методи мають аргументи, наприклад, для методу Rows потрібен індекс рядка для вибору окремого рядка із сімейства Rows. Якщо метод є частиною формули, то аргументи повинні перебувати в круглих дужках. Для того щоб одержати значення властивості RowHeight другого рядка в сімействі Rows, використовується наступний оператор:

***TheHeight = Selection.Rows(2).RowHeight***

Якщо розробляється тільки виконання методу, і він не є частиною формули, то аргументи розміщаються праворуч від посилання на цей метод. Наприклад, метод Insert при використанні для об'єкта Range вимагає відомостей про те, як переміщати виділені осередки. При використанні методу Insert для вставки порожніх осередків на місце поточного виділення й переміщення його долілиць для одержання вільного простору можна використовувати наступний оператор:

***Selection.Insert xlDown***

Аргумент у дійсності являє собою ціле число, але тут використана вбудована константа для того, щоб зробити код більше наочним. Опис вбудованих констант методу приводиться в довідковій системі.

**Створення додатків**

Створення додатків в Visual Basic включає два базових кроки:

- ✧ побудова користувальницького інтерфейсу додатка, що дозволяє вводити й виводити інформацію;
- ✧ написання програмного коду, що забезпечує обробку інформації.

В Visual Basic користувальницький інтерфейс і частину коду представлені у вигляді екранної форми. Користувальницький інтерфейс програм для Windows звичайно складається з меню, однієї або декількох панелей інструментів для прискорення дій і робочої області, представленої у вигляді документа або розгорнутого аркуша. Екранна форма – це те, що утворює вікно (або вікна), що користувач програми бачить на екрані.

Незважаючи на те, що розроблювальний користувальницький інтерфейс може відрізнятися від інтерфейсу інших програм для Windows, є деякі елементи, які повинні бути виконані так само, як і в інших програм. До них відносяться нижчеперераховані елементи:

Меню “File”, що дозволяє відкривати, зберігати й закривати файли.

Меню “Edit”, де втримуються операції видалення, копіювання, вставки й очищення.

Меню “Help”, куди користувач звертається за допомогою.

Інструментальна лінійка, на якій розташовані кнопки, використовувані для прискореного запуску виконуваних операцій.

Інші елементи інтерфейсу варто підібрати, використовуючи різні об’єкти керування. Об’єкти керування являють собою основу для побудови всіх програм на Visual Basic. Об’єкти керування дозволяють створювати програму, не вимагаючи від розробника всіх деталей, відповідальних за роботу цих об’єктів.

Деякі об’єкти керування (такі як текстові вікна, смуги прокручування, вікна списків, кнопки-перемикачі) використовуються для забезпечення користувальницького уведення. Інші елементи керування, такі як командні кнопки, дозволяють користувачеві виконати передбачені в програмі дії.

Незважаючи на важливість, користувальницький інтерфейс при розробці програми сам по собі не забезпечує рішення всіх завдань. Тільки програмний код реально робить програму, що розробляється робочим інструментом. Кодом називається мова, що використовується для спілкування з комп’ютером.

Visual Basic – мова програмування, що підтримує програмні конструкції, властивій більшості інших мов програмування.

Правильно написана програма створюється за модульним принципом. Більшість програм створюється із блоків, відомих як процедури й функції. У мові Visual Basic модуль – набір оголошень і процедур мовою VB для додатків, зібраних в одну програмну одиницю.

В Visual Basic є два типи процедур:

1) процедури обробки подій;
2) загальні процедури.

Visual Basic автоматично викликає процедури обробки подій у відповідь на дії клавіатури, мишею або системні дії. Наприклад, командні кнопки мають процедуру обробки події Click (клацання миші). Код, що міститься в процедуру обробки події Click, буде виконаний, коли користувач клацне кнопку.

Кожний елемент керування має фіксований набір процедур обробки подій (для кожного підтримуваного об’єктом події). Ці процедури для кожного елемента керування перераховані в списку, що розкривається, Procedure у вікні коду Visual Basic.

Наприклад, що випливає фрагмент коду – процедура обробки подій Click для командної кнопки cmdOK:

```
Private Sub cmdOK_Click()
MsgBox «Visual Basic» ‘Відкрити вікно повідомлення
```

***End Sub***

Після того, як до додатка доданий деякий програмний код, у вікні "Object Browser" можна побачити всі створені процедури, швидко перейти до певної процедури або вставити виклик процедури в текст програми у вікні коду.

У вікні "Object Browser" показані класи, доступні із всіх об'єктних бібліотек, підключених до проекту.

Модулі класів в VB – модулі, пов'язані з певною формою або звітом, які містять процедури обробки подій, що запускаються у відповідь на подію у формі або звіті.

Загальні процедури – це процедури, створювані для виконання певних завдань. На відміну від процедур обробки подій, щоб виконати загальну процедуру, її треба явно викликати. Викликати процедуру можна, специфицируя тільки ім'я процедури або використовуючи оператор Call з ім'ям процедури й параметрами. Наприклад,

***Call sortList ("name1")***

Створити загальну процедуру можна за допомогою оператора Sub або за допомогою діалогового вікна "Insert Procedure" або "Add Procedure" залежно від версії Visual Basic. Процедура Sub не повертає значень, вона тільки виконує деяку обробку або дію. Хід виконання програми із процедурою Sub показаний на рис. 3.40.

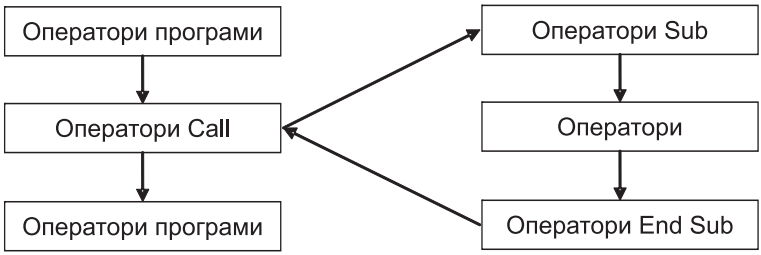



Рисунок 3.40. Хід виконання програми із процедурою Sub

Процедура Function виконує дію й повертає значення. Наприклад, у наступному прикладі функція одержує число й повертає квадрат цього числа:

***Function Square (I As Integer) As Integer***
***Square = I * I***
***End Function***

Якщо потрібно зберегти повертається значение, що, при виклику функції необхідно використовувати круглі дужки, як у наступному операторі:

***J = Square (5)***

Якщо круглі дужки опущені, що повертається значення можна ігнорувати або не зберігати його в зміною. Це може бути корисно, якщо потрібно лише виконати обробку, пов'язану з функцією. Наприклад:

***Square 5***

Крім додавання коду до модуля, асоційованому з формою, можна також повідомляти процедури в стандартних модулях. Стандартні модулі містять тільки код Visual Basic. Вони призначені для зберігання коду, що не привязується до певної форми. Стандартні модулі в VB – модулі, у яких утримуються загальні процедури, не зв'язані ні з яким об'єктом.

Процедури можуть бути оголошені як Private або Public. До класу Private ставляться змінні, використовувані тільки усередині процедури або функції, у якій вони визначені. До класу Public ставляться загальні змінні, які доступні у всіх процедурах або функціях програми. Їх можна використовувати для зберігання інформації, необхідної в декількох процедурах.

Отже, процедури, оголошені як Private, можуть викликатися тільки іншими процедурами, розміщеними в даній формі, модулі або класі.

Процедури Public у модулі-формі стають методами форми. Такі процедури можна викликати незалежно від місця в додатку, специфікуя імена процедури й форми.

Наприклад нижче представлена процедура Public :

***Public Sub Proc1 ()***

***...***

***End Sub***

Якщо ця процедура оголошена в модулі форми, викликається вона в такий спосіб:

***Form1.Proc1***

Якщо процедура оголошена в стандартному модулі, викликається вона так:

***Proc1***

Функцію, як і процедуру, можна створити за допомогою діалогового вікна або за допомогою оператора Function. Як і процедура, функція може бути оголошена як Public або Private і може бути поміщена на екранну форму або в окремий модуль. Синтаксис звертання до функції практично ідентичний синтаксису виклику процедури.

Розробка програм у вигляді набору невеликих фрагментів (модулів) спрощує їхнє написання й налагодження.

**Налагодження й виконання програм**

Незалежно від рівня кваліфікації програміста поява помилок у програмі неминуче. Оскільки виключити поява помилок неможливо, основою успішної розробки програм є можливість виявлення й усунення цих помилок. Для цих цілей в Visual Basic передбачений ряд інструментів.

В Visual Basic є блок синтаксичного контролю кожного нового рядка коду, що вводитися. Блок синтаксичного контролю виявляє помилки при записі ключових слів або операторів, наприклад, пропуск дужок. При виявленні синтаксичної помилки Visual Basic виділяє помилковий рядок червоним кольором і виводить повідомлення про можливу причину помилки. Блок синтаксичного контролю звичайно підключається при початковій установці Visual Basic.

Існує одна сервісна функція Visual Basic, призначена для виділення ключових слів синім кольором і написання їх із заголовної букви. Це дозволяє візуально контролювати правильність уведення команд. Visual Basic також контролює правильність написання імен властивостей об'єктів.

При запуску програми із середовища проектування Visual Basic зустрічаються помилки, перераховані в посібнику з програмування. У цьому випадку виводиться повідомлення, у якому приводиться номер помилки і її текстовий опис. Іноді помилка буває очевидної, але часто для виявлення помилки потрібно більше ретельний аналіз коду.

Вікно "Debug" є основним засобом для пошуку помилок. Воно передбачає набір засобів для пошуку помилок. Це вікно дозволяє виконувати команди програми й визначати як викликаний, так і зухвалий модуль.

Наявні у вікні "Debug" засобу дозволяють стежити за значенням змінної при виконанні програми. Це дозволяє, наприклад, виявляти нескінченні цикли.

Існує кілька способів визначення значення змінної:

- ✧ установка спостереження у вікні.
- ✧ використання разового спостереження.
- ✧ присвоювання значення змінної в текстовому вікні.

Іншою можливістю вікна "Debug" є можливість прямого уведення команд, які виконуються відразу ж після натискання клавіші <Enter>. Користаючись із цієї можливості, можна вивести змінну на екран або змінити її значення за допомогою оператора присвоювання. Можна також використовувати це вікно для уведення команд, що змінюють властивості об'єктів, з якими виконується програма, наприклад, можна змінити шрифти на формі.

Ще одна можливість вікна "Debug" – дана можливість індикації поточної виконуваної процедури. Крім того, можна викликати діалог, що показує всю послідовність викликів від вихідної процедури до поточної.

Це допомагає виявити ситуації, при яких виклик процедури здійснюється з тих частин програми, де він не передбачався.

Існують і інші можливості контролю виконання програми. Наприклад, в Visual Basic існує три способи виконання переривання програми:

1) установити в програмі крапку переривання, у якій виконання програми буде припинено, якщо відбулася зміна значення змінної або задане вираження прийняла значення True.
2) клацнути мишею на піктограмі Break (переривання) на лінійці інструментів.
3) установити в програмі крапку переривання на певному рядку.

Однак один з найефективніших засобів налагодження в Visual Basic – є можливість покрокового виконання програми й спостереження результатів виконання кожного оператора. Такий спосіб дуже корисний при визначенні правильності роботи умовних операторів.

На закінчення можна сказати, що налагодження програм вимагає часу, терпіння й використання всіх сервісних засобів налагодження, наявних в Visual Basic.

Після того як закінчена побудова інтерфейсу, написання й налагодження програмного коду додатка, можна створювати для користувачів файлу, що виконується.

Створення файлу, що виконується, в Visual Basic – простий процес. Для цього потрібно:

- ✧ в меню “File” вибрати команду “Make Ім’я_ проекту.exe”.
- ✧ увести ім’я для файлу, що виконується.
- ✧ щоб додати інформацію з версії програми: у вікні діалогу “Make Exe File” нажати кнопку “Options”. Відкриється вікно діалогу “Project Properties”. На вкладці “Make” увести номер версії й текст інформації версії, клацнути “ОК”.

Освоєння програмування в середовищі MS Office представляє певні труднощі як для початківців, так і для досвідчених розробників. Спочатку потрібно вивчити незнайомий світ програмування, потім – оволодіти функціональними можливостями офісних пакетів. І як з’ясовується на практиці, для цього потрібно вивчати не тільки книги з “програмуванню в Office”, але і літературу з роботи з офісними пакетами.

## 3.3 Програмні засоби роботи зі структурованими документами

Дана інформація носить досить загальний характер і спрямована на формування уявлення про можливості текстових процесорів та електронної таблиці і їх використанні. Основні функціональні можливості й технологічні операції розглядаються без прив’язки до конкретного типу процесора. Отримані знання є базовими при практичному знайомстві з будь-яким новим видом текстового та табличного процесора, що дозволить самостійно опанувати новий тип процесору.

### 3.3.1 Призначення й класифікація текстових редакторів

Робота з текстами – важлива складова частина діяльності людей багатьох професій: письменників, юристів, учених, керівників, інженерів, секретарів. Обробка текстів, підготовка різного виду документів становлять значну частину робіт, виконуваних у цей час на ПК.

До складу підготовлюваних на ПК документів можуть входити текстові дані, таблиці, математичні формули, графічні об'єкти й т.д. Головне завдання будь-якого текстового редактора полягає в забезпеченні оптимальних для користувача умов по створенню й обробці документів.

Сучасний **текстовий редактор** являє собою програмний продукт, що забезпечує користувача ПК засобами створення, обробки й зберігання документів різного ступеня складності. Останнім часом текстові редактори витісняються текстовими процесорами, які дозволяють тільки набирати не відформатований текст. Однак без шкоди для розуміння можна рівноцінно використати обидва редактори.

Текстовий редактор дозволяє робити все те, що може робити друкарниця за допомогою друкарської машинки. Але, окрім традиційних можливостей ПК дозволяє здійснювати якісно нові способи обробки текстових документів: вставку повторюваних фрагментів, зміна довжини рядків, автоматичний перенос слів, виділення потрібних частин тексту нестандартним шрифтом при друку, тощо.

Помилка при підготовці документа на ПК не створює труднощів при переробці, тому що автоматичне виконання рутинних робіт полегшує зміну та переміщення фрагментів тексту. Можливість попереднього перегляду отриманого документа на екрані дисплея допомагає уникнути непродуктивних витрат праці й паперу при друку.

Достаток різних типів документів привело до урізноманітнення існуючих текстових редакторів. Ознакою класифікації, за допомогою якої можна розділити всю безліч текстових редакторів на групи, приймемо ***тип оброблюваного документа***.

1. **Редактори текстів** – призначені для створення й редагування нескладних текстів і текстів програм (Brief, Norton Editor, Quick).

2. **Редактори документів** – призначені для роботи з документами, що структурно складаються із вкладених розділів, сторінок, абзаців і т.д. У структуру документа можуть входити таблиці, графічні образи, які можуть створюватися в інших додатках. Серед редакторів, призначених для роботи з текстовими документами, можна виділити Лексикон, AmiPro, MultiEdit, Word Perfect, Microsoft Word. Визнаючи спільність останніх по типі створюваного й оброблюваного об'єкта, не можна не відзначити їхнього розходження за можливостями, надаваним користувачеві.

3. **Редактори наукових текстів**. Особливість даного класу редакторів у тому, що вони забезпечують підготовку й редагування наукових текстів,

які містять велику кількість математичних формул, графіків, спеціальних символів і т.д. Серед найбільш відомих редакторів таких текстів можна виділити системи ТИХ й MathOr. Необхідно відзначити, що сучасні текстові редактори містять у собі засобу підготовки документів з формулами, використання редакторів наукових текстів доречно лише при підготовці тексту, що містить багато формул.

4. **Видавничі системи** – використаються для підготовки більш складних документів (книги, альбоми, журнали, газети, буклети). Як приклади видавничих систем можна назвати Corel Ventura Publisher, Adobe PageMaker, QuarkXPress. Необхідно відзначити, що робота з видавничими системами припускає використання редакторів документів на етапі попередньої підготовки матеріалів. Видавничі системи призначені для верстки тексту.

**Верстка тексту** полягає в розміщенні тексту по сторінках створюваного документа, вставці малюнків, використанні різних шрифтів відповідно до документів, уже створених за допомогою редакторів текстів. У цілому, на наш погляд, розходження між редакторами документів і видавничими системами поступово стираються. За загальною думкою, останнім часом редактори документів за своїми можливостями не тільки наблизилися до видавничих систем, але навіть у деяких питаннях, стали їх перевершувати. Прикладом такого текстового редактора може служити Microsoft Word.

Отже, після викладення короткого огляду текстових редакторів можна зробити висновок, що можливі розходження між ними в цей час перестали бути принциповими. Тенденція в розвитку текстових редакторів полягає в створенні редакторів, що дозволяють користувачам одночасно працювати з текстами, що містять об'єкти й фрагменти різної природи. Особливо чітко це простежується в редакторах документів.

Як правило, вихідним моментом роботи з документом є текст, до якого додаються об'єкти й фрагменти різної природи, наприклад, діаграми, таблиці й т.д. Робота фахівця з документами в першу чергу має на меті розширення інформаційної бази прийнятих рішень. Тому бажано, щоб при роботі з документом основний робочий час витрачався фахівцем не на створення документа (набір), а на додання йому необхідного вигляду та його аналу.

### 3.3.2 Базові можливості текстових редакторів

3.3.2.1 Типова структура інтерфейсу

Типова структура інтерфейсу сучасного текстового редактора наведена на рис. 3.41.

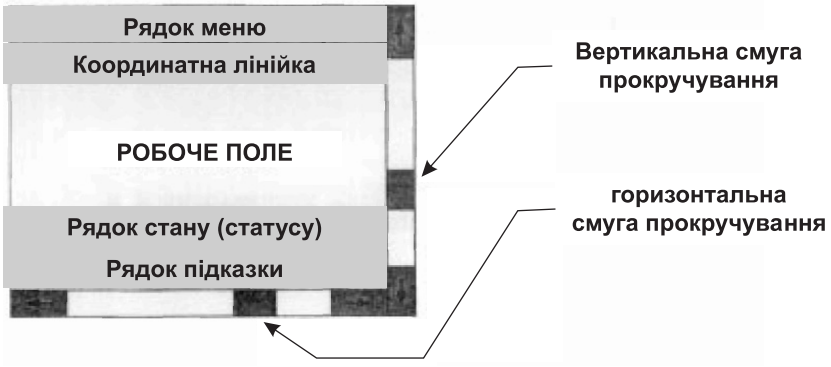



Рисунок 3.41. Вид типового інтерфейсу

**Рядок меню** містить імена груп команд, об'єднаних за функціональною ознакою. Рядок меню розташований у верхній частині екрана. Вибір команди меню відкриває відповідне підменю, а вибір певної опції в ньому забезпечує доступ до меню більше низького рівня. Така система вкладених (спадаючих) меню становить основу інтерфейсу текстового процесора. Команди меню вибираються за допомогою миші, клавіш керування курсором або натисканням комбінацій певних клавіш ("гарячих клавіш" або клавіш швидкого доступу).

**Рядок стану** (статусу) містить ім'я документа, що редагується, і визначає поточне положення курсору в цьому документі. У рядку перебувають індикатори режиму роботи редактори й виводиться довідкова інформація.

**Рядок підказки** містить інформацію про можливі дії користувача в даний момент.

**Робоче поле** – це простір на екрані дисплея для створення документа й роботи з ним. Максимальний розмір робочого поля визначається стандартними параметрами монітора й становить 25 рядків по 80 знаків кожна.

**Координатна лінійка** визначає границі документа й позиції табуляції. Розрізняють вертикальну й горизонтальну лінійки. За замовчуванням координатна лінійка градуйована в сантиметрах. Нульова крапка координатної лінійки вирівняна по першому абзаці тексту.

**Смуга (лінійка) прокручування** використовується для переміщення тексту документа в робочому полі вікна. Смуга, що забезпечує вертикальне переміщення тексту, називається вертикальною смугою прокручування, а горизонтальне переміщення – горизонтальною смугою прокручування.

### 3.3.2.2 Керування курсором

**Курсор –** світлова відмітка, що означає поточну позицію на екрані, в яку буде зроблено введення символу при натисненні клавіші на клавіатурі.

Покажчик миші має вигляд латинської букви **I**, коли він знаходиться в межах вікна документу. Якщо вимагається перемістити курсор до деякої ділянки тексту документу, слід помістити покажчик в потрібне місце і клацнути кнопкою миші.

В інформаційному рядку (рядку стану) редактора на екрані дисплея виводиться номер рядка документа й номер позиції в ній, де в сучасний момент перебуває курсор. У текстовому режимі курсор горизонтальний, що розміщений внизу знакомісця на яке вказує. У графічному режимі – вертикальний, перебуває в лівому місці вставки чергового символу. Кожен текстовий процесор має свої можливості для забезпечення руху курсору (як і керування інтерфейсом взагалі).

Керування курсором може здійснюватися за допомогою клавіатури й миші. Так, клавіші управління купсором (стрілки) переміщують курсор на одну позицію в напрямку стрілки. Клавіші Ноті й End – переміщають курсор у початок і кінець рядка відповідно. Клавіші PageUp й PageDn – переміщають курсор і текст на одну сторінку (екран) вгору або вниз.

Сучасні текстові процесори, використовують різні комбінації клавіш, дають можливість переміщувати курсор на одне слово, пропозицію або абзац, направляти його в початок або кінець тексту.

### 3.3.2.3 Перегляд тексту

Редагуємий текстовий документ, як правило, має розміри, що перевищують розмір екрану дисплея, тому користувач бачить тільки частину тексту. Можна розглядати екран дисплея як деяке вікно, через яке відображено фрагмент документа.

Текстовий процесор забезпечує переміщення тексту у вікні. Цей режим включається автоматично, як тільки курсор досягає границь екрану. Так, якщо курсор перебуває на нижньому рядку екрану, при натисненні на клавішу переміщення курсору вниз, унизу екрана з'явиться новий рядок документа, а самий верхній рядок екрана зникне. У такий же спосіб можна переглядати текстовий документ за напрямком вгору. Якщо ширина документа більша за ширину екрану монітора, то клавішою переміщення курсору вліво й вправо переміщають текст документа у вікні по горизонталі.

Як уже вказувалося раніше, для переміщення тексту документа в робочому полі вікна в багатьох текстових процесорах використаються вертикальна й горизонтальна смуги прокручування (див. рис. 3.41). Клавіші PageUp й PageDn дозволяють переміщуватися на одну сторінку (екран) вгору або вниз. Спеціальні команди забезпечують перехід, з будь-якої позиції, в кінець або в початок документу.

3.3.2.4 Режими роботи текстового процесора “вставка” й “заміна”

Текстовий процесор завжди перебуває в одному із двох режимів вставка або заміна. Індикація режиму здійснюється в інформаційному службовому рядку (рядку стану) процесора на екрані дисплея.

**Режим вставки** – режим роботи з документом, при якому додавання нових символів здійснюється за рахунок зрушення вправо тексту, якій находится за курсором.

**Режим заміни** – режим роботи з документом, при якому символ, що вводить із клавіатури, заміняє символ, що розміщується в позиції курсору.

Для перемикання з одного режиму на інший звичайно використається клавіша Ins.

Режимом вставки користуються, коли текст уводиться вперше, а також при редагуванні тексту. Якщо ж треба замінити один символ іншим, використають режим заміни.

3.3.2.5 Богатовіконий режим роботи

Характерною рисою сучасних текстових процесорів є їх *богатовіконисть*, що дозволяє одночасно працювати з декількома документами в різних вікнах. При уведенні й редагуванні тексту користувач працює в активному вікні, у якому можливий доступ до команд меню. Спеціальні команди дають можливість упорядковувати вікна документів, створювати нові й переходити з одного вікна в інше.

Тут розрізняють два режими: повноекранний і неповноекранний. У *повноекранном* режимі знову відкрите вікно є поточним (активним) і перекриває інші відкриті вікна. Перехід до інших вікон здійснюється по певній команді. У *неповноекранном* режимі команди впорядкування вікон дають можливість розташувати всі відкриті вікна на екрані в зменшеному форматі без перекриття. Вікно стає активним у результаті фіксації на ньому в довільному місці покажчика миші. Команди Розгорнути й Згорнути дають можливість розгорнути активне вікно у весь екран або згорнути його.

За допомогою команди **Розділити** робоча область розділяється на дві частини. Це зручно, якщо необхідно працювати одночасно із двома різними частинами великого документа.

### 3.3.3 Підготовка текстового документа

Основними етапами підготовки текстових документів є:

- ✧ набір тексту;
- ✧ редагування тексту;
- ✧ друк тексту;
- ✧ ведення архіву текстів.

Кожен етап складається з виконання певних операцій. Склад операцій залежить від використовуваного текстового редактора. Як правило, збільшення обсягу можливих операцій текстового редактора, які б задовольняли вимоги працівників різних професій, з одного боку, приводить до ускладнення його освоєння, підвищенню вимог до встаткування ПК.

Обобщенная блок-схема подготовки текстового документа представлена на рис. 3.42.

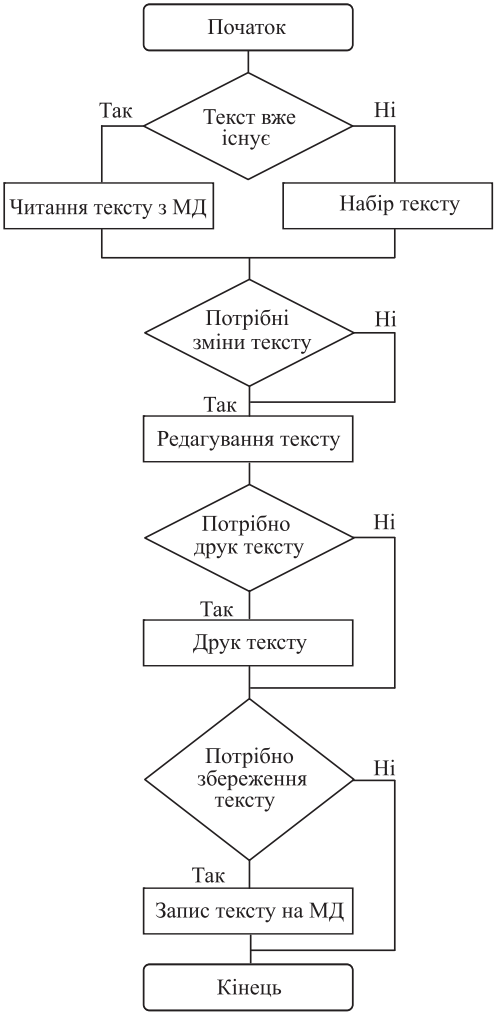



Рисунок 3.42. Узагальнена блок-схема підготовки текстового документу

Точно так існують “м’які” й “тверді” ознаки кінця рядка. “М’який” ознака кінця рядка створюється текстовим редактором автоматично при перенесенні тексту з одного рядка екрана на іншу в процесі досягнення позиції правої границі рядка. “Твердий” ознака кінця рядка створюється при натисканні користувачем клавіші Enter для вказівки завершення абзацу. “М’які” ознаки кінця рядка при редагуванні можуть автоматично віддалятися, додаватися, міняти своє місце розташування на відміну від “твердих” ознак кінця рядка.

Деякі текстові редактори не вказують на екрані ознака кінця рядка, але проте він існує в оперативній пам’яті ПК і записується на магнітний диск.

Такий принцип набору тексту на ПК істотно полегшує й прискорює процес уведення документів у порівнянні із друкарською машинкою, дозволяє робити його, не дивлячись на екран дисплея. При наборі тексту варто памятати, що ознакою для відділення слова від слова є пробіл, тому розділовий знак не треба відокремлювати пробілом від попереднього слова, а після розділового знака вводити пробіл треба обов’язково.

### 3.3.3.1 Набір тексту

Перед початком роботи курсор, зазвичай, перебуває у верхньому лівому куті порожньої області екрана, відведеної для набору тексту. Набір тексту здійснюється натисканням потрібних клавіш на клавіатурі ПК, при цьому черговий символ відображається на екрані в позиції курсору, а курсор переміщається на одну позицію вправо.

Набір тексту на ПК за допомогою текстового редактора нагадує роботу на друкарській машинці, але надає більші можливості. Зокрема, текстовий редактор не вимагає від користувача необхідності стежити за близькістю тексту до правої границі й здійснювати дії для переведення курсору в початкову позицію чергового рядка. При цьому забезпечується автоматичне вирівнювання лівих й, якщо необхідно, правих границь рядків. Вирівнювання правих границь рядків досягається за рахунок вставки пробілів між словами рядка таким чином, щоб самий правий символ рядка перебував у позиції, що відповідає правій границі документа.

Ті пробіли, які текстовий редактор автоматично встановлює для вирівнювання рядків, називають “м’якими” на відміну від “твердих” пробілів, внесених у текст при натисканні користувачем клавіші Пробіл на клавіатурі ПК. Термін “м’який” указує на те, що при редагуванні документа текстовий редактор при необхідності автоматично видаляє зайві або вставляє додаткові “м’які” пробіли для вирівнювання рядків. “Тверді” пробіли в текст автоматично не включаються й не вилучаються з нього.

3.3.3.2 Редагування тексту

Створений на етапі набору тексту документ, у більшості випадків піддається змінам, пов'язаним з виявленням помилок, коректуванням інформації, реорганізацією його фрагментів і тому подібного. При роботі зі звичайною друкарською машинкою проведення таких змін вимагає більших витрат, які пов'язані з багаторазовим передруковуванням матеріалів, що найчастіше незначно відрізняються один від одного.

Використання ПК і текстових редакторів відокремлює етап друку документа від етапів його набору й редагування. У принципі можна зробити всі зміни, не виводячи документ на папір, а переглядаючи його на екрані дисплея. Використання такої нової інформаційної технології підготовки текстових документів дає можливість багаторазової переробки й "шліфування" матеріалів, поки не буде досягнутий необхідний рівень якості. При цьому забезпечується економія часу, трудових і матеріальних ресурсів.

Операції редагування виконуються при подачі користувачем команд текстового редактора. Команди подаються вибором їх з меню або натисканням потрібних клавіш на клавіатурі ПК.

**Операції редагування над символами**

Щоб вставити один або кілька символів, текстовий редактор повинен перебувати в режимі вставки, а не в режимі заміни. Перемикання між цими режимами здійснює клавіша Ins. Перед вставкою необхідно підвести курсор у потрібну позицію й нажати клавішу з необхідним символом. Символ вставиться в позицію курсору, а частина рядка праворуч (включаючи курсор) зрушиться на одну позицію вправо.

Щоб видалити один або кілька символів, використаються клавіші Del й Backspace. При натисканні клавіші Del віддаляється символ у позиції курсору, права частина рядка зміщуються вліво, курсор залишається на місці. При натисканні клавіші Backspace віддаляється символ у позиції ліворуч від курсору, курсор і права частина рядки зміщуються вліво. Цю клавішу зручно використати, коли треба видалити один або декількох, щойно набраних символів.

Щоб замінити один або кілька символів, текстовий редактор повинен перебувати в режимі заміни, а не вставки. При цьому знову набраний символ просто заміняє символ, що перебував у позиції курсору. Такий спосіб заміни зручний, якщо кількість замінних символів дорівнює кількості що заміняють, а також при редагуванні різних таблиць.

**Операції редагування над рядками**

Основними операціями редагування для рядків є: видалення рядка, поділ одного рядка на дві, злиття двох рядків в одну, вставка порожнього рядка.

Щоб видалити рядок, як правило, треба встановити на неї курсор і подати спеціальну команду. Рядок вилучається з тексту, нижня частина тексту зміщується на один рядок, уникаючи проміжку.

Щоб розділити один рядок на два, як правило, треба встановити курсор у позицію поділу й нажати клавішу Enter, при цьому в позицію, де перебував курсор, буде вміщено "твердий" ознака кінця рядка, рядок розділиться на два, розташованих один під одним.

Щоб з'єднати два рядки в один, як правило, треба встановити курсор у будь-яку позицію за останнім символом першого рядка й нажати клавішу Del. При цьому видаляється невидимий на екрані роздільник рядків і два рядки з'єднуються в один.

Щоб вставити між двома рядками порожній рядок, як правило, треба встановити курсор у будь-яку позицію за останнім символом першого рядка й нажати клавішу Enter, що приведе до появи "твердого" ознаки кінця рядка, між рядками виникне новий порожній рядок.

**Операції редагування фрагментів**

Фрагментом називається безперервна частина тексту. Виділення фрагменту робить його об'єктом наступної команди. Виділити фрагмент – значить "підсвітити" його за допомогою миші або клавіатури. Даний фрагмент може складатися з одного або декількох символів, слова, рядка, пропозиції, абзацу або навіть усього документа.

Фрагмент може бути рядковим, блоковим або лінійним. Рядковий фрагмент складається з послідовності цілих рядків. Блоковий фрагмент звичайно поєднує частини рядків, що утворять у сукупності прямокутник. Лінійний фрагмент може містити в собі послідовність (необов'язково) цілих рядків (рис. 3.43). Існують спеціальні способи виділення для слова, рядка, пропозиції, абзацу, наприклад, подвійний або потрійний щиглик мишею в редакторі Word.

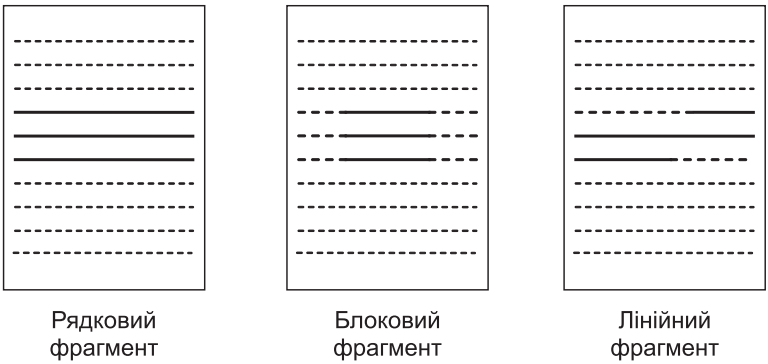



Рисунок 3.43. Види фрагментів тексту

Найчастіше використовувані операції в текстових процесорах – копіювання, переміщення й видалення фрагментів тексту. Ці операції можуть виконуватися як з буфером проміжного зберігання – буфером обміну, так і без нього.

Буфером проміжного зберігання (буфером обміну) називається частина оперативної пам'яті для тимчасового зберігання фрагмента, що копіюється чи переміщується.

У Лексиконі для DOS, наприклад, такі операції виконуються тільки через буфер. У процесорах MultiEdit й Word – як за допомогою буфера, так і без нього. У текстовому процесорі Word можливе перетаскування фрагмента документа мишею на нове місце без допомоги буфера обміну.

Технологія виконання операцій, що використають буфер проміжного зберігання, містить у собі кілька етапів:

- ✧ виділення фрагмента тексту;
- ✧ перенос або копіювання виділеного фрагмента в буфер проміжного зберігання;
- ✧ переміщення курсору в потрібне місце документа;
- ✧ вставка фрагмента з буфера в місце документа, зазначене курсором. Необхідно пам'ятати, що в буфері проміжного зберігання може вміщувативатися тільки один фрагмент тексту й кожен новий фрагмент, поміщений у буфер, заміняє попередній.

На рис. 3.44 ілюструється робота буфера проміжного зберігання: фрагмент тексту переноситься в нове місце документа.

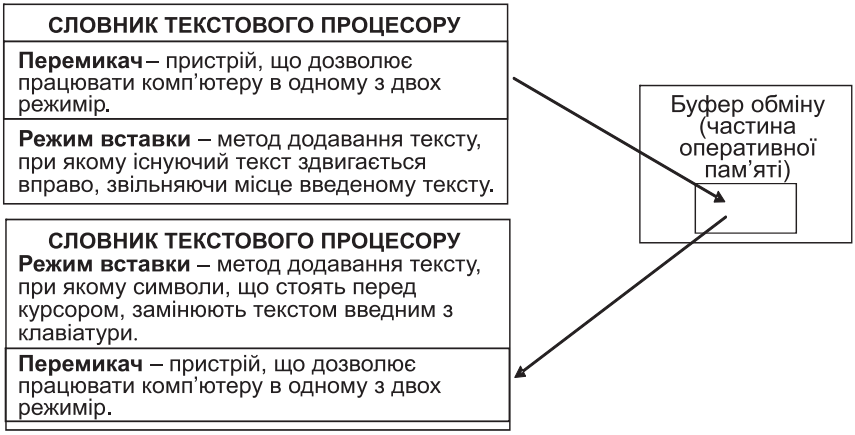



Рисунок 3.44. Переміщення виділеного фрагмента тексту за допомогою буфера проміжного зберігання

Оскільки всі додатки Windows працюють через загальний буфер обміну, операції із фрагментами тексту можуть зачіпати не тільки доку-

менти, створювані в текстовому процесорі Word, але й в інших додатках. Так, ви можете перенести (скопіювати) позначений фрагмент із вашого поточного документа, створюваного в редакторі Word, у який-небудь документ, створений у табличному процесорі Excel, і навпаки.

**Видалення тексту**

У сучасних текстових процесорах можливо видалити символ, слово, рядок, фрагмент тексту (рядковий або блоковий).

При цьому видалення символу здійснюється за допомогою клавіш Backspace та Delete.

Видаленню слова, рядка, пропозиції або будь-якого іншого фрагмента тексту передує попереднє виділення відповідного елемента тексту, а потім застосування або клавіші Delete, або операції видалення. Місце, в якому розташовувався вилучений елемент тексту, автоматично заповниться розміщеним після нього текстом.

Вилучений фрагмент тексту може бути поміщений у буфер проміжного зберігання (див. рис. 3.44), як і при виконанні операції переміщення. Тоді при необхідності його можна відновити, якщо в буфер не був зроблений новий запис.

**Операція відкочування**

Багато програм, у тому числі й текстові процесори, дозволяють виконувати операцію відкочування. Для реалізації цієї операції, текстовий процесор фіксує послідовність дій зі зміни тексту. Спеціальною командою відкочування Undo (Скасувати) користувач може повернути документ у стан, який був кілька кроків назад, тобто "відкотитися". Глибина можливого відкочування залежить від використовуваної програми. Поряд з командою відкочування існує команда скасування відкочування Redo (Повторити).

### 3.3.3.3 Форматування тексту

Операції форматування містять у собі розбивку тексту на рядки (у рамках абзацу) і сторінки, вибір розміру, виду й накреслення шрифту, розташування абзаців, відступів і відбиттів між абзацами. Ці операції виконуються різними текстовими процесорами з різним ступенем автоматизації. Наприклад, розбивку на рядки й сторінки Word виконує автоматично (ця властивість – відмітна риса потужних текстових процесорів). Лексикон автоматично розбиває текст тільки на рядки, а розбивку на сторінки виконує лише за спеціальної команди. MultiEdit і те й інше виконує за відповідної команди.

У чому суть форматування? Це здатність текстового процесора створювати оформлення документа:

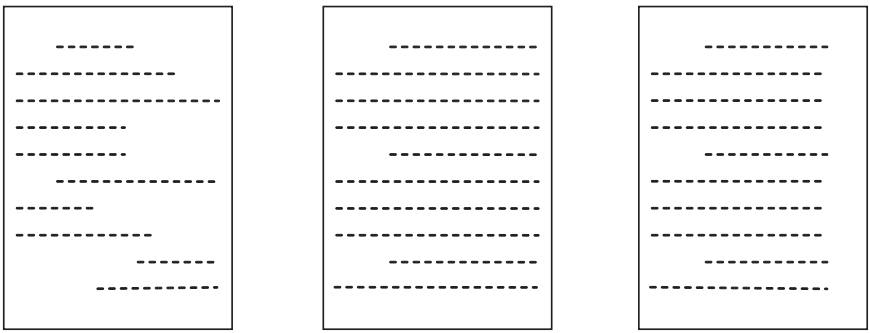
– задавати границі робочого поля (смуги набору), визначаючи поля зверху, знизу, ліворуч, праворуч;

– встановлювати міжрядковий інтервал (розрідженість рядків на сторінці) і міжлітерний інтервал у слові;

урівнювати текст – центрувати, притискати до лівої або правої границі, рівномірно розподіляти слова в рядку; використати різне шрифтове оформлення. У результаті подачі команд форматування можна вирівняти ліві й праві границі рядків за рахунок вставки й видалення "м'яких" пробілів між словами, зміни місця розташування "м'яких" знаків кінця рядка. Перед використанням даної команди потрібно встановити бажані границі розміщення тексту. Команда може виконуватися як для одного абзацу, так і для всього текстового документа. На рис. 3.45 (а) показаний текст до форматування (рядка мають різну довжину), на рис. 3.45 (б) цей текст уже відформатований (праві границі рядків вирівняні), на рис. 3.45 (в) текст переформатирован з метою зміни (зменшення) довжини рядка.

Форматування нерідко застосовується стосовно абзацу.

**Абзац** – фрагмент тексту, процес уведення якого закінчується натисканням на клавіші Enter. Таке натискання варто розглядати як команду, за якою здійснюється закриття поточного абзацу. У більшості текстових процесорів при завершенні абзацу натисненням клавіші Enter новий абзац успадковує стиль попереднього. Форматування абзацу впливає на весь абзац. Воно ніби зберігається в символі, яким закінчується кожен абзац. Якщо видалити цей символ, то попередній йому текст стане частиною наступного абзацу. У текстовому редакторі Word цей символ має вигляд "¶". Він може бути видний при відображенні на екрані символів, що не друкують.



а) б) в)

Рисунок 3.45. Фрагмент тексту до і після форматування

### 3.3.3.4 Друк тексту

Одержання видрукуваного тексту є, як правило, головною метою підготовки документу за допомогою текстового редактора на ПК. Етап друку складається з операцій підготовки тексту до друку та самого процесу друку.

До операцій підготовки тексту до друку ставляться поділ на сторінки, нумерація сторінок, зміна шрифту, виділення елементів тексту при друку, завдання заголовка й підложки сторінок.

Для поділу тексту на сторінки користувач повинен задати параметри сторінки й відстань між рядками при друці. Спеціальні команди редактора управляють розташуванням текстового матеріалу на аркуші паперу. З їхньою допомогою можна задати розмір полів ліворуч і праворуч від тексту, відстань між рядками, кількість порожніх рядків у верхній і нижній частинах аркуша. На екрані дисплея границя сторінки вказується найчастіше горизонтальною лінією із крапок або рисок.

За аналогією із пробілами й роздільниками рядків існують “м’які” й “тверді” роздільники сторінок. Установка “м’яких” роздільників сторінок виробляється текстовим редактором, при редагуванні вони можуть змінювати своє положення, наприклад, після видалення частини тексту.

Встановлення “твердих” роздільників сторінок робить користувач. “Тверді” роздільники сторінок потрібні в тих випадках, коли подальший друк повинен здійснюватися з нової сторінки, наприклад наприкінці глави, перед початком таблиці, яку треба обов’язково вмістити на одній сторінці, тощо. Багато текстових редакторів здійснюють нумерацію сторінок автоматично або за певною командою. При цьому номер може розташовуватися у верхній або нижній частині сторінки, ліворуч, праворуч або в центрі рядка. Деякі редактори дозволяють здійснити нумерацію, що починається не з одиниці.

Спектр можливостей, що надається текстовими редакторами для підготовки до документів до друку досить широкий. Склад операцій і спосіб їх задання повністю визначаються конкретним текстовим редактором.

При підготовці тексту до друку корисно переглянути документ на екрані. Це дає можливість виявити невдалі результати роботи виконавця (наприклад, коли заголовок розміщується на одній сторінці, а сам розділ – на іншій) і виконати додаткове редагування тексту (наприклад, вставку зайвого порожнього рядка перед заголовком з наступним новим розміщенням “м’яких” роздільників сторінок).

Властиво друк є заключною операцією етапу друку документа. Перед її виконанням може виникнути необхідність подачі спеціальних команд редактори й встанолення режимів друку на панелі управління принтера.

Установка режимів друку на принтері залежить від технічних можливостей самого принтера й докладно описується в посібнику з його експлуатації.

Надається можливість задання певних режимів друку: із зупинкою після кожного аркуша, безперервного друку чи друку частини документу (виділеного фрагменту, декількох аркушів). Універсальні текстові редактори надають можливості для налаштування відповдіних параметрів для різних моделів принтерів.

3.3.3.5 Ведення архіву текстів

Використання ПК для підготовки текстових документів ставить перед користувачами завдання створення й ведення архіву текстів на магнітних дисках (МД). Ведення архіву містить у собі операції: перегляд змісту архіву; читання та запис текстів в архів; копіювання, перейменування, видалення текстів.

*Перегляд змісту архіву на МД.* Перегляд змісту – нерідко використовувана операція, буває необхідна для пошуку дискети, що містить потрібний текст, для аналізу вмісту МД перед виконанням сервісних функцій над файлами. На екран дисплея видаються короткі відомості про вміст потрібного диска. Як правило, ці відомості обмежуються ім'ям файлу разом з розширенням.

*Читання тексту із МД.* Операція виконується при введенні користувачем спеціальної команди редактору. Для операції читання документа із МД першорядним є правильне завдання імені файлу. Як правило, якщо файл розташований у поточному підкаталозі, то досить задати тільки ім'я файлу. У іншому випадку треба задати повний шлях до файлу, наприклад:

C:\KNIGA\REDAKTOR\GLAVA1 .TXT

*Запис тексту на МД.* Операція виконується при подачі користувачем спеціальної команди редактору. Існує можливість записати на МД весь текст або його частину. Після подачі команди текстовий редактор запитує в користувача ім'я, під яким треба записати текст у вигляді файлу на МД. Особливості завдання імені такі ж, як і для операції читання із МД. Вибір розширення імені залежить від типу текстового документа.

### 3.3.4 Робота з текстом

3.3.4.1 Мінімальний набір типових операцій

Мінімальний набір типових операцій включає операції, вироблені над документом у цілому, над абзацами документа й над його фрагментами.

**Операції, вироблені над документом у цілому**

До операцій, виробленим над документом, ставляться (рис. 3.46):

- ✧ створення нового документа;

- ✧ завантаження попередньо створеного документу;
- ✧ збереження документа;
- ✧ видалення документа;
- ✧ роздруківка документа.




Рисунок 3.46. Операції вироблені над документом

**Операції, вироблені над абзацами документа**

Абзац є ключовим елементом у структурі документа для багатьох текстових процесорів.

До операцій, виробленим над абзацами відносять установку границь абзаців й абзацних відступів, вирівнювання, а також включення переносу слів (рис. 3.47).

*Установку границь абзаців* здійснюється за допомогою маркерів відступів, що перебувають на координатній лінійці, або відповідними командами меню.

*Вирівнювання (виключка)*. Розрізняють чотири види горизонтального (уліво, вправо, по центру, по ширині) і три види вертикального вирівнювання (нагору, униз, по висоті).

*Перенос*. При виключеному режимі автоматичного переносу слово, що не помістилося на рядку, повністю переноситься на наступний рядок. Для поліпшення зовнішнього вигляду тексту використають режим переносу. При ручному варіанті переносу користувач сам визначає місце переносу, уводячи дефіс, і твердим перекладом каретки (натисканням на клавішу Enter) переходить на наступний рядок. Використання такого режиму переносу приводить до необхідності видалення дефісів при повторному форматуванні тексту документа.

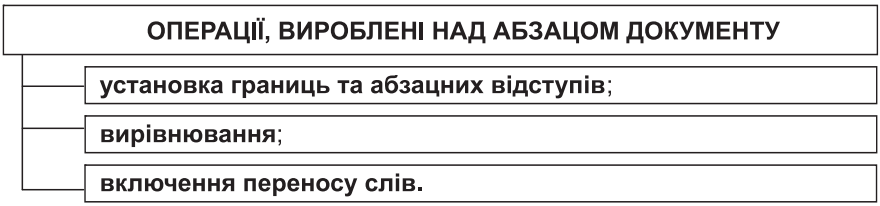



Рисунок 3.47. Операції вироблені над абзацом документу

При включеному режимі автоматичного переносу реалізується м'який варіант переносу: текстовий процесор сам ділить слово на склади й переносить його найкращим способом. Цей режим не створює ніяких труднощів при повторному форматуванні.

**Операції, вироблені із фрагментами тексту** (рис. 3.48)

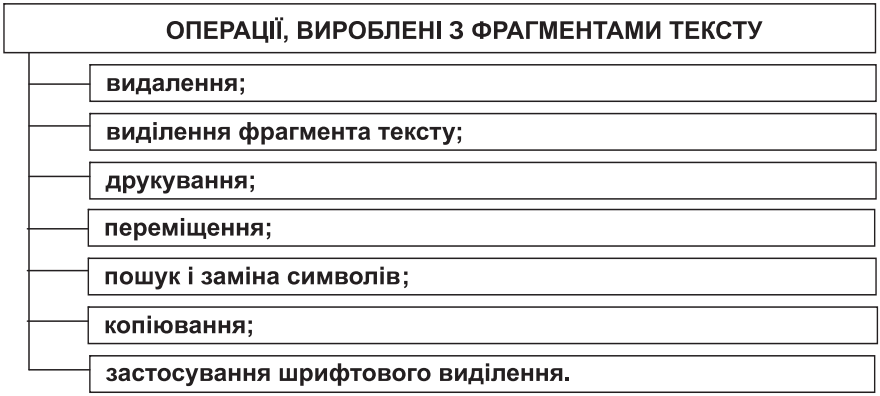



Рисунок 3.48. Операції, вироблені із фрагментами тексту

Ці операції включають виділення фрагмента тексту, його переміщення, копіювання або видалення, які були розглянуті в попередньому розділі глави. Крім того, виділений фрагмент тексту можна надрукувати, здійснювати пошук і заміну символів, застосувати шрифтове виділення й ряд інших операцій.

3.3.4.2 Розширений набір типових операцій

**Контекстний пошук і заміна**

Один з видів пошуку – ***за місцем*** – полягає в попередній розмітці тексту, це зручно при роботі, що вимагає багаторазового звертання до певних місць документа. Для такого пошуку використаються, за аналогією із за-

кладками у звичайній книзі, команди типу “створити закладку”, “знайти потрібну закладку”, “забрати закладку”. Такі “закладки” найчастіше позначаються цифрами або буквами алфавіту.

Інший вид пошуку – ***за зразком*** – полягає в тому, що задається деякий зразок (символ, слово, група слів або ланцюжок символів) і задається команда пошуку. Текстовий редактор починає перегляд документа. Перегляд припиняється щораз при виявленні заданого зразка, на екран видається відповідний фрагмент тексту й користувач може зробити потрібні операції. Потім можна подати команду на продовження або припинення пошуку.

Пошук за зразком зручний, наприклад, при заміні терміна в документі. Так, якщо потрібно замінити в тексті слово “особистий” на слово “персональний”, як зразок для пошуку можна задати “особ”, щоб пошук не залежав від відмінкових закінчень. Текстовий редактор буде послідовно підводити курсор до кожного знайденого зразка, а заміну потрібно буде робити вручну з урахуванням роду, числа, відмінку й інших граматичних особливостей. Команда заміни використовується для заміни одного контексту на інший. Перед подачею задаються зразок пошуку й зразок для заміни. Команда може виконуватися як за всим текстом, так і за його виділеним фрагментом. Така команда може бути використана, наприклад, для заміни в текстовому документі слова “ЕОМ” на слово “ПК” (рис.3.49).

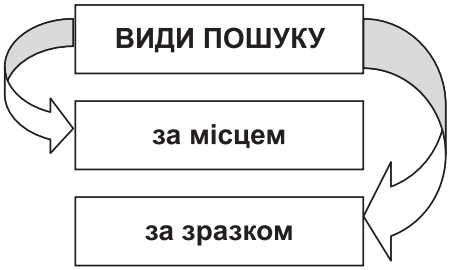



Рисунок 3.49. Контекстний пошук

Команда заміни часто має різні режими виконання (рис. 3.50):

- одноразова й глобальна заміна. Режим одноразової заміни припиняє пошук після знаходження першого замінного елемента. Глобальна заміна заміняє всі входження замінного елемента.
- автоматична й ручна заміна. Режим ручної заміни вимагає підтвердження користувача на заміну після знаходження кожного замінного елемента. Автоматичний режим такого підтвердження не вимагає.
- чутливість і нечутливість до рядкових і прописних символів. При виборі режиму, чутливого до рядкових і прописних символів, регістр, у якому відображений замінний елемент, має значення

при його пошуку. Іншими словами, якщо ви задали замінний елемент як ibm, текстовий процесор не зверне уваги на встретившееся йому в тексті слово IBM.

– напрямок пошуку. Звичайно команда пошуку й заміни реалізує пошук, починаючи від позиції тексту, займаної в цей час курсором, і до кінця документа. У ряду текстових процесорів напрямок пошуку можна змінити на зворотне.

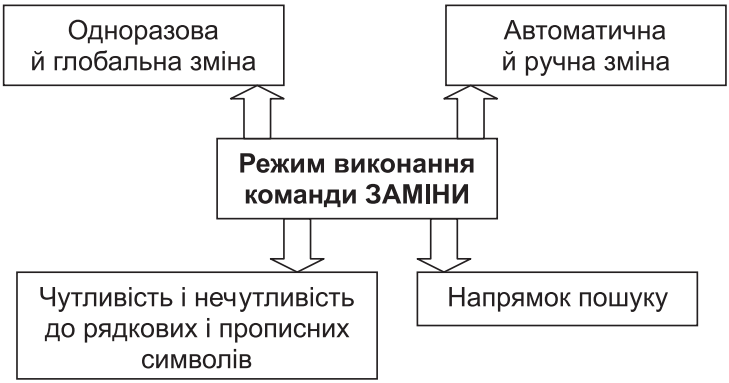



Рисунок 3.50. Режими виконання команди заміни

У ряді текстових процесорів (наприклад, Word) за допомогою спеціальних символів реалізується мова запитів, що дає можливість здійснювати контекстний пошук за складними критеріями.

**Операції збереження**

Операція збереження записує відредагований документ, що перебуває в оперативній пам'яті, на диск для постійного зберігання. Тип збереженого документа звичайно привласнюється текстовим процесором автоматично. Наприклад, у текстовому процесорі Word документу привласнюється тип .DOC. Більшість текстових процесорів використають наступні три операції збереження даних:

1) Зберегти й продовжити редагування. Ця операція виконується періодично, гарантуючи схоронність більше свіжої копії створюваного документа на випадок можливої його втрати.

2) Зберегти й вийти. Ми використаємо цю операцію для збереження відредагованого документа й виходу в операційну систему.

3) Вийти без збереження. Ця операція дозволяє вийти в операційну систему без збереження документа, з яким працювали.

У текстових процесорах, що мають багатовіконний режим роботи, передбачаються одна операція виходу й окремі операції збереження й за-

криття вікон. Якщо при виході користувач закриває вікно та залишає не збереженим змінений документ, то текстовий процесор додатково запитує, чи варто зберегти чи документ ні.

Текстові процесори за допомогою резервних файлів забезпечують захист створених документів від можливої втрати. Для цього спеціальною командою збереження забезпечується режим, коли одночасно зберігаються два файли того самого документа – поточний і резервний. Після внесення змін у документ попередня його версія автоматично зберігається як резервний файл: йому привласнюється розширення .ВАК, а відредагована версія розглядається як поточний файл. Тип поточного файлу визначає сам користувач або він привласнюється за замовчуванням (наприклад, .TXT, або .DOC). У такий спосіб зберігається тільки остання, змінена версія документа; більш ранні версії губляться. Хоча зміст поточних і резервного файлів не однаково, можливість використання останнього (у випадку ушкодження або втрати поточного файлу) представляється дуже ціной. Автоматичне збереження резервних файлів може виявитися недостатнім для забезпечення захисту важливих документів і програм. У цьому випадку створюються додаткові копії, що зберігаються окремо від основних.

Важливим фактором захисту створюваних документів є функція автозбереження, що може виконуватися як звичайна операція збереження або як спеціальна операція збереження поточного стану текстового процесора в спеціальному файлі. В останньому випадку при аварійному припиненні роботи цей стан може бути відновлено, включаючи вміст всіх вікон, положення курсорів у вікнах і т.п.

Зберігайте результати своєї роботи. Для цього кожні 10-15 хвилин виконуйте операцію збереження або встановите режим автозбереження, якщо він є в текстовому процесорі.

**Перевірка правопису слів і синтаксису**

Режим перевірки правопису й синтаксису виконується спеціальними програмами (Speller / checker), які можуть бути автономними, наприклад Орфо, або убудованими в текстовий процесор. Ці програми значно розрізняються по своїх можливостях. Найбільш потужні з них перевіряють не тільки правопис, але й відмінювання, дієвідміну, пунктуацію й навіть стиль. Зазначений режим використають для контролю одного слова, сторінки або цілого документа. Зазначене слово зіставляється з його написанням у словнику й у випадку будь-яких невідповідностей видається на екран для редагування. При цьому користувачеві пропонується наступний вибір:

1) провести виправлення;
2) ігнорувати помилку;
3) додати дане слово в допоміжний словник.

Багато текстових редакторів пропонують додаткові послуги (наприклад, варіанти написання слова), що полегшують виправлення помилок. Але пам'ятайте, що можливості програми залежать від повноти словника. Тому постійно поповнюйте допоміжний словник, вносячись у нього слова, відсутні у вихідному словнику.

Крім перевірки помилок пунктуації й видачі пропозицій щодо їхнього усуненню цей режим забезпечує виявлення деяких помилок стилю, зокрема неправильне використання заголовних і малих літер, повторення того самого слова кілька разів підряд, відсутність пробілу між словами, відсутність других лапок і т.п. Всі зазначені помилки виявляються на основі порівняння розробленого тексту з основними правилами, що зберігаються в пам'яті. Помітимо, що використовуваний у рамках текстового процесора набір таких правил істотно обмежений.

**Словник синонімів**

Словник синонімів допоможе уникнути повторень і зробить елегантним ваш стиль викладу. Його використання надзвичайно просто. Наведіть курсор на слово, що цікавить вас, і введіть команду перегляду словника синонімів. Текстовий редактор видасть на екран кілька синонімів. Виберіть той, котрий вам більше подобається, і він автоматично буде перенесений у документ.

**Установка загальних параметрів сторінки**

Розрізняють логічну й фізичну сторінки. Фізична сторінка (у деяких текстових редакторах замість цього терміна використається термін "розмір паперу") звичайно має деякий стандартний розмір, наприклад 210x297 мм (формат А4), а логічна сторінка утвориться на поле фізичної за винятком установлених користувачем границь (рис. 3.51). Кількість даних на логічній сторінці визначається, з одного боку, щільністю печатки (кількістю знаків на рядку), а з іншого боку – розрідженістю рядків (інтервалом між рядками). Якщо ви збираєтеся працювати з подвійними аркушами, конвертами або наклейками, вам належить установити нові розміри фізичної сторінки.

Аналогічно тому, як здійснюється твердий і м'який переклад каретки на новий рядок, текстовий процесор здійснює м'який і твердий перехід на нову сторінку.

Існує також *команда заборони розриву сторінок*, що використається, коли ви хочете, щоб певна частина документа (наприклад, таблиця) перебувала на одній сторінці. У цьому випадку ви повинні поставити команду заборони розриву сторінок перед таблицею, що цікавить вас, для запобігтання її розриву.

При розриві абзацу багато текстових процесорів забезпечують контроль за, так званими, "висячими рядками". **Висячим рядком** називається

перший рядок або заголовок нового абзацу, що виявився на останньому рядку сторінки (Widow), або останній рядок абзацу, що виявився на початку сторінки (Orphan). Розміщення абзацу при його розриві може регулюватися по-різному. Наприклад, не менш двох рядків наприкінці сторінки й не менш трьох рядків на початку.

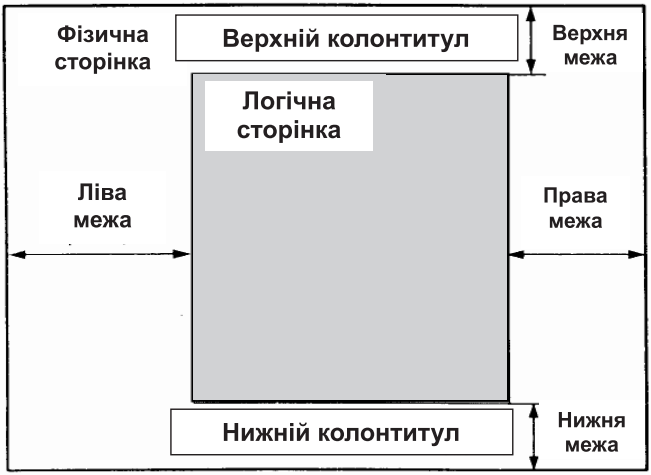



Рисунок 3.51. Співвідношення логічної й фізичної сторінок

Для введення *нумерації сторінок* у створюваному вами документі текстовий процесор запропонує спеціальне меню, у якому ви зможете вказати всі умови нумерації, що цікавлять вас: місце розташування на аркуші номера сторінки, відмова від нумерації першої сторінки, використання колонтитулів й інших. Номера сторінок проставляються в колонтитулі.

**Колонтитулом** називається заголовне дане, таке, що поміщається на початку або кінці кожної сторінки документу. Колонтитули звичайно містять номера сторінок, назва глав і параграфів, назва й адреса фірми й т.п. Колонтитули можуть розрізнятися для парних і непарних сторінок, а також для першої сторінки й наступних. Використання колонтитулів дозволяє краще орієнтуватися в документі, а також використати додаткові можливості.

**Використання шаблонів і стилів**

Для уніфікації структури й зовнішнього вигляду документів використаються шаблони. **Шаблоном** називають спеціальний вид документа, що представляє основні засоби форматування створюваного документа.

Шаблон як поняття містить у собі ряд елементів:

- ✧ текст або форматування, які однакові в кожному документі певного типу;
- ✧ стилі;
- ✧ елементи автотекста;
- ✧ макрокоманди;
- ✧ панелі інструментів;
- ✧ набір меню й "гарячих клавіш".

Шаблон можна вважати свого роду порожнім документом із заданими властивостями, що "накладаються" на створюваний документ (або на основі якого будується новий документ). Задаються як властивості самого документа, так і властивості й способи роботи з документом. Наявність шаблона полегшує створення, редагування й оформлення документа, дозволяє порівняно просто створювати однотипні документи.

Шаблони дозволяють становити й зберігати універсальні бланки документів різного типу: листів, службових записок, доручень, платіжних доручень. Складовою частиною шаблонів є стилі. Стиль як елемент шаблона призначений для зовнішнього оформлення документа і його абзаців. Стиль має ім'я, задаючи яке користувач задає певний набір параметрів форматування, зібраних разом.

**Стиль –** це набір форматуючих команд, що зберігається під своїм ім'ям для багатократного використання. Стилі спрощують завдання оформлення й зміни виду документа, забезпечують погодженість подання окремих частин документа або документів одного типу, заощаджують час на процес форматування.

Текстовий процесор Word має більше число стандартних (вбудованих) стилів, частина з них є стилями символів, інші – стилями абзаців. Список доступних стилів залежить від шаблона, обраного в момент створення документа. За замовчуванням у всіх нових документах, заснованих на шаблоні Normal, Word використає стиль Звичайний, у якому встановлені основні параметри форматування: шрифт – Times New Roman, розмір шрифту – 10 пунктів, вирівнювання по лівому краї, одинарний міжрядковий інтервал.

Починаючи становити певний документ, ви спочатку викликаєте шаблон цього типу документів, а вже потім заповнюєте його. Складання документа при цьому зводиться до заповнення його певних полів текстом. Один раз зроблений на основі стандартів шаблон може надалі багаторазово використатися для створення документів певного виду.

**Використання макросів**

**Макросом** називають файл, у якому зберігається програма послідовності дій, задана користувачем. Макрос має унікальне ім'я.

За допомогою макросів можна автоматизувати багато типових технологічних етапів при роботі з документами, наприклад, макрос, що виконує послідовність команд по створенню стилю для кожного абзацу документа. Після виклику макросу записана в ньому послідовність дій або команд буде в точності відтворена.

Макрос створюють двома способами:

- ✧ автоматично в спеціальному режимі текстового процесора шляхом запису послідовності дій користувача;
- ✧ програмуванням послідовності команд, що підлягають виконанню.

Макрос може зберігатися в самому файлі документа або в спеціальному стильовому файлі як елемент загального документа.

**Злиття документів**

У потужних текстових процесорах є можливість *злиття документів*. Для виконання цієї процедури необхідно мати:

- ✧ головний документ, що містить постійну інформацію;
- ✧ документ-джерело для зберігання змінної інформації.

Головним документом може бути стандартна форма довідки або запрошення, стандартний лист, написи на конверті, тобто форма будь-якого стандартного тексту й так далі.

Документ-джерело повинен являти собою базу даних (або таблицю), що складається з однотипних записів. У документі-джерелі втримуються дані, якими буде заповнюватися головний документ.

Розглянемо ідею злиття документів на прикладі.

Писати листа численним адресатам того самого змісту – малоцікаве заняття. Текстовий процесор автоматизує цей трудомісткий процес, з'єднуючи постійну частину листа з даними, що характеризують адресатів. При цьому користувач повинен створити спеціальний адресний файл. Адресний файл складається з однотипних записів з іменами й адресами клієнтів. Кожен запис містить у собі певне число полів, у які записуються дані кожного клієнта. Так, у перше поле завжди записується прізвище, у друге – ім'я, у третє – по батькові, у четверте – адресу й т.п..

Допустимо, ви хочете розіслати листа всім абітурієнтам, що здали іспити й надійшов в Університет цього року. Для цього треба підготувати загальну частину листа й адресний файл всіх зарахованих абітурієнтів (рис. 3.52). У загальній частині листа спеціальними операторами (на рис. 3.52 – це F1", F2", F3" й F4") ви вкажете, де і які поля з адресного файлу повинні бути поміщені для кожного одержувача. У результаті текстовий процесор автоматично надрукує всі листи, що містять крім основної частини адреса й дані кожного абітурієнта.

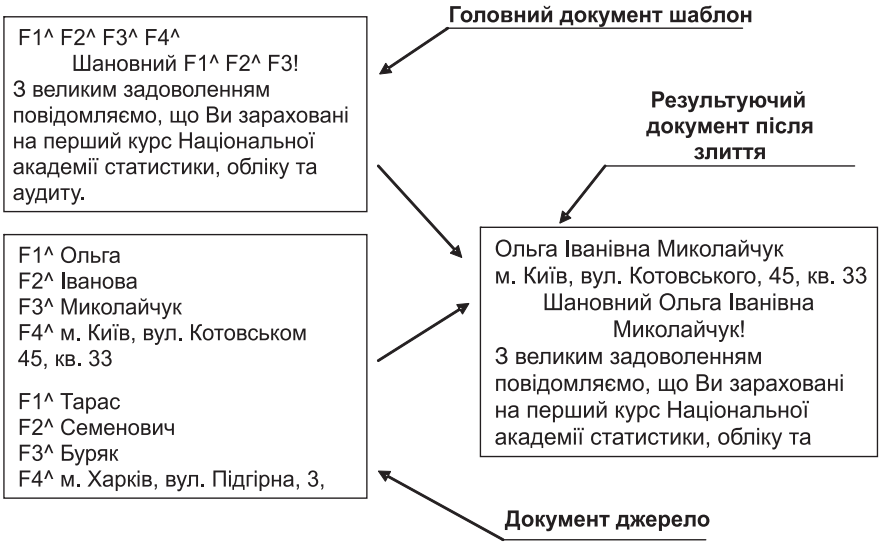



Рисунок 3.52. Злиття листів

### 3.3.5 Призначення й області застосування табличних процесорів

Електронна таблиця є одним з найпоширеніших і зручних інформаційних технологій для обробки даних, що вимагають арифметичних обчислень і представлення в табличному вигляді. Типовим прикладом застосування електронних таблиць є складання кошторисів, фінансових планів і бюджетів, а також матричні обчислення в інженерних розрахунках.

Технології електронних таблиць реалізуються в спеціальних програмних продуктах – табличних процесорах.

Практично, в будь-якій області діяльності людини, особливо при рішенні планово-економічних завдань, бухгалтерському й банківському обліку, проектно-кошторисних роботах і т.п., виникає необхідність представляти дані у вигляді таблиць; при цьому частина даних періодично змінюється, а частина – розраховується за формулами. Для проведення таких робіт на комп'ютері були розроблені електронні таблиці.

Таким чином, електронні таблиці призначені для зберігання й обробки інформації, представленої в табличній формі.

**Електронні таблиці** – це двовимірні масиви, що складаються зі стовпців і рядків. Для керування електронною таблицею створені спеціальні програмні продукти – табличні процесори, що дозволяють не тільки створювати електронні таблиці, але й автоматизувати обробку табличних даних.

За допомогою електронних таблиць можна виконувати різні економічні, бухгалтерські й інженерні розрахунки, а також будувати різного

роду діаграми, проводити складний економічний аналіз, моделювати й оптимізувати рішення різних господарських ситуацій, тощо.

Табличні процесори забезпечують:

- ✧ введення, зберігання й корегування даних;
- ✧ автоматичне проведення обчислень за заданими формулами;
- ✧ наочність і природну форму документів, що представляються користувачеві на екрані;
- ✧ побудова різного роду діаграм і графіків на основі табличних даних, що особливо важливо при вирішенні деяких завдань економічного характеру;
- ✧ оформлення й виведення на друк електронних таблиць;
- ✧ створення підсумкових і зведених таблиць;
- ✧ роботу з електронними таблицями як з базами даних: сортування таблиць, вибірка даних за допомогою запитів;
- ✧ дружній інтерфейс.

Сучасні табличні процесори реалізують цілий ряд додаткових функцій:

- ✧ можливість роботи в локальній мережі;
- ✧ організація обміну даними з іншими програмними засобами, наприклад, із системами керування базами даних;
- ✧ можливість роботи з тривимірною організацією електронних таблиць;
- ✧ розробка макрокоманд, налаштування середовища відповідно до потреб користувача й т. ін.

### 3.3.5.1 Поняття та призначення процесора електронних таблиць Microsoft Excel

**Microsoft Excel –** табличний процесор (редактор), програма для створення й обробки електронних таблиць. (**Excel –** назва електронних таблиць фірми **Microsoft** (від англійського слова ***"excel"*** – перевершувати)). **MS Excel** призначений для: створення та редагування табличнообчислювальних службових документів, оформлення звітів, створення систем пакетів таблично-розрахункових баз даних, програм проведення різноманітних розрахунків, зв'язаних між собою, і їх графічного подання в діаграмах та графіках. Редактор має всі засоби для редагування, збереження та друку створених документів у вибраних форматах. Крім цього, **Excel** дозволяє: спрогнозувати розвиток подій ("що буде, якщо"); підбирати параметри для отримання заданих результатів; знаходити рішення задач і т. ін.

**Електронні таблиці** це аркуші, розліняні на рядки та стовпці, з великою кількістю чисел, з якими проводяться різноманітні розрахунки.

**Важливо**, що розрахунки можуть багато разів уточнюватися з іншими вхідними даними. На папері, кожен раз при заміні даних це потребує переробки всієї таблиці. В **Excel** – достатньо ввести нові дані автоматично отримати новий результат.

Електронні таблиці **особливо зручні для обробки різноманітних списків та відомостей. Ви можете:**

- ✧ сортувати списки, наприклад, за прізвищами;
- ✧ проводити пошук потрібної інформації;
- ✧ вивести на екран тільки потрібну інформацію.

Файли електронних **таблиць Excel** називаються **робочими книгами,** кожна з яких може вміщати **понад 256 робочих таблиць,** принаймні до 65 535 ($2^{16}$-1)**).** Кожен лист електронної книги – сукупність клітин (осередків), утворених 16 384 стовпцями ($2^{14}$) і 1 048 576 рядками ($2^{20}$). Дані параметри змінюються в залежності від версії MS Excel. Кожна таблиця розміщується на окремому аркуші та має ярлик з іменем (Sheet 1, Аркуш 1 і т. ін.).

Після завантаження табличного процесора Excel 2010 користувачеві для обробки даних надається книга (рис. 3.53).

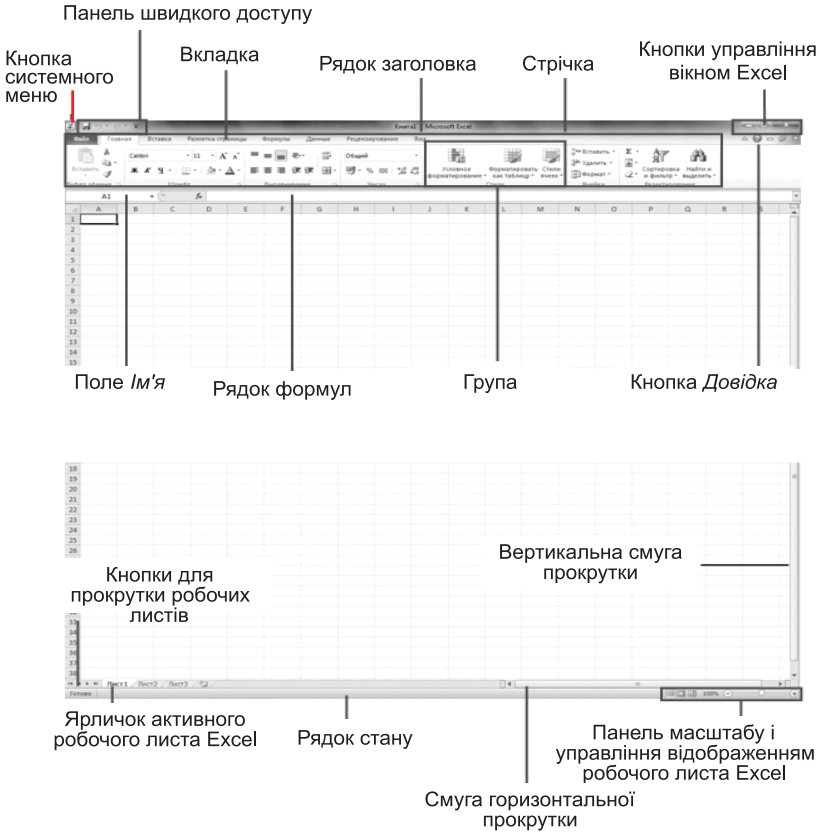



Рисунок 3.53. Елементи вікна табличного процесора Excel 2010

За умовчанням вона складається з трьох електронних листів (у налаштуваннях Excel можна змінити число листів нової книги, встановлене за умовчанням, виконавши команди: команда **Файл → Параметри →** Вкладка **Загальні →** Вказати число листів при створенні нової книги : від 1 до 255).

3.3.5.2 Історія й тенденції розвитку

Табличні процесори розрізняються, в основному, набором виконуваних функцій і зручністю інтерфейсу. Найбільш поширеними електронними таблицями для персональних комп'ютерів є табличні процесори Excel, Lotus 1-2-3, Quattro Pro і SuperCalc.

Ідея створення електронної таблиці виникла в студента Гарвардського університету (США) Дэна Брикліна (Dan Bricklin) у 1979 р. при виконанні нудних обчислень економічного характеру за допомогою бухгалтерських книг. Він і його друг Боб Франкстон (Bob Frankston), який розумівся в програмуванні, розробили першу програму електронної таблиці, названу ними VisiCalc.

VisiCalc незабаром стала однією з найбільш популярних програм. Спочатку вона призначалася для комп'ютерів типу Apple, але потім була трансформована для всіх типів комп'ютерів. Багато хто вважає, що різке підвищення продажів комп'ютерів типу Apple, у той час, було саме пов'язане з можливістю використання на них табличного процесора VisiCalc.

У електронних таблицях – аналогах, що незабаром з'явилися, (наприклад, SuperCalc) основні ідеї VisiCalc були досить вдосконалені.

Новий істотний крок у розвитку електронних таблиць відбувся з появою в 1982 р. на ринку програмних засобів Lotus 1-2-3. Lotus був першим табличним процесором, що інтегрував у своєму складі, крім звичайних інструментів, графіку та можливість роботи із системами керування базами даних. Оскільки Lotus був розроблений для комп'ютерів типу IBM, він зробив для даної фірми те ж, що VisiCalc у свій час зробив для фірми Apple. Після розробки Lotus 1-2-3 компанія Lotus у перший же рік підвила обсяг продажів до 50 млн. дол. і стала найбільшою незалежною компанією – виробником програмних засобів.

Успіх компанії Lotus привів до жорсткості конкуренції, викликаної появою на ринку нових електронних таблиць, таких як VP Planner компанії Paperback Software і Quattro Pro компанії Borland International, які запропонували користувачеві практично той же набір інструментарію, але за значно низькими цінами.

Наступний кроком в розвитку електронних таблиць, була поява в 1987 р. табличного процесора Excel фірми Microsoft. Розробниками було запропоновано користувачам більш простий графічний

інтерфейс у комбінації зі спадаючим меню, значно розширивши при цьому функціональні можливості пакета й підвищивши якість вихідної інформації. Розширення спектра функціональних можливостей електронної таблиці, як правило, веде до ускладнення роботи із програмою.

Розробниками Excel вдалося знайти золоту середину – максимально полегшивши користувачеві освоєння програми й роботу з нею, завдяки чому Excel швидко отримала популярність серед широкого кола користувачів. У даний час Excel займає провідне місце на ринку табличних процесорів, 80% усіх користувачів електронних таблиць воліють Excel, на другому місці за обсязі продажів – Lotus 1-2-3 та потім Quattro Pro.

Наявні на сучасному ринку табличні процесори, здатні працювати в широкому колі економічних і інших додатків та можуть задовольнити потреби практично будь-якого користувача.

### 3.3.6 Основні поняття. Типова структура інтерфейсу

#### 3.3.6.1 Основні поняття

**Електронна таблиця** – автоматизований еквівалент звичайної таблиці, в чарунках якої розміщують або дані (тексти, дати, формули, числа й т.д.), або результати розрахунку за формулами. При зміні цих даних розрахунок виконується автоматично.

***Головна перевага електронної таблиці*** – це можливість миттєвого перерахування всіх даних, зв'язаних формульними залежностями при зміні значення будь-якого операнда.

**Робоча область електронної таблиці** складається з рядків і стовпців, що мають свої імена. **Імена рядків** – це їх номера. Нумерація рядків починається з 1 і закінчується максимальним числом, установленим для даної програми. **Імена стовпців** – це букви латинського алфавіту спочатку від A до Z, потім від AA до AZ, BA до BZ і т.д.

Сучасні програми дають можливість створювати електронні таблиці, що мають більше 1 млн. осередків, хоча для практичних цілей, у більшості випадків, цього не потрібно.

Перетинання рядка й стовпця утворить чарунку таблиці, які мають унікальну адресу. Для вказівки адрес чарунок у формулах використовуються посилання (наприклад, A2 або 34).

**Чарунка (осередок, комірка, клітинка)** – область, обумовлена перетинанням стовпця й рядка електронної таблиці, що має свою унікальну адресу.

**Адреса чарункі** визначається іменем (номером) стовпця та іменем (номером) рядка, на перетинанні яких перебуває чарунка, наприклад A1.

**Посилання** – вказівка адреси чарункі.

В електронних таблицях можна працювати як з окремими чарунками, так і із групами чарунок, які утворять блок.

**Блок чарунок** – група суміжних чарунок, обумовлена за допомогою адреси. Блок чарунок може складатися з однієї чарунки, рядка (або її частини), стовпця (або його частини), а також послідовності рядків і стовпців (або їхніх частин).

**Адреса блоку чарунок** задається вказівкою посилань першої і останньої його чарунки, між якими ставиться розділовий символ – двокрапка ":" або дві крапки підряд "..". Якщо блок має вигляд прямокутника, то його адреса задається адресами лівих верхніх і правої нижньої чарунок, що входять у блок.

Приклад вказування адрес чарунків і блоків (рис. 3.54):

- ✧ адреса чарункі, що розміщена на перетинанні стовпця G і рядка 3, виражається посиланням G3;
- ✧ адреса блоку, утвореного у вигляді частини рядка 1 – A1:H1;
- ✧ адреса блоку, утвореного у вигляді стовпця B – B1:B10;
- ✧ адреса блоку, утвореного у вигляді прямокутника – D4:F5.

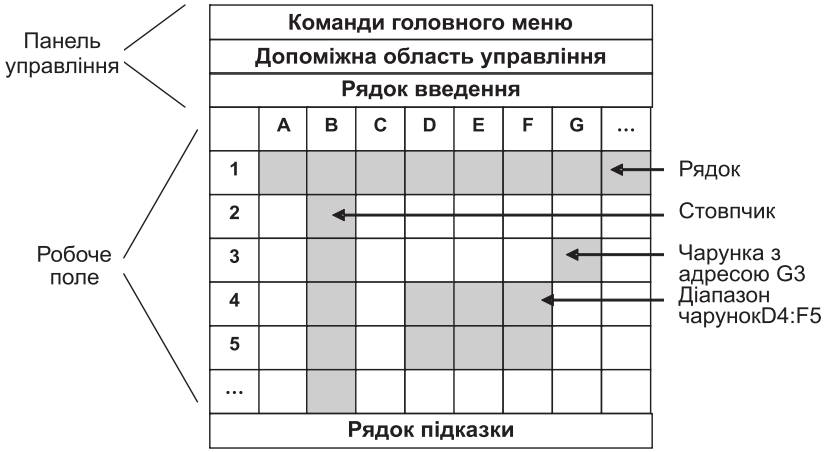



Рисунок 3.54. Вид електронної таблиці на екрані

Кожна команда електронної таблиці зазвичай вимагає вказування блоку (діапазону) чарунок, з використанням яких вона повинна бути виконана.

Блок чарунок може бути зазначений двома шляхами: або заданням з клавіатури початкової й кінцевої адреси чарунок блоку, або виділенням відповідної частини таблиці за допомогою клавіш управління курсором або лінії.

Установками, прийнятими за замовчуванням на рівні всіх чарунок таблиці, як правило, є:

- ✧ ширина чарункі – 9 розрядів;
- ✧ ліве вирівнювання для символьних даних;
- ✧ основний формат для цифрових даних з вирівнюванням вправо.

3.3.6.2 Типова структура інтерфейсу

Відповідно до рис. 3.54, при роботі з електронною таблицею на екран виводяться робоче поле таблиці й панель керування.

**Робоче поле** – простір електронної таблиці, що складає з чарунок, назв стовпців і рядків.

**Панель керування** – частина екрана, що надає користувачеві інформацію про активну чарунку та її вміст, меню та режими роботи.

Панель керування зазвичан включає: головне меню, допоміжну область керування, рядок уведення й рядок підказки. Розташування цих областей на екрані може бути довільним і залежить від особливостей конкретного табличного процесора.

Рядок головного меню містить імена меню основних режимів програми. Вибравши один з них, користувач одержує доступ до спадаючого меню, що містить перелік команд. Після вибору деяких команд спадаючого меню з'являються додаткові підменю.

Допоміжна область керування включає:

- ✧ рядок стану;
- ✧ панелі інструментів;
- ✧ вертикальну й горизонтальну лінійки прокручування.

У **рядку стану** (статусному рядку) розміщено відомості про поточний режим роботи програми, ім'я файлу поточної електронної таблиці, номер поточного вікна й т. ін.

**Панель інструментів** (піктографічне меню) містить певну кількість кнопок (піктограм), призначених для швидкої активізації виконання певних команд меню й функцій програми. Розмір електронної таблиці іноді істотно перевищує ту її частину, що може одночасно розміститися на екрані комп'ютера. Щоб викликати на екран ті області таблиці, які на ньому в даний момент не відображені, використовуються вертикальну та горизонтальну смуги прокручування. Бігунок (движок) смуги прокручування показує відносну позицію активної чарункі в таблиці й використовується для швидкого переміщення по ній. У деяких табличних процесорах на екрані утворяться спеціальні зони швидкого виклику. При натисненні миші в такій зоні викликається відповідна функція. Наприклад, при клацанні миші на координатній лінійці викликається діалог задання параметрів сторінки.

**Рядок уведення** відображає дані, що вводяться в осередок. У ній користувач може переглядати або редагувати зміст поточного осередку. Особливість рядка уведення – можливість бачити, що розміщено у поточній чарункі – формула або функція, а не результат розрахунку. Рядок уведення зручно використовувати для перегляду або редагування текстових і інших даних.

**Рядок підказки** призначений для надання користувачеві інформації щодо можливих дій у цей момент.

Наведена структура інтерфейсу є типовою для табличних процесорів, призначених для роботи в середовищі Windows. Для табличних процесорів, що є в DOS, найчастіше відсутні командні кнопки панелей інструментів і смуги прокручування.

**Поточною** (активною) називається чарунка електронної таблиці, у якій у цей момент перебуває курсор. Активна чарунка звичайно виділяється жирною рамкою. Адреса і вміст поточної чарункі виводяться в рядку уведення електронної таблиці. Переміщення курсору як по рядку уведення, так і по екрану здійснюється за допомогою клавіш управління курсором або миші.

Частина електронної таблиці, що ми бачимо на екрані монітора, називається **поточним** (активним) екраном.

Основні об'єкти обробки інформації – електронні таблиці, розміщаються табличним процесором у самостійних вікнах, і відкриття або закриття цих таблиць є, по суті, відкриття або закриття вікон, у яких вони розміщені. Табличний процесор дає можливість відкривати одночасно безліч вікон, організувати тим самим "богатовіконний режим" роботи. Існують спеціальні команди, що дозволяють змінювати взаємне розташування й розміри вікон на екрані. Не кожний табличний процесор дозволяє працювати в богатовіконному режимі.

**Робоча книга** являє собою документ, що містить кілька аркушів, у які можуть входити таблиці, діаграми або макроси (див. нижче). Ви можете створити книгу для спільного зберігання в пам'яті аркушів, що вас цікавлять, і вказати, яку кількість аркушів вона повинна містити. Робоча книга, як правило, зберігається в одному файлі. Зазначимо, що термін "робоча книга" не є загальноприйнятим. Так, наприклад, табличний процесор Framework замість нього використовує поняття Frame (рамка).

### 3.3.6.3 Інтерфейс MS Excel

У вікні MS Excel розташовується (рис. 3.55):

***рядок заголовку*** в якому розміщується ім'я файлу робочої книги (при розкритті співпадає із рядком головного вікна), кнопка виклику системного меню, кнопки зміни розмірів, кнопка закриття;

***рядок меню;***

***панель інструментів***;

***рядок формул*** (знаходиться під панеллю інструментів), який складається з двох полів: *поля імені (зліва); поля формул.*

**16** ***аркушів робочих таблиць*** з ярликами;

***поле вводу – чистий аркуш***, розбитий на стовпці (по літерах) та рядки (по числах).

***смуги прокрутки***;

***рядок стану*** (інформаційний рядок), який знаходиться в нижній частині екрану та містить інформацію за вибраними параметрами: наприклад, кількість сторінок, значення (середнє, сума, кількість значень, мінімальне чи максимальне значення) виділених чарунок.

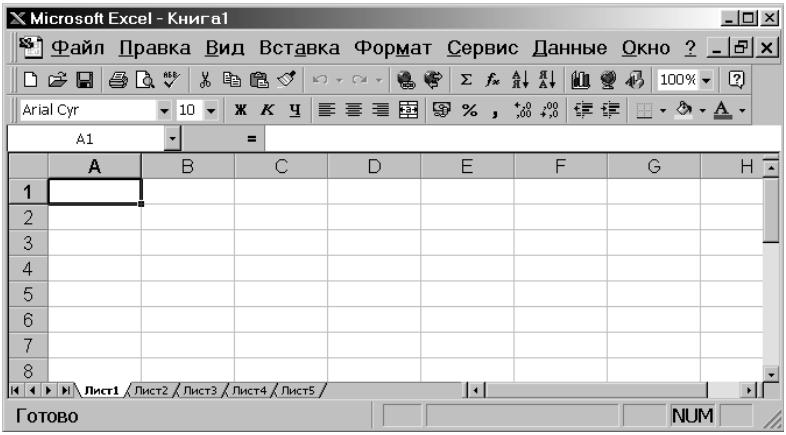



Рисунок 3.55. Вікно MS Excel 2003

**Рядок формул** використовується для введення та редагування даних в чарунках таблиці. В лівій частині рядка формул відображається ім'я активної чарунки. Між полями при вводі інформації в чарунки з'являються кнопки:

- ✧ відміни (червоний хрестик),
- ✧ вводу (зелена галочка),
- ✧ виклику майстра функції.

Щоб закінчить ввід або редагування, потрібно натиснути **Enter** або ☑ в формульному рядку.




Щоб відмовитися від зроблених в чарунках змін (видалити знову введений текст або відмінити редагування), потрібно натиснути **Esc** або **x** в рядку формул.

Стовпці таблиці іменуються латинськими літерами A, B, C,…Z, AA, BB і т.ін. (таблиця може вміщувати до 256 стовпців, останній стовпчик має ім'я IV).

Рядки нумеруються числами від 1 до 16384 (65536).

Перетинання ***j***-го стовпця з ***k***-м рядком створює чарунку з іменем (для посилання) ***jk***. Наприклад, A1, B3, AA184 і т.ін. (використовуються тільки літери латинської абетки).

### 3.3.6.4 Дані, збережені в осередках електронної таблиці. Типи вхідних даних

У кожну чарунку користувач може ввести дані одного з наступних можливих типів: символьні, числові, формули й функції, а також дати, таким чином в чарунку можна помістити значення або формулу.

**Значення** може бути:

- ✧ число;
- ✧ час;
- ✧ дата;
- ✧ текст;
- ✧ логічна константа;
- ✧ код помилок.

**Символьні (текстові) дані** – це будь-яка послідовність символів. Символьні дані можуть містити в собі алфавітні, числові та спеціальні символи. У якості їхнього першого символу часто використовується апостроф, іноді – лапки. Дані текстового типу використовуються для заголовків таблиць, рядків і стовпців, а також для коментарів.

Приклад символьних даних: Відомість по нарахуванню премії; Група № 115.

Якщо в запису присутні інші символи, крім числових, запис сприймається, як текст.

При введенні тексту запис зміщується до лівої сторони чарунки.

**Числові дані** – це числові константи. Вони не можуть містити алфавітних і спеціальних символів, оскільки з ними здійснюються математичні операції. Єдиними виключеннями є десяткова крапка (кома) і знак числа, що розміщується перед ним.

Приклад числових даних: -20; 256,28; 251.

**Excel** розуміє запис, як **число**, якщо в ньому присутні тільки такі символи: *цифри; знак "-"; розподільна буква "E" або "$"*.

При введенні чисел в чарунку запис зміщується до правої сторони чарунки.

Особливим типом вхідних даних є **дати**. Цей тип даних забезпечує виконання таких функцій, як додавання до дати числа (перерахування дати вперед та назад) або обчислення різниці двох дат (тривалості періоду), тобто можуть використовуватися в розрахунках. Дати мають **внутрішній** (наприклад, дата може виражатися кількістю днів від початку 1900 року або порядковим номером дня за Юліанським календарем) і **зовнішній формат**. **Зовнішній формат** використовується для уведення й відображення дат, наприклад: ДД-ММ-ГГ ( 04-10-98); ДД.ММ.ГГ (04.10.98); ДД-МММ (04-янв) і т.д.

Є декілька способів запису дат. Числа розділяються: "." , "/" , "-" , відповідно до правил різних країнах. Наприклад: **19.08.99**, **19/08/99** або **19-08-99.**

Після запису року ніякі символи не вводяться.

Крім даних кожній чарунці можуть бути приписані формати і для неї створені примітки (**спливаючі підказування**).

**Формула** вказує спосіб створення нового значення і складається з:

- ✧ операндів;
- ✧ знаків операцій.

Використання **формул** є головною особливістю електронних таблиць і відкриває практично необмежені можливості їх застосування для різних розрахунків. З їхньою допомогою задається алгоритм отримання нових значень.

**Формули** – це вираз, що складаються із числових величин, адреси осередків і функцій, з'єднаних знаками арифметичних операцій.

В чарунці, яка містить формула, відображено тільки результат обчислень над даними, а формула, за якою здійснюється підрахунок можна відобразити в рядку введення, коли чарунка стане активною.

Формули завжди *починаються з символу* "= " або "+" і містять **операнди** та **знаки операцій**.

В якості **операндів** можуть використовуватися:

- ✧ посилання на чарунки;
- ✧ імена діапазонів;
- ✧ константи;
- ✧ вбудовані функції.

Знаки операцій перераховані в табл. 3.3.

Таблиця 3.3

**Знаки операцій, які можуть використовуватися в формулах**

| Знак | Дія | Знак | Дія |
|---|---|---|---|
| + | додавання | = | дорівнює |
| - | віднімання | ≠ | не дорівнює |
| * | множення | > | більше |
| / | ділення | >= | більше або дорівнює |
| % | відсоток | < | менше |
| ^ | приведення в ступінь | <= | менше або дорівнює |
| & | зчеплення рядків | : | інтервал |
| ; | об'єднання | пробіл | перетинання |

Знак % визначає унарну операцію, тобто потребує одного операнда (наприклад, 16,66 %, де в якості операнда виступає число 16,66), інші знаки визначають бінарні операції та потребують двох і більше операндів (знаки "+" і "–" можуть використовуватися і як унарні, якщо вони стоять перед першим операндом у формулі для вказівки на те, що розміщений за ним операнд має позитивне або від'ємне значення). Зміст більшості операцій такий саме, що й у аналогічних математичних операцій.

Пояснення потребують тільки деякі з них:

Зчеплення рядків (знак амперсанд &) – до рядка першого операнда приєднується з правого боку рядок другого операнда. Наприклад, якщо чарунка **B2** вміщує текст **“Електронна таблиця”**, то записавши в яку не будь іншу чарунку формулу **=B2&”Excel**” і натиснувши **Enter**, отримаємо в цій чарунці результат **“Електронна таблиця Excel”**. В даному прикладі перший операнд – це **B2**, котрий є посиланням на чарунку **B2**, тобто вказує на необхідність переписати інформацію чарунки **B2** в дану чарунку. Другим операндом є вислів “**Excel”** .

**Інтервал (знак :)** – утворює посилання на всі чарунки інтервалу, включаючи граничні. **Наприклад**: **A1:D1** визначає групу чарунок **A1**, **B1**, **C1**, **D1**.

**Об’єднання (**знак **;)** – утворює посилання на всі чарунки операндів. Наприклад, терміни **Продаж** та **Прибуток** це імена, присвоєні діапазону чарунок. Так, якщо ім’я **Продаж** присвоєне чарункам **B2:D2**, а ім’я **Прибуток** – діапазону **C2:C10,** тоді **“Прибуток ; Продаж”** утворюють посилання на чарунки **B2:D2 і C2:C10**.

**Перетинання (знак пробіл )** – утворює посилання на чарунки, *загальні для двох операндів*, наприклад, якщо **Продаж** посилається на чарунки **B2:D2**, а **Прибуток** – на чарунки **C2:C10**, то **“Прибуток␣Продаж”** визначає посилання на чарунку **C2** (знак ␣ визначає натискання на клавіш **Пробіл**).

В **формулі**, як і в звичайних математичних прикладах, можуть використовуватися *декілька знаків операцій*. В цих випадках вони виконуються в порядку, який визначений наступною **таблицею пріоритетів (**табл. 3.4):

Таблиця 3.4

**Таблиця пріоритетів**

| Знак | Дія |
|---|---|
| : | інтервал |
| Пробіл | перетинання |
| ; | об’єднання |
| –, + | унарні операції |
| % | відсоток |
| ^ | приведення в степінь |
| *, / | множення і ділення |
| +, – | додавання і віднімання |
| & | зчеплення рядків |
| =, <>, >, >=, <, <= | порівняння |

**Пріоритет** в цій таблиці *зменшується зверху донизу*, тобто *найвищим пріоритет* має операція **Інтервал**, а *найнижчий* – операція **Порівняння**.

Якщо в формулі зустрічаються декілька *операцій рівного пріоритету*, вони розраховуються *зліва направо*.

Пріоритет операції можна *змінити за допомогою круглих дужок*. **Наприклад, 3+4*5 буде 23, а (3+4)*5 буде 35.**

Під **функцією** розуміють залежність однієї змінної (у) від однієї (х) або декількох змінних ($x_1$, $x_2$, ..., $x_n$), причому кожному набору значень змінних $x_1$, $x_2$, ..., $x_n$ буде відповідати єдине значення залежної змінної певного типу.

**Функція** являє собою програму з унікальним ім'ям, для якої користувач повинен задати конкретні значення аргументів, що розміщуються у дужках після її імені. Функцію (так само, як і число) можна вважати частиною формули. Розрізняють статистичні, логічні, фінансові й інші функції.

Функції вводять у таблицю в складі формул або окремо.

В **Excel** визначено більш ніж **500** різноманітних **вбудованих функцій**, які можна розподілити на **9 категорій**:

*фінансові функції;*
*функції дати та часу;*
*інформаційні функції;*
*математичні та тригонометричні функції;*
*статистичні функції;*
*функції перегляду та посилань;*
*функції баз даних;*
*текстові функції;*
*логічні функції.*

Математичні функції виконують різні математичні операції, наприклад, обчислення логарифмів, тригонометричних функцій, перетворення радіан у градуси й т. ін.

Статистичні функції виконують операції по обчисленню параметрів випадкових величин або їхніх розподілів, представлених безліччю чисел, наприклад, стандартного відхилення, середнього значення й т. ін.

Текстові функції виконують операції над текстовими рядками або послідовністю символів, обчислюючи довжину рядка, перетворюючи заголовні букви в рядкові й т. ін.

Логічні функції використовуються для побудови логічних виражень, результат яких залежить від істинності умови, що перевіряється.

Фінансові функції використовуються в складних фінансових розрахунках, наприклад, визначення норми дисконту, розміру щомісячних виплат для погашення кредиту, амортизаційних відрахувань і ін.

Всі функції мають однаковий формат запису й включають ім'я функції й перелік, що перебуває в круглих дужках, аргументів, розділених комами.

Приведемо приклади функцій, що виконуються найчастіше.

SUM(Список) – статистична функція визначення суми всіх числових значень у Списку. Список може складатися з адрес ча-

рунок і блоків (посилань), а також числових значень. Наприклад: SUM(B5:E5); SUM(A3:E3;230).

AУЕВАСЕ(Список) – статистична функція визначення середнього арифметичного значення всіх перерахованих у Списку величин. Наприклад: AVERAGE(5;20;10;5); AVERAGE(B10:B13;B17).

MAX(Список) – статистична функція, результатом якої є максимальне значення в зазначеному Списку. Приклад: MAX(B3:B8;A3:A6).

IF(Умова, Істинно, Ложно) – логічна функція, що перевіряє на істинність заданої логічної умови. Якщо умова виконується, то результатом функції є значення аргументу “ИСТИНА”. Якщо умова не виконується, то результатом функції стає значення аргументу “ЛОЖЬ”. Приклад: IF(B4<100;100;200) – якщо чарунка B4 містить число менше 100, то функція має значення 100, якщо умова не виконується (тобто вміст чарункі B4 більше або дорівнює 100) – функції привласнюється значення 200.

## КОНТРОЛЬНІ ПИТАННЯ

1. Звідки походить термін “алгоритм”?
2. Яким вимогам повинен відповідати алгоритм?
3. Які способи запису алгоритмів вам відомі?
4. Які недоліки словесного способу запису алгоритму?
5. Які переваги і недоліки представлення алгоритму у вигляді блок-схеми?
6. На які групи можна підрозділити мови програмування високого рівня?
7. Які базові алгоритмічні конструкції вам відомі?
8. Що таке “лінійний алгоритм”?
9. Що таке “алгоритм, що розгалужується”? Чим обумовлюється вибір шляху при галуженні?
10. Що таке “циклічний алгоритм”? Чим відрізняється цикл з передумовою від циклу з постумовою?
11. Які основні функції текстового процесора?
12. Яким вимогам повинна відповідати.сучасна електронна таблиця?
13. Чим табличний процесор відрізняється від бази даних?
14. Назвіть функціональність, важливу для сучасного текстового процесора?
15. 4. Перерахуєте головні функції табличного процесора?
16. У чому полягає додаткова функціональність табличного процесора?
17. Чим табличний процесор відрізняється від бази даних?
18. Що таке “зведені таблиці”, для чого вони потрібні?
19. Дайте визначення поняття “мова програмування”.
20. Що таке “комп’ютерна програма”?
21. У чому полягає процес програмування?
22. Що таке “трансляція” та з яких етапів вона складається?
23. Чи створюється в результаті роботи компілятора програма, яку можна завантажити в постійну пам’ять комп’ютера і виконати?
24. За якими ознаками ви можете класифікувати мови програмування?
25. У чому полягає відмінність структурного програмування від модульного?
26. Назвіть этапи розвитку мов програмування.
27. Назвіть концепції об’єктно-орієнтованого програмування.
28. Дайте визначення абстракції.

## ТЕСТИ ДЛЯ САМОКОНТРОЛЮ

1. Буфер обміну – це:
- ✧ файл певного вигляду, який створюється водночас з відкриттям нового документу та призначений для зберігання його копії протягом усієї роботи з ним
- ✧ файл певного вигляду, який створюється водночас з відкриттям нового документу та призначений для зберігання інформації про нього
- ✧ файл певного вигляду, який створюється водночас з відкриттям нового документу та призначений для зберігання перекладу на іноземній мові
- ✧ частина оперативної пам`яті для тимчасового зберігання фрагментів документу будь-якого вмісту та подальшого їх застосування

2. У вікні MS WORD знаходяться:
- ✧ рядок заголовка
- ✧ рядок меню
- ✧ панелі інструментів
- ✧ лінійка
- ✧ смуги прокрутки
- ✧ корінці вкладок
- ✧ прапорці
- ✧ лічильники

3. Рядок стану визначає:
- ✧ номер сторінки документа
- ✧ кількість сторінок документа
- ✧ відстань в міліметрах до поточного місця курсору від лівого краю аркуша
- ✧ номер розділу документу
- ✧ відстань в сантиметрах до поточного місця курсору від нижнього краю
- ✧ аркуша
- ✧ номер рядка поточного місця курсору
- ✧ номер стовпця знаходження курсору
- ✧ кількість знакомісць в рядку від початку до курсору

4. Адреси комірок таблиці позначаються:
- ✧ A2, D4, K6
- ✧ 2A, 4D, 6K
- ✧ 11E, 56L, 90H

5. Які з цих класифікацій не є візуальними?
   Сайти в стилі GUI
   Метафоричні сайти
   Комерційні сайти
   Експериментальні сайти
   Філантропічні сайти

6. З чого складається макрос?
   З набору тегів
   Із сукупності операторів Visual Basic
   З набору гіперпосилань
   З набору макрокоманд

7. Алгоритм – це:
- ✧ правила виконання певних дій
- ✧ орієнтований граф, який вказує порядок виконання певного набору команд
- ✧ зрозуміле і точне розпорядження виконавцю здійснити послідовність дій, спрямованих на досягнення поставлених цілей
- ✧ набір команд для комп'ютера
- ✧ протокол обчислювальної мережі

8. Алгоритм називається лінійним:
- ✧ якщо він складений так, що його виконання передбачає багаторазове повторення одних і тих же дій
- ✧ якщо хід його виконання залежить від істинності тих чи інших умов
- ✧ якщо його команди виконуються в порядку їх природного слідування один за одним незалежно від будь-яких умов
- ✧ якщо він включає в себе допоміжний алгоритм
- ✧ якщо він розв'язує систему лінійних рівнянь

9. Основні переваги компіляції:
- ✧ Швидкість виконання готової програми
- ✧ Незалежність програми від системи реалізації
- ✧ Простота зміни програмного коду

10. Які компоненти включає інтерфейс VBA?
- ✧ вікно проекту
- ✧ вікно редагування коду
- ✧ вікно редагування форм
- ✧ панель елементів
- ✧ вікно властивостей

- ✧ меню та панель інструментів
- ✧ вікно компіляції
- ✧ інтерпретатор

11. Функція на VBA повертає значення через
- ✧ ім'я функції
- ✧ процедуру
- ✧ змінну

12. MS EXCEL за призначенням є:
- ✧ операційною системою, подібною WINDOWS
- ✧ графічним редактором
- ✧ процесором електронних таблиць
- ✧ правильної відповіді немає

13. Функції МИН, МАХ відносяться до:
- ✧ математичних
- ✧ статистичних
- ✧ фінансових
- ✧ текстових

# РОЗДІЛ 4.
# БАЗИ ДАНИХ, ПРЕДСТАВЛЕННЯ ЗНАНЬ ТА ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ

## 4.1 Програмні засоби роботи з базами та сховищами даних.

**28 лютого – свято IT-специалистів**
Ще одне неофіційне свято IT-специалистів, приурочене до дня винаходу мережевого кабелю.

### 4.1.1 Основні поняття теорії систем управління базами даних. Реляційні бази

#### 4.1.1.1 Поняття бази даних, об'єкта, класу об'єктів, атрибутів, доменів, кортежів

Сучасні інформаційні системи, засновані на концепції інтеграції даних, характеризуються величезними обсягами збережених даних, складною організацією, необхідністю задовольняти різноманітні вимоги численних користувачів.

Інформаційна система – система, що реалізує автоматизований збір, обробку і маніпулювання даними і включає технічні засоби обробки даних, програмне забезпечення і відповідний персонал.

Мета будь-якої інформаційної системи – обробка даних про об'єкти реального світу.

**Базове визначення даних.**

***Дані*** – інформація, що відображує визначений стан деякої предметної галузі в конкретній формі подання і містить лише найбільш суттєві з погляду цілей і задач збору та обробки інформації елементи образа фрагмента дійсності, що відображується.

З розвитком обчислювальної техніки (і програмних засобів) посилились можливості представлення даних, ми тепер можемо використовувати як найпростіші (неструктуровані дані), так і дані більш складних типів, отримані за допомогою комбінації найпростіших даних. Такі дані називають ***структурованими*****,** оскільки вони мають певну організацію.

Однією з найважливіших функцій в діяльності органів управління є вироблення пропозицій для прийняття обґрунтованого рішення керівником. Основою для виконання цієї функції є різного роду інформація (про ціни, курси валют, попит та ін.). Під інформацією далі будемо розуміти сукупність свідчень про процеси, явища, події та об'єкти реального світу, накопичені

та систематизовані за визначеними ознаками чи закономірностями для їх подальшого використання. Тобто, перш ніж інформацію можна було б використовувати, її необхідно зібрати, систематизувати, обробити і тільки після цього в потрібному вигляді видати користувачу. Для вирішення цих задач і створюються автоматизовані інформаційні системи (АІС). Слід відзначити, що поняття АІС є базовим загальноприйнятим поняттям. На практиці дуже часто можна зустрітися і з іншими назвами систем (АІРС – автоматизована інформаційно-розрахункова система, просто ІРС і т. ін.), які тільки визначають специфіку їх конкретного призначення. За основними принципами ж побудови такі системи нічим не відрізняються від АІС, а мають лише деякі відмінності функціонального характеру.

Під **автоматизованою інформаційною системою** розуміють систему, призначену для накопичення, збереження, пошуку та видачі інформації за запитами користувачів для наступного її використання, обробки та обміну в різноманітних прикладних програмах.

На даний час АІС підрозділяють на документографічні та фактографічні автоматизовані інформаційні системи.

**Документографічна АІС** характерна тим, що інформаційним масивом для неї є різноманітні неформалізовані документи, оформлені на природній мові. Прикладом таких інформаційних масивів можуть бути накази, директиви, розпорядження і т. ін. Для повного вирішення інформаційних задач такою системою необхідно, щоб вона володіла властивостями інтелектуальності, тобто в ній повинна бути організована природно-мовна взаємодія користувача з АІС, і сама система повинна "розуміти" суть інформації, викладеної природною мовою, що на сьогоднішній день становить певні труднощі. Тому на практиці більш широке розповсюдження отримали фактографічні АІС.

**Фактографічні АІС** – це автоматизовані інформаційні системи, масивом інформації для яких є формалізовані записи, приведені до встановленої форми. Це, як правило, різного роду таблиці. Прикладом найпростішої фактографічної АІС може бути автоматизована система обліку товару, кадрового обліку тощо. Інформація в таких АІС для більш ефективної організації її обробки подається формалізованою мовою і циркулює у вигляді даних. Саме фактографічні АІС і є основою для створення інформаційно-розрахункових систем органів управління.

***Дані*** – це інформація, яка подана в формалізованому вигляді, придатному для її наступної обробки, збереження та передачі.

Наприклад: відкриття 2 конференції розпочнеться 12.03.08 року о 9 годині ранку.

Наведені свідчення є просто інформацією, яка в такому вигляді не може використовуватися в фактографічній АІС.

Вирази: відкриття, 2 конференція, 9.00, 12.03.08 – вхідні дані для обліку їх в АІС.

Структурна схема найпростішої АІС наведена на рис. 4.1.




Рисунок 4.1. Структурна схема найпростішої АІС.

Фахівці взаємодіють з основними модулями АІС через спеціальні програми-інтерфейси, які входять до складу їх автоматизованих робочих місць. Інтерфейси дозволяють здійснювати запити до системи та видавати результат інформаційного пошуку і обробки інформації прикладними програмами в зручному для користувача виді, а також здійснювати введення та корегування даних.

Прикладні програми призначені для виконання різних операцій над даними (проведення оперативно-тактичних розрахунків, моделювання процесів тощо) та видачі результатів на АРМ користувачів.

Основною підсистемою АІС є **банк даних –** комплекс програмних, мовних, організаційних та технічних засобів, призначених для централізованого накопичення та колективного використання даних, а також самі дані, що зберігаються в базах даних.

Банк даних поєднує в собі систему управління базою даних (СУБД) та саму базу даних (БД).

**База даних –** це поіменоване, незалежне від прикладного програмного забезпечення інформаційне середовище, організоване за встановленими правилами для збереження, оновлення та маніпулювання сукупністю свідчень про стан та властивості об'єктів та їх взаємовідношення в предметній області, що розглядається.

База даних включає дані, метадані і процедури.

**Дані.** Під термінами "дані", "інформація" або "відомості" у даному контексті розуміється увесь фактичний матеріал, що зберігається у базі даних. Дані є необробленою сировиною, яка підлягає відповідній структуризації. Ухвалення рішення про те, яку інформацію помістити у БД, яким чином її упорядкувати і структурувати, є найважливішою частиною роботи системних архітекторів (проектувальників) бази даних.

**Метадані** складають вміст системного каталогу бази даних і є відомостями про імена і структуру таблиць, імена і права користувачів, найменування і типи обмежень, про процедури, функції і інші об'єкти бази даних.

**Процедури *є*** важливим компонентом системи. Вони встановлюють стандарти ведення комерційної, технологічної і виробничо-технічної діяльності у рамках підприємства і в стосунках з клієнтами. Процедури також використовуються для організації спостереження і аудиту як за інформацією, що вводиться у БД, так і за інформацією, що породжується на основі витягуваних даних.

**Система управління базою даних –** це комплекс мовних та програмних засобів, призначений для забезпечення функціонування бази даних.

СУБД забезпечує взаємодію користувача і прикладних програм з базами даних, а також виконання спеціальних функцій в середовищі БД.

Для виконання цих функцій СУБД використовує різні мови та спеціальні прикладні програми, найважливіше значення серед яких мають:

**мова описання даних** (**МОД**), яка служить для забезпечення визначених форматів даних в БД. База даних у процесі експлуатації може змінюватися, доповнюватися, але всі ці зміни не можуть бути довільними, тобто повинен витримуватися певний формат даних у відповідності до прийнятої мови їх описання;

**мова маніпулювання даними** (**ММД**), яка забезпечує порядок доступу до даних та їх обробки. Вона представляється словником ММД (переліком команд-операторів) на алгоритмічних мовах, якими описана БД;

**мова запитів,** яка визначає порядок отримання інформації з БД для розв'язання розрахункових задач та одержання довідок;

**доповнюючі компоненти** – спеціальні програми, призначені для вирішення таких завдань: завантаження та ведення БД; копіювання вмісту БД; реорганізації БД; друку вмісту БД; ведення обліку використання даних та їх захисту (включаючи їх відновлення в аварійних ситуаціях).

Бази даних використовуються при проведенні оперативно-тактичних розрахунків або моделювання у визначеній предметній області.

**Предметна область** – це відображення в базі даних сукупності об'єктів реального світу з їх зв'язками, які належать до певної області знань і мають практичну цінність для користувачів.

Описання предметної області здійснюється на таких рівнях:

- ✧ абстракції реального світу;
- ✧ уявлення деякого користувача;
- ✧ подання бази даних на фізичних носіях.

Відповідно йдеться про три рівні абстракції: концептуальний, зовнішній та внутрішній (фізичний). При цьому лише фізична база даних існує матеріально, інші рівні описуються абстрактно і використовуються під час роботи з базою даних.

**Концептуальний рівень**. На даному рівні база даних описується концептуальною моделлю (або просто моделлю даних). Концептуальна модель визначає інформацію про предметну область у такій формі даних, в якій вони існують у реальному світі. Концептуальна модель не відображує уявлення користувача про дані з точки зору їх обробки та збереження на фізичних носіях. Ця модель складається з множини концептуальних записів, які в природно-мовному поданні дають повну характеристику об'єктів реального світу та їх взаємозв'язків.

**Зовнішній рівень**. Зовнішній рівень, найбільш близький користувачам, пов'язаний з тим, як окремі користувачі уявляють собі дані. На цьому рівні дані подаються за допомогою деякої зовнішньої моделі даних, яка є, з одного боку, інформаційним змістом бази даних і, з іншого – незалежним інформаційним середовищем прикладних програм. Для забезпечення такого обміну даними між прикладною програмою та базою даних у зовнішній моделі використовується своя мова описання даних та мова маніпулювання даними.

Записи моделі створюються системою на той момент, коли вони викликані програмою користувача (при читанні даних з бази) або формуються в прикладній програмі, надалі дані з цих записів переносяться в базу даних для їх збереження (при введенні даних у базу). Такі процедури задаються у вигляді спеціального описання відображення-перетворення даних з фізичної бази даних в дані для прийнятої моделі та навпаки.

**Внутрішній рівень**. На внутрішньому рівні база даних подається у вигляді сукупності наборів даних (файлів). Набір даних складається з логічних записів. Логічні записи набору даних на відміну від звичайного набору даних називаються внутрішніми записами.

Основними характеристиками набору даних бази даних є: тип організації та метод доступу, формат внутрішнього запису та його довжина, розмір блока та ін.

Кожному запису на концептуальному рівні в природно-мовному поданні відповідає внутрішній запис на внутрішньому рівні в формалізованому вигляді.

Найбільш простою для створення та використання є **реляційна база даних** – модель безпосереднього доступу до даних, які об'єднуються

на найнижчому рівні в двомірні масиви і мають між собою встановлені зв'язки. Пошук потрібних даних спрощується до визначення назви таблиці, рядка і стовпця, які і визначають адресу потрібного значення з бази даних. Сукупність таких пойменованих таблиць і створює основу єдиної бази даних, що використовується в ІРС.

Основу структури бази даних складають певні елементи, визначення та поняття, які вводяться при формалізації природних записів концептуальної моделі у внутрішні записи.

Будь-який об'єкт може бути охарактеризований за допомогою сукупності кількісних та якісних прикмет – показників об'єкта. Ці прикмети називають **атрибутами**. **Атрибут** – це логічно неподільний елемент, що належить до властивості деякого об'єкта чи процесу. Якість визначають атрибути-прикмети, а кількість – атрибути-основи.

**Атрибути-прикмети** є якісною характеристикою об'єкта і застосовуються в логічних операціях, таких як порівняння, сортування, компонування та редагування. Як атрибути-прикмети можуть виступати, наприклад, код заказу, товар, ціна та ін.

**Атрибути-основи** характеризують кількісний бік об'єкта, залежать від атрибутів-прикмет та застосовуються в обчислювальних операціях. Атрибути-основи, наприклад кількість, дальність, час, без атрибутів-прикмет не дають визначення об'єкта. Кожний об'єкт характеризується сукупністю атрибутів-прикмет та атрибутів-основ.

Для кожного атрибута існує множина його значень. Так, *атрибут* сторона світу може мати чотири значення: південь, північ, схід або захід. Область можливих значень атрибутів називається **доменом**.

Один і той же домен може служити областю визначення різних атрибутів. Наприклад, на домені населений пункт визначаються такі атрибути: пункт розташування, станція, місце призначення і т. ін. Кожному домену та атрибуту присвоюється унікальне ім'я. Допускається збіг імені атрибута та імені відповідного йому домену.

Для запису значень атрибутів у базі даних відводять поля. **Поле –** це найменша кількість символів, що визначає властивості об'єкта, тобто це найменша пойменована одиниця даних.

Сукупність полів (сукупність атрибутів) визначеного типу (об'єкта) називається **записом**. Записи в базі даних об'єднуються за типами об'єктів, їх призначенням в так звані **файли** (пойменовані таблиці). Сукупність усіх файлів, які визначають предметну область, утворює **бібліотеку** бази даних.

Дані про об'єкти в базах даних узгоджені між собою зв'язками. Зв'язки встановлюються між всіма значеннями атрибутів в запису, записами та іменами файлів. Такий послідовний пойменований набір утворює так званий **кортеж даних.**

**Внутрішня структура даних.**

Дані подаються у вигляді пов'язаних списків:

$$d < N_в, M_d >, \tag{4.1}$$

де $N_в$ – ім'я пов'язаного списку;

$M_d$ – список значень атрибутів (список даних).

Список даних $M_d$ являє собою список такого вигляду:

$$M_d = [ D_1, D_2, \ldots D_i, \ldots D_n ], \tag{4.2}$$

де $D_i$ – елемент даних;

$n$ – кількість атрибутів.

Структура кортежу даних має вигляд:

$$d < N_в, [ D_1, D_2, \ldots D_i, \ldots D_n ] >. \tag{4.3}$$

При такому поданні даних $N_в$ може розглядатися як ім'я розділу в базі даних. Це дозволяє структурувати дані згідно з їх тематикою.

Наприклад, утворюється база даних про автомастила, в якій визначаються атрибути: "Код заказу", "Фірма", "Ціна", "Кількість".

Кортеж даних буде мати вигляд:

$d <$ "Автомастила" ["Код заказу", "Фірма", "Ціна", "Кількість"]>.

При заповненні бази даних отримаємо кортеж даних:

$d <$ "Автомастила" ["78653", Лукойл , 100 , 1005 ]>.

З наведеного прикладу видно, що значення атрибутів визначаються не тільки іменем пов'язаного списку, але й порядковим номером їх у списку $M_d$. Так, атрибуту з номером 1 відповідає код автомастила, атрибуту з номером 2 – назва фірми тощо.

Для того, щоб звільнити користувача від необхідності контролювати порядок слідування атрибутів у списку, введена внутрішня інтерпретація даних (семантика).

З цією метою визначений ще один кортеж даних – кортеж імен атрибутів:

$$a < N_в, [ A_1, A_2, \ldots A_i, \ldots A_n ] >, \tag{4.4}$$

де $A_i$ – ім'я атрибута, який відповідає $D_i$.

Тоді порядок чергування даних не має значення, оскільки встановлюється однозначна відповідність $A_i \rightarrow D_i$.

Встановлення зв'язків між елементами реляційної бази даних зображене на рис. 4.2.

Такий підхід до організації даних дозволяє змінювати кількість атрибутів, переставляти їх місцями, не порушуючи при цьому цілісності та однозначності даних, що значною мірою визначає простоту роботи в середовищі бази даних.

Створюючи базу даних, користувач прагне упорядкувати інформацію за різними ознаками і швидко робити вибірку з довільним сполученням ознак. Велике значення при цьому здобуває структурування даних.

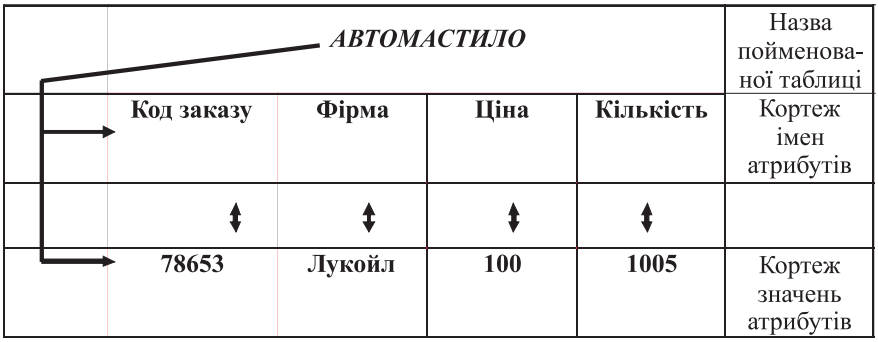



Рисунок 4.2. Встановлення зв'язків у базі даних.

Структурування даних – це введення угод про способи представлення даних.

Неструктурованими називають дані, записані, наприклад, у текстовому файлі.

Нижче приведений приклад неструктурованих і структурованих даних, що містять зведення про студентів (номер залікової книжки, прізвище, ім'я, по батькові і дату народження).

Неструктуровані дані:

Номер залікової книжки № 16493. Сергєєв Петро Михайлович, дата народження 1 січня 1976 р.; номер залікової книжки № 16593, Петров Анатолій Іванович, дата народж. 15 березня 1975 р.; номер залікової книжки 16693, д.н. 14.04.76, Анохін Андрій Борисович.

Легко переконатися, як складно організувати пошук необхідних даних, що зберігаються в неструктурованому вигляді.

Структуровані дані мають такий вигляд:

| Номер залікової книжки | Прізвище | Ім'я | По батькові | Дата народження |
|---|---|---|---|---|
| 16493 | Сегєєв | Петро | Михайлович | 01.01.76 |
| 16593 | Петров | Анатолій | Іванович | 15.03.75 |
| 16993 | Анохін | Андрій | Борисович | 14.04.76. |

Щоб автоматизувати пошук і систематизувати ці дані, необхідно виробити визначені угоди про способи представлення даних, тобто дату народження потрібно записувати однаково для кожного студента, вона

повинна мати однакову довжину і визначене місце серед іншої інформації. Ці ж зауваження справедливі і для інших даних (номер особової справи, прізвище, ім'я, по батькові).

4.1.1.2 Поняття моделі даних, типи моделей даних

Під *інформаційною моделлю* розуміють інформацію про об'єкт, відібрану і структуровану у відповідності з заданою метою.

Центральним поняттям в області баз даних є поняття моделі даних. Термін "модель" використовується в декількох значеннях.

***Модель даних*** – це опис методів представлення і обробки даних в СУБД, в тому числі методів визначення типів і логічених структур в базі даних, методів маніпулювання даними і методів визначення і підтримки цілісності бази даних.

Іншим його значенням є опис на різних рівнях абстракції схеми конкретної бази даних, призначеної для роботи в певних умовах. Незважаючи на те що терміни однакові, в другому випадку мається на увазі моделювання з точки зору розробників інформаційної системи (бази даних).

Для того, щоб спроектувати і реалізувати реляційну базу даних, що складається з трьох таблиць, немає необхідності прибігати до спеціальних технологій і прийомів, такого роду роботу можна виконати безпосередньо. Але коли йдеться про базу даних для інформаційної системи підприємства, таке "пряме" проектування стає не лише трудомістким, але у багатьох випадках просто неможливим. Для полегшення роботи проектувальників, в процес проектування включається етап моделювання даних. На цьому етапі структури даних спочатку представляються у вигляді графічних схем і діаграм, що полегшують загальне розуміння зв'язків і взаємодії об'єктів бази даних, а також сприяючих встановленню більшої відповідності бізнес-процесів підприємства, бізнес-правил інформаційної системи і структури даних, а потім вже реалізують базу даних у вигляді набору реляційних таблиць і об'єктів.

На рис. 4.3 представлена класифікація моделей даних відповідно до трирівневої архітектури, запропонованої в 1975 р. Комітетом планування стандартів і норм (Standards Planning and Requirements Committee, SPARC) Національного інституту стандартизації США (American National Standard Institute, ANSI), ANSI/X3/SPARC (мал. 6.8). Так, моделі даних, позначені на рис. 4.3 як фізичні, відповідають першому рівню архітектури ANSI/X3/SPARC, даталогические моделі можна віднести до другого, внутрішнього рівня архітектури, а инфологические моделі відповідають концептуальному рівню архітектури.

***Модель даних (МД)*** являє собою сукупність структур даних і операцій над цими структурами, призначених для визначення логічної організації БД і динамічного моделювання станів предметної області.

Структури даних відображена на рис. 4.4.

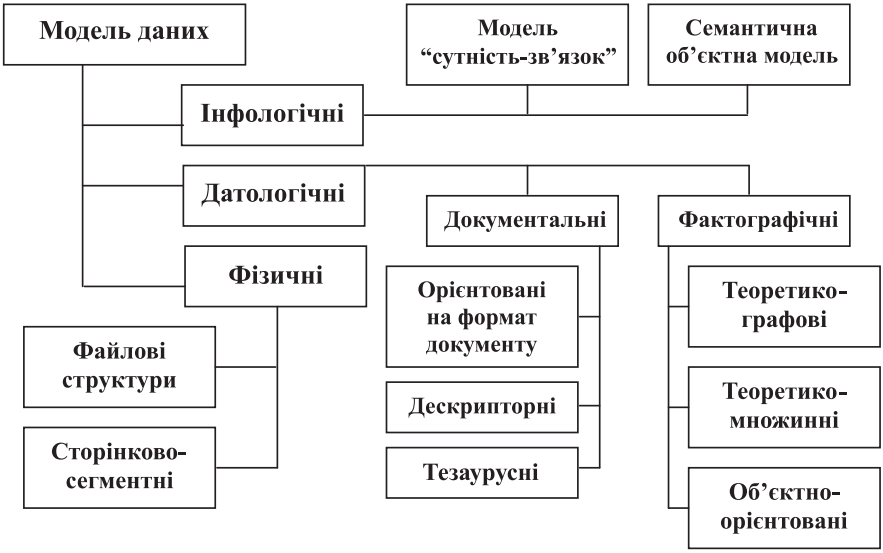



Рисунок 4.3. Класифікація моделей даних.

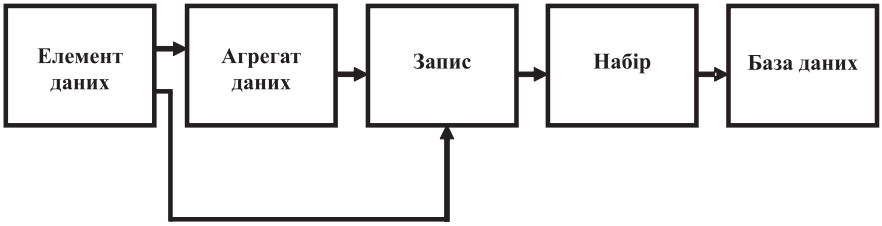



Рисунок 4.4. Модель КОДАСИЛ.

Порядок композиції між типами структур даних моделі КОДАСИЛ. Основні компоненти моделі:

– *Елемент даних* – найменша пойменована одиниця даних. *Елемент даних* визначає тип даних. Поняття елементу даних застосовується як при концептуальному моделюванні, так і в ході створення фізичної моделі даних. На етапі концептуального моделювання елемент даних – це елемент *абстрактного типу даних*, а під час створення фізичної моделі даних це вже елемент *базового типу* конкретної СУБД. *Елемент даних* – це мінімальна одиниця даних, до якої СУБД може адресуватися безпосередньо і за допомогою якої виконується побудова всіх інших структур.

***Приклад.*** Під час концептуального моделювання елементу даних, в якому треба зберегти рядок, привласнюється абстрактний тип String. Якщо ця концептуальна модель буде реалізована на СУБД MS SQL Server, то елемент абстрактного типу String перетвориться до базового типу строкових даних, прийнятого в MS SQL Server, тобто до типу varchar. При реалізації цього ж елементу на СУБД Oracle він отримає тип char, що є базовим типом для рядка в Oracle.

- Агрегат даних. Пойменована сукупність елементів даних усередині запису, яку можна розглядати як єдине ціле.
- *Запис*. Пойменована сукупність елементів даних або елементів даних і агрегатів. Поняття *запису* у базах даних близьке до цього поняття в мовах програмування, але не в усьому співпадає з ним. *Запис* – це одиниця незалежного зберігання, що унікально ідентифікується, в системі баз даних. *Схема запису* – це внутрішня структура запису, що включає відомості про кількість полів і тип кожного поля запису.
- Набір. Пойменована сукупність записів, що утворюють дворівневу ієрархічну структуру.

Схема (тип) запису визначає зв'язну послідовність *полів* – позицій в структурах зберігання записів. Внутрішня структура кожного поля визначається типом даних, заданим в оголошенні кожного запису.

Для унікальної ідентифікації записів одно або більше полий записи мають бути оголошені явно як *ключ запису*. Значеннями полів є конкретні дані (числа, символьні рядки, слова і ін.).

Приклади простого (а) і складеного (б) агрегатів

| Дата | | |
|---|---|---|
| Число | Місяць | Рік |

а)

| Вищий навчальний заклад | | | |
|---|---|---|---|
| Назва | Адреса | | |
| | Поштовий індекс | Місто | Вулиця і номер будинка |

б)

Користувачами бази даних можуть бути різні прикладні програми, програмні комплекси, а також фахівці предметної області, що виступають у ролі споживачів або джерел даних, називані кінцевими користувачами.

Створення бази даних, її підтримка і забезпечення доступу користувачів до неї здійснюються централізовано за допомогою спеціального програмного інструментарію – системи управління базами даних.

**Операції над даними.**

За характером виробленої дії розрізняють наступні види операцій:

- ✧ ідентифікацію даного і знаходження його позиції в БД;
- ✧ вибірку (читання) даного з БД;
- ✧ включення (запис) даного в БД;
- ✧ видалення даного з БД;
- ✧ модифікацію (зміну) даного в БД.

За способом одержання результату розрізняють:

- ✧ навігаційні – якщо результат операції одержується шляхом проходження по зв'язках, реалізованих у структурі БД;
- ✧ специфікаційні – якщо в операції визначаються тільки вимоги до результату, але не задається спосіб його одержання.

Іноді необхідно застосовувати більш узагальнені операції, що реалізують виконання більш складних дій над даними, які називають процедурами БД.

**Процедура БД** – це послідовність операцій, що дозволяють реалізувати визначений алгоритм обробки даних.

**Обмеження цілісності.**

Логічні обмеження, що накладаються на дані називаються обмеженнями цілісності. Обмеження вводяться в моделі даних з метою підвищення їх семантичносты і розширення можливості підтримки цілісності даних. Перший аспект зв'язаний з адекватністю відображення реального світу в схемі, а другий – з можливостями СУБД забезпечувати несуперечність даних заданим обмеженням при переведенні БД з одного стану в інше.

4.1.1.3 Характеристики баз даних

**Класифікація баз даних**

За технологією обробки дані бази даних підрозділяються на централізовані і розподілені.

*Централізована база даних* зберігається в пам'яті однієї обчислювальної системи. Якщо ця обчислювальна система є компонентом мережі ЕОМ, можливий розподілений доступ до такої бази. Такий спосіб використання баз даних часто застосовують у локальних мережах ПК.

*Розподілена база даних* складається з декількох частин, які можливо перетинаються або навіть дублюються, збережених у різних ЕОМ обчислювальної мережі. Робота з такою базою здійснюється за допомогою системи управління розподіленою базою даних (СУРБД).

За способом доступу до даних бази даних розділяються на бази даних з локальним доступом і бази даних з віддаленим (мережним) доступом.

Системи централізованих баз даних з мережним доступом припускають різні архітектури подібних систем:

- – файл-сервер;
- – клієнт-сервер.

*Файл-сервер*. Архітектура систем БД із мережним доступом припускає виділення однієї з машин мережі в якості центральної (сервер файлів). На такому комп'ютері зберігається спільно використовувана централізована БД. Всі інші комп'ютери мережі виконують функції робочих станцій, за допомогою яких підтримується доступ користувальницької системи до централізованої бази даних. Файли бази даних відповідно до користувальницьких запитів передаються на робочі станції, де в основному і реалізується обробка.

При великій інтенсивності доступу до однакових даних продуктивність інформаційної системи падає. Користувачі можуть також створювати на робочих станціях локальні БД, що використовуються ними монопольно.

Концепція файл-сервер умовно відображена на рисунку 4.5.

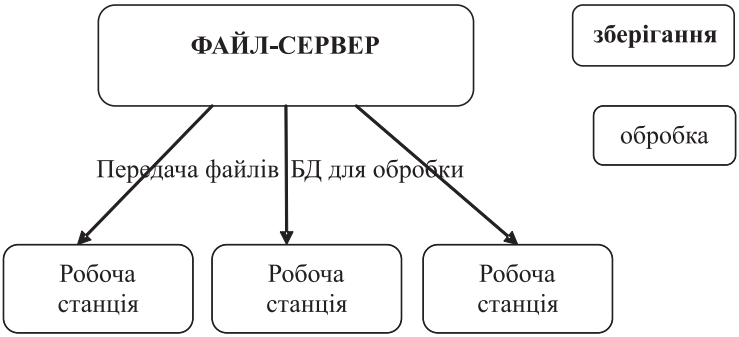



Рисунок 4.5. Схема обробки інформації в БД за принципом файл-сервер

Недоліки архітектури файл-сервер полягають у наступному:

- ✧ вся вага обчислювальної роботи лягає на комп'ютер клієнта; наприклад, якщо в результаті запиту клієнт повинен одержати 2 записи з таблиці обсягом 100 000 записів, всі 100 000 записів будуть скопійовані з файлу-сервера на клієнтський комп'ютер; у результаті зростає завантаження мережі (мережний трафік) і збільшуються вимоги до апаратних потужностей користувальницького комп'ютера; потреби в постійному збільшенні обчислювальних потужностей клієнтського комп'ютера обумовлюються постійно зростаючим обсягом оброблюваної інформації;
- ✧ оскільки БД являє собою набір файлів на мережному сервері, доступ до таблиць регулюється тільки мережною операційною системою, що робить таку БД по суті беззахисної від випадкового або навмисного перекручування інформації, що зберігається в ній, знищення або розкрадання;
- ✧ перевірки коректності даних у системах файл-сервер реалізуються в програмі клієнта, що в принципі не виключає проектування су-

перечних перевірок у різних програмах; значеннєва цілісність інформації при цьому може порушуватися;

- ✧ внесення змін у БД є потенційним джерелом помилок як при одночасному внесенні змін у той самий запис, так і при реалізації відкату результатів серії об'єднаних за змістом у єдине ціле операцій над БД, коли деякі з них завершилися успішно, а деякі – ні; це може порушувати посилальну й значеннєву цілісність БД.

*Клієнт-сервер.* У цій концепції мається на увазі, що крім збереження централізованої бази даних центральна машина (сервер бази даних) повинна забезпечувати виконання основного обсягу обробки даних. Запит на дані, видаваний клієнтом (робочою станцією), породжує пошук і витяг даних на сервері. Витягнуті дані (але не файли) транспортуються по мережі від сервера до клієнта. Специфікою архітектури клієнт-сервер є використання мови запитів SQL.

Концепція клієнт-сервер умовно зображена на рис. 4.6.

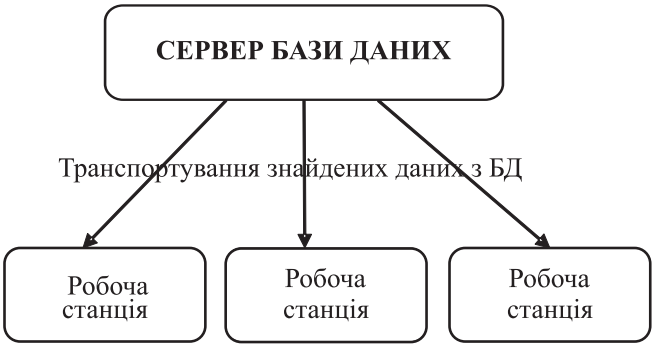



Рисунок 4.6. Схема обробки інформації в БД за принципом клієнт-сервер.

Переваги архітектури клієнт-сервер:

- ✧ більшість обчислювальних процесів відбувається на сервері, що знижує вимоги до обчислювальних потужностей комп'ютера клієнта; збільшення обчислювальної потужності одного сервера еквівалентно одночасному збільшенню потужності всіх клієнтських місць;
- ✧ знижується мережний трафік за рахунок посилки сервером клієнтові тільки тих даних, які він запитував; наприклад, якщо необхідно зробити з таблиці обсягом 100 000 записів вибірку, результатом якої будуть усього 2 запису, сервер виконає запит і перешле клієнтові тільки потрібні йому 2 запису;
- ✧ БД на сервері являє собою, як правило, єдиний файл, у якому містяться таблиці, обмеження цілісності та інші компоненти

БД; зламати, викрасти або зіпсувати таку БД значно складніше; істотно збільшується захищеність БД від уведення неправильних значень, оскільки сервер БД проводить автоматичну перевірку відповідності значень, що вводяться, накладеним обмеженням; крім того, сервер відслідковує рівні доступу для кожного користувача й блокує спроби виконання не дозволених для користувача дій; все це дозволяє говорити про значно вищий рівень забезпечення безпеки БД, посилальної й значеннєвої цілісності інформації;

- ✧ сервер реалізує управління змінами даних і запобігає спробі одночасної зміни тих самих даних;
- ✧ безпека системи зростає за рахунок переносу більшої частини перевірок коректності даних на сервер; падає питома вага суперечних один одному перевірок даних у клієнтських додатках, що виконують різні дії над БД.

Для реалізації архітектури клієнта-сервер застосовують так називані промислові сервери баз даних, такі, як InterBase, Oracle, Informix, IBM DB2, MS SQL Server.

**Класифікація за моделлю даних**

Ядром будь-якої бази даних є модель даних. СУБД ґрунтується на використанні ієрархічної, мережної або реляційної моделі, на комбінації цих моделей або на деякій їх підмножині.

Розглянемо три основних типи моделей даних: ієрархічну, мережну і реляційну.

*Ієрархічна модель даних*

Ієрархічна модель організує дані у вигляді деревоподібної структури.

До основних понять ієрархічної структури відносяться: рівень, елемент (вузол), зв'язок. Дерево являє собою ієрархію елементів, називаних вузлами. Вузол – це сукупність атрибутів даних, що описують деякий об'єкт. На самому верхньому рівні ієрархії є один і тільки один вузол – корінь. Кожен вузол, крім кореня, зв'язаний з одним вузлом на більш високому рівні, називаним вихідним для даного вузла. Жоден елемент не має більш одного вихідного. Кожен елемент може бути зв'язаний з одним чи декількома елементами на більш низькому рівні. Вони називаються породженими.

До кожного запису бази даних існує тільки один (ієрархічний) шлях від кореневого запису. Наприклад, як видно з рис.4.7, для запису С4 шлях проходить через записи А й В3.

Приклад, представлений на наступному рис. 4.8, ілюструє використання ієрархічної моделі бази даних. Для розглянутого прикладу ієрархічна структура правомірна, тому що кожен студент учиться у визначеній (тільки одній) групі, що відноситься до визначеного (тільки одного) інституту.

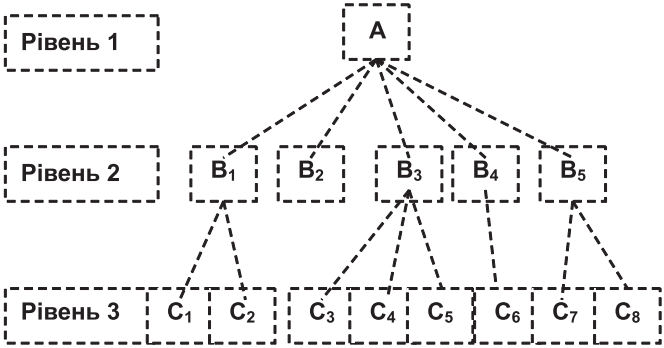



Рисунок 4.7. Схема ієрархічної моделі даних.

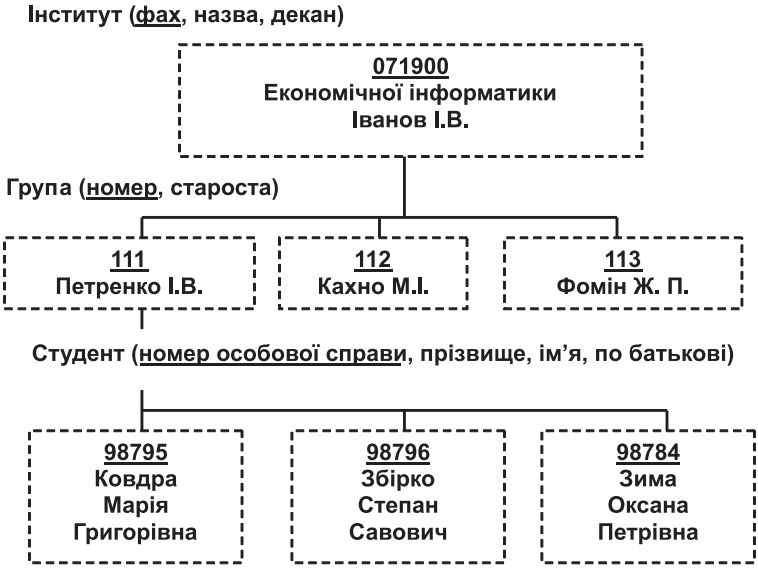



Рисунок 4.8. Приклад ієрархічної структури БД

**Мережна модель даних**

Мережна модель організує дані у вигляді мережної структури.

Структура називається мережною, якщо у відносинах між даними породжений елемент має більш одного вихідного. У мережній структурі при тих же основних поняттях (рівень, вузол, зв'язок) кожен елемент може бути зв'язаний з будь-яким іншим елементом.

На рисунку 4.9 зображена мережна структура бази даних у вигляді графа.

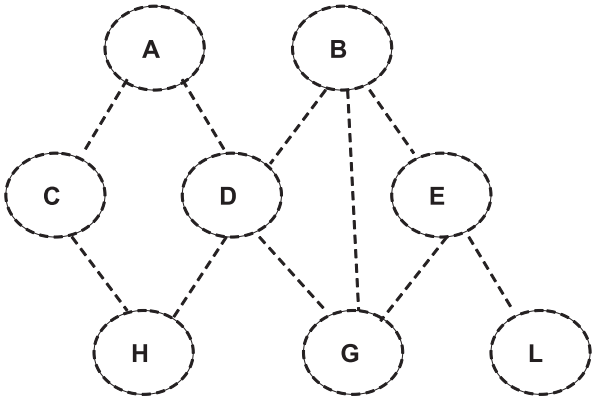



Рисунок 4.9. Графічне зображення мережної структури

Прикладом мережної структури може служити структура бази даних, що містить зведення про студентів, які беруть участь у науково-дослідних роботах (НДР). Можлива участь одного студента в декількох НДР, а також участь декількох студентів у розробці однієї НДР. Графічне зображення описаної в прикладі мережної структури, що складається тільки з двох типів записів, показане на рис. 4.10. Єдине відношення являє собою складний зв'язок між записами в обох напрямках.




Рисунок 4.10. Приклад мережної структури БД.

Реляційну модель даних зараз є найбільш розповсюдженою моделлю даних, тому розглянемо її більш детально.

4.1.1.4. Реляційні бази даних.
Поняття та вимоги до реляційної бази даних.
Типи зв'язків у базах даних, їх характеристика

**Реляційна модель даних**

Поняття реляційний (англ. relation – відношення) пов'язане з розробками відомого американського фахівця в області систем баз даних Е. Кодда. Справа в тому, що математична структура, що відповідає таблиці називається відношенням. Ці моделі характеризуються простотою структури даних, зручним для користувача табличним представленням і можливістю використання формального апарата алгебри відносин і реляційного числення для обробки даних.

Реляційна модель орієнтована на організацію даних у вигляді двовимірних таблиць. Наприклад, реляційною таблицею можна представити інформацію про студентів, що навчаються у ВНЗ (рис.4.11).

| № особової справи | Прізвище | Ім'я | По батькові | Дата народження | Група |
|---|---|---|---|---|---|
| 16493 | Кучма | Петро | Іванович | 01.01.76 | 111 |
| 16593 | Сліченко | Лариса | Петрівна | 15.03.75. | 112 |
| 16693 | Гнатюк | Микола | Гнатович | 14.04.76. | 111 |

Рисунок 4.11. Приклад реляційної таблиці.

Реляційна модель даних є сукупністю взаємозалежних двовимірних таблиць – об'єктів моделі.

Зв'язки між двома логічно зв'язаними таблицями в реляційній моделі встановлюються за рівністю значень однакових атрибутів цих таблиць.

Кожна реляційна таблиця являє собою двовимірний масив і має наступні властивості:

- ✧ кожен елемент таблиці – один елемент даних;
- ✧ усі стовпці в таблиці однорідні, тобто всі елементи в стовпці мають однаковий тип (числовий, символьний і т.інш.) і довжину;
- ✧ кожен стовпець має унікальне ім'я;
- ✧ однакові рядки в таблиці відсутні;
- ✧ порядок проходження рядків і стовпців може бути довільним.

При описі реляційної моделі часто використовують наступні терміни: відношення, кортеж, домен.

Відношення представлені у виді таблиць, рядки яких відповідають записам (кортежам), а стовпці – полям, атрибутам відносин (доменам).

Одна з основних властивостей реляційних БД – наявність певних типів таблиць і ключів, що забезпечують взаємозв'язок між таблицями. Існують наступні типи реляційних ключів і таблиць.

***Базова таблиця***. Складається з одного або декількох стовпців властивостей об'єкту, що повністю визначають ці об'єкти як інформаційні елементи. У ній обов'язково повинен знаходитися первинний ключ, тому їх називають ***первинними***. У міжтабличних зв'язках вони є головними таблицями відносин.

***Пов'язуюча таблиця***. Пов'язуючею називається таблиця, яка використовується для установки зв'язків між іншими таблицями і в той же час не є базовою таблицею (оскільки не містить опису властивостей об'єктів і полів первинного ключа). Кожне ключове поле пов'язуючої таблиці є зовнішнім ключем, пов'язаним з первинним ключем головної таблиці.

З технічної точки зору справжня пов'язуюча таблиця складається тільки із зовнішніх ключів і не містить незалежних інформаційних елементів.

***Первинний ключ (ПК)*** – це одне чи кілька полів, що однозначно ідентифікують запис. Первинний ключ містить набір значень, що повністю визначають рядок базової таблиці, яка в Access є основною. Якщо первинний ключ складається з одного поля, він називається ***простим***, якщо з декількох полів – ***складеним ключем***. Кожному значенню первинного ключа відповідає ***один і лише один*** рядок таблиці. Якщо значення осередків даних якого-небудь поля ніколи не повторюються, то створити первинний ключ можна і на базі одного цього поля. Значення первинного ключа в таблиці БД повинне бути унікальним, тобто в таблиці не повинне існувати двох або більше записів з однаковим значенням первинного ключа.

***Потенційний ключ***. Будь-який стовпець або група стовпців, що задовольняє вимогам первинного ключа, є потенційним кандидатом на первинний ключ таблиці і називається потенційним ключем. Потенційним ключем для ідентифікації громадян, наприклад, може бути податковий ідентифікаційний код.

***Зовнішній ключ***. Зовнішній ключ є полем, значення якого відповідають значенням первинного ключа або частини складеного первинного ключа іншої таблиці, пов'язаної з тією, що розглядається. До складу зовнішнього ключа може входити як одне, так і декілька полів (***складений зовнішній ключ***). Якщо довжина зовнішнього ключа менша, ніж відповідний первинний ключ, то він називається ***частковим*** або ***усіченим зовнішнім ключем***.

***Вторинний ключ (ВК)*** (індекси) – це одне чи декілька полів, що виконують роль пошукових ознак або ознак групування. Вторинні ключі установлюються по полях, які часто використаються при пошуку й сортуванні даних: вторинні ключі допоможуть системі значно швидше знайти потрібні дані. На відміну від первинного, значення вторинного ключа може повторюватися в декількох записах файлу, тобто він не є унікальним – у цьому, власне, і полягає головна різниця між первинними й вторинними ключами. Якщо за значенням первинного ключа може бути знайдений один єдиний екземпляр запису, то по вторинному – декілька.

У прикладі, показаному на рисунку 4.11, ключовим полем таблиці є № особової справи.

Різні таблиці, можуть бути пов'язані між собою через загальне поле даних.

На рис.4.12 показаний приклад реляційної моделі, побудованої на основі відношень: **СТУДЕНТ, СЕСІЯ, ЗАОХОЧЕННЯ.**

Таблиця СТУДЕНТ має поля: Номер, Прізвище, Ім'я, По батькові, Дата народження, Група;

СЕСІЯ – Номер, Оцінка 1, Оцінка 2, Оцінка 3, Оцінка 4, Результат;

ЗАОХОЧЕННЯ – Результат, Відсоток.

Таблиці СТУДЕНТ і СЕСІЯ мають співпадаючі ключі (Номер), що дає можливість легко організувати зв'язок між ними. Таблиця СЕСІЯ має первинний ключ Номер і містить зовнішній ключ Результат, що забезпечує її зв'язок з таблицею ЗАОХОЧЕННЯ.

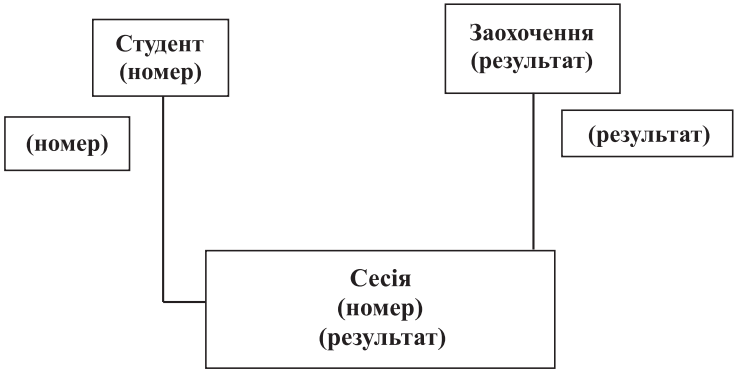



Рисунок 4.12. Приклад реляційної БД.

Завдяки наявним зв'язкам досягаються наступні переваги:

а) вдається уникнути дублювання інформації. Усі необхідні дані можна зберігати тільки в одній таблиці. Так, наприклад, немає необхідності в таблиці СЕСІЯ зберігати номер групи кожного студента, що здає іспити, досить задати зв'язок з таблицею СТУДЕНТ.

б) у реляційних базах даних легко робити зміни. Якщо в таблиці СЕСІЯ змінити які-небудь значення, то правильна інформація автоматично буде змінена в інших таблицях, що посилаються на першу (наприклад, таблиця ЗАОХОЧЕННЯ).

в) у нереляційних базах даних складно передати всі наявні залежності, тобто зв'язати між собою дані з різних таблиць. Реляційна база даних виконує всі ці дії автоматично.

У реляційних базах даних удається легко уникнути встановлення помилкових зв'язків між різними таблицями даних.

***Приклад***. Представимо, що ви хочете перетворити список розсилки в список учасників і відслідковувати надходження добровільних пожертвувань, внесених учасниками. Кожен учасник може сплатити внесок будь-яку кількість разів. Можна додати в первісну таблицю стовпець внесків і щоразу додавати запис при внесенні учасником нового внеску, однак це може привести до чисельного дублюванню інформації (ім'я й адреса неминуче будуть повторюватися при фіксуванні чергового внеску).

Більш ефективне рішення полягає в тому, щоб присвоїти кожному учаснику унікальний номер і додати стовпець із цими номерами у базу даних списків розсилання. Після цього необхідно створити іншу таблицю, що складається з номерів учасників і сум внесків, як це показано на рис. 4.13. У цьому випадку будь-який запис про внесок від учасника буде зв'язаний з окремим учасником і в той же час дублювання буде зведено до мінімуму.

| ID | Ім'я | Прізвище | E-Mail Адреса | Телефон |
|---|---|---|---|---|
| 1 | Юрій | Кулаков | kulakov@coms.ntu-kpi.kiev.ua | 241-76-93 2 |
| 2 | Сергій | Омелянский | sergey@coms.ntu-kpi.kiev.ua | – \|\| – |



Зв'язок за ключовим полем

| ID | Дата | Сума |
|---|---|---|
| 1 | 20 листопада 2001 року | 250 грн. |
| 1 | 8 грудня 2001 року | 170 грн. |
| 2 | 18 січня 2002 року | 230 грн. |

Рисунок 4.13. Приклад організації зв'язку за ключовим полем.
ID – ідентифікатор учасника (ключове поле для зв'язування).

У прикладі, показаному на рис. 4.13, будь-який запис у таблиці учасників може бути зв'язаний з декількома записами в таблиці внесків. Даний приклад ілюструє зв'язок "один-до-багатьох", у якому кожен запис про учасника може бути зв'язаний з довільною кількістю записів про внески. Незважаючи на те що, строго говорячи, ця база даних не є реляційною, тут присутні чіткі зв'язки між записами.

Основним ***обмеженням*** реляційної моделі є неможливість представлення у відношенні кортежів дублікатів**.**

***Переваги*** реляційної моделі:

✧ простота;

- ✧ гарне теоретичне обґрунтування (засноване на добре проробленій теорії відносин реляційної алгебри);
- ✧ забезпечує більш високий ступінь логічної незалежності даних у порівнянні з іншими моделями.

***Недолік.*** Продуктивність існуючих реляційних СУБД нижче, ніж у СУБД інших типів.

**Правила нормалізації**

*Нормалізація* – це формалізована процедура, в процесі виконання якої атрибути даних (поля) групуються в таблиці, а таблиці, у свою чергу, – в бази даних. Цілі нормалізації такі:

- ✧ виключити дублювання інформації в таблицях;
- ✧ забезпечити можливість змін в структурі таблиць;
- ✧ зменшити вплив структурних змін бази даних на роботу додатків, що забезпечують користувачам доступ до даних.

Процес нормалізації складається з декількох етапів. Перших три з них, виконуваних найчастіше, були описані в 1972 році доктором Коддом (Codd). Призначення цих етапів можна зрозуміти з рис. 4.15.

**Перша нормальна форма**

Правила побудови *першої нормальної форми* вимагають, щоб всі таблиці даних були плоскими і не містили даних, що повторюються, в різних рядках. Під плоскою розуміється таблиця, що має тільки два вимірювання: довжина (число записів або рядків) і ширина (число полів або стовпців). Її осередки не можуть містити більше одного значення. Якщо хоч би один елемент таблиці містить більше одного значення, для представлення її вмісту вже потрібне третє вимірювання – глибина.

За допомогою таких додатків електронних таблиць, як Microsoft Excel, можна значно прискорити процес нормалізації даних, особливо якщо в цих даних зустрічаються групи, що повторюються. Імпортуйте дані в робочий лист, а потім виріжте групи, що повторюються, і вставте їх в новий робочий лист. Тепер, коли дані обох таблиць нормалізовані, збережіть робочі листи і імпортуйте їх в таблиці Access. Такий підхід часто набагато швидший, ніж розробка запитів на створення таблиць для їх нормалізації.

**Друга нормальна форма**

Для приведення таблиць до другої *нормальної форми необхідно* забезпечити повну залежність стовпців, які не є ключовими, від первинного ключа, а якщо цей ключ складений, то від кожного його елементу. Під повною *залежністю розуміється* можливість однозначного визначення значення кожного неключового поля за допомогою значення первинного ключа. Якщо для однозначного визначення використовується складений первинний ключ, то це правило застосовується до кожного значення з полів,

що входять в складений ключ. Перед переходом до другої нормальної форми необхідно привести дані до першої нормальної форми. В процесі створення другої нормальної форми велика частина даних, що повторюються, залишилися в таблиці після приведення її до першої нормальної форми, буде видалена.

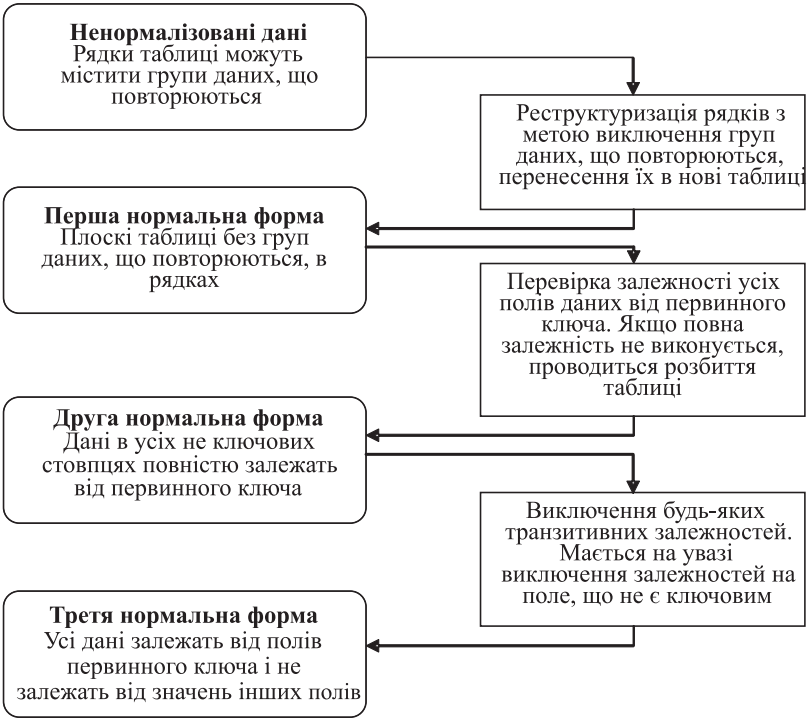



Рисунок 4.15. Процес нормалізації реляційної бази даних до третьої нормальної форми.

**Третя нормальна форма**

У третій нормальній формі стовпці, що не є ключовими, залежать від первинного ключа таблиці і не залежать від всієї решти стовпців. Перш ніж перейти до третьої нормальної форми, приведіть свої дані до першої, а потім – до другої.

**Четверта нормальна форма**

Щоб база даних знаходилася в *четвертій нормальній формі*, необхідно, щоб незалежні елементи даних, між якими існує зв'язок типу багато-до-багатьох, не зберігалися в одній таблиці.

Багато розробників додатків баз даних ігнорують четверту і п'яту нормальні форми в своїх програмних продуктах, оскільки вважають їх вельми специфічними. Результатом цього часто є створення бази даних неправильної структури, хоча це зовсім ще не означає, що вона не функціонуватиме.

**П'ята нормальна форма і комбіновані елементи**

П'ята нормальна форма вимагає забезпечення можливості точного відновлення початкової таблиці з таблиць, на яких вона заснована. Побудова п'ятої нормальної форми вимагає задоволення вимог третьої нормальної форми і, за наявності зв'язків *багато-до-багатьох*, відповідності правилам четвертої (рис.4.16).

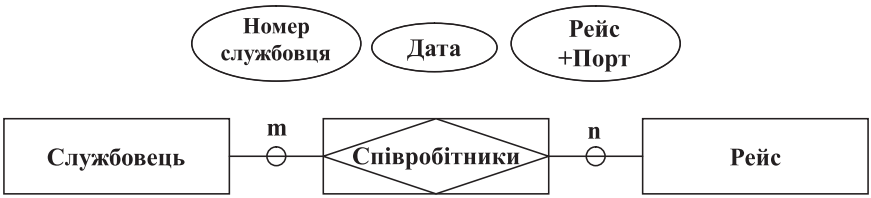



Рисунок 4.16. Схема "об'єкти-відносини" для зв'язку *багато-до-багатьох*.

Використання графічних схем і схем "об'єкти-відносини" в процесі створення баз даних Access дозволяє переконатися, чи відповідає база даних вимогам, що пред'являються вами. Крім усього іншого, схема корисна для пояснення функціонування бази даних тим співробітникам, які її використовуватимуть. Схема "об'єкти-відносини" допомагає знайти структурні помилки, такі як помилки нормалізації таблиць принаймні до четвертої нормальної форми. Набагато простіше небагато поекспериментувати на цьому етапі, чим потім повністю змінювати структуру великої таблиці із-за неправильної її нормалізації. Одним словом, *ретельне планування* – найважливіший етап успішного створення структури бази даних.

**Типи зв'язків**

Між окремими таблицями БД можуть існувати зв'язки. Зв'язані таблиці взаємодіють за принципом головна-підлегла. Головну таблицю називають *батьківською*, а підлеглу – *дочірньої*. Одна і та сама таблиця може бути головною стосовно однієї таблиці БД і дочірньою стосовно іншої.

Звичайно зв'язки розглядаються перед процесом нормалізації. Проте, зараз опис зв'язків слідує за описом процесу нормалізації. Це пояснюється тим, що зв'язки створюються між таблицями, структура яких розробляється відповідно до перших трьох правил нормалізації, розглянутих раніше. Пропонується опис чотирьох основних типів зв'язків між таблицями. А

також ви дізнаєтеся, як за допомогою діаграм "об'єкти-відносини" графічно представити зв'язок.

**Зв'язки один-до-одного**

Зв'язки **один-до-одного** – найпростіші зв'язки, які можна встановити між таблицями. При такому типі зв'язку кожному запису однієї таблиці (батьківської) відповідає тільки один запис іншої таблиці (дочірньої), і навпаки. Зв'язані таким чином таблиці легко об'єднати в одну, що містить стовпці всіх об'єднуваних таблиць.

Зв'язки **один-до-одного** часто використовуються для розбиття дуже широких базових таблиць на декілька вужчих. Розбиття великої таблиці на декілька маленьких дозволить зменшити час проглядання полів з найбільш важливою інформацією. Досить часто необхідно управляти доступом до певних частин таблиць, що містять важливу інформацію. Прикладом тут може служити база даних співробітників: кожен має можливість дістати доступ до таблиці імен співробітників, але тільки співробітники відділу кадрів можуть проглянути інформацію про їх зарплати (рис.4.17).

| Служ-бовець | Посада | Прізвище | Ім'я |
|---|---|---|---|
| 00688 | Капітан | Студень | Андрій |
| 00721 | Капітан | Лопата | Павло |
| 00885 | Старший помошник | Наділяй | Ігор |
| 00912 | Старший помошник | Мурашов | Олег |
| 00955 | Капітан | Почемуйло | Михайло |
| 01023 | Капітан | Завитайло | Максим |
| 01156 | Старший помошник | Дербенев | Анатолій |
| 01203 | Старший помошник | Приходько | Сергій |

| Служ-бовець | Зарплата |
|---|---|
| 00688 | 6500.00 |
| 00721 | 6250.00 |
| 00885 | 5100.00 |
| 00912 | 5000.00 |
| 00955 | 6100.00 |
| 01023 | 6050.00 |
| 01156 | 4900.00 |
| 01203 | 4850.00 |

Рисунок 4.17. Таблиці Співробітники і Зарплата.

На рис. 4.18 показана схема "об'єкти-відносини" для таблиць Співробітники і Зарплата. По обидві сторони ромба зв'язку встановлений символ "1". Це указує на тип зв'язку **один-до-одного**. Участь елементів таблиці в зв'язках може бути як обов'язковою, так і необов'язковою. Необов'язкові зв'язки визначаються за допомогою круга, намальованого на лінії, що сполучає необов'язковий елемент з ромбом зв'язку. Показаний

на рисунку зв'язок Оплачено-зарплата – приклад необов'язкового зв'язку. Річ у тому, що деяким співробітникам може видаватися почасова оплата. В цьому випадку доведеться встановлювати зв'язок з таблицею почасова оплата. З декількох таблиць, зв'язаних відношенням **один-до-одного**, таблиця з обов'язковими записами називається головною; таблиці з необов'язковими зв'язками по відношенню до головної таблиці – *зв'язаними*. Якщо між групами полів таблиці існують необов'язкові зв'язки **один-до-одного**, розбиття її на декілька таблиць, зв'язаних таким відношенням, дозволить значно скоротити використання дискового простору.

Таблиці Співробітники і Зарплата, зв'язані відношенням **один-до-одного.**




Рисунок 4.18. Схема "об'єкти-відносини" для необов'язкового зв'язку один-до-одного.

**Зв'язки один-до-багатьох**

Зв'язки **один-до-багатьох** пов'язують один запис якої-небудь таблиці з двома або декількома записами іншої таблиці. Зв'язок при цьому встановлюється між первинним ключем основної таблиці і відповідним зовнішнім ключем зв'язаної таблиці. Хоча зовнішній ключ зв'язаної таблиці (тобто таблиці із сторони багато) може бути частиною її власного складеного первинного ключа, для даного зв'язку він все ж таки є зовнішнім. Зв'язки **один-до-багатьох** найбільше поширені між таблицями, вони дозволяють моделювати ієрархічні структури даних (рис.4.19).

| Судно | Назва судна | Судно | Рейс | Порт | Тип | Прибуття | Відправлення |
|---|---|---|---|---|---|---|---|
| 528 | Japan Bear | 528 | 9203W | HNL | S | 6/6/92 | 6/8/92 |
| | | 528 | 9204W | HNL | S | 8/27/92 | 8/29/92 |
| | | 528 | 9203W | OSA | S | 7/15/92 | 7/18/92 |
| | | 528 | 9204W | OSA | S | 9/30/92 | 10/2/92 |
| | | 528 | 9203W | SFO | E | | 5/31/92 |
| | | 528 | 9204W | SFO | E | | 8/20/92 |

Рисунок 4.19. Зв'язок **один-до-богатьох** між таблицями **Судно** і **Порти.**

Показаний на рис. 4.20 зв'язок **один-до-багатьох** ставить всі записи таблиці Порти у відповідність одного єдиного запису таблиці Судно. За допомогою цього зв'язку можна відібрати і відобразити на екрані всі порти, в які заходитиме вибраний корабель, – в даному випадку відображені всі записи таблиці Порти, що відносяться до корабля Japan Bear.

На рис. 4.20 представлена діаграма "об'єкти-відносини", що ілюструє відношення **один-до-багатьох**. Кількість записів, що беруть участь в зв'язку між двома таблицями з кожною із сторін, представлено у вигляді символів "1" і "m". Вікно **Relationships** (**Схема даних**) Access використовує символ F (нескінченність) для представлення сторони відносин багато.




Рис. 4.20. Схема "об'єкти-відносини" для зв'язку **один-до-богатьох**, показаного в таблиці.

**Зв'язки багато-до-одного**

Використання зв'язку цього типа дає можливість відобразити назву судна для будь-якого запису таблиці **Порти**. Зв'язок **багато-до-одного** є "дзеркальним віддзеркаленням" зв'язку **один-до-багатьох**, тому говорять, що цей зв'язок – ***рефлексія*** (рис. 4.21). У Access всі зв'язки **богато-до-одного** є рефлексіями. Тому, використовуючи опції діалогового вікна **Edit Relationships** (**Зміна зв'язків**), між головною і зв'язаною таблицями можна визначити тільки два типу зв'язку – **один-до-одного** або **один-до-багатьох**.

| Судно | Рейс | Порт | Тип | Прибуття | Відправлення |
|---|---|---|---|---|---|
| 528 | 9203W | HNL | S | 6/6/92 | 6/8/92 |
| 528 | 9204W | HNL | S | 8/27/92 | 8/29/92 |
| 528 | 9203W | OSA | S | 7/15/92 | 7/18/92 |
| 528 | 9204W | OSA | S | 9/30/93 | 10/2/92 |
| 528 | 9203W | SFO | E | | 5/31/92 |
| 528 | 9204W | SFO | E | | 8/20/92 |

| Судно | Назва судна |
|---|---|
| 528 | Japan Bear |
| | |
| | |
| | |
| | |
| | |

Рисунок. 4.21. Зв'язок між таблицями **Порти** і **Судно** з боку таблиці Порти є зв'язком **богато-до-одного.**

Якщо ви виберете запис в таблиці із сторони багато, то відобразите запис, відповідний її зовнішньому ключу по іншу сторону зв'язку. На рис. 4.22 представлена схема "об'єкти-відносини" для зв'язків, рефлексій. На рефлексивність зв'язку указує дієслово поряд з ромбом зв'язку, залежне від напряму зв'язку.

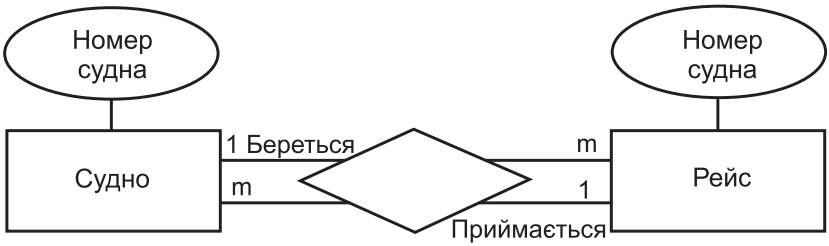



Рисунок 4.22. Схема "об'єкти-відносини" для зв'язку, рефлексії **богато-до-одного.**

**Зв'язки багато-до-багатьох і четверта нормальна форма**

Відношення "багато-до-багатьох" означає, що кожному запису першої таблиці може відповідати кілька записів другої таблиці, а кожному запису другої таблиці може відповідати кілька записів першої таблиці.Зв'язок типу **багато-до-багатьох** не можна встановити між двома таблицями безпосередньо. Вона встановлюється після створення третьої таблиці, пов'язаної з двома основними таблицями відношенням **багато-до-одного**.

На рис. 4.23 ви бачите реляційну таблицю Команди, призначену для відбору членів команди на конкретний рейс. Ця таблиця встановлює зв'язок **багато-до-багатьох** між таблицями **Судна** (за допомогою поля **Судно**) і **Співробітники** (за допомогою номерів співробітників в полях **Капітан**, **Помічник** і **Старший помічник**).

| Судно | Рейс | Порт | Відправлення | Капітан | Старший помічник | Перший помічник |
|---|---|---|---|---|---|---|
| 528 | 9203W | SFO | HNL | 01023 | 01156 | 01367 |
| 528 | 9203W | HNL | HNL | 01023 | 01156 | 01367 |
| 528 | 9203W | HNL | OSA | 01023 | 01156 | 01367 |
| 528 | 9203W | OSA | OSA | 01023 | 01156 | 01367 |
| 528 | 9203W | OSA | INC | 01023 | 01156 | 01367 |

Рисунок 4.23. Перша версія пов'язуючої таблиці Команди.

Показана на рис. 4.23 таблиця зв'язана відношенням **багато-до-одного** між таблицями **Судно** і **Співробітники**. В результаті таблиці Судно і Співробітники виявляються зв'язаними відношенням **богато-до-богатьох**. Хоча найчастіше службовці весь рейс знаходяться на одному судні, тео-

ретично кожний з них може бути призначений на будь-який етап рейса будь-якого судна. Таким чином члени команди судна не залежать один від одного. Відмітимо, що таблиця на рис. 4.23 не задовольняє правилам четвертої нормальної форми, оскільки містить незалежні елементи.

Реструктуризована таблиця Команди показана на рис. 4.24. Вона використовується для призначення службовців на етапи рейса і містить один запис для кожного службовця на кожен етап.

У цю таблицю можна додавати нові елементи, забезпечуючи їх повну залежність від всіх полів зовнішнього ключа. Прикладом такого залежного елементу виступає інформація про оплату. До його атрибутів відноситься кількість відпрацьованого і наднормового годинника і понесені за час плавання витрати. Такі елементи називаються слабкими, або асоціативними, тому що вони використовуються тільки спільно з інформацією з інших основних таблиць. Як тільки ви додасте в згадувану таблицю Команди **асоціативні** елементи, вона перестане бути такою, що виключно пов'язує, оскільки не всі її поля будуть зовнішніми ключами.

| Служащий | Судно | Рейс | Порт | Відправлено в |
|---|---|---|---|---|
| 01023 | 528 | 9203W | SFO | HNL |
| 01156 | 528 | 9203W | SFO | HNL |
| 01367 | 528 | 9203W | SFO | HNL |
| 01023 | 528 | 9203W | HNL | HNL |
| 01056 | 528 | 9203W | HNL | HNL |
| 01367 | 528 | 9203W | HNL | HNL |
| 01123 | 528 | 9203W | HNL | OSA |
| 01158 | 528 | 9203W | HNL | OSA |
| 01367 | 528 | 9203W | HNL | OSA |
| 01023 | 528 | 9203W | OSA | OSA |
| 01156 | 528 | 9203W | OSA | OSA |
| 01367 | 528 | 9203W | OSA | OSA |
| 01023 | 528 | 9203W | OSA | INC |
| 01156 | 528 | 9203W | OSA | INC |
| 01367 | 528 | 9203W | OSA | HNL |

Рисунок 4.24. Таблиця Команди для четвертої нормальної форми.

На рис. 4.25 показана схема "об'єкти-відносини", що ілюструє зв'язок **багато-до-багатьох**, встановлений між двома основними таблицями за допомогою третьої, пов'язуючої, в якій вказані службовці і етапи рейсів, на які ці службовці призначаються. У схемі в таблицю Команди додане ще одне корисне поле – Дата. Його включення дозволить гарантувати, що один службовець в кожен конкретний день може бути призначений тільки на один рейс. Такий зв'язок побудований на датах відправлення і

прибуття. Автоматичне виконання умови, коли один службовець не може бути в декількох місцях в один і той же момент часу, виконується після створення індексу, що не повторюється, містить поля коду співробітника і дати з таблиці Команди. Асоціативні елементи показані в схемі "об'єкти-відносини" у вигляді ромба зв'язку, ув'язненого в прямокутник інформаційного елементу. Саме такий елемент буде створений після додавання в таблицю Команди інформації про оплату. Круги на лініях показують, що призначення службовців на судно є не обов'язковим, оскільки вони можуть піти у відпустку, захворіти або залишитися на березі.

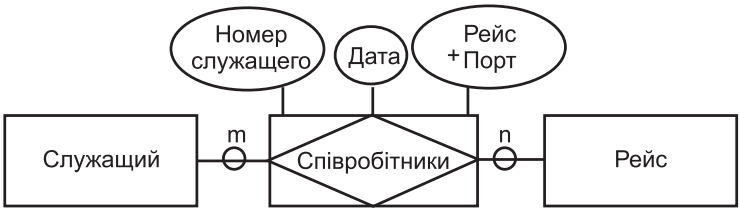



Рисунок 4.25. Схема "об'єкти-відносини" для зв'язку **багато-до-багатьох.**

Використання графічних схем і схем "об'єкти-відносини" в процесі створення баз даних Access дозволяє переконатися, чи відповідає база даних вимогам, що пред'являються вами. Крім усього іншого, схема корисна для пояснення функціонування бази даних тим співробітникам, які її використовуватимуть. Схема "об'єкти-відносини" допомагає знайти структурні помилки, такі як помилки нормалізації таблиць принаймні до четвертої нормальної форми. Набагато простіше небагато поекспериментувати на цьому етапі, чим потім повністю змінювати структуру великої таблиці із-за неправильної її нормалізації. Одним словом, ретельне планування – найважливіший етап успішного створення структури бази даних.

### 4.1.2 Системи управління базами даних. Основні поняття СУБД

#### 4.1.2.1 Типи СУБД їх коротка характеристика

**Система управління базою даних** є пакетом програм, за допомогою якого реалізується централізоване управління базою даних і забезпечується доступ до даних. СУБД виступає як інтерфейс між користувачами і БД.

СУБД забезпечує програмні засоби для створення, завантаження, пошуку і відновлення даних, контролює дії, пов'язані з уведенням-виводом даних, вирішує питання їх спільного використання і захисту.

СУБД служить для підтримки бази даних в актуальному стані і забезпечує ефективний доступ користувачів до даних, що містяться в ній, у рамках наданих користувачам повноважень.

При класифікації сучасних СУБД найчастіше використовують дві ознаки: архітектура та модель організації даних, що використовується. За архітектурою СУБД поділяються на локальні (ті, що підтримують роботу БД на локальних ЕОМ) і розподілені (ті, що можуть використовуватися в мережних БД).

СУБД підтримують один з можливих типів моделей даних – ***мережну***, ***ієрархічну*** чи ***реляційну***, що є одним з найважливіших ознак класифікації СУБД.

В залежності від моделі організації даних виділяють наступні типи СУБД.

***Плоскі бази даних***. Плоскою називається база даних, що складається з однієї таблиці. Як приклад плоскої бази даних можна привести список розсилання, що складається тільки з імен і адрес. Інформація на одному аркуші електронної таблиці, яка не зв'язана з інформацією на іншому аркуші, також є плоскою базою даних. Плоскі бази даних зазвичай легко обробляються і швидко працюють, але їхнє застосування через примітивність обмежене.

***Ієрархічні бази даних***. Ієрархічна база даних – це набір зв'язаних між собою таблиць, що об'єднані за допомогою односпрямованих відносин. Як приклад ієрархічної бази даних можна привести структуру каталогів, у якій один кореневий каталог містить файли і підкаталоги, у свою чергу, що містять файли й підкаталоги, і т.д.

***Мережні бази даних***. Мережна база даних характеризується більшою гнучкістю зв'язків між таблицями. Як приклад можна привести структуру каталогів, у якій один файл чи підкаталог може знаходитися в декількох "батьківських" каталогах. У цьому контексті термін "мережа" позначає спосіб зв'язку таблиць між собою, а не різновид комп'ютерних мереж. Обробка мережних баз даних може бути дуже складною.

***Реляційна база даних*** є найпростішим випадком мережної бази даних; у ній складність зведена до мінімуму завдяки використанню суворих правил збереження інформації й організації внутрішньої структури.

***Об'єктно-орієнтовані бази даних***. Цей тип бази являє собою розширення реляційної моделі. У той час, як у традиційній базі даних зберігаються винятково дані, в об'єктно-орієнтованій базі дані зв'язані з кодом, що їх обробляє. Цей код може виконуватися при звертанні користувача до даних. Зв'язані з даними інструкції називаються методами. Об'єднання даних і методів зветься об'єктом.

За ***ступенем універсальності*** розрізняють два класи СУБД:

1) системи загального призначення;

2) спеціалізовані системи.

СУБД ***загального призначення*** не орієнтовані на яку-небудь предметну область або на інформаційні потреби якої-небудь групи користувачів.

Кожна система такого роду реалізується як програмний продукт, здатний функціонувати на деякій моделі ЕОМ у визначеній операційній системі, і поставляється багатьом користувачам як комерційний виріб. Такі СУБД мають засоби настроювання на роботу з конкретною базою даних. Використання СУБД загального призначення як інструментального засобу для створення автоматизованих інформаційних систем, заснованих на технології баз даних, дозволяє істотно скорочувати терміни розробки, заощаджувати трудові ресурси. В цих СУБД присутні розвиті функціональні можливості і навіть визначена функціональна надмірність.

***Спеціалізовані*** СУБД створюються в рідких випадках при неможливості або недоцільності використання СУБД загального призначення.

СУБД ***загального призначення*** – це складні програмні комплекси, призначені для виконання всієї сукупності функцій, пов'язаних зі створенням і експлуатацією бази даних інформаційної системи.

– Вони дозволяють визначати структуру створюваної бази, ініціалізувати її і робити початкове завантаження даних.

– Вони керують повноваженнями користувачів на доступ до БД, організують паралельний доступ до неї декількох користувачів.

– Вони забезпечують захист логічної і фізичної цілісності даних – захист від руйнувань.

СУБД забезпечують багатоцільовий характер використання бази даних, захист і відновлення даних. Наявність розвинених діалогових засобів і мови запитів високого рівня робить СУБД зручним засобом для кінцевого користувача.

Основними ***засобами*** СУБД є:

- ✧ засоби завдання (опису) структури бази даних;
- ✧ засоби конструювання екранних форм, призначених для введення даних, перегляду і їх обробки в діалоговому режимі;
- ✧ засоби створення запитів для вибірки даних при заданих умовах, а також виконання операцій по їх обробці;
- ✧ засоби створення звітів з бази даних для виведення на друк результатів обробки в зручному для користувача вигляді;
- ✧ мовні засоби – макроси, убудована алгоритмічна мова (Dbase, Visual Basic або інша), мова запитів (QBE – Query By Example, SQL) і т.ін., що використовуються для реалізації нестандартних алгоритмів обробки даних, а також процедур обробки подій у задачах користувача;
- ✧ засоби створення додатків користувача (генератори додатків, засоби створення меню і панелей управління додатками), що дозволяють об'єднати різні операції роботи з базою даних у єдиний технологічний процес.

**СУБД у системах з багатьма користувачами**

База даних, як правило, містить дані, необхідні багатьом користувачам. Одержання одночасного доступу декількох користувачів до загальної бази даних можливо при установці СУБД у локальній мережі персональних комп'ютерів і створенні бази даних для багатьох користувачів.

У мережі СУБД стежить за розмежуванням доступу різних користувачів до загальної бази даних і забезпечує захист даних при одночасній роботі користувачів із загальними даними. Автоматично забезпечується захист даних від одночасного їх корегування декількома користувачами-клієнтами.

У мережі з *файловим сервером* база даних може розміщатися на сервері. При цьому СУБД завантажується і здійснює обробку бази на робочих станціях користувачів. Концепція файлового сервера в локальній мережі забезпечується низкою мережних операційних систем. Найбільш популярними є Microsoft Windows NT і NetWare Novell.

У мережі, що підтримує концепцію *"клієнт-серве"*, використовується сервер баз даних, що розташовується на потужній машині, виконує обробку даних, розміщених на сервері, і відповідає за їх цілісність і збереженість.

Для управління базою даних на сервері використовується мова структурованих запитів SQL (Structured Queries Language). На робочих станціях-клієнтах працює СУБД-клієнт. Користувачі можуть взаємодіяти не тільки зі своїми локальними базами, але і з даними, розташованими на сервері. СУБД-клієнт, у якій підтримується SQL, у повному обсязі може посилати на сервер запити SQL, одержувати необхідні дані, а також посилати оновлені дані.

При цьому з загальною базою даних можуть працювати СУБД різного типу, установлені на робочих станціях, якщо в них підтримується SQL.

Підключення із СУБД до серверів баз даних SQL може бути здійснене за допомогою драйверів ODBC. ODBC (Open Database Connectivity – відкритий стандарт доступу до баз даних) підтримує стандартний протокол для серверів баз даних SQL.

**Властивості СУБД і бази даних**

До основних властивостей СУБД і бази даних можна віднести:

- ✧ відсутність дублювання даних у різних об'єктах моделі, що забезпечує однократне введення даних і простоту їхнього корегування;
- ✧ несуперечність даних;
- ✧ цілісність БД;
- ✧ можливість багатоаспектного доступу;
- ✧ всілякі вибірки даних та їх використання різними задачами і додатками користувача;

- ✧ захист і відновлення даних при аварійних ситуаціях, апаратних і програмних збоях, помилках користувача;
- ✧ захист даних від несанкціонованого доступу засобами розмежування доступу для різних користувачів;
- ✧ можливість модифікації структури бази даних без повторного завантаження даних;
- ✧ забезпечення незалежності програм від даних, що дозволяє зберегти програми при модифікації структури бази даних;
- ✧ реорганізацію розміщення дані бази на машинному носії для поліпшення об'ємно-часових характеристик БД;
- ✧ наявність мови запитів високого рівня, орієнтованої на кінцевого користувача, що забезпечує виведення інформації з бази даних на будь-який запит і надання її у вигляді відповідних звітних форм, зручних для користувача.

**Технологія використання СУБД**

СУБД є основою створення практичних додатків користувача для різних предметних галузей.

*Критерії вибору СУБД користувачем*. Вибір СУБД для практичних додатків користувачем визначається багатьма факторами, до яких відносяться:

- ✧ наявне технічне і базове програмне забезпечення, їх конфігурація, оперативна і дискова пам'ять;
- ✧ потреби розроблювальних додатків користувача;
- ✧ тип підтримуваної моделі даних, специфіка предметної області, топологія інформаційно-логічної моделі;
- ✧ вимоги до продуктивності при обробці даних;
- ✧ наявність у СУБД необхідних функціональних засобів;
- ✧ наявність потрібної мовної версії СУБД;
- ✧ рівень кваліфікації користувачів і наявність у СУБД діалогових засобів розробки і взаємодії з БД.

*Установка СУБД*. СУБД є програмним продуктом, що поставляється у вигляді пакета прикладних програм, який повинене бути встановлений (інстальований) на комп'ютер з урахуванням його конфігурації, ресурсів і операційної системи, а також вимог до набору функцій.

*Процес поетапного впровадження.* Після установки СУБД можна здійснювати створення БД, у тому числі задавати структуру БД, робити введення даних, а також виконувати будь-які дії, передбачені функціональними можливостями СУБД. Варто відмітити, що сучасні СУБД для ПК мають достатню гнучкість. Це дозволяє на самих ранніх етапах розробки додатків користувача приступати до створення окремих частин БД. Така БД у міру поглиблення розробки може легко розширюва-

тися і модифікуватися. Таким чином, полегшується прискорене освоєння персоналом технології роботи з БД, вивчення можливостей СУБД і поетапне впровадження.

*Розробка структури бази даних*. Розробка додатків на основі СУБД припускає підготовку рішень за структурою БД. Ці рішення безпосередньо зв'язані з позамашинною сферою – з описом позамашинної інформаційної бази, її документів, що містять необхідну інформацію, а також з постановкою й алгоритмізацією задач по обробці цієї інформації.

На початковому етапі розробки структури БД доцільна побудова інформаційно-логічної моделі, що відбиває логічну структуру інформації предметної області. Така модель, що відповідає вимогам нормалізації даних, є основою створення реляційних баз даних.

*Створення бази даних засобами СУБД*. Відповідно до розробленої структури бази даних здійснюється її створення засобами СУБД на машинному носії і введення в експлуатацію. Для забезпечення процесів створення БД та її експлуатації необхідне знання можливостей інструментальних засобів СУБД. При цьому варто керуватися рекомендаціями з технології використання засобів СУБД. Така технологія повинна визначати всі необхідні процеси, включаючи первісне введення, завантаження БД і контроль даних, виконання операцій по внесенню змін, реалізацію запитів для одержання потрібних довідок, відновлення БД і т.ін. Одним з найважливіших етапів цієї технології є підготовка екранних форм введення-виведення для завантаження інформації з документів позамашинної сфери в базу даних, корегування даних і їх перегляду.

*Обробка даних засобами СУБД*. Додавання, видалення, зміна і вибірка даних виробляється за допомогою мови запитів, вбудованої алгоритмічної мови й інших засобів СУБД. Реалізація запитів забезпечується або діалоговою системою команд або запитами за прикладом QBE (Query By Example). У першому випадку окремий запит виконується однією або декількома командами мови СУБД.

В другому – для виконання запиту користувач вибирає послідовно один або кілька пунктів меню або вказує в запиті приклад (зразок), за яким складається запит, а також при необхідності вказує умови вибору й операції обчислення, які необхідно виконувати з даними (СУБД Paradox, Access). Послідовність команд меню і запитів може бути збережена як програма-макрос і надалі виконуватись так само, як командний файл.

СУБД може мати включену або базову мову програмування. У СУБД із включеною мовою використовується одна з універсальних алгоритмічних мов (C, Pascal тощо). Прикладна програма, написана включеною мовою, може ініціювати команди СУБД. У СУБД із базовою мовою застосовується власна алгоритмічна мова, що дозволяє крім операцій маніпулювання даними виконувати різні обчислення й обробку даних.

*Стандартною реляційною мовою запитів* є мова структурованих запитів SQL (Structured Queries Language).

Мова запитів SQL реалізована у цілій низці популярних СУБД для різних типів ЕОМ або як базова, або як альтернативна. У силу свого широкого використання вона є міжнародним стандартом мови запитів. Мова SQL надає розвинуті можливості як кінцевим користувачам, так і фахівцям в області обробки даних.

Сумісність з SQL-системами відіграє велику роль, коли передбачається проведення роботи з корпоративними даними. Access і Paradox for Windows працюють із джерелами SQL-даних, сумісних із системою ODBC (Open Database Connectivity – відкрите з'єднання баз даних). FoxPro (для DOS і для Windows) поставляються з додатковими бібліотеками, що забезпечують доступ до SQL-баз даних, здатних працювати разом із системою ODBC.

Можна прямо керувати базами даних Access за допомогою мови SQL і передавати наскрізні SQL-запити сумісним за специфікацією ODBC SQL-базам даних, таким, як MS SQL Server і Oracle.

**Найпоширеніші СУБД**

Ринок програмного забезпечення ПК має у своєму розпорядженні велике число різноманітних за своїми функціональними можливостями комерційних систем управління базами даних загального призначення, а також засобами їх оточення практично для всіх масових моделей машин і для різних операційних систем.

СУБД, що використовуються в даний час, мають засоби забезпечення цілісності даних і безпеки, що дає можливість розроблювачам гарантувати велику безпеку даних при менших витратах сил на низькорівневе програмування.

Продукти, що функціонують у середовищі WINDOWS, вигідно відрізняються зручністю користувальницького інтерфейсу й убудованих засобів підвищення продуктивності.

Серед СУБД загального призначення найбільш відомими є:

- ✧ Microsoft Access;
- ✧ Microsoft Visual FoxPro;
- ✧ Paradox 9;
- ✧ IBM DB2 Universal Database;
- ✧ Microsoft SQL Server 2000;
- ✧ Oracle 8й.

Ще однією характеристикою є продуктивність СУБД, яка оцінюється за:

- ✧ часом виконання запитів;
- ✧ швидкістю пошуку інформації;
- ✧ часом виконання операцій імпортування бази даних з інших форматів;

- ✧ швидкістю створення індексів і виконання таких масових операцій, як відновлення, вставка, видалення даних;
- ✧ максимальним числом паралельних звертань до даних у режимі з багатьма користувачами;
- ✧ часом генерації звіту.

Так, наприклад, досить швидкою СУБД є FoxPro 2.6, однак вона не має засобів дотримання цілісності даних на відміну від більш повільної СУБД Access 2.0.

*Забезпечення цілісності даних на рівні бази даних*. Ця характеристика має на увазі наявність засобів, що дозволяють упевнитися, що інформація в базі даних завжди залишається коректною і повною. Повинні бути встановлені правила цілісності, і вони повинні зберігатися разом з базою даних і дотримуватися на глобальному рівні.

Access і Paradox for Windows краще інших СУБД забезпечують надійність збереження цілісності даних на рівні бази даних; правила зберігаються разом з базою даних і автоматично дотримуються.

*Забезпечення безпеки*. Деякі СУБД передбачають засоби забезпечення безпеки даних. Такі засоби забезпечують виконання наступних операцій:

- ✧ шифрування прикладних програм;
- ✧ шифрування даних;
- ✧ захист паролем;
- ✧ обмеження рівня доступу (до бази даних, до таблиці, до словника, для користувача).

Найвищий рівень безпеки даних реалізований у СУБД dBASE IV.

Адміністратор може призначити системі різні права доступу на рівні файлу, поля, а також організувати автоматичне шифрування даних.

Не поганими характеристиками забезпечення безпеки відрізняється Access 2.0. Вона передбачає призначення паролів для індивідуальних користувачів або груп користувачів і присвоєння різних прав доступу окремо таблицям, запитам, звітам, макрокомандам або новим об'єктам на рівні користувача або групи.

*Робота в середовищах з багатьма користувачами*. Практично всі розглянуті СУБД призначені для роботи в середовищах з багатьма користувачами, але мають для цього різні можливості.

Обробка даних у середовищах з багатьма користувачами припускає виконання програмним продуктом наступних функцій:

- ✧ блокування бази даних, файлу, запису, поля;
- ✧ ідентифікацію станції, що установила блокування;
- ✧ відновлення інформації після модифікації;
- ✧ контроль за часом і повторення звернення;
- ✧ обробку транзакцій (транзакція – послідовність операцій користувача над базою даних, що зберігає її логічну цілісність);

- ✧ роботу з мережними системами (LAN Manager, NetWare, Unix).

Непогані можливості для роботи в середовищах з багатьма користувачами мають СУБД Paradox for DOS 4.5, Access 2.0 і dBASE IV.

*Імпорт-експорт*. Ця характеристика відображує:

- ✧ можливість обробки СУБД інформації, підготовленої іншими програмними засобами;
- ✧ можливість використання іншими програмами даних, сформованих засобами розглянутої СУБД.

Особливий інтерес представляють наступні формати файлів: ASCII-файли, .DBF, .WK*, .XLS.

Усі розглянуті тут СУБД мають гарні можливості імпорту-експорту даних.

### 4.1.2.2 Призначення СУБД Microsoft Access та її характеристика. Об'єкти СУБД Microsoft Access

Система управління базами даних Microsoft® Access, що з'явилася в 1992 році, швидко завоювала популярність серед користувачів самих різних категорій. Її версія Access 1.0 стала першою реляційною СУБД, спеціально розробленою для операційної системи Microsoft Windows®. Знавців подібних продуктів уже тоді вразило, що потужна база даних для настільних систем може бути настільки простою у роботі. Поява Access 2.0 ще сильніше змінила підхід кінцевих користувачів до розуміння й застосування баз даних. Коли Access була включена в Microsoft Office, користувачі цього комплекту почали усвідомлювати сильну потребу в реляційній СУБД для пошуку й управління даними, що допомагає приймати більше вдалі ділові рішення.

Ріст популярності цієї СУБД продовжився в 1995 з випуском Access 95 – першої у світі 32-розрядної реляційної системи управління базами даних. Access 97, що з'явилася в січні 1997 року, поєднала кращі засоби роботи з базами даних і з Інтернетом, пропонуючи прості можливості обміну статичними й динамічними даними по корпоративній інтрамережі.

Сьогодні Access знаходить все більше поширення, її застосовують не тільки досвідчені, але й користувачі-початківці. У версії Access 2000 здійснюється тісна інтеграція з іншими додатками комплекту Office і пропонується знайомий інтерфейс, що допомагає швидко приступитися до роботи. У той же час досвідчені користувачі й розроблювачі відкриють для себе нові переваги, пов'язані із широким поширенням цієї СУБД на робочих місцях, оскільки масштабованість Access 2000 збільшилася завдяки більше тісною інтеграцією з базами даних масштабу підприємства. Незалежно від того, чи створює користувач базу даних для управління контактами й клієнтами або систему управління складськими запасами,

Access пропонує користувачам з будь-яким рівнем підготовки зручні засоби пошуку, управління й обміну даними.

Призначення СУБД Access.

Access є функціонально повною реляційною СУБД, яка надає користувачу широкі можливості зі створення структур даних, їх описання, роботи з ними й організації колективного доступу до інформації. Можливості СКБД значною мірою розширені стандартними елементами управління Windows та впровадженими об'єктно-орієнтованими середовищами.

СУБД Access надає необхідні засоби для роботи з базами даних недосвідченому користувачеві, дозволяючи йому легко й просто створювати бази даних, уводити в них інформацію, обробляти запити й формувати звіти.

База даних в MS Access – це сукупність інструментів для уведення, зберігання, перегляду, вибірки й управління інформацією. До цих засобів відносяться таблиці, форми, звіти, запити. В MS Access підтримуються два способи створення бази даних. Можна створити порожню базу даних, а потім додати в неї таблиці, форми, звіти та інші об'єкти. Такий спосіб є найбільш гнучким, але вимагає окремого визначення кожного елемента бази даних. Крім цього, є можливість створити за допомогою майстра базу даних певного типу з усіма необхідними таблицями, формами й звітами. MS Access містить великий вибір підготовлених баз даних. Другий спосіб у багатьох випадках може виявитися кращим. В обох випадках залишається можливість у будь-який час змінити й розширити створену вами базу даних.

Розглянемо об'єкти СУБД Access (рис. 4.26). До них належать:

1. Таблиці, які містять дані.
2. Запити, які дозволяють створювати умови для відбору даних і вносить зміни в дані.
3. Форми, які дозволяють переглядати й редагувати інформацію.
4. Сторінки (починаючи з Access 2000) – файли у форматі HTML, що дозволяють переглядати дані Access засобами Internet Explorer.
5. Звіти, які дозволяють узагальнювати й роздруковувати інформацію.
6. Макроси, які виконують одну чи кілька операцій автоматично.
7. Модуль – програма автоматизації і налаштування функцій БД, написана на мові VBA.

Як правило, за етапом визначення об'єктів, які будуть входити в таблиці бази даних, наступає етап побудови таблиць із цими об'єктами. Для створення таблиць, що відповідають моделі реляційній бази даних, використається процес, називаний нормалізацією (normalization). Нормалізація – це видалення з таблиць повторюваних даних у результаті їхнього переносу в нові таблиці, записи яких не містять повторюваних значень. Після цього за допомогою загальних значень полів між новими

таблицями встановлюються зв'язки – так утвориться структура реляційній бази даних.

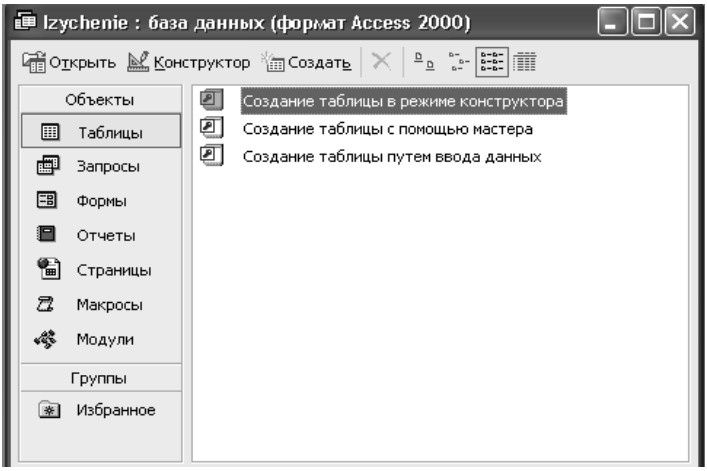



Рисунок 4.26. Вікно робочого поля бази даних.

**Створення таблиць.**

Створення таблиці в MS Access здійснюється у вікні бази даних.

Розглянемо послідовність дій при створенні таблиці в новій базі даних:

1. Відкрийте вікно створеної Вами бази даних і перейдіть на вкладку "Таблиці".

2. Натисніть кнопку Створити у вікні бази даних.

3. Відкриється вікно діалогу "Нова таблиця", у правій частині якого перебуває список варіантів подальшої роботи:

- ✧ Режим таблиці дозволяє створити нову таблицю в режимі таблиці;
- ✧ Конструктор дозволяє створити нову таблицю в конструкторі таблиць;
- ✧ Майстер таблиць дозволяє створити нову таблицю за допомогою майстра;
- ✧ Імпорт таблиць дозволяє здійснити імпорт таблиць із зовнішнього файлу в поточну базу даних;
- ✧ Зв'язок з таблицями дозволяє здійснити створення таблиць, пов'язаних з таблицями із зовнішніх файлів.

4. Виберіть із цієї таблиці потрібний варіант створення таблиці й натисніть кнопку ОК.

При використанні майстра таблиць Вам немає необхідності піклуватися про типи даних. Коли Ви вибираєте поле зі списку Зразки полів, майстер припускає, що він знає, який тип даних вам підійде. Після

того як таблиця створена, можна перейти в режим конструктора таблиць і подивитися, які типи даних вибрав майстер для Ваших полів.

Створення таблиць у вікні конструктора надає більш широкі можливості по визначенню параметрів створюваної таблиці.

У верхній частині вікна діалогу знаходиться таблиця, що містить такі атрибути створюваної таблиці: найменування поля, тип даних й опис. Крім цих основних атрибутів, кожне поле таблиці має додаткові властивості, які відображуються в нижній частині конструктора й визначальної умови уведення даних. Ім'я кожного з полів таблиці, як правило, вибирається довільно, але таким чином, щоб відбити характер інформації, яка буде зберігатися в даному полі. Тип поля визначається типом даних, що зберігаються в цьому полі.

При завданні імен полів потрібно додержуватися таких правил:

1. Ім'я поля може містити до 64 символів, але не слід зловживати цією можливістю, задаючи занадто довгі імена.
2. Ім'я поля може містити букви, цифри, пробіли й спеціальні символи, за винятком крапки (.), знак оклику (!), прямих дужок ([]) і деяких керуючих символів (з кодами ASCII 0-31).
3. Ім'я поля не може починатися із пробілу.
4. Два поля в одній таблиці не можуть мати однакових імен.
5. Як імена полів не рекомендується використати абревіатури або короткі назви.

Недотримання цих правил відслідковується засобами СУБД MS Access, але в деяких випадках це може привести до важко обумовлених помилок, тому рекомендується самостійно контролювати виконання перерахованим вище правилам у практичній роботі.

Бажано намагатися використати імена, що відрізняються стислістю, для полегшення їхньої ідентифікації при перегляді таблиць.

Типи даних. Ім'я поля використовується для посилання на дані таблиці. Для визначення типу даних, що зберігаються, використовується тип даних. Тип поля вводиться в поле уведення стовпця Тип даних. В MS Access припустимими є дані таких типів (таблиця 4.1):

Кожний з типів даних наділений власними властивостями, які відображаються в розділі "Властивості поля" вікна конструктора.

При уведенні імені поля за замовчуванням MS Access присвоює йому текстовий тип даних із шириною поля, рівною 50.

Текстові поля можуть містити букви, цифри й спеціальні символи. Максимальна ширина поля становить 255 символів.

Для зміни ширини поля потрібно в рядку Розмір поля роздягнула "Властивості поля" задати число, що визначає ширину поля (від 1 до 255). Наприклад, якщо Ви задасте ширину поля рівної 25, те це означає, що в таке поле не можуть бути уведені значення, довжина яких перевищує 25 символів.

Числові поля. При введенні даних числового типу автоматично виробляється перевірка даних. Якщо дані містять текстові або спеціальні символи, MS Access видасть попередження й помилкові дані не будуть уведені. Над числовими полями можна виконувати математичні операції.

Таблиця 4.1

**Типи даних в MS Access**

| Тип даних | Коротка характеристика |
|---|---|
| **Текстовый** | Текст та числа (може містити до 255 символів) |
| **Поле МЕМО** | Довгий текст та числа (коментарі, пояснення тощо). Може містити до 64 000 символів |
| **Числовой** | Загальний тип для числових даних. Допускає проведення математичних розрахунків та встановлення різних типів числових даних у полі **Свойства** (клітина **Размер поля**) |
| **Дата/время** | Значення дати та часу |
| **Денежный** | Грошові значення |
| **Счетчик** | Послідовні номери, починаючи з 1 (установлюються автоматично) |
| **Логический** | Значення “**Да”**/“**Нет”**, “**Истина”**/“**Ложь”**, “**Вкл”**/“**Выкл”** |
| **Поле объекта OLE** | Об’єкти, створені в інших програмах та прив’язані чи впроваджені в базу даних *Microsoft Access* |

Для вказівки кількості десяткових знаків використовується властивість поля Число десяткових знаків. Воно може приймати значення від 0 до 15. Атрибут Авто даної властивості служить для автоматичної установки кількості знаків після коми.

Використовуючи значення властивості Розмір поля, можна встановити такий формат для полів числового типу (таблиці 4.2):

*Поля типу Лічильник*. Поля типу Лічильник призначені для зберігання даних, значення яких не редагуються, а встановлюються автоматично при додаванні кожного нового запису в таблицю.

Їхні значення є унікальними, вони послідовно зростають на 1 при додаванні кожного нового запису або можуть бути будь-якими випадковими числами.

Такі поля дозволяють проводити автоматичну перенумерацію даних, що буває дуже зручно при веденні різних кодифікаторів, ключів й упорядкованих масивів даних, тому що створює умови для застосування алгоритмів швидкого сортування й пошуку даних.

*Поля дата/час*. Використовуються для зберігання дат і часу. Дані представлені в спеціальному форматі, що надає ряд додаткових переваг.

MS Access дозволяє вводити й відображати дати в декількох форматах. Конкретний варіант відображення дати/часу встановлюється у властивості Формат поля вікна конструктора таблиць. При зберіганні даних ці формати автоматично переводяться у внутрішнє подання даних.

При уведенні дат MS Access здійснює перевірку формату даних і допускає уведення тільки правильних дат.

У полі даного типу може знаходитися будь-яка дата й будь-який час: від 1 січня 100 р. до 31 грудня 9999 р. Дані цього типу займають 8 байт.

*Логічні поля*. Логічні поля використаються для зберігання даних, які можуть приймати одне із двох можливих значень. Властивість Формат поля логічного поля дозволяє використати спеціальні формати або один із трьох вбудованих: Истина/Ложь, Да/Нет або Вкл/Выкл. При цьому значення Истина, Да й Вкл еквівалентні логічному значенню True, а значення Ложь, Нет й Выкл еквівалентні логічному значенню False.

Таблиця 4.2

**Властивості числового типу даних**

| Властивості | Можливі значення (призначення) властивостей |
|---|---|
| **Размер поля (Fiele Size)** | Максимальна довжина тексту або тип подання чисел |
| **Формат поля (Format)** | Формат подання даних (використовується один із стандартних або створюється спеціальний) |
| **Дробная часть (Decimal Places)** | Число знаків справа від коми |
| **Маска ввода (Input Mask)** | Символи форматування для введення даних (використовуються стандартні або створюються спеціальні) |
| **Подпись (Caption)** | Стандартний підпис поля у формі або звіті |
| **Значение по умолчанию (Default Value)** | Значення, що автоматично вводиться в поле при створенні запису |
| **Условие на значение (Validation Rule)** | Вираз, що визначає умову для введення даних |
| **Сообщение об ошибке (Validation Text)** | Текст повідомлення, який виводиться при введенні даних у поле з порушенням умови на їх значення |
| **Обязательное поле (Required)** | Значення, що визначає можливість введення пустих значень |

Якщо Ви виберете вбудований формат, а потім будете вводити еквівалентне логічне значення, то введене значення буде відображатися

в обраному форматі. Наприклад, якщо значення True або Вкл уводиться в елемент управління типу Поле, для якого у властивості Формат поля зазначений Да/Нет, то уведене значення відразу перетвориться в Да.

При створенні спеціального формату видалите поточне значення із властивості Формат й уведіть свій власний формат.

*Текстові поля довільної довжини*. Текстові поля довільної довжини (поля МЕМО) можуть містити ті ж типи даних, що й прості текстові поля. Відмінність між цими полями полягає в тім, що довжина полів типу МЕМО може дорівнює 64000 символів.

*Поля об'єкта OLE*. MS Access дозволяє зберігати в таблицях зображення й інші двійкові дані (наприклад, електронну таблицю MS Excel, документ MS Word, малюнок, звукозапис). Для цих цілей служить тип даних Поле об'єкта OLE. Фактичний обсяг даних, що можна ввести в поле даного типу, визначається обсягом жорсткого диска комп'ютера.

*Тип даних майстер підстановок*. Вибір цього типу даних запускає майстер підстановок, що створює поле, у якому пропонується вибір значень зі списку, що розкривається, з постійним набором значень або значень із іншої таблиці.

Розмір даного поля збігається з розміром ключового поля, використовуваного як підстановка (звичайно, 4 байти).

Запити створюються для виклику даних з таблиць з метою їх перегляду, використання та зміни за вимогами користувача. Відповідно до функціонального призначення в системі управління базою даних **Access** вони класифікуються таким чином:

- ✧ запит-вибірка задає запит про дані таблиці бази даних та видає вказаний динамічний набір в режимі таблиці (або форми); зміни, що вносяться у динамічному наборі, вносяться та зберігаються в базових таблицях;
- ✧ запит-зміна використовується для зміни або переміщення даних; до цього типу належать запити на доповнення записів та їх видалення, створення чи оновлення таблиці;
- ✧ перехресний запит здійснює групування даних та їх видачу в компактному вигляді;
- ✧ запит з параметрами дозволяє визначати одну або декілька умов відбору даних при виконанні запиту;
- ✧ запит SQL створюється за допомогою мови SQL в режимі SQL та використовується для об'єднання даних різних полів таблиць, створення, видалення й корегування таблиць та використання даних баз даних, що зберігаються на файл-серверах.

На практиці часто потрібно вибрати з вихідної таблиці частину записів, що задовольняють певним критеріям, і впорядкувати вибірку. Критерії можуть визначатися сполученням ряду умов. Для рішення таких

завдань призначені майстри запитів і конструктор запитів, за допомогою яких можна:

- ✧ формувати складні критерії для вибору записів з однієї або декількох таблиць;
- ✧ вказати поля, відображувані для обраних записів;
- ✧ виконувати обчислення з використанням обраних даних.

Запит за зразком (QBE) – це інтерактивний засіб для вибору даних з однієї або декількох таблиць. При формуванні запиту Вам необхідно вказати критерії вибірки записів у вихідній таблиці. При цьому замість того, щоб друкувати пропозиції спеціальною мовою, потрібно просто заповнити бланк запиту, що розташовується у вікні конструктора запитів. Метод формування запиту шляхом заповнення бланка простий для вивчення й розуміння. Він сприяє ефективному використанню можливостей MS Access користувачами, що мають навіть мінімальну навичку роботи з додатком або не мають його зовсім.

Для створення найпростіших запитів можна використати майстер запитів, що послідовно запитує найменування таблиць, використовуваних у запиті, перелік полів таблиць і деякі додаткові параметри.

Для деяких користувачів табличне подання даних може бути незручним. Часто дані необхідно розташовувати у такому вигляді, в якому зберігаються відповідні документи. Система Access дозволяє розташовувати на екрані дані так, як того бажає користувач, тобто у вигляді різноманітних форм. Форма може використовуватись також для пошуку даних.

Система Access може автоматично створювати три види форм:

- ✧ стовпцеву,
- ✧ стрічкову,
- ✧ табличну.

При роботі з базами досить часто виникає проблема виведення даних, які були оброблені в БД. Рішення даної проблеми зв'язано з створенням і формуванням звітів. В звіті користувач може вибирати дані з однієї або кількох таблиць чи запитів БД, робити необхідні обчислення та оформляти його в найбільш зручній формі. В MS Access можна використовувати зразки стандартних структур звітів (в основному це більш прості форми) або будувати свій звіт будь-якої складності. Якщо у користувача ще немає навиків побудови звітів, краще користуватись програмними засобами MS Access для побудови звітів (що має назву Мастер Отчетов).

Для створення звітів потрібно вибрати один із варіантів їх створення, які означають:

- – Конструктор – побудова звіту користувачем;
- – Мастер отчетов – побудова звіту за допомогою програмних засобів MS Access;

- Автоотчет: в столбец – вибрані або розрахункові поля видаються в звіті в колонку;
- Автоотчет: ленточный – відрізняється від попереднього тим, що при цьому варіанті не встановлюються розміри сторінки;
- Диаграмма – інформація вибраних полів використовується для побудови діаграм або графіків;
- Почтовые наклейки – використовується для розсилки пошти шляхом підготовки адресних етикеток з наступним наклеюванням їх на поштові конверти.

### 4.1.3 Етапи проектування бази даних

4.1.3.1 Мета та підходи до проектування бази даних

**Основна мета проектування бази даних –** це скорочення надмірності збережених даних, а отже, економія об'єму використовуваної пам'яті, зменшення витрат на багаторазові операції відновлення надлишкових копій і усунення можливості виникнення протиріч через збереження в різних місцях відомостей про один й той самий об'єкт. Результатом проектування БД є перетворення опису предметної галузі у внутрішню схему БД.

При проектуванні бази даних вирішуються три основних проблеми:

Проблема адекватного відображення предметної галузі та інформаційних потреб користувачів у концептуальній моделі. Цю проблему називають проблемою інфологічного проектування баз даних, на якому створюють змістовні концептуальні моделі

Проблема знаходження способу адекватного відображення об'єктів предметної галузі в абстрактні об'єкти моделі даних так, щоб це відображення не суперечило семантиці предметної галузі, і було, по можливості, найкращим (ефективним, зручним тощо). Цю проблему називають проблемою логічного проектування баз даних Ціль логічного етапу проектування – організація даних, виділених на попередньому етапі, у форму, прийняту в обраній СУБД чи СУБД обраного класу.

Проблема забезпечення ефективності виконання запитів до БД, тобто як, з урахуванням особливостей конкретної СУБД, розташувати дані у зовнішній пам'яті. Це проблема фізичного проектування баз даних. Мета цього етапу – вибір раціональної структури збереження даних і методів доступа до них.

У ході проектування БД доводиться вирішувати такі **задачі**:

- ✧ виявлення і представлення даних і зв'язків між ними, необхідних для всіх основних областей застосування даного додатка і будь-яких існуючих груп його користувачі;
- ✧ створення моделі даних, здатних підтримувати виконання будь-яких необхідних трансакцій операцій обробки даних;

- ✧ розробка попереднього варіанта проекту, структура якого дозволяє задовольнити всі основні вимоги з боку продуктивності системи.

**Основні підходи до проектування БД**

Існують два основних підходи до проектування БД – висхідний і нисхідний.

При висхідному підході робота починається із самого нижчого рівня – рівня визначення атрибутів (тобто опису властивостей елементів предметної галузі, відображуваних у БД), що на основі аналізу існуючих між ними зв'язків групуються в певні відношення між сутностями. Підхід добре підходить для проектування невеликих БД, у яких атрибутів небагато і їх легко установити відразу.

Для складних БД більш придатною стратегією є нисхідний підхід. Він починається з розробки моделей даних, утримуючих кілька високорівневих сутностей і зв'язків і полягає в поступовому переході до сутностей більш низького рівня і зв'язкам між ними. Цей підхід демонструється в концепції "сутність-зв'язок".

Найчастіше використовуються комбіновані стратегії проектування, наприклад, підхід "зсередини назовні" і "змішана стратегія проектування". Підхід "зсередини назовні" схожий на висхідний, але відрізняється від нього початковою ідентифікацією набору всіх основних сутностей з наступним розширенням кола сутностей, атрибутів і зв'язків між ними за рахунок тих, котрі взаємодіють із початково визначеними сутностями. У змішаній стратегії висхідний і нисхідний підходи використовуються для різних частин моделі, що потім збираються в одне ціле.

4.1.3.2 Етапи проектування баз даних

Задачі й етапи проектування БД випливають із трирівневої схеми (так називаної моделі ANSI/SPARC) даних (зовнішня – концептуальна – внутрішня).

Укрупнено виділяють три основних етапи проектування БД (рис. 4.27):

- ✧ етап інфологічного проектування;
- ✧ етап логічного проектування;
- ✧ етап фізичного проектування.

**Концептуальне, чи інфологічне проектування** – це процес створення моделі використовуваної на підприємстві інформації, що не залежить від будь-яких фізичних аспектів її представлення. Інфологічне проектування має своїм результатом одержання семантичних моделей, що відбивають інформаційний зміст конкретної предметної області, тобто концептуальної моделі даних для аналізованої частини підприємства. Ця модель створюється на основі аналізу інформації, записаної в

специфікації користувача, і сформованої в ході підготовки до проектування користувальницької моделі БД. Ця модель абсолютно не залежить від логічних і фізичних подробиць реалізації БД. На цьому етапі виконується сприйняття реальної дійсності, абстрагування, вивчення й опис предметної області. У результаті цього визначаються об'єкти, властивості і зв'язки об'єктів, які мають відбиватися при проектуванні БД.

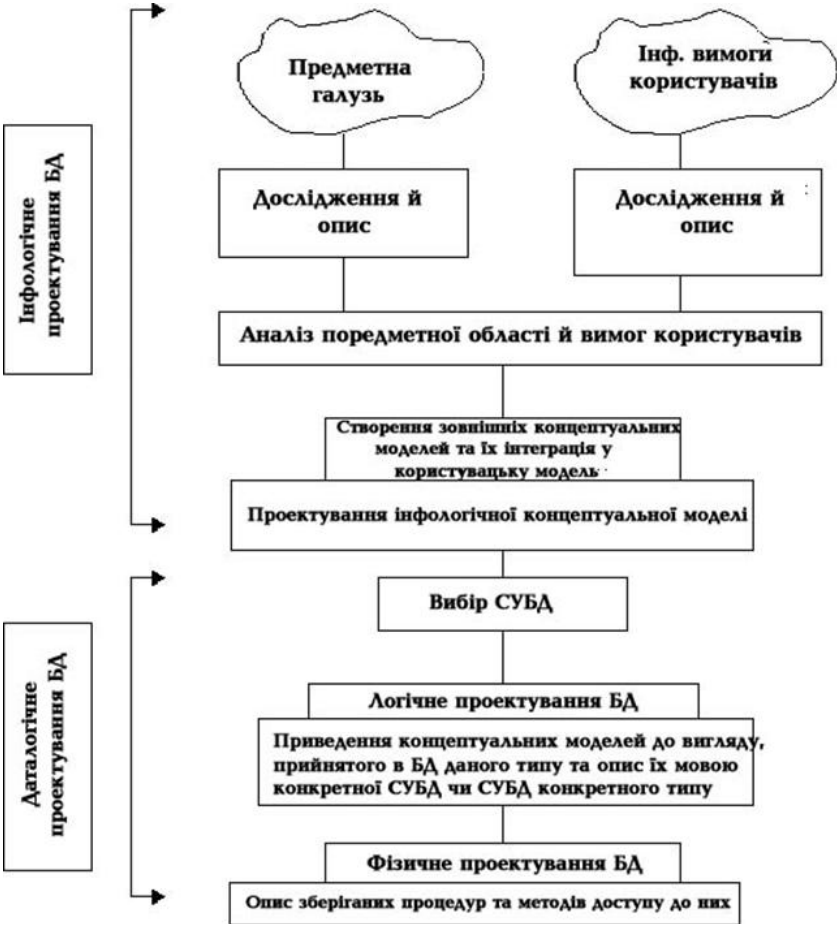



Рисунок 4.27. Схема проектування бази даних.

**Концептуальне проектування** передбачає виконання наступних кроків:

а) аналіз вимог до БД із боку користувачів і побудову так називаної користувацької моделі (вивчення і систематизація інформації про

користувачів, їхні інформаційні потреби, джерела інформації, реальні інформаційні потоки і бізнес-правила);

b) виявлення основних об'єктів (сутностей), що підлягають відображенню в моделі, виходячи зі специфіки предметної області, цілей замовника і наявного ділового регламенту;
c) визначення атрибутів сутностей і їхнє приведення у відповідність із вимогами майбутньої об'єктної моделі;
d) визначення зв'язків між сутностями і властивостей сутностей;
e) вибір типу інфологічної (об'єктної) моделі і її побудова. Найчастіше це модель "сутність – зв'язок" (ER-діаграма);
f) перевірку моделі даних;
g) побудову розподіленої моделі даних;
h) вибір моделі організації даних у концептуальній схемі.

**Логічне проектування** – процес створення моделі використовуваної на підприємстві інформації з урахуванням обраної моделі організації даних, але незалежно від типу цільової СУБД і інших фізичних аспектів реалізації. Ціль етапу – створення логічної моделі даних на рівні моделі записів для досліджуваної частини підприємства шляхом уточнення і перетворення концептуальної моделі з урахуванням особливостей обраної організації даних у цільовий СУБД (наприклад, реляційна чи мережева модель). Для реляційних моделей на цьому етапі фіксується набір і структура таблиць, що представляють сутності, визначаються ключі і проводиться нормалізація таблиць.

Якщо концептуальна модель даних не залежить від будь-яких фізичних аспектів реалізації, то логічна модель даних створюється на основі обраної моделі організації даних у цільовий СУБД. На цьому етапі вже повинне бути відомо, яка СУБД буде використовуватися в якості цільової – реляційна, мережева, ієрархічна чи об'єктно-орієнтована. Однак на цьому етапі ігноруються всі інші аспекти обраної СУБД – наприклад, будь-які аспекти фізичної організації її структур збереження даних і побудови індексів.

У процесі розробки логічна модель постійно тестується і перевіряється на відповідність вимогам користувачів. Для перевірки коректності логічних моделей використовуються спеціальні вимоги. Наприклад, для реляційних моделей – це метод нормалізації. Результатом логічного проектування є організація даних, виділених на попередньому етапі проектування у форму, прийняту для моделі організації даних в СУБД обраного типу. Визначаються також зовнішні обмеження по використанню СУБД.

Логічна модель є основним джерелом інформації на етапі фізичного проектування, а також полегшує експлуатацію і супровід БД, тому що логічна модель дозволяє наочно представити будь-які внесені в БД зміни й оцінити їхній вплив на прикладні програми і використання даних.

**Фізичне проектування** – це процес створення опису реалізації БД на вторинних запам'ятовуючих пристроях із указівкою структур збереження і методів доступу, використовуваних для організації ефективної обробки даних. Фізичний етап проектування забезпечує вибір раціональної структури збереження даних і методів доступу до них, виходячи з арсеналу методів і засобів, що надаються розроблювачу конкретної СУБД.

Логічна модель даних залежить тільки від прийнятої в СУБД моделі організації даних (реляційна, мережева тощо), фізична – від конкретної СУБД, з вибору якої власне і починається етап фізичного проектування. Між логічним і фізичним проектуванням існує нерозривний зворотний зв'язок: рішення, прийняті на етапі фізичного проектування для підвищення продуктивності системи можуть впливати на структуру логічної моделі даних.

Основною метою фізичного проектування БД є опис способу фізичної реалізації логічного проекту БД. У випадку реляційної моделі даних під цим мається на увазі наступне:

- створення набору реляційних таблиць і обмежень на них на основі інформації, представленої в глобальній логічній моделі даних;
- визначення конкретних структур збереження даних і методів доступу до них, що забезпечують оптимальну продуктивність системи з БД;
- розробка засобів захисту створюваної системи.

### 4.1.3.3 Поняття предметної області й технологія її аналізу

Кожна інформаційна система в залежності від її призначення має справу з частиною реального світу, що прийнято називати предметною областю (ПО) системи. ПО може відноситься до будь-якого типу організацій: банк, військовий університет, завод, склад, магазин і т. ін.

*Предметна область інформаційної системи* – це сукупність реальних об'єктів (сутностей), що становлять інтерес для користувачів.

*Об'єкт (сутність) – предмет, процес або явище, про яке збирається інформація, необхідна для рішення задачі*. Об'єктом може бути людина, предмет, подія.

Кожен об'єкт характеризується низкою основних властивостей – атрибутів.

*Атрибутом* називається *поіменована характеристика об'єкта*. Атрибут показує, яка інформація повинна бути зібрана про об'єкт. Наприклад, об'єкт – клієнт банку.

Атрибути – номер рахунка, адреса, сума внеску.

Першим етапом проектування БД будь-якого типу є *аналіз предметної області*, що закінчується побудовою інформаційної структури (концептуальної схеми). На даному етапі аналізуються запити

користувачів, вибираються інформаційні об'єкти і їх характеристики, що визначають зміст проектованої БД. На основі проведеного аналізу структурується предметна область. Аналіз предметної області не залежить від програмного і технічного середовищ, у яких буде реалізовуватися БД.

Аналіз предметної області доцільно розбити *на три фази*:

1) аналіз концептуальних вимог і інформаційних потреб;

2) виявлення інформаційних об'єктів і зв'язків між ними;

3) побудова концептуальної моделі предметної області і проектування концептуальної схеми БД.

Вимоги користувачів до розроблювального БД являють собою список запитів із указівкою їх інтенсивності й обсягів даних. Ці зведення розроблювачі БД одержують у діалозі з її майбутніми користувачами.

Тут же з'ясовуються вимоги до введення, відновлення і корегування інформації. Вимоги користувачів уточнюються і доповнюються при аналізі наявних і перспективних задач.

Розглянемо зразковий склад запитальника при аналізі різних предметних областей.

***Приклад 1***. Пропонується розробити БД для обліку слухачів вузу.

Аналіз предметної області:

1. Скільки слухачів учиться у вузі?
2. Скільки факультетів і відділень у вузі?
3. Як розподілені слухачі по факультетах відділень і курсах?
4. Скільки дисциплін читається на кожнім курсі по кожній спеціальності?
5. Як часто обновлюється інформація в БД?
6. Скільки викладачів у вузі?
7. Скільки іногородніх слухачів живе в гуртожитку, на приватних квартирах?
8. Скільки лекційних аудиторій і аудиторій для проведення практичних занять, лабораторій?
9. Яка наступність існує між курсами, що читаються?
10. Як інформація, представлена в п.п. 1 – 9, використовується в даний час (розклад занять, іспитів, заліків і т.інш.) і як її збираються використовувати?
11. Скільки разів у день, скільки чоловік і хто користаються БД?

***Приклад 2***. Розробити вимоги до локального БД "Аеропорт".

Питання 1. Для яких типів задач (додатків) проектується БД?

Відповідь. Для трьох типів задач:

Задача 1. Інформація про обслуговуючий персонал.

Задача 2. Інформація про польотні засоби.

Задача 3. Інформація про графік руху літаків.

Питання 2. Якими інформаційними об'єктами характеризуються ці задачі?

Відповідь. Задача 1 характеризується трьома інформаційними об'єктами: льотний склад, диспетчери, технічний персонал.

Задача 2 характеризується двома інформаційними об'єктами: літак, злітне поле.

Задача 3 характеризується одним інформаційним об'єктом – рейси.

Питання 3. Яким поточним запитам повинні задовольняти дані інформаційні об'єкти?

Відповідь.

1. ПІБ, звання, посада членів екіпажа літака.
2. Обліковий склад диспетчерів.
3. Склад зміни технічного персоналу.
4. Тип літака, що може обслуговувати той чи інший пілот.
5. Номер літака, що обслуговує даний пілот, дана зміна диспетчерів і технічного персоналу.
6. Номер особової справи співробітника аеропорту.
7. Номер зміни диспетчерів і технічного персоналу, що обслуговує аеропорт у заданому інтервалі часу.
8. Готовність літака №... до польоту.
9. Кількість годин нальоту літака №...
10. Готовність даної злітної смуги в даний час.
11. Довжина даної смуги.
12. Номер (номера) рейса до даного пункту призначення.
13. Які проміжні посадки робить рейс №... ?
14. Час вильоту і розрахунковий час прибуття рейса №... .
15. Час і місце реєстрації рейса №... .
16. Час посадки на рейс №... .
17. До якого часу затримується рейс №... ?
18. Які типи літаків обслуговують рейс №... ?
19. Який номер літака обслуговує рейс №... ?

Питання 4. Яким перспективним запитам повинні задовольняти інформаційні об'єкти в БД "Аеропорт"?

1. З якого року використовується літак з №... в аеропорті, тип літака?

2. Яка кількість годин польоту в члена екіпажа, ПІБ?

3. Розрахунковий час відпустки члена екіпажа, диспетчера, технічного працівника.

***Приклад 3***. Розробити БД "Відеомагнітофони".

Питання 1. На кого розрахована ця БД?

Відповідь. На покупця відеосистем.

Питання 2. Що цікавить покупця?

Відповідь. Покупця цікавлять технічні характеристики системи, її ціна, фірма-виготовлювач, технічні характеристики і ціна відеокасет, фірма-виготовлювач касет.

Далі проектувальник вибирає по технічній документації параметри відеосистем, розробляє перелік запитів і уточнює його з майбутнім користувачем БД. Однак БД поки немає.

Є тільки пропозиції проектувальника і схвалення майбутнього користувача. Користувачу здається, що всі проблеми позаду, а проектувальника ще чекає дуже велика робота.

Виявлені запити представлені наступним переліком:

1. Видати дані на визначену модель системи.
2. Яка ціна тієї чи іншої системи?
3. Видати системи визначеної країни-виготовлювача, ціни яких не перевищують задану.
4. Видати останні моделі визначеної фірми.
5. Видати моделі визначеної фірми, габаритні розміри яких не перевищують задані.
6. Видати № моделей визначеної країни-виготовлювача, ціни яких не перевищують задані.
7. Яка ціна останніх моделей визначеної фірми?
8. Видати № моделі і фірму-виготовлювача найдешевшого відеоплейєру, випущеного у визначеному році.
9. Видати характеристики відеомагнітофонів, випущених у визначеному році.
10. Видати найменування моделі і фірми-виготовлювача відеокасет з найбільшим часом програванняll. Видати дані на найдешевшу відеокасету.
12. Видати дані на найдорожчу відеокасету.
13. Скільки коштує відеокасета визначеного найменування і визначеної фірми?
14. Видати дані на відеокасету, що має визначений час програвання

Друга фаза аналізу предметної області полягає у виборі інформаційних об'єктів, завданні необхідних властивостей для кожного об'єкта, виявленні зв'язків між об'єктами, визначенні обмежень, що накладаються на інформаційні об'єкти, типи зв'язків між ними, характеристики інформаційних об'єктів.

Проаналізуємо предметну область на прикладі БД "Відеомагнітофони".

При виборі інформаційних об'єктів постараємося відповісти на ряд питань:

На які класи можна розбити дані, що підлягають збереженню в БД?

Яке ім'я можна привласнити кожному класу даних?

Які найбільш цікаві характеристики (з погляду користувача) кожного класу даних можна виділити?

Які імена можна привласнити обраним наборам характеристик?

***Приклад***. Продовжимо створення БД "Відеомагнітофони", розрахованої на користувачів, що хочуть придбати даний вид техніки.

Після бесіди з різними користувачами і перегляду каталогів було з'ясовано, що інтерес представляють три інформаційних об'єкти: *відеомагнітофон, відеоплейєр, відеокасета*. Розглянемо найбільш істотні характеристики кожного інформаційного об'єкта.

Об'єкт – ВІДЕОМАГНІТОФОН.

Атрибути – країна-виготовлювач, фірма-виготовлювач, № моделі, телевізійні системи, число касетних гнізд, ресурс безупинної роботи, система автопошуку, напруга в мережі, наявність таймера, число програм, габаритні розміри, маса, ціна в доларах, рік випуску.

Об'єкт – ВІДЕОПЛЕЙЄР.

Атрибути – країна-виготовлювач, фірма-виготовлювач, №° моделі, телевізійні системи, число відтворюючих голівок, ресурс безупинної роботи, напруга в мережі, наявність таймера, габаритні розміри, маса, ціна в доларах, рік випуску.

Об'єкт – ВІДЕОКАСЕТА.

Атрибути – найменування, країна-виготовлювач, фірма-виготовлювач, тип касети, час програвання, ціна в доларах.

Далі виділимо зв'язки між інформаційними об'єктами. У ході цього процесу постараємося відповісти на наступні питання:

Які типи зв'язків між інформаційними об'єктами?

Яке ім'я можна привласнити кожному типу зв'язків?

Які можливі типи зв'язків, що можуть бути використані згодом?

Чи мають зміст які-небудь комбінації типів зв'язків?

Спробуємо задати обмеження на об'єкти і їх характеристики.

Під обмеженням цілісності звичайно розуміють логічні обмеження, що накладаються на дані. Обмеження цілісності – це така властивість, що ми задаємо для деякого інформаційного об'єкта або його характеристики і яке повинно зберігатися для кожного їх стану.

Введемо наступні обмеження:

1. Значення атрибута "число касетних гнізд" змінюється від 1 до 2.
2. Значение атрибута "ресурс безупинної роботи" змінюється від 4 до 24.
3. Значення атрибута "напруга в мережі" змінюється від 110 до 240 В.
4. Значення атрибута "число програм" змінюється від 1 до 20 і т.ін.

**Побудова концептуальної моделі предметної області.**

Заключна фаза аналізу предметної області складається в проектуванні її інформаційної структури або концептуальної моделі.

Концептуальна модель включає описи об'єктів і їх взаємозв'язків, що представляють інтерес у розглянутій предметній області (ПО) і даних, що виявляються в результаті аналізу.

Концептуальна модель застосовується для структурування предметної області з урахуванням інформаційних інтересів користувачів системи. Вона дає можливість систематизувати інформаційний зміст предметної області, дозволяє як би "піднятися вгору" над ПО і побачити її окремі елементи. При цьому, рівень деталізації залежить від обраної моделі.

Концептуальна модель є представленням точки зору користувача на предметну область і не залежить ні від програмного забезпечення СУБД, ні від технічних рішень.

Концептуальна модель повинна бути стабільною. Можуть мінятися прикладні програми, що обробляють дані, може мінятися організація їх фізичного збереження, концептуальна модель залишається незмінною чи збільшується з метою включення додаткових даних.

Однією з розповсюджених моделей концептуальної схеми є модель "сутність – зв'язок" (ER-модель). Основними конструкціями даної моделі є сутності і зв'язки.

Під сутністю розуміють основний зміст об'єкта ПО, про який збирають інформацію. Як сутність можуть виступати місце, річ, особистість, явище.

Екземпляр сутності – конкретний об'єкт.

Наприклад:

сутність (об'єкт) – службовець

екземпляр сутності – Іванов А.В.;

сутність (об'єкт) – вищий навчальний заклад

екземпляр сутності – ХВУ.

Сутність прийнято визначати атрибутами – пойменованими характеристиками. Наприклад:

*сутність – службовець*

*атрибути: ПІБ, рік народження, адреса, організація і т.ін*.

Щоб задати атрибут у моделі, йому треба привласнити ім'я і визначити область припустимих значень. Одне з призначень атрибута – ідентифікувати сутність. Зв'язок визначає відношення між сутностями.

При побудові моделі "сутність – зв'язок" використовують графічні діаграми. При цьому позначають:

сутності – прямокутниками,

атрибути – овалами,

зв'язки – ромбами.

Приклад такої моделі наведений на рис. 4.28.




Рисунок 4.28. Приклад моделі "сутність – зв'язок".

На практиці доводиться будувати кілька варіантів моделей, з яких вибирається одна, що найбільше повно відображає предметну область.

***Приклад***. Спроектувати БД "Сесія". База даних повинна видавати оперативну інформацію про успішність студентів на факультетах у семестрі. Результатами сесії вважати тільки іспити.

По суті справи в БД, виходячи з формулювання завдання, можна виділити лише один додаток. Мова йде про успішність студентів різних факультетів по тим чи іншим дисциплінам. Більш конкретно мова йде про видачу довідок за результатами сесії кожного студента, навчальної групи, курсу, факультету, а також про автоматизоване складання відомості.

Виберемо наступні сутності:

ІНСТИТУТ, ФАКУЛЬТЕТ, СТУДЕНТ, ВИКЛАДАЧ, ДИСЦИПЛІНА.

У даному прикладі також можна виділити сутність ІСПИТ або ВІДОМІСТЬ, але можна не виділяти, а сформувати відомість з наявних даних по засобах зв'язків.

Задамо кожну сутність набором атрибутів:

ІНСТИТУТ (назва, підпорядкованість, адреса, телефон, ФИО ректора).

ФАКУЛЬТЕТ (назва, код спеціальності, дані про кафедри, число випускників, декан).

СТУДЕНТ (ПІБ, група, курс, номер поточного семестру, стать).

ВИКЛАДАЧ (ПІБ, посада, звання, кафедра, стаж).

ДИСЦИПЛІНА (назва, число годин, код дисципліни, види занять, число семестрів, що читаються, номера поточних семестрів, на яких курсах викладається)

У кожному наборі атрибутів, що характеризують сутність, необхідно вибрати ключові атрибути, тобто атрибути, що роблять сутність унікальною.

При завданні атрибутів ключові атрибути підкреслювалися.

Визначимо зв'язки між сутностями

| Назва зв'язку | Зв'язок між сутностями |
|---|---|
| вчиться | студент, факультет |
| вивчає | студент, дисципліна |
| має | інститут, факультет |
| працює | викладач, факультет |
| викладає | викладач, дисципліна |
| іспит | студент, дисципліна, викладач |

Після вибору сутностей, завдання атрибутів і аналізу зв'язків можна перейти до проектування інформаційної (концептуальної) схеми БД (рис. 4.29).

Атрибути сутностей на рисунку не показані.

Розглянемо деякі обмеження в розглянутому прикладі:

1. Значення атрибута "телефон" (сутність – ІНСТИТУТ) задається цілим позитивним шестизначним числом.
2. Значення атрибута "код факультету" (сутність – ФАКУЛЬТЕТ) лежить в інтервалі 1 – 10.
3. Значення атрибута "курс" (сутність – СТУДЕНТ) лежить в інтервалі 1 – 6.
4. Значення атрибута "семестр" (сутність – СТУДЕНТ, ДИСЦИПЛІНА) лежить в інтервалі 1 – 12.
5. Значення атрибута "число годин" (сутність – ДИСЦИПЛІНА) лежить в інтервалі 1 – 300.
6. Одному студенту може бути приписана тільки одна група.
7. Один студент може учитися тільки на одному факультеті.
8. Один студент у семестрі здає від 3 до 5 дисциплін.

9. Один студент вивчає в семестрі від 6 до 12 дисциплін.
10. Одному викладачу приписується тільки одна кафедра.
11. Один студент може перездавати одну дисципліну не більш трьох разів.
12. Ключі: назва інституту, назва факультету, ПІБ і група студента.

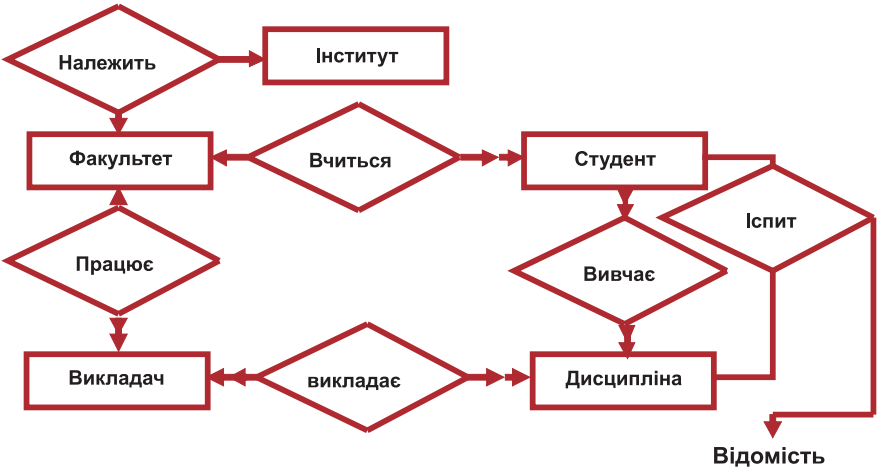



Рисунок 4.29. Концептуальна схема БД “Успішність”.

4.1.3.4 Логічне проектування

Логічне проектування являє собою необхідний етап при створенні БД. Основною задачею логічного проектування є розробка логічної схеми, орієнтованої на обрану систему управління базами даних (СУБД). Етап логічного проектування на відміну від концептуального проектування цілком орієнтований на інструментальні засоби комп’ютера.

Процес логічного проектування складається з наступних етапів:

*1. Вибір конкретної СУБД.*

*2. Відображення концептуальної схеми на логічну схему.*

*3. Вибір ключів.*

*4. Опис мови запитів.*

Одним з основних критеріїв вибору СУБД є оцінка того, наскільки ефективно внутрішня модель даних, підтримувана системою, здатна описати концептуальну схему. Існуючі СУБД поділяються по типах моделей даних на реляційні, ієрархічні і мережні. СУБД, орієнтовані на персональні комп’ютери, як правило, підтримують реляційну модель даних. Переважна більшість сучасних СУБД – реляційні. Якщо обрана реляційна система, то концептуальну схему БД треба буде відобразити на реляційну.

При відображенні концептуальної схеми на реляційну модель даних кожен прямокутник схеми відображається в таблицю. При цьому варто враховувати обмеження на розміри таблиць, що накладає обрана СУБД.

Відобразимо концептуальну схему, зображену на останньому рис., на реляційну модель. Кожен прямокутник (сутність) цієї схеми буде представлений у вигляді таблиці. Кожен стовпець таблиці призначений для запису одного атрибута і має своє унікальне ім'я.

Представимо сутність ВИКЛАДАЧ (ПІБ, посада, звання, кафедра, стаж) у вигляді таблиці.

**ВИКЛАДАЧ**

| ПІБ | Посада | Звання | Кафедра | Стаж |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |

Визначимо структуру кожної таблиці, тобто задамо типи і розміри полів.

| Ознака ключа | Поле | Тип поля | Розмір поля |
|---|---|---|---|
| ключ | ПІБ | символьний | 21 |
| | Посада | символьний | 15 |
| | Звання | символьний | 10 |
| ключ | Кафедра | символьний | 20 |
| | Стаж | числовий | 2 |

Можна зробити оцінку необхідного обсягу пам'яті для збереження даної таблиці.

Для збереження одного запису необхідно 21+15+10+20+2=68 байт. Усього передбачено 100 записів, тобто для збереження таблиці буде потрібно 6800 байт.

Аналогічно поводяться із всіма іншими сутностями (об'єктами) концептуальної схеми.

Кожна конкретна СУБД має свої особливості, які необхідно враховувати. Знаючи функціональні можливості будь-якої СУБД, можна представити узагальнену технологію роботи користувача в цьому середовищі.

Як основні етапи узагальненої технології роботи із СУБД, можна виділити наступні (рис. 4.30):

- ✧ створення структури таблиць бази даних;
- ✧ уведення й редагування даних у таблицях;
- ✧ обробка даних, що містяться в таблицях;
- ✧ виведення інформації з бази даних.

Розглянемо виділені етапи більш докладно.

*Створення структури таблиць бази даних*. При формуванні нової таблиці бази даних робота із СУБД починається зі створення структури таблиці. Цей процес включає визначення переліку полів, з яких складається кожен запис таблиці, а також типів і розмірів полів.

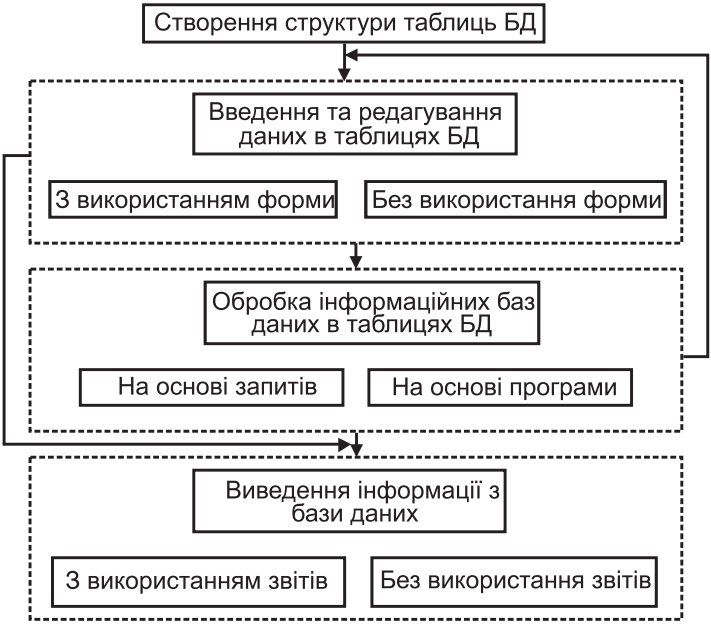



Рисунок 4.30. Схема узагальненої технології роботи в СУБД.

Практично всі сучасні СУБД зберігають дані наступних типів: текстовий (символьний), числовий, календарний, логічний. Деякі СУБД формують поля спеціального типу, що містять унікальні номери записів, які використовуються для визначення ключа.

Якщо оброблювана база даних включає кілька взаємозалежних таблиць, то необхідно визначення ключового поля в кожній таблиці, а також полів, за допомогою яких буде організований зв'язок між таблицями.

Створення структури таблиці не пов'язане із заповненням таблиць даними, тому ці дві операції можна рознести в часі.

*Введення й редагування даних*. Заповнювати таблиці даними можна як безпосереднім введенням даних, так й в результаті виконання програм і запитів.

Практично всі СУБД дозволяють вводити й корегувати дані в таблицях двома способами:

– за допомогою надаваної за замовчуванням стандартної форми у вигляді таблиці;
– за допомогою екранних форм, спеціально створених для цього користувачем.

Можлива побудова форм, найбільш зручних для роботи користувача, що включають записи різних зв'язаних таблиць бази даних.

*Обробка даних, що містяться в таблицях*. Обробляти інформацію, що міститься в таблицях бази даних, можна шляхом використання запитів або в процесі виконання спеціально розробленої програми.

Кінцевий користувач одержує при роботі із СУБД такий зручний засіб обробки інформації, як запити. Запит – це інструкція на відбір записів.

Більшість СУБД дозволяють використати запити наступних типів: запит-вибірка, запити на виконання дії, запит з параметрами.

Найпоширенішим типом запиту є запит на вибірку, призначений для відбору даних, що зберігаються в таблицях, і не змінює ці дані. Результатом виконання запиту є таблиця з тимчасовим набором даних (динамічний набір).

Вибірка – динамічна таблиця із записами даних, які задовольняють певним умовам запиту.

Стандартні запити вибору – запити, використовувані для того, щоб відібрати й представити у вигляді зручної динамічної таблиці дані, що цікавлять користувача, з таблиць БД.

Перехресний запит – запит, що дозволяє представити дані в досить наочній, компактній формі зведених кросів-таблиць, здійснюючи при цьому різноманітне угруповання записів і групові обчислення при обробці даних.

Запит-зміна призначена для зміни або переміщення даних; до цього типу запитів ставляться: запит на додавання записів, запит на видалення записів, запит на створення таблиці, запит на відновлення.

Запити на виконання дії – запити, які використаються для створення нових реальних таблиць даних, що надалі існують уже незалежно від тих таблиць бази даних, які були використані для їхньої побудови.

Запит з параметром дозволяє визначити одне або кілька умов відбору під час виконання запиту.

Запит з параметрами – запит, при якому СУБД запитує в користувача значення параметрів для визначення умов вибірки.

*Виведення інформації з бази даних*. Практично будь-яка СУБД дозволяє вивести на екран і принтер інформацію, що втримується в базі даних, з режимів таблиці або форми. Такий порядок виведення даних може використовуватись тільки як чорновий варіант, тому що дозволяє виводити дані тільки в такому виді, в якому вони містяться в таблиці або формі.

Кожен користувач, що працює із СУБД, має можливість використання спеціальних засобів побудови звітів для виведення даних. Використовую-

чи спеціальні засоби створення звітів, користувач одержує такі додаткові можливості:

- ✧ включати у звіт вибіркову інформацію з таблиць бази даних;
- ✧ додавати інформацію, що не міститься в базі даних;
- ✧ при необхідності виводити підсумкові дані на основі інформації бази даних;
- ✧ розташовувати виведену у звіті інформацію в будь-якому, зручному для користувача виді (вертикальне або горизонтальне розташування полів);
- ✧ включати у звіт інформацію з різних зв'язаних таблиць бази даних.

### 4.1.4 Поняття сховищ даних та передумови їх створення

#### 4.1.4.1 Поняття, передумови створення та основні характеристики сховищ даних

***Сховище даних*** *(Data Warehouse)* – це предметне орієнтована, інтегрована, прив'язана до часу та незмінна сукупність даних, призначена для підтримки прийняття рішень.

Необхідність розробки нової концепції сховищ даних обумовлена такими факторами:

– Системи підтримки прийняття рішень, що ґрунтуються на формуванні аналітичних запитів, почали конфліктувати з трансакційними системами оперативної обробки даних (OLTP-системами). Одночасне вирішення оперативних та аналітичних запитів на одній базі даних часто призводить до нестачі ресурсів.

– Реалізація аналітичних звітів на основі традиційних баз даних, які містять оперативну інформацію, займає дуже багато часу. Це пов'язано з тим, що для аналітичних звітів переважно потрібні не первинні оперативні дані, а певним чином узагальнені, тобто агреговані дані. Причому витрати часу, необхідні на формування аналітичних звітів, невпинно зростають по мірі зростання обсягів оперативної інформації в базі даних. Це призводить до затримок при реалізації аналітичних запитів.

– Дуже часто на підприємстві чи в організації функціонує кілька OLTP-систем, кожна з яких має свою окрему базу даних. У них використовуються різні структури даних, способи кодування, одиниці вимірювання. Побудова зведеного аналітичного запиту на основі кількох баз даних є дуже складною проблемою, яка спочатку потребує вирішення проблеми узгодженості даних, що зберігаються в різних базах даних.

– Для вирішення оперативних аналітичних задач недостатньо інформації, що зберігається в базі даних. Необхідні архівні дані, що містять результати роботи за попередні календарні періоди. Крім того, дуже часто

виникає потреба в зовнішніх джерелах (дані про клієнтів, конкурентів, політичні, соціологічні, демографічні та ін.).

Сховища даних характеризуються предметною орієнтацією, інтегрованістю, підтримкою хронології, незмінністю і мінімальною надлишковістю. Ці основні особливості сховищ даних були визначені в 1992р. їх винахідником Біллом Інмоном. Вони незалежно від реалізації властиві всім сховищам даних і полягають у такому:

*Предметна орієнтація.* Дані в сховищі даних організовані відповідно до основних напрямів діяльності підприємства чи фірми (замовники, продажі, склад і т. п.). Це – відмінність сховищ даних від організації оперативної БД, в якій дані організуються відповідно до процесів (відвантаження товару, виписка рахунків і т. п.). Предметна організація даних не лише спрощує проведення аналізу, але й значно прискорює виконання аналітичних розрахунків. Тобто сховища орієнтовані на бізнес-поняття, а не на бізнес-процеси. Приклад побудови сховища наведений на рис.4.31.

*Інтегрованість.* Дані у сховище надходять з різних джерел, де вони можуть мати різні імена, формати, одиниці вимірювання і способи кодування. Перш ніж завантажити дані до сховища, вони перевіряються, певним чином відбираються, приводяться до одного єдиного способу кодування, виду та формату і в необхідній мірі агрегуються (тобто обраховуються сумарні показники). З цього моменту вони представляються користувачеві у вигляді єдиного інформаційного простору, які набагато простіше аналізувати.

Якщо, наприклад, у чотирьох різних базах даних код товару кодувався чотирма різними способами, то в сховищі даних буде використана єдина система кодування.

*Підтримка хронології.* Дані в сховищі даних зберігаються у вигляді "історичних пластів", кожен з яких характеризує певний календарний період. Це дозволяє проводити аналіз зміни показників у часі. В OLTP-системах істинність даних гарантована тільки в момент читання, оскільки вже в наступну мить вони можуть змінитися внаслідок чергової трансакції. Важливою відмінністю сховищ від OLTP-систем є те, що дані в них зберігають свою істинність у будь-який момент процесу читання.

В OLTP-системах інформація часто модифікується як результат виконання яких-небудь трансакцій. Часова інваріантність даних у сховищі даних досягається за рахунок введення полів, що характеризують час (день, тиждень, місяць) у ключі таблиць. У СД містяться начебто моментальні знімки даних. Кожний елемент у своєму ключі явно або непрямо зберігає часовий параметр, наприклад день, місяць або рік.

*Незмінність.* Дані сховища даних, що характеризують кожен "історичний пласт", не можуть змінюватись. Це теж є суттєвою відмінністю даних, що зберігаються у сховищі даних, від оперативних даних. Останні дані в базі даних постійно змінюються. З даними сховища даних можливі лише операції їх первинного завантаження, пошуку та читання.

Якщо при створенні OLTP-систем розробники повинні враховувати такі моменти, як відкоти трансакцій після збою сервера, боротьба із взаємним блокуванням процесів *(deadlocks),* збереження цілісності даних, то для сховища даних ці проблеми не так актуальні. Перед розробниками стоять інші задачі, пов'язані, наприклад, із забезпеченням високої швидкості доступу до даних.

*Мінімальна надлишковість.* Незважаючи на те, що інформація . до сховищ даних завантажується з БД OLTP-систем, це не призводить до надлишковості даних. Мінімум надлишковості даних забезпечується тим, що перш ніж завантажувати дані до сховищ, вони фільтруються і певним чином очищаються від тих даних, які не потрібні і не можуть бути використаними в бізнес-аналізі.




Рисунок 4.31. Приклад сховища даних.

### 4.1.4.2 Основні компоненти сховища даних, архітектура сховищ даних

Враховуючи те, що сховища даних є новою технологію, яка розвивається, існують різні підходи до викладення структури. Згідно з схеми основних компонентів сховища даних включає до свого складу такі компоненти.

Менеджер завантаження

**Менеджер завантаження** виконує функції диспетчеризації щодо регулярності завантаження нових даних до сховища даних згідно зі встановленим регламентом. Оскільки джерелом даних можуть бути різні OLTP-системи, функції менеджера завантаження також полягають в очищенні, конвертації та приведенні даних до заданого виду їх представлення в СД.

**Менеджер сховища** виконує операції аналізу та управління даними. Це такі основні операції: аналіз узгодженості та несуперечності даних; перетворення та переміщення даних з тимчасового сховища в основні таблиці СД; створення індексів; денормалізація даних у разі її необхідності; агрегація (узагальнення) даних; резервне копіювання та архівування даних.

Одним з основних компонентів сховища даних є дані. Дані в сховищі даних представлені в трьох видах:

- ✧ детальні дані;
- ✧ агреговані дані;
- ✧ метадані.

**Детальні (оперативні) дані** – ця складова містить усі детальні дані, які визначені схемою сховища даних. Це можуть бути як первинні дані найнижчого рівня деталізації, так і узагальнені до певного рівня деталізації.

**Агреговані дані** – ця компонента містить дані, які попередньо оброблені менеджером сховища з метою їх часткового чи глибокого узагальнення. У цій частині зберігаються певним чином відсортовані та згруповані дані, необхідні для виконання запитів.

Ця частина сховища є тимчасовою і змінною, оскільки вона постійно модифікується у відповідь на зміни запитів. Необхідність цієї компоненти пов'язана з підвищенням продуктивності виконання запитів. Узагальнені дані поновлюються по мірі надходження нових даних до системи. Частково узагальнені дані – це результат певного узагальнення та агрегації деталь-них даних. Глибоко узагальнені дані отримуються на основі узагальнення частково узагальнених даних.

**Репозитарій метаданих** – це інформація про дані, що зберігаються в СД.

**Метадані – це дані про дані, які визначають джерело, приймач та алгоритм трансформації даних під час перенесення їх від джерела до приймача.**

Метадані містять:

- ✧ описи структур даних та їхніх взаємозв'язків;
- ✧ інформацію про джерела даних і про ступінь їх вірогідності;
- ✧ інформацію про власників даних, права доступу;
- ✧ схему перетворення стовпців вхідних таблиць у стовпці кінцевих таблиць;
- ✧ правила підсумовування, консолідації та агрегування даних;
- ✧ інформацію про періодичність оновлення даних;
- ✧ каталог використаних таблиць, стовпців та ключів;
- ✧ фізичні атрибути стовпців;
- ✧ кількість табличних рядків та обсяг даних;
- ✧ часові ярлики (дата та час створення/модифікації записів);
- ✧ статистичні оцінки часу виконання запитів.

Структура метаданих може відрізнятися залежно від їх призначення. Метадані використовуються для таких основних цілей:

*Вибірка і завантаження даних.* Метадані містять інформацію про джерела даних, способи та періодичність їх вибірки і завантаження в СД.

*Обслуговування сховища.* Метадані використовуються для автоматизації процедур узагальнення даних.

*Обслуговування запитів.* Метадані використовуються для визначення переліку таблиць для виконання запитів.

**Менеджер запитів** – це складова сховища, яка виконує операції, пов'язані з управлінням запитів користувачів. Ця компонента реалізується, як правило, на базі СУБД, що підтримує сховище даних, а також сховища даних і програм власної розробки.

Користувачі спілкуються і працюють зі сховищем за допомогою спеціальних засобів. До них можуть бути віднесені OLAP інструменти, засоби, що підтримують технологію Data Mining, та різні засоби доступу кінцевого користувача: створення звітів і запитів; Інструмент що належать до систем підтримки прийняття рішень *(Executive Information System, EIS).*

При визначенні програмно-технологічної архітектури сховища потрібно мати на увазі, що система прийняття рішень, на які б візуальні засоби вона не спиралася, повинна надавати користувачеві можливість деталізації інформації. Керівник підприємства або фірми, отримавши інтегроване представлення даних і висновки, зроблені на їх основі, може зажадати детальніші дані, що уточнюють джерело даних або причини висновків. З погляду проектувальника сховищ даних це означає, що необхідно забезпечити його взаємодію в деяких випадках з БД OLTP-систем.

### 4.1.4.3 Архітектура сховищ даних

Розрізняють такі види сховищ даних: корпоративні і кіоски, або вітрини, даних.

*Корпоративні сховища даних (enterprise data warehouses)* вміщують інтегровану інформацію, зібрану з певної множини оперативних БД, яка характеризує всю корпорацію і необхідна для виконання консолідованого аналізу діяльності корпорації в цілому. Такі сховища охоплюють усі багаточисленні напрями діяльності корпорації і використовуються для прийняття як тактичних, так і стратегічних рішень. Розробка корпоративного сховища даних – дуже трудомісткий процес, який може тривати від одного до кількох років, а обсяги сховища можуть досягати від 50 Гбайт до кількох терабайт.

*Кіоски, або вітрини, даних (data marts)* – це певна підмножина корпоративних даних, які характеризують конкретний аспект діяльності корпорації, наприклад роботу конкретного підрозділу. Кіоск може

вміщувати як агреговані, так і первинні дані певної предметної області. Кіоск може отримувати дані з корпоративного сховища даних (залежний кіоск) чи бути незалежним і тоді джерелом поповнення його даними будуть оперативні БД. Розробка кіоску даних потребує значно менше часу і в середньому займає приблизно 3–4 місяці.

Корпоративні сховища даних і кіоски будуються за подібними принципами і використовують практично одинакові технології.

Останнім часом з'явилось поняття *глобального сховища даних,* в якому сховище даних розглядається як єдине джерело інтегрованих даних для всіх вітрин даних.

Успіх розробки та впровадження сховища даних залежить від обґрунтованості вибору його архітектури. Щодо концептуальної архітектури сховищ даних, то згідно з сховища даних залежно від підходів до побудови їх архітектури поділяються на:

- ✧ віртуальне СД;
- ✧ СД на основі семантичної інтеграції предметних областей;
- ✧ СД із системою управління запитами до предметних областей;
- ✧ Монолітне сховище;
- ✧ СД на основі стандартного архіву даних.
- ✧ Зупинимося детальніше на характеристиці кожної архітектури.

*Віртуальне сховище даних.* Основою віртуального сховища даних є репозитарій метаданих, який описує місце розташування даних в оперативних системах, структуру даних, методи агрегації та завантаження даних, відомості про структуру бізнес-понять та інші дані. Архітектурно-віртуальне сховище даних складається з оперативних систем та системи управління запитами, що зберігає свій репозиторій метаданих

Коли користувач запускає на виконання запит, то засоби управління запитами, використовуючи метадані, виконують пошук та вибірку необхідних даних з оперативних систем. Потім ці дані об'єднуються необхідним чином і видаються користувачеві. Тобто при виконанні запитів щоразу дані об'єднуються з різних оперативних систем, тому системи, побудовані за цими принципами, характеризуються низькою продуктивність виконання запитів. Інший недолік такої архітектури – можливість отримання різних результатів на одні й ті ж запити, виконані у різний час, оскільки дані оперативних систем постійно змінюються і не зберігають історичних даних. Ще одна суттєва перепона, яка стоїть на шляху побудови такої архітектури, – це відсутність єдиної структури полів, кодів і ключів. Тому для організацій з розгалуженою організаційною структурою та наявністю великої кількості інформаційних систем, кожна з яких працює зі своєю базою даних, такий підхід до архітектури сховища даних не може бути прийнятним, оскільки кожного разу для виконання запитів потрібно буде об'єднувати дані з різних систем замість того, щоб це зробити один раз при створенні сховища даних.

Переваги цієї архітектури полягають у тому, що максимально використовується існуюче апаратне забезпечення і основне навантаження по виконанню запитів беруть на себе системи-джерела, а система управління запитами лише управляє результатом та надає його користувачеві. Така архітектура побудови сховища даних може бути рекомендована для організацій, інформаційні системи-джерела якої побудовані з урахуванням системного підходу. Тобто вхідні дані для виконання запитів характеризуються високою якістю, наявністю спільних ключів і систем класифікації та кодування. Крім того, інформаційні системи-джерела мають засоби обмежень ресурсів, які виділяються для виконання запитів, що надходять з віртуального сховища, щоб уникнути ефекту "затоплення" запитами сховища.

*Архітектура сховища на основі семантичної інтеграції предметних областей.* Сховище, побудоване за цією архітектурою, поділено на певні розділи, кожен з яких характеризує окрему предметну область, але проектуються вони за єдиними правилами, що гарантує легкість їх об'єднання. Тобто спочатку проектується єдина корпоративна модель сховища, яка забезпечує ідентичність структур полів, кодів і первинних ключів.

При використанні цієї архітектури процес семантичної інтеграції і проміжне сховище заміняються стандартним архівом. Стандартний архів – це стаціонарне інтегроване сховище, що вміщує інформацію для всіх СППР, і яке можна використовувати для заповнення сховищ окремих предметних областей і вітрин даних. Недоліком цього типу архітектури є високі витрати на пам'ять та підвищені вимоги до супроводження. Великі витрати пам'яті пов'язані з тим, що дані у стандартному архіві повинні зберігатись в такому вигляді, щоб легко забезпечувати обробку оперативних деталізованих запитів до сховища даних.

Ще одним суттєвим недоліком архітектури цього типу є відсутність інтеграції окремих предметних областей. Тому включення додаткового блоку управління запитами підвищує зручність використання сховищ даних при виконанні інтегрованих запитів, для реалізації яких потрібні дані з кількох предметних областей.

#### 4.1.4.4 Основні елементи сховища даних та підходи до проектування сховищ

Основними елементами даних, які зберігаються в сховищі даних, є:

- ✧ показники (змінні);
- ✧ виміри та їх ієрархія;
- ✧ факти.

Основними задачами проектування сховищ даних є визначення кандидатів на змінні, факти та виміри. Для цього застосовуються кілька підходів, кожен з яких характеризується певною послідовністю ідентифікації основних елементів моделі. Згідно з послідовністю

ідентифікації елементів сховища даних можна визначити такі підходи до визначення основних елементів сховища даних:

**Підхід "від запиту". ** Передусім визначаються змінні, потім виміри, пов'язані зі змінними, а надалі формуються факти. Цей підхід називається "від запиту", оскільки він орієнтований насамперед на аналітичні запити до сховища даних. Тобто визначення елементів сховища даних виконується на основі аналізу запитів користувачів-аналітиків.

**Підхід, орієнтований на бізнес.** Визначаються факти, потім виміри, а на завершення змінні. Цей підхід називається орієнтованим на бізнес, оскільки спочатку аналізується предметна область.

Проектування сховищ даних має свою особливість і специфіку, що робить цей процес відмінним від проектування баз даних трансакційних систем.

Сховища даних за своїм призначенням не схожі на бази даних OLTP-систем, які обслуговують поточну оперативну діяльність підприємств та організацій. Сховища даних використовуються для підтримки прийняття стратегічних рішень, що дозволяє аналітикам виявляти тенденції, проводити порівняння і прогнозувати майбутні результати.

Системи OLTP не можуть безпосередньо виконувати функції OLAP. Зокрема, в системах OLTP зберігається тільки "моментальний знімок" найостаннішої оперативної інформації, а для аналізу OLAP потрібне знання попередньої історії трансакцій за тривалий період. Бази даних OLTP-систем безперервно поновлюються, проте іноді містять пропуски і помилкові дані; для OLAP потрібні статичні, вичерпні дані, з виправленими помилками. І нарешті, системи OLTP обробляють тисячі (часом мільйони) трансакцій у день; системи OLAP виконують тільки одну трансакцію за один цикл завантаження: коли дані в систему завантажуються в пакетному режимі, але можуть одночасно реалізовувати та охоплювати тисячі запитів у день. (Під трансакцією тут розуміємо операцію, внаслідок якої база даних переходить з одного стану в інший.)

Ці відмінності у функціях бази даних і сховища даних відображено в структурних відмінностях. Основними з них при проектуванні сховищ даних порівняно з проектуванням баз даних є такі:

1. У базі даних OLTP-систем дані нормалізуються з метою запобігання виникненню аномалій надмірності та порушення цілісності й узгодженості бази даних при внесенні змін до бази даних. Дані в сховищі даних є статичними, тобто в них практично не вносять зміни, тому для сховищ не таке актуальне питання нормалізації.

2. При проектуванні баз даних не враховуються процедури їх подальшої обробки. При проектуванні сховищ даних потрібно знати і враховувати процедури агрегування даних, причому ступінь узагальнення для різних даних може бути різним.

3. При проектуванні сховищ даних необхідно враховувати "часовий" фактор, тобто ступінь деталізації фактів.

Існує два способи проектування сховищ даних: спадний і висхідний.

*Спадний спосіб* – це такий спосіб, коли спочатку проектується корпоративне сховище, а потім воно стає джерелом інформації для кіосків даних, тобто кіоски є залежними.

*Висхідний спосіб* полягає в тому, що на початку проектуються кіоски даних, які охоплюють окремі напрями діяльності корпорації чи певні його підрозділи.

За умови узгодженого форматування даних можна на основі семантичного об'єднання кіосків створювати розподілені корпоративні сховища даних. Такий поетапний підхід дає можливість у дуже стислі терміни отримати відчутні вигоди, оскільки для формування корпоративного сховища потрібно кілька років, а окремі кіоски можна сформувати за кілька місяців.

Можна виокремити такі три підходи до проектування сховищ даних: метод реконструкції, проектування за шаблоном, проектування під замовлення.

**Метод реконструкцій** – це проектування сховища на основі використання існуючих моделей даних OLTP-систем. Тобто основним джерелом даних для сховища виступають дані, що зберігаються в базах даних трансакційних систем. Іноді цей підхід ще називається "від джерела". Він полягає в трансформації існуючої моделі бази даних у модель сховища даних. Проектування за допомогою реконструкції рекомендується у випадках:

- ✧ коли модель OLTP-системи відносно нова та охоплює всі предметні області, для яких планується розробка сховища даних;
- ✧ коли багато таблиць фактів та вимірів присутні в моделі бази даних.

Проте цей підхід має теж ряд недоліків. Насамперед переважна більшість сутностей бази даних орієнтована на бізнес-процеси, а не на бізнес-поняття, що вимагає суттєвого перепроектування існуючої моделі до потреб сховища даних. Так, база даних може містити такі сутності, що характеризують бізнес-процеси, як ВІДВАНТАЖЕННЯ та ВИПИСКА. Перша сутність характеризує такий бізнес-процес, як відвантаження продукції споживачу, а друга – містить дані банківської виписки про перерахування коштів за відвантажену продукцію. Представлена в такому розрізі оперативна інформація для сховища даних не є актуальною. Для аналізу та прийняття рішень потрібні дані про реалізацію продукції, тобто про ту частину продукції, яка була не лише відвантажена, але й оплачена, тобто за яку надійшли виписки про перерахування коштів на рахунок у банку. Таким чином, у сховищі даних повинні зберігатись не первинні оперативні дані, а результати їх певної обробки, які виконуються при вирішенні задачі "Автоматизація обліку реалізації готової продукції". Тобто використання

методу реконструкції потребує глибоких знань предметної області та специфіки і особливостей функціонуючих OLTP-систем.

Крім того, суттєвим недоліком зазначеного підходу до проектування сховищ даних є те, що він орієнтується на існуючі джерела OLTP-систем і не враховує зовнішніх даних, які можуть виявитись досить суттєвими для бізнес-аналізу, який потрібно виконувати в даній предметній області.

**Проектування за шаблоном** – це такий підхід до розробки сховища даних, коли за основу для проектування береться існуюча, функціонуюча модель сховища даних у подібній предметній області. В цю модель вносяться зміни, що відображають специфіку конкретного об'єкта управління. Кращим варіантом цього підходу є придбання стандартної галузевої моделі побудови сховища даних, якщо така є на підприємствах галузі.

**Проектування під замовлення** – це проектування з "чистого аркуша". Цей підхід ігнорує врахування існуючих галузевих моделей сховищ даних і моделей OLTP-систем. При цьому варіанті проектування розробники цілком зосереджують свою увагу на потребах користувачів у результатах бізнес-аналізу та особливостях предметної області, що досліджується. Тобто при цьому підході акцент робиться на дослідженні бізнес-вимог до сховища з боку користувачів. Тому цей підхід ще можна назвати "від запиту". Проте цей підхід теж має недоліки, пов'язані з тим, що користувач, висуваючи вимоги, керується поточними потребами і не враховує перспективи розвитку. Тобто важко визначитись з повнотою та ефективним складом вимог до сховища з боку користувачів. Крім того, користувач може поставити такі вимоги, які не забезпечені потрібними даними.

Враховуючи недоліки кожного з підходів до проектування сховищ даних, можна запропонувати комбінований підхід, який включає елементи двох підходів до проектування сховищ даних – "від запиту" та "від джерела". Це дасть змогу врахувати ті дані, які накопичені в трансакційних системах, шляхом їх певної реконструкції та поповнити сховище даних недостатніми даними, які потрібні з позицій користувача для проведення бізнес-аналізу.

Тепер про різні **варіанти зберігання інформації**. Як детальні дані, так і агрегати можуть зберігатися або в реляційних, або в багатовимірних структурах. Багатовимірне зберігання дозволяє поводитися з даними як з багатовимірним масивом, завдяки чому забезпечуються однаково швидкі обчислення сумарних показників і різні багатовимірні перетворення по будь-якому з вимірювань.

Якийсь час назад OLAP-продукти підтримували або реляційне, або багатовимірне зберігання. Сьогодні, як правило, один і той же продукт забезпечує обидва ці види зберігання, а також третій вигляд – змішаний. Застосовуються наступні терміни:

MOLAP (Multidimensional OLAP) – і детальні дані, і агрегати зберігаються в багатовимірній БД. В цьому випадку виходить найбільша надмірність, оскільки багатовимірні дані повністю містять реляційні.

ROLAP (Relational OLAP) – детальні дані залишаються там, де вони "жили" спочатку – в реляційній БД; агрегати зберігаються в тій же БД в спеціально створених службових таблицях.

HOLAP (Hybrid OLAP) – детальні дані залишаються на місці (в реляційній БД), а агрегати зберігаються в багатовимірній БД.

Кожний з цих способів має свої переваги і недоліки і повинен застосовуватися залежно від умов – об'єму даних, потужності реляційної СУБД і т. ін. При зберіганні даних в багатовимірних структурах виникає потенційна проблема "розбухання" за рахунок зберігання порожніх значень. Адже якщо в багатовимірному масиві зарезервовано місце під все можливі комбінації міток вимірювань, а реально заповнена лише мала частина (наприклад, ряд продуктів продається тільки в невеликому числі регіонів), то велика частина куба порожнітиме, хоча місце буде зайнято. Сучасні OLAP-продукти уміють справлятися з цією проблемою.

## 4.2 Експертні і навчальні системи

### 4.2.1 Штучний інтелект – вершина розвитку інформаційних технологій

4.2.1.1 Поняття "штучний інтелект", терміні й визначення

Еволюція інформаційних технологій і систем все більшою мірою визначається їхньою інтелектуалізацією. ***Інтелектуальні інформаційні технології*** – одна з найбільш перспективних наукових і прикладних областей інформатики, яка постійно розвивається. Вона впливає на всі наукові й технологічні напрямки, пов'язані з використанням комп'ютерів і вже сьогодні дає суспільству те, що воно чекає від науки, – практично значимі результати.

Цілями інтелектуальних інформаційних технологій є, по-перше, розширення кола завдань, розв'язуваних за допомогою комп'ютерів, особливо в слабоструктурованих предметних областях і по-друге, підвищення рівня інтелектуальної інформаційної підтримки сучасного фахівця.

Ключовім компонентом наукового фундаменту інтелектуальних інформаційних технологій є ***штучний інтелект*** (***ШІ***).

Штучній інтелект, як і будь-який напрямок сучасної науки, має багату передісторію. Знайомство з іменами й працями основоположників, їхніми науковими поглядами, навіть біографіями вчених має практичний сенс, тому що в багатьох випадках дозволяє глибше зрозуміти зміст тої або іншої наукової моделі.

Штучній інтелект – новітня інформаційна технологія, що має специфічне коло завдань і базується на раніше існуючих технологіях.

Сучасній етап розвитку інформатики характеризується необхідністю вирішення управлінських завдань, які значно ускладнилися. Труднощі

вирішення цих завдань пояснюються тим, що об'єктам і процесам у економічної сфері, які досліджуються, окрім всіх основних рис, якими володіють складні системи, притаманні такі особливості:

- ✧ велика кількість слабоформалізованих і часто суперечливих цілей функціонування, їх мінливість у часі;
- ✧ конфліктний і багатоаспектний характер взаємовідносин як між елементами всередині об'єкта (процесу), так і з навколишніми об'єктами (процесами) при сильному впливі людського чинника;
- ✧ здебільшого понятійний характер і суперечливість вихідних описів функціонування і можливих обмежень.

За наявності перерахованих властивостей неможливо побудувати жорстку формальну модель об'єкта, орієнтовану на дані і рішення завдань необхідно шукати у класі інших методів, що розробляються в рамках теорії штучного інтелекту.

**Штучній інтелект (ШІ)** – область інформатики, що займається науковими дослідженнями і розробкою методів і засобів для правдоподібної імітації окремих функцій людського інтелекту за допомогою автоматизованих систем.

**ШІ** – це метафорична назва нових інформаційних технологій обробки інформації, створених фахівцями різноманітних галузей науки і техніки (математиками інженерами, психологами, лінгвістами, військовими вченими та ін.).

Існує декілька визначень інтелектуальної системи:

1. ***Інтелектуальною*** називається система (рис. 4.32), здатна цілеспрямовано, залежно від стану інформаційних входів, змінювати не тільки параметри функціонування, але і сам спосіб своєї поведінки, причому спосіб поведінки залежить не тільки від поточного стану інформаційних входів, але також і від попередніх станів системи.




Рисунок 4.32. Інтелектуальна система як "чорний ящик".

Будь-який живий організм – інтелектуальна система. Він володіє довготривалою пам'яттю і здібністю до самонавчання. Приведемо декілька прикладів: дитина, щеня.

Технічні ж системи найчастіше не є інтелектуальними, тобто їх реакція на одну і ту ж подію не може змінитися кардинально.

2. **Інтелектуальною** називається система, що моделює на комп'ютері мислення людини.

Друге визначення з'явилося в 60-х рр., коли вважалося, що мозок людини можна змоделювати на комп'ютері. Клітини мозку – нейрони – програмно описувалися спеціальними математичними методами. Комп'ютерна програма, таким чином, представляла як би шматочок мозку людини.

Ідея про можливість повторити мозок на комп'ютері до 80-х рр. XX ст. потерпіла повну невдачу, проте теорія нейронних мереж, нейромережевий підхід довели свою корисність в цілому ряді практичних застосувань. Добрі результати отримані насамперед при застосуванні нейронних мереж до завдань прогнозу значень параметрів і розпізнавання образів.

3. **Інтелектуальною** називається система, що дозволяє підсилити інтелектуальну діяльність людини за рахунок ведення з ним осмисленого діалогу.

На кінець 80-х рр. стало очевидно, що створити універсальний штучний розум неможливо. Більш того, з'ясувалося, що це абсолютно не потрібно. Слід створювати вузькоспеціалізовані інтелектуальні системи, які не замінюють людину, але доповнюють її.

Таким чином, під ***інтелектуальними системами*** розуміють будь-які біологічні, штучні або формальні системи, що проявляють здібність до цілеспрямованої поведінки. Останнє включає властивості (прояви) спілкування, накопичення знань, ухвалення рішень, навчання, адаптації і так далі.

В даний час існує стійка тенденція *інтелектуалізації* комп'ютерів і їх програмного забезпечення (ПО). Основними функціями майбутніх комп'ютерів будуть вирішення завдань все більшою мірою необчислювального характеру, зокрема логічний вивід, управління базами знань (БЗ), забезпечення інтелектуальних інтерфейсів тощо. Інтелектуалізація комп'ютерів здійснюється за рахунок розробки як спеціальної апаратури (наприклад, нейрокомп'ютери), так і ПО (експертні системи, бази знань, вирішувачі завдань тощо).

Сьогодні **штучний інтелект** – це самонавчальний інструмент, що підсилює діяльність людини по генерації і ухваленню рішень.

Зупинимось на деяких поняттях, без яких неможливе створення штучного інтелекту.

***Знання*** – основний термін теорії штучного інтелекту. Дуже інтуїтивно і досить спірно знання можна визначити як відношення між елементами даних.

Робота програми штучного інтелекту полягає в "виводі на знаннях" або "виведенні нових знань". Цей процес схематично представлений на рис. 4.33.

***Стан*** – другий найважливіший термін. Кожна інформаційна одиниця, як і вся система в цілому, може знаходитися в одному із станів.

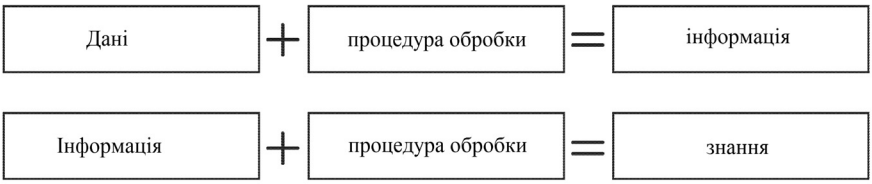



Рисунок 4.33. Вивід на знаннях.

Перехід системи із стану в стан може володіти властивістю симетричності. В цьому випадку система здатна повернутися в початковий стан (а комп'ютерна програма – зациклитися).

***Мета*** – основна характеристика системи ШІ. Система штучного інтелекту (інтелектуальна програма) відрізняється від традиційних програм властивістю цілеспрямованості. Інтелектуальна програма на відміну від звичайної не містить алгоритм, а містить тільки мету, яку потрібно досягти, і правила руху до цієї мети. Можна сказати, що програма ШІ – це програма "Що потрібно зробити", а традиційна – "як потрібно робити". У традиційній програмі алгоритм досягнення мети заданий спочатку і виконується неминуче, в інтелектуальній програмі алгоритм будується по ходу досягнення мети з урахуванням ситуації, що складається.

Як цілі в інтелектуальних програмах найчастіше задаються вимоги переходу системи в заданий стан, а як критерій ефективності – число кроків рішення. Очевидно, досягнення мети не завжди можливе без вказівки початкового стану.

***Модель представлення знань*** – основний тип моделей ШІ. Реалізація конкретних систем, заснованих на знаннях, відбувається в рамках однієї з моделей знань або мови представлення знань. Таких моделей небагато і нижче ми їх розглянемо. Реальні системи рідко засновані на одній з моделей в її класичному розумінні. Найчастіше реальна система – це гібрид з класичних моделей із значною часткою власних припущень, обмежень і умовностей. Найбільш корисні з них називають евристиками.

***Наочна область*** – це область людської діяльності, до вирішення завдань якої застосовується теорія ШІ. Наприклад, якщо Ви створюєте експертну систему, яка по набору результатів аналізів ставить діагноз хворому, то Ваша наочна область – медицина. Якщо Ви створюєте інтелектуальну систему запобігання зіткненню кораблів, то наочна область – навігація, і так далі

4.2.1.2 Сучасні напрями досліджень у області штучного інтелекту

Можна виділити такі важливі напрями розвитку інтелектуальних систем, які показані на рис. 4.34.

1. Системи, що імітують творчі процеси. Створення музичних творів, вирішення ігрових завдань (шахи, шашки, доміно), автоматичний переклад, доказ теорем, розпізнавання образів, імітація мислення і тому подібне
2. Інформаційні системи, засновані на знаннях (експертні системи), тобто консультування недосвідчених користувачів, налаштування устаткування, навчання і ін.
3. Інтелектуальні інформаційні системи – великі і дуже великі програми, призначені для вирішення завдань в наочній області на основі математичних і алгоритмічних моделей і що володіють здатністю вести осмислений діалог з користувачем з метою спростити управління, скоротити об'єм роботи людини, підвищити якість і тому подібне
4. Робототехника. З погляду "інтелектуальності" розрізняють декілька поколінь роботів.

Перше покоління – ***роботи-маніпулятори***, що діють за заздалегідь затвердженою і незмінною програмою (наприклад, що подають заготовки до верстата). Якщо в процесі роботи по яких-небудь причинах зміниться відстань до заготівки, робот її втратить.

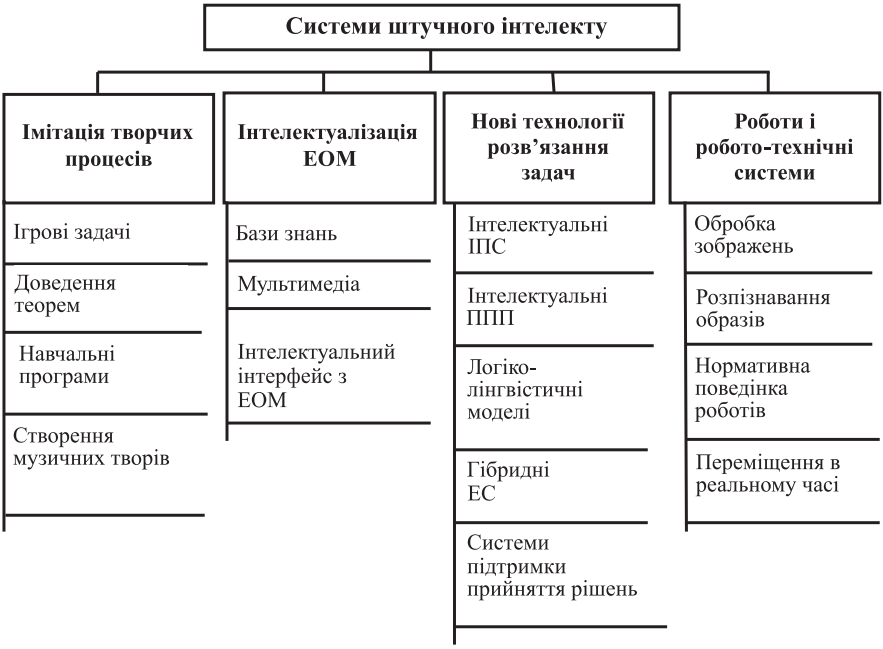



Рисунок 4.34. Основні напрямки розвитку і застосування систем ШІ.

Друге покоління – ***адаптивні роботи***. Складові частини таких роботів оснащені великою кількістю датчиків: кутомірів, тензометрів, газових аналізаторів і ін. Такі роботи застосовуються, наприклад, для зварки кузовів автомобілів. Достатньо крупні технічні вироби, що навіть випускаються серійно, все одно виходять унікальними. Довжина двох автомобілів однієї і тієї ж марки різна. Адаптивний робот робить зварний шов не в абсолютних координатах, а відносно деякої точки початку відліку, яку сам же і знаходить на кузові автомобіля. При необхідності людина-зварювач бере управління на себе, а робот, запам'ятовуючи його дії в новій ситуації, "навчається".

Перші два види роботів – промислові роботи, сконструйовані для роботи в заздалегідь визначеному середовищі, – в цеху заводу. Для орієнтації і точного позиціонування в просторі в цеху є велика кількість контрольних точок – датчиків, координати яких незмінні. Набагато складніше роботові, який знаходиться в реальному світі, наприклад, обстежує затонулий корабель. Доводиться відповідати на питання: "де я? ", "що переді мною? ", "чи можна пройти до?" та ін. Інтелектуальним роботам доводиться обробляти дані своїх датчиків і команди людини в реальному масштабі часу, а якщо передбачений ще і прогноз подій, то в "наднатуральному" (застосовуються, наприклад, для управління транспортними засобами).

Схожі проблеми виникають у ***пошукових роботів*** – особливого класу програм, призначених для індексування документів в глобальній мережі Інтернет, що забезпечує роботу пошукових машин.

Системі ШІ, що використовуються в економіці, в основному є системами, які грунтуються на знаннях (від англійського knowlege based systems).

В майбутньому можливі такі області застосування ШІ:

У гірській промисловості комп'ютери покликані працювати там, де виникають занадто небезпечні умови для людей.

У сфері виробництва ЕОМ повинні виконувати різного виду задачі по зборці і технічному контролю.

В установах ЕОМ зобов'язані займатися складанням розкладів для колективів і окремих людей, робити коротке зведення новин.

У навчальних закладах ЕОМ повинні розглядати задачі, що вирішують студенти, здійснювати пошук помилок, подібно тому як відшуковуються помилки в програмі і усувати їх. Вони повинні забезпечувати студентів книгами, що зберігаються в пам'яті обчислювальних систем.

У лікарнях ЕОМ повинні допомагати ставити діагноз, направляти хворих у відповідні відділення, контролювати хід лікування.

У сільському господарстві комп'ютери повинні оберігати посіви від шкідників, підрізати дерева.

У домашньому господарстві ЕОМ повинні допомагати порадами щодо готування їжі, закупівлі продуктів, стежити за станом підлоги в квартирі і газону в саду.

Звичайно, у даний час жодний з цих варіантів не представляється можливим, але дослідження в області ШІ можуть сприяти їхньої реалізації.

Проведемо короткий історичний огляд розвитку робіт в області ШІ.

Штучній інтелект як науковий напрям, пов'язаний із спробами формалізувати мислення людини, здавна привертав увагу учених. Платон, Арістотель, Р. Декарт, Г.В. Лейбніц, Дж. Буль і багато інших дослідників на рівні знань свого годині намагалися описати мислення як сукупність деяких елементарних операцій, правив і процедур. Якісно новий період розвитку ШІ пов'язаний з появою в наукових лабораторіях ЕОМ і публікацією книги Н. Вінера "Кібернетика або управління і зв'язок у тварині і машині".

Початок досліджень в області ШІ (кінець 50-х років) зв'язують з роботами Ньюєлла, Саймона і Шоу, що досліджували процеси рішення різних завдань. Результатом їх робіт були такі програми як "ЛОГІК-ТЕОРЕТИК", яка призначена для доказу теорем у вирахуванні висловлювань і "ЗАГАЛЬНІЙ ВИРІШУВАЧ ЗАВДАНЬ". Ці роботи поклали початок першому етапу досліджень в області ШІ, зв'язаному з розробкою програм, що вирішують задачі на основі застосування різноманітних евристичних методів.

Евристичний метод рішення задачі при цьому розглядався як властивий людському мисленню "узагалі", для якого характерне виникнення здогадів про шляхи рішення задачі з наступною перевіркою їх. Йому протиставлявся алгоритмічний метод, який використовується в ЕОМ, що інтерпретувалося як механічне здійснення заданої послідовності кроків, який детерсіновано приводив до правильної відповіді. Трактування евристичних методів рішення завдань як суто людської діяльності й обумовили появу і подальше поширення терміна ШІ. Так, при описі своїх програм Ньюєлл і Саймон приводили докази як підтвердження факту, що їх програми моделюють людське мислення, результати порівняння записів доведення теорем у виді програм із записами міркування *мислячої вголос* людини.

У 1950 р., тобто в період розробки перших ЕОМ, англійський математик Т'юрінг опублікував статтю, що стосується області науки, яку згодом стали називати "штучним інтелектом". На питання: "Чи можуть машини мислити?". Він запропонував таку відповідь: "Так, я думаю, що це буде можливо через п'ятдесят років". Основна ідея Т'юрінга полягала в досить простому визначенні штучного інтелекту як множини наукових дисциплін, які за допомогою ЕОМ імітують інтелектуальні здібності людини.

Термін "штучний інтелект" (artificial intelligence) був запропонований в 1956 році на семінарі з аналогічною назвою, який відбувся в США і був присвячений рішенню логічних завдань.

На кінець 70-х років Ньюєлл і Саймон опублікували багато даних і запропонували загальну методику складання програм, що моделюють мислення. Приблизно тоді, коли роботи Ньюєлла і Саймона стали привертати до собі увагу, у Массачусетському технологічному інституті,

Стенфордському університеті і Стенфордському дослідницькому інституті також сформувалися дослідницькі групи в області ШІ. На противагу раннім роботам Ньюєлла і Саймона ці дослідження більше стосуються формальних математичних представлень. Способи рішення завдань у цих дослідженнях розвивалися на основі розширення математичної і символічної логіки. Моделюванню ж людського мислення надавалося другорядне значення.

На подальші дослідження в області ШІ великий вплив мала поява методу резолюцій Робінсона, заснованого на доказі теорем у логіці предикатів, який є вичерпним методом. При цьому визначення терміна ШІ досить суттєво змінилося. Метою досліджень, проведених у напрямку ШІ, стала розробка програм, здатних вирішувати "людські задачі". Так, один з відомих дослідників ШІ того часу Р. Бенерджі в 1969 році писав: "Область досліджень, яку звичайно називають ШІ ймовірно, можна представити як сукупність методів і засобів аналізу і конструювання машин, здатних виконувати завдання, з якими донедавна могла справитися тільки людина. При цьому за швидкістю й ефективністю машини повинні бути порівнювані з людиною". Функціональній підхід до спрямованості досліджень із ШІ зберігся в основному дотепер, хоча ще і зараз ряд учених, особливо психологів, намагаються оцінювати результати робіт з ШІ з позицій їх відповідності людському мисленню.

Дослідницьким полігоном для розвитку методів ШІ на першому етапі були всілякі ігри, головоломки, математичні задачі. Деякі з цих завдань стали класичними в літературі з ШІ (задачі про мавпу і банани, місіонерів і людожерів, Ханойські вежі, гра в 15 і інші). Вибір таких завдань обумовлювався простотою і ясністю проблемного середовища (середовища, у якому розвертається рішення задачі) їх відносно невеликим розміром, можливістю досить легкого підбору і навіть штучного конструювання "під метод". Основний розквіт такого роду досліджень припадає на кінець 60-х років, після чого стали здійснюватися перші спроби застосування розроблених методів для завдань, розв'язуваних не в штучних, а в реальних проблемних середовищах.

Необхідність дослідження систем ШІ при їх функціонуванні в реальному світі призвело до постановки задачі *створення інтегральних роботів*. Проведення таких робіт можна вважати наступним етапом досліджень із ШІ. У Стенфордському університеті, Стенфордському дослідницькому інституті і деяких інших місцях були розроблені експериментальні роботи, що функціонують у лабораторних умовах. Проведення цих експериментів показало необхідність рішення кардинальних питань, пов'язаних із проблемою представлення знань про середовище функціонування і одночасно недостатню дослідженість таких проблем, як зорове сприйняття, побудова складних планів поводження в динамічних середовищах, спілкування з

роботами природною мовою. Ці проблеми були більш ясно сформульовані і поставлені перед дослідниками в середині 70-х рр, зв'язаних з початком третього етапу досліджень систем ШІ.

Його характерною рисою став зсув центру уваги дослідників зі створення автономно функціонуючих систем, що самостійно вирішують у реальному середовищі поставлені перед ними задачі, до створення людино-машинних систем, що інтегрують у єдине ціле інтелект людини і здатності ЕОМ для досягнення загальної мети – рішення задачі, поставленої перед інтегральною людино-машинною вирішальною системою. Такий зсув обумовлювався двома причинами:

– до цього часу з'ясувалося, що навіть прості на перший погляд задачі, що виникають перед інтегральним роботом при його функціонуванні в реальному часі, не можуть бути вирішені методами, розробленими для експериментальних завдань у спеціально сформованих проблемних середовищах;
– стало зрозуміло, що сполучення доповнюючих одна одну можливостей людини й ЕОМ дозволяє обійти гострі кути шляхом перекладання на людину тих функцій, які поки ще не доступні для ЕОМ. На перший план висувалася не розробка окремих методів машинного рішення завдань, а розробка методів та засобів, що забезпечують тісну взаємодію людини й обчислювальної системи протягом усього процесу рішення задачі з можливістю оперативного внесення людиною змін у ході цього процесу.

Розвиток досліджень з ШІ в даному напрямку обумовлювалося також різким зростанням виробництва засобів обчислювальної техніки і також різким їх здешевленням, що зробило їх потенційно доступними для більш широких кіл користувачів.

Отже, в 60-х – 70-х рр., відбулося розділення на два самостійні напрями: машинний інтелект і штучний розум. Це розділення зберігається і до теперішнього часу.

Розділення робіт по ШІ на два напрями пов'язане з існуванням двох точок зору на побудову системи ШІ. *Програмно-прагматічне* ("не має значення, як влаштований "мислячий" пристрій, головний, щоб на задані вхідні дії воно реагувало, як людський мозок"). Піхильникі цієї точки зору переконані в тому, що найважливішим є результат, тобто хороший збіг поведінки штучно створених і природних інтелектуальних систем, а що стосується внутрішніх механізмів формування поведінки, то розробник ШІ не повинний копіювати або навіть приймати до уваги особливості природних, живих аналогів.

*Біонічне* ("єдиний об'єкт, здатний мислити – це людський мозок, тому будь-який «мислячий» пристрій повинен якимось чином відтворювати

його структуру") – саме вивчення механізмів природного мислення і аналіз даних про способи формування розумної поведінки людини можуть створити основу для побудови систем ШІ, причому побудова ця повинна здійснюватися, перш за все, як моделювання, відтворення технічними засобами принципів і конкретних особливостей функціонування біологічних об'єктів.

Обидва напрями ШІ пов'язані з моделюванням. У першому випадку з моделюванням феноменологічним імітаційним, а в іншому – із структурним. У спрощеному вигляді структура основних напрямів, що існують у ШІ, зображена на рис. 4.35.

Фахівці у області штучного інтелекту запропонували такі моделі:

- ✧ *модель лабіринтового пошуку*. Цей підхід представляє завдання як деякий граф, що відображає простір станів і у цьому графі проводитися пошук оптимального шляху від вхідних даних до результуючих;
- ✧ *евристичне програмування*. *Евріcтика* – правило, теоретично не обгрунтоване, але яке дозволяє скоротити кількість переборів в просторі пошуку;
- ✧ *використання методів математичної логіки*. На основі методу резолюцій, що дозволили автоматично доводити теореми за наявності початкових аксіом, в 1973 р. була створена мова Пролог.

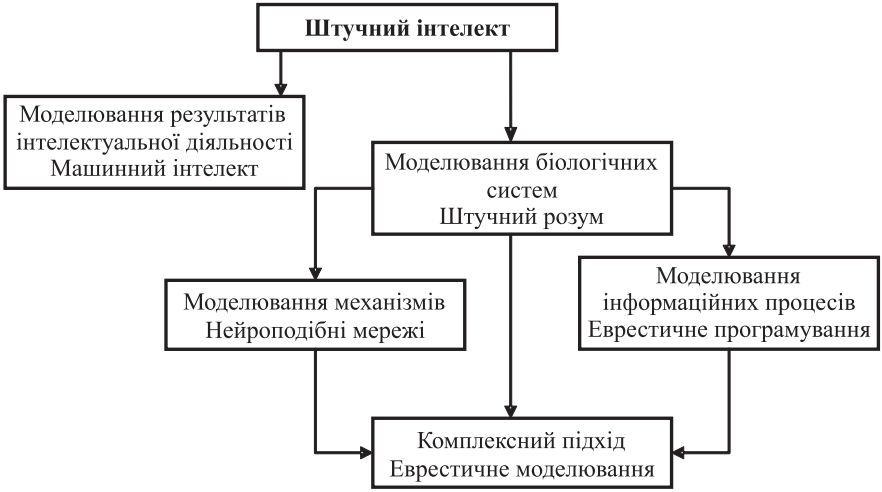



Рисунок 4.35. Основні напрямки досліджень у області штучного інтелекту.

4.2.1.3 Категорія знання, класифікація.
Проблема представлення знань

**Інтелектуальний інтерфейс інформаційної системи**

Аналіз розвитку засобів обчислювальної техніки дозволяє стверджувати, що вона постійно еволюціонує в двох напрямах. Перший з них зв'язаний з поліпшенням параметрів існуючих комп'ютерів, підвищенням їх швидкодії, збільшенням об'ємів їх оперативної і дискової пам'яті, а також з вдосконаленням і модифікацією програмних засобів, орієнтованими на підвищення ефективності виконання ними своїх функцій. Це можна назвати розвитком по горизонталі.

Другий напрям визначає зміни в технології обробки інформації, що приводять до поліпшення використання комп'ютерних систем. Розвиток в цьому напрямі пов'язаний з появою нових типів комп'ютерів і якісно нових програмних засобів, доповнюючих що вже існують. Такий розвиток можна назвати розвитком по вертикалі.

Інтелектуальний інтерфейс, що забезпечує безпосередню взаємодію кінцевого користувача і комп'ютера при рішенні задачі у складі людино-машинної системи, повинний виконувати три групи функцій:

- забезпечення користувачу можливості постановки завдання для ЕОМ шляхом сполучення тільки її умови (без завдання програми рішення);
- забезпечення для користувача можливості формування середовищ рішення задачі з використанням тільки термінів і понять з області професійної діяльності користувача, природних форм представлення інформації;
- забезпечення гнучкого діалогу з використанням різноманітних засобів, зокрема тих, що не регламентуються наперед, з корекцією можливих помилок користувача.

Структура системи (рис. 4.36), яка задовольняє вимогам нової технології рішення завдань, складається з трьох компонент:

- виконавчої системи, що є сукупністю засобів, які забезпечують виконання програм;
- бази знань, яка містить систему знань про проблемне середовище;
- інтелектуального інтерфейсу, що забезпечує можливість адаптації обчислювальної системи до користувача.

Шлях реалізації новітніх інформаційних технологій припускає використання обчислювальних систем, побудованих на основі представлення знань предметної області завдання і інтелектуального інтерфейсу.

**Категорія знання**

У основі досліджень в області ШІ лежить підхід, пов'язаний із знаннями. Опора на знання – базова парадигма ШІ. У Великій радянській енциклопедії дається наступне тлумачення: "Знання – перевірений прак-

тикою результат пізнання дійсності, вірне її віддзеркалення в свідомості людини. Знання бувають життєвими, донауковими, художніми, науковими (теоретичними і емпіричними) ".

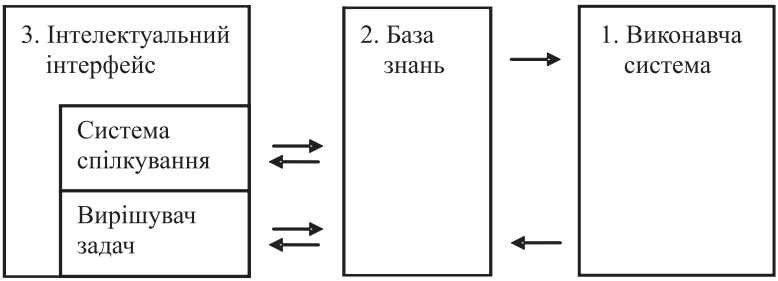



Рисунок 4.36. Структура сучасної системи рішення прикладних завдань.

*Знання про деяку предметну область (ПрО)* є сукупністю відомостей про об'єкти цієї ПрО, їх істотні властивості та відношення, що зв'язують їх, процеси, що протікають в даній ПРО, а також методи аналізу ситуацій, що виникають в ній, і способах рішення асоційованих з ними проблем. Враховуючи широту подібного тлумачення знань, коротко розглянемо інші трактування цієї базової категорії.

Трактування знань можуть бути об'єднані в чотири групи (або рівня): психологічну, інтелектуальну, формально-логічну та інформаційно-технологічну (рис. 4.37).

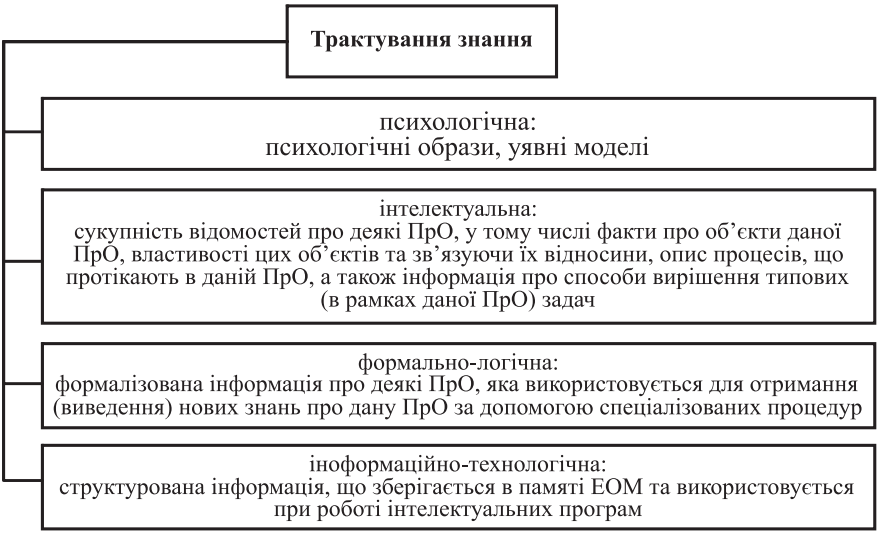



Рисунок 4.37. Трактування знань

Ряд аспектів класифікації знань ілюструє рис. 4.38.

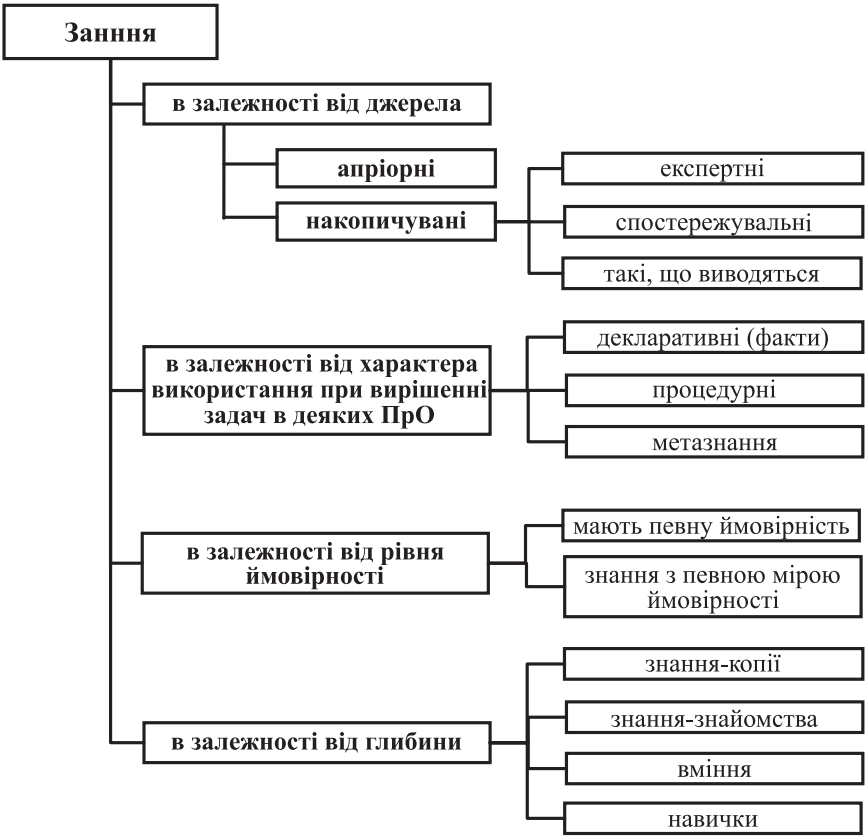



Рисунок 4.38. Класифікація знань.

Знання у контексті, який розглядається, – це основні закономірності, що дозволяють людині вирішувати конкретні виробничі, наукові та інші задачі, тобто факти, поняття, взаємозв'язки, оцінки, правила, а також стратегії прийняття рішень.

У цілому відмінності між даними і знаннями не мають чітких меж. Але можна говорити, що дані – це ізольовані факти, відношення між якими та з зовнішнім світом не зафіксовані. ***Знання*** – це елементи інформації, пов'язані між собою та з зовнішнім світом.

Знання (на відміну від даних):

- ✧ інтерпретуються; мають класифікуючі зв'язки (родо-видові, елемент – клас, клас – підклас, тип – підтип, ситуація – підситуація та ін.);

- ✧ мають ситуативні відношення, що визначають ситуативну сумісність знань, які зберігаються в пам'яті ЕОМ;
- ✧ активні (знання можуть породжувати нові знання).

Моделі, побудовані на знаннях, мають синтаксичний, семантичний і прагматичний аспекти. Жорсткі ж формальні моделі, які орієнтовані на використання даних, мають тільки синтаксичний аспект.

Перехід від даних до знань – логічний наслідок розвитку і ускладнення інформаційно-логічних структур, що обробляються на ЕОМ. Поняття знання не має якого-небудь вичерпного визначення.

Виділяють наступні ***види знань***:

- ✧ інтерфейсні – знання про взаємодію з навколишнім середовищем;
- ✧ проблемні – знання про наочну область (ПрО);
- ✧ процедурні – знання про методи вирішення завдань;
- ✧ структурні – знання про операційне середовище;
- ✧ метазнання – знання про властивості знань.

*Знання* можна визначити як сукупність відомостей, які створюють цілісний опис, відповідний деякому рівню обізнаності про описуване питання, предмет, проблему і т.д. Іншими словами, ***знання*** – це виявлені закономірності наочної області (принципи, зв'язки, закони), що дозволяють вирішувати задачі в цій області. З погляду ***штучного інтелекту знання*** можна визначити як ***формалізовану інформацію, на яку і посилаються в процесі логічного висновку***.

Для зберігання даних використовуються бази даних (для них характерні великий об'єм і відносно невелика питома вартість інформації), для зберігання знань – бази знань (невеликого об'єму, але виключно дорогі інформаційні масиви).

**База знань –** це сукупність знань, описаних з використанням вибраної форми їх подання. База знань є основою будь-якої інтелектуальної системи.

Якщо розглядати знання з погляду рішення завдань в деякій наочній області, то їх зручно розділити на дві великі категорії – ***факти*** і ***евристику***. Перша категорія вказує звичайно на добре відомі в даній наочній області обставини, тому знання цієї категорії іноді називають текстовими, підкреслюючи їх достатню освітленість в спеціальній літературі або підручниках. Друга категорія знань грунтується на власному досвіді фахівця (експерта) в даній наочній області, накопиченому в результаті багаторічної практики.

Знання можна розділити на ***процедурні*** і ***декларативні***. Історично першими були ***процедурні знання***, тобто знання, "розчинені" в алгоритмах. Вони управляли даними. Для їх зміни потрібно було змінювати програми, проте з розвитком штучного інтелекту пріоритет даних поступово змінився і все більша частина знань зосереджувалася в структурах даних (таблиці, списки, абстрактні типи даних), тобто збільшувалася роль декларативних

знань. Іншими словами, відбулося перенесення центру тяжіння з машинного представлення процедур на машинне представлення знань.

Традиційно структури даних розуміються як декларативні знання, що несуть тільки функцію відображення наочної області. Над структурами даних може здійснюватися ***впорядкована послідовність операцій*** – програма (процес), що реалізовує деякий алгоритм. Результатом роботи програми завжди є декларативне знання, а сама програма є процедурним знанням.

***Декларативні знання*** – це сукупність відомостей про якісні і кількісні характеристики конкретних об'єктів, явищ і їх елементів, представлених у вигляді фактів і евристик. Традиційно такі знання накопичувалися у вигляді різноманітних таблиць і довідників, з появою ЕОМ придбали форму інформаційних масивів (файлів) і баз даних. Декларативні знання часто називають просто даними. Декларативні знання зберігаються в пам'яті ІС так, що вони безпосередньо доступні для використання. У вигляді декларативного знання записується інформація про властивості наочної області, факти, що мають в ній місце і тому подібна інформація.

***Процедурні знання*** зберігаються в пам'яті ІС у вигляді описів процедур, за допомогою яких їх можна одержати. У вигляді процедурних знань звичайно описується інформація про наочну область, що характеризує способи рішення завдань в цій області, а також різні інструкції, методики і тому подібна інформація. Іншими словами, ***процедурні знання*** – це методи, алгоритми, програми рішення різних завдань, послідовності дій (у вибраній проблемній області) – вони складають ядро баз знань. Наприклад, в продукційних моделях – це множина продукційних правил виду "ЯКЩО ТО"; у виробничій сфері аналог процедурних знань – технологічні знання про способи організації і здійснення різноманітних виробничих процесів. Процедурні знання утворюються в результаті здійснення процедур (алгоритмів, програм, аналітичних перетворень і т. інш.) над фактами як початковими даними.

Подальший розвиток структур даних в рамках досліджень по штучному інтелекту привів до появи спеціальних структур даних: фреймів, семантичних мереж, продукцій, названих знаннями. З появою систем, заснованих на знаннях, знаннями вважаються пропозиції, записані на мовах представлення знань, наближених до природних і зрозумілих неспеціалістам. У семантичному плані обробка інформації одержує нове забарвлення, пов'язане вже з уявленням і обробкою знань, з отриманням необхідних знань, але не з процесом.

Однією з найбільш важливих проблем, характерних для систем штучного інтелекту, є представлення знань. Це пояснюється тим, що форма представлення знань істотно впливає на характеристики і властивості системи. Для того, щоб маніпулювати знаннями з реального світу за допомогою комп'ютера, необхідно здійснювати їх моделювання.

***Проблема представлення знань*** – це проблема представлення взаємозв'язків в конкретній наочній області у формі, зрозумілій системі штучного інтелекту. Представлення знань – це їх формалізація і структуризація (у цілях полегшення рішення задачі), за допомогою яких відображаються характерні ознаки знань: внутрішня інтерпретованість, структурованість, зв'язність, семантична метрика і активність. Іншими словами, представлення знань – це угода про те, як описувати реальний світ. В рамках цього напряму розв'язуються завдання, пов'язані з формалізацією і представленням знань в пам'яті інтелектуальної системи за допомогою спеціально розроблених моделей представлення знань.

На відміну від методів представлення даних, що базуються на строгих алгоритмах, моделі представлення знань мають справу з інформацією, що одержується від фахівця в конкретній наочній області (експерта), яка часто носить якісний і навіть суперечливий характер. Проте, подібна інформація повинна бути приведена до формалізованого вигляду. Це здійснюється з використанням різних методів і прийомів, зокрема, на основі ідей багатозначної логіки, теорії нечітких множин і інших математичних моделей.

При роботі із знаннями використовуються два основні підходи:

***Логічній ( формальний)*** підхід, при якому основна увага приділяється вивченню і застосуванню теоретичних методів представлення знань, формалізації, а також логічній повноті (наприклад, створення моделей представлення знань на основі деяких логічних числень).

***Евристичний (когнітивний)*** підхід, який орієнтується на забезпечення можливостей рішення завдань. При цьому наголос робиться на принцип організації людської пам'яті та евристичне моделювання. На відміну від формальних, евристичні моделі мають різноманітний набір засобів, які передають специфічні особливості тієї або іншої області. Саме тому евристичні моделі перевершують логічні по виразності і можливості адекватно представити наочну область.

Одна з ключових проблем, що виникають при побудові ІАС, полягає в необхідності вибору і реалізації способу представлення знань. Важливість даного завдання обумовлюється тим, що саме представлення знань зрештою визначає характеристики системи. Використаний метод представлення знань впливає на рішення таких питань інженерії знань, як витягання знань з різних джерел, придбання знань від експертів, інтеграція і узгодження знань тощо.

Диференціація ментальних (уявних) представлень людини і формально-логічного представлення знань служить фундаментом для виділення поняття моделі знань, що визначає спосіб формального опису знань в БЗ. **Модель –** об'єкт, реальний, знаковий або уявний, відмінний від початкового, але здатний замінити його при рішенні задачі. Вона визначає форму представлення знань в БЗ.

Застосування систем ШІ в професійній справі – це не випадкова подія, а вимоги часу. Їх використання для надання допомоги керівнику пояснюється ускладненням завдань управління в економічної сфері на сучасному етапі. Тому принципи, на яких будуються системи ШІ, необхідно знати і у подальшому використовувати при розробці програмних систем і комплексів, що призначені для обгрунтування рішень, що приймаються.

### 4.2.2 Моделювання знань про наочні області як основа інтелектуальних автоматизованих систем

#### 4.2.2.1 Моделі представлення знань

Як показано раніше, центральною частиною інтелектуальної системи служить база знань, яка повинна містити в собі всі необхідні знання про проблемну область і способи рішення характерних для даної області завдань.

При формуванні бази знань виникають три питання:

- ✧ як добути потрібні знання;
- ✧ як сформувати модель проблемної області:
- ✧ як підтримувати цю модель в процесі функціонування системи.

Вірішують ці проблеми фахівці нового типу – **інженери знань**.

Професійні знання діляться на дві частини:

- ✧ знання, що легко описуються словами і добуваються відносно легко з підручників, монографій, звітів худе;
- ✧ знання, що неможливо висловити словами (формалізувати).

Розглянемо моделі представлення знань.

Перелік реальних моделей, вживаних в системах ШІ, широкий і важко класифікується. У сучасній теорії ШІ використовуються декілька основних груп моделей: логіка, фрейми, семантичні мережі, нейронні мережі та ін. Більшість моделей в реальних системах є гібридними, зі значною часткою евристик.

Класифікація найбільш відомих моделей наведена на схемі (рис. 4.39).

#### 4.2.2.2 Логічна модель для представлення знань

Логічна модель за своєю практичною результативністю і ступенем впровадження в реальні технічні пристрої сьогодні займає центральне місце. Сама по собі модель не однорідна і включає вже близько десятка істотно різних теорій (У штучному інтелекті говорять про “моделі представлення знань”, а в логіці – про “формальні теорії”. Стосовно даного контексту це одне і те ж.) (рис. 4.40).

Історично першою моделлю для представлення знань (серед моделей ШІ взагалі) стала класична (формальна) логіка Аристотеля, що заклала фундамент всіх подальших логічних теорій.

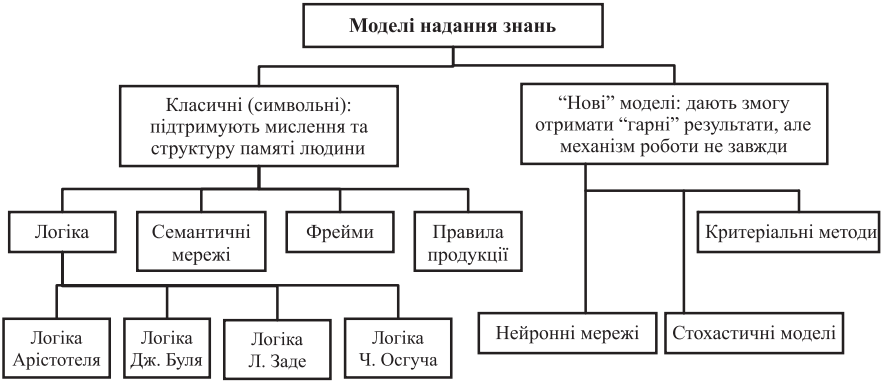



Рисунок 4.39. Деякі найбільш відомі моделі надання знань.

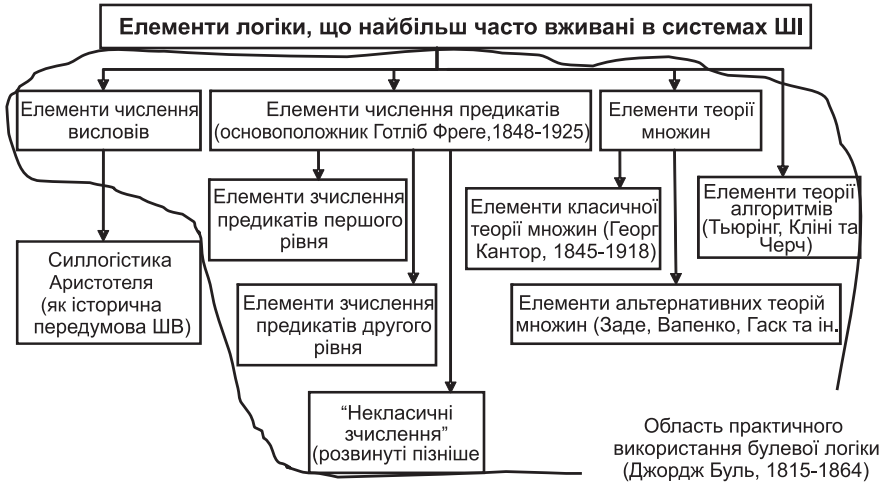



Рисунок 4.40. Елементи логіки, найбільш часто вживані в системах ШІ.

Аристотель (вчитель Александра Македонського) народився в 384 р. до н.е. в селі Стагире в Халкидіке (північ Греції). У 17 років прийшов до Афін (щоб вчитися у Платона), де і став відомий. Пізніше заснував бібліотеку і ліцей (філософську школу). Зробив значний вплив на подальший розвиток всіх наук. Твори відносяться до всіх областей знання того часу: "Фізика", "Про походження і знищення", "Про небо", "Про метеорологічні питання", "Механіка". За первинні якості матерії вважав дві пари протилежностей "тепле-холодне" і "сухе-вологе". Основними (нижчими) елементами, або

стихіями, – землю, повітря, воду і вогонь. Елементи миру розглядав як комбінації первинних якостей: з'єднанню холодного з сухим відповідає земля, холодного з вологим – вода, теплого з сухим – вогонь. П'ятим, за найбільш досконалий елемент вважав ефір. В середні віки Аристотелівське вчення про Всесвіт було канонізоване церквою. Сьогодні пам'ятники Аристотелю встановлені по всій Греції.

Одним з напрямків досліджень Аристотеля стали дослідження міркувань, що отримали надалі назву формальної логіки (наука про закони і операції правильного мислення (логічного виводу, міркування). Основне завдання логіки – відділення правильних способів міркування від неправильних. Логіка Аристотеля називається формальною, тому що з її точки зору правильність міркування залежить тільки від логічної форми цього міркування і не залежить від його змісту. Зміст думки не чинить жодного впливу на правильність міркувань.

Подальший розвиток моделі пов'язаний з іменами Канта (семантичний аналіз на основі категорій), Буля ("математизація" логіки), Фреге, Заді (ввів поняття "Нечіткої логіки"). Останні десятиліття активно розвиваються «некласичні» напрями: модальна, тимчасова логіки і ін.

Сенс побудови будь-якої формальної теорії полягає в тому, щоб виразити розумові процеси (а в ідеалі – створити універсальну модель світу) формально (тобто записати формулами).

Для того, щоб задати формальну логічну теорію, необхідно визначити алфавіт (множину символів, які використовуються для запису), правила синтаксису (правила запису формул), аксіоматику (особлива підмножина формул) і правила виводу (множину відношень на множині формул). Різні логічні теорії мають свої переваги при реалізації моделей в різних мікросвітах[1] .У XIX столітті (через 2000 років після Аристотеля) англійський математик Джордж Буль[2] удосконалив логічну систему позначень.

[1] Так, наприклад, семантика силлогистики Арістотеля дуже близька до природної мови, тому результати формального виводу легко інтерпретувати. Виводи, що отримуються на основі числення предикатів, вже не повною мірою збігаються з семантикою природної мови, їх важче інтерпретувати, проте моделі на основі числення предикатів виходять набагато компактнішими і, як наслідок, осяжними. Моделі на основі нечіткої логіки Л. Заде дозволяють оперувати розмитими поняттями, проте такі результати і інтерпретувати важче і навіть не завжди можливо.

[2] Джордж Буль В. 11.1815–8.12.1864) – отець математичної логіки. Професор математики, самоучка (закінчив тільки школу). Знав грецьку, французьку, німецьку і італійську мови. Вчився по книгах Ньютона, Лапласа, Лагранжа. Був відмічений як талановитий математик професором Августом Де Морганом. У 1854 г. опублікував роботу "Дослідження законів мислення, що базуються на математичній логіці і теорії вірогідності". Першим показав, що існує аналогія між діями алгебри і логічних, спробував сформулювати загальний метод вірогідності, за допомогою якого із заданої системи вірогідних подій можна було б визначити вірогідність наступної події, логічно пов'язаної з ними.

Він запропонував записувати істинність як 1, а брехню як 0. Буля не цікавило все можливе різноманіття відношень природної мови. Він розглянув тільки відношення "+" об'єднання, "*" перетину, виключення.

Вдосконалена система запису Буля дозволила поставити абстрактне логічне завдання у вигляді системи рівнянь:

$x_1 + x_2 = 1$
$x_2 * x_3 = 0.$

Хто з'їв шоколад? Кішка Ліза ($x_1$) або ("+") щеня Кинг ($x_2$)? Якщо господар ($x_3$) не їв, а гуляв в цей час з щеням, означає це була Ліза.

Буль ввів спеціальні позначення (табл.4.3).

Таблиця 4.3

**Деякі позначення булевої логіки**

| ***Операція*** | ***Обчислення*** |
|---|---|
| Коньюнкція | «і», « $\wedge$ », « $\times$ » |
| Диз'юнкція | «чи», « $\vee$ », «+» |
| Еквівалентність | «*A* тоді і тільки тоді, коли *B*», « $\leftrightarrow$ » чи « $\equiv$ » |
| Імплікація | «якщо *A*, то *B*, *A* тягне *B*», « $\rightarrow$ » |
| Заперечення | «ні», « $\neg$ » |

За часів Дж. Буля математична логіка не мала того найважливішого прикладного значення, яке їй надається зараз. З винаходом в 1918г. Бонч-Бруєвичем[3] принципу тригерного кільця або просто тригера з'явилася можливість реалізувати рівняння Буля, а отже, і частину логіки Арістотеля в електричних і електронних пристроях.

Прикладів такого застосування дуже багато, найголовніший – це, звичайно ж, створення комп'ютерів. Набор мікросхем, на яких вони будуються, називають елементарною логікою або елементною базою. Комп'ютер – це дуже велика і складна система логічних рівнянь.

[3] Михайло Олександрович Бонч-бруєвіч 9.02.1888-07.03.1940). Винахідник тригерного кільця А918. Незалежно від нього відкриття повторили англійські учені У. Екклз і Ф. Джордан А919. У 1909 г. закінчив Інженерне училище в Петербурзі, в 1914 г. – Офіцерську електротехнічну школу. З 1922 г. професор Московського вищого технічного училища, з 1932 г. професор Ленінградського інституту інженерів зв'язку, що нині носить його ім'я. У 1916-19 рр. створив вітчизняне виробництво електронних ламп. У 1918 г. очолив Ніжегородську радіолабораторію, що об'єднала кращих радіо фахівців того часу. У 1919-25 рр. створив конструкцію могутньої генераторної радіолампи з водяним охолоджуванням і схеми радіотелефонних станцій. У 1924-30 рр. керував розробкою першої в світі короткохвильової направленої антени і короткохвильових лінії дальнього радіозв'язку. Займався питаннями фізики верхніх шарів атмосфери і їх практичним застосуванням, зокрема в області радіолокації.

У ***логічних моделях*** знання представляються у вигляді сукупності правильно побудованих формул якої-небудь формальної системи (ФС), яка задається четвіркою

$$(T,P,A,R), \tag{4.5}$$

де Т – множина базових (термінальних) елементів, з яких формуються всі вирази ФС;

Р – множина синтаксичних правил, що визначають синтаксично правильні вирази з термінальних елементів ФС;

А – множина аксіом ФС, відповідних синтаксично правильним виразам, які в рамках даної ФС апріорі вважаються за істинних;

R – кінцева множина правил виводу, що дозволяють отримувати з одних синтаксично правильних виразів інші.

Простою логічною моделлю є *числення висловів*.

***Числення висловів*** є одним з початкових розділів математичної логіки, служить основою для побудови складніших формальних виразів. У практичному плані числення висловів застосовується у ряді наочних областей (ПрО) (зокрема, при проектуванні цифрових електронних схем).

Розвиток логіки висловів знайшло віддзеркалення в *численні предикатів першого порядку*. В основі ***числення предикатів*** лежить ототожнення властивостей і відношень з особливими функціями, які називаються логічними або пропозиційними. Аргументами пропозиційних функцій (або предикатів) є терми, відповідні об'єктам-носіям властивостей і відношень, а значеннями цих функцій можуть бути істина або брехня. У разі виразу властивості істина інтерпретується як факт приналежності даної властивості терму-аргументу, брехню позначає заперечення цього факту.

Представлення знань в рамках логіки предикатів служить основою напряму ШІ, яке називається *логічним програмуванням*. Методи логічного програмування в даний час широко використовуються на практиці при створенні ІС в ряді ПРО.

Відзначимо наступні *недоліки* логічних моделей.

По-перше, представлення знань в таких моделях ненаглядно: логічні формули важко читаються і сприймаються.

По-друге, обмеження числення предикатів першого порядку не допускають квантифікації предикатів і використання їх як змінних.

По-третє, обгрунтованість позначення властивостей і відношень однотипними пропозиційними функціями викликає сумніви.

По-четверте, опис знань у вигляді логічних формул не дозволяє застосувати переваги, які є при автоматизованій обробці структур даних.

Шляхи підвищення ефективності логічних моделей знань зв'язані з використанням багаторівневих і спеціальних логік.

4.2.2.3 Продукційна модель для представлення знань

Для того, щоб реалізувати описані вище принципи логічного виводу, в автоматизованій системі необхідна спеціальна машинно-орієнтована мова. Однією з найбільш простих і ефективних машинно-орієнтованих мов для опису логічних завдань є правила продукцій.

"Продукція" є виразом вигляду:

$$\underbrace{\text{якщо } A \text{ (умова), то } B \text{ (дія),}}_{\text{ядро}} \text{ постумова } C$$

Під *умовою* розуміється деяка пропозиція-зразок, по якій здійснюється пошук в базі знань, а під *дією* – дії, що виконуються при успішному результаті пошуку (вони можуть бути проміжними, використовуватися далі як умови, і цільовими (термінальними), завершуючими роботу системи). *Постумова* описує дії і процедури, які необхідно виконати після реалізації дії. Наприклад, після покупки деякої речі в магазині необхідно в описі товарів зменшити кількість речей такого типу на 1.

Центральною ланкою продукційної моделі є множина продукцій або правил виводу. Кожна така продукція в загальному вигляді може бути представлена виразом

$$(W_i, U_i, P_i, A_i \rightarrow B_i, C_i), \tag{4.6}$$

де $W_i$ – сфера застосування *i-ї* продукції, що визначає клас ситуацій в деякій ПрО (або фрагменті робочої ПрО), в яких застосування даної продукції правомірне;

$U_i$, – передумова *i-ї* продукції, що містить інформацію про істинність даної продукції, її значущість щодо інших продукцій тощо;

$P_i$ – умова і-ї продукції, яка визначається чинниками, що безпосередньо не входять в $A_i$, істинне значення якої дозволяє застосовувати дану продукцію;

$A_i \rightarrow B_i$ – ядро і-ї продукції, відповідне правилу "якщо..., то... ";

$C_i$ – постумова і-ї продукції, що визначає зміни, які необхідно внести до системи продукцій після виконання даної продукції.

Реалізація логічних і продукційних моделей знань базується на мовах типу Прологу.

***Продукційна система*** є базою знань (правил продукцій) і машиною виводу – спеціальною програмою "Зіставлення за зразком". Залежно від того, які використані продукції і які правила виводу, виходять різні продукційні системи.

Ядра продукцій можна класифікувати на різних підставах. Перш за все розрізняють ***детерміновані*** і ***недетерміновані*** ядра. При актуалізації

детермінованого ядра і здійснимості умови права частина (дія) виконується з неминучістю (і ступенем впевненості), в недетермінованих – з вірогідністю.

Якщо А, то, можливо, В з вірогідністю б.

Якщо А, то В з коефіцієнтом упевненості б (детермінована продукція).

Продукції також можуть бути ***однозначними*** і ***альтернативними***. Для альтернативних правил в правій частині ядра указуються "альтернативні можливості вибору", які оцінюються "вагами вибору, – $б_1$, $б_2$ ., $б_n$".

Окремо виділяють прогнозні продукції, в яких описуються наслідки, очікувані при актуалізації А, наприклад:

якщо А, то з вірогідністю р можна чекати Ст.

**Продукційні правила –** найбільш простий спосіб представлення знань. Він заснований на представленні знань у формі правив, структурованих відповідно до зразка "ЯКЩО – ТО". Частина правила "ЯКЩО" називається посилкою, а "ТО" – висновком або дією. Правило в загальному вигляді записується так:

ЯКЩО *A1*, *A2...,A*n ТО *Ст.*

Такий запис означає, що "якщо всі умови від *A1* до *An* є істинними, то *B* також істинно" або "коли всі умови від *A1* до *An* виконуються, то слід виконати дію".

Розглянемо правило:

ЯКЩО (1) *y* є батьком *x*
*z* є братом *y*
ТО (2) *z* є дядьком *x*

У випадку *n=0* продукція описує знання, що складається тільки з висновку, тобто *факт*. Прикладом такого знання є факт – "атомна маса заліза 55,847 а.е.м. ".

У інтелектуальну систему входити також **механізм висновків –** частина продукційної системи, яка дозволяє на основі знань, наявних в базі знань, одержувати нові знання.

Проілюструємо сказане. Покладемо, що в БЗ разом з описаним вище правилом містяться і такі знання:

ЯКЩО (1) *z* є батьком *x*
(2) *z* є батьком *y*
(3) *x* і у не є однією і тією ж людиною
ТО *x* і *y* є братами
Іван є батьком Сергія
Іван є батьком Павла
Сергій є батьком Миколи

З представлених знань можна формально вивести висновок, що Павло – дядько Миколи. При цьому вважається, що однакові змінні, що входять

в різні правила, незалежні; об'єкти імена яких ці змінні можуть одержувати, ніяк не зв'язані між собою. Формалізована процедура, що використовує зіставлення (при якому встановлюється, чи співпадають між собою дві форми уявлення, включаючи підстановку можливих значень змінних), пошук в БЗ, повернення до початкового стану при невдалій спробі рішення є механізм висновків.

Продукційна база знань (БЗ) складається з набору фактів і правил. Програма, що управляє перебором правил, називається ***машиною виводу***. Найчастіше вивід буває прямим (від даних до пошуку мети) або зворотний (від мети для її підтвердження – до даних). Дані – це початкові факти, на підставі яких запускається машина виводу, тобто програма, що перебирає правила з бази.

Продукційна модель (все ще) частіше за інших застосовується в промислових експертних системах. Вона привертає розробників своїми ***позитивними сторонами***:

- ✧ ясністю і наочністю;
- ✧ високою модульністю;
- ✧ легкістю внесення доповнень і змін (модифікації БЗ);
- ✧ простотою механізму логічного виводу (виконання продукций).

Є велике число програмних засобів, що дозволяють реалізувати продукційний підхід:

- ✧ мови Prolog, Lisp[4], LOGO[5], OPS;
- ✧ "оболонки" або "порожні" ЕС, а саме EMYCIN, EXSYS, ESISP, ЕКСПЕРТ;
- ✧ інструментальні системи ПІЕС, СПЕІС і ін.

Серед основних досягнень радянської науки в даній області – програма, що автоматично доводить теореми ("АЛІЄВ ЛОМІ" – Ленінградське відділення математичного інституту ім. В.А. Стеклова. Вона заснована на оригінальному зворотному виведенні С.Ю.Маслова, аналогічному методу резолюцій Робінсона).

***Недоліками*** продукційної моделі є:

- ✧ складність управління виводом, неоднозначність вибору конкуруючих правил, тобто при великому числі продукцій стає складною перевірка несуперечності системи продукцій. Це примушує при додаванні нових продукцій витрачати багато часу на перевірку;
- ✧ неоднозначність обліку взаємозв'язку окремих продукцій, тобто системі властива недетермінованість (неоднозначність вибору виконуваної продукції зі списку продукцій, що активізуються), виникають принципові труднощі при перевірці коректності ро-

[4] Розроблений проф. Джоном Маккартні (США, 1961 г.). LISP – List processing language – мова обробки списків.

[5] Створений проф. математики М.І.Т. Сеймуром Пайпертом на основі LISP, 1969 г.

боти системи. Вважається, що якщо в ІС число продукцій досягне тисячі (за іншими джерелами, півтора тисяч), то мало шансів, що система продукцій у всіх випадках правильно функціонуватиме;

- ✧ невідповідність психологічним аспектам уявлення і обробки знань людиною;
- ✧ складність оцінки цілісного уявлення ПрО.

4.2.2.4 Фрейми для представлення знань

Одну з найбільш психологічно обгрунтованих і практично цінних моделей запропонував Марвін Мінськи. У 1974г. вийшла його книга "Фрейми для представлення знань" (Frame (анг.) – кадр, каркас, рамка, скелет) Фрейм – це структура даних для представлення стереотипної ситуації. З кожним фреймом асоційована інформація різних видів. Одна її частина указує, яким чином слід використовувати даний фрейм, інша – що імовірно може спричинити його виконання, третя – що слід зробити, якщо ці очікування не підтвердяться. Ця концепція стала в певному значенні "підведенням межі" під роботами практиків і крупних теоретиків як в області психології, так і комп'ютерного моделювання психіки (рис. 4.41).

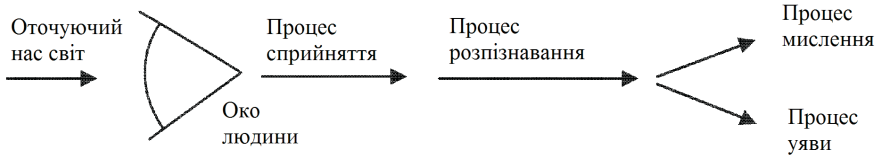



Рисунок 4.41. Психіка з погляду теорії фреймів.

Marvin Minsky – професор комп'ютерних наук в Массачусетському технологічному інституті, США. Фахівець в області штучного інтелекту, когнітивної психології, програмування для машини Тюрінга. Працював також в області обчислювальної геометрії, обчислювальної семантики, машинного сприйняття, символічного навчання. Притягувався до космічних проектів. Один із засновників робототехники і видаленого управління: особисто розробив тактильні і зорові датчики, механічні частини перших роботів. У 1951 г. створив одну з перших нейронних мереж. У 1959 г. заснував лабораторію ШІ – М.І.Т. Багато працював над проблемами представлення знань для розуміння мов і візуального сприйняття. У 70-і рр. створив власну психологічну теорію інтелекту. Всього опублікував близько 10 книг, кожна з яких мала успіх, але найбільший слід визнати, мабуть, за монографією "Фрейми для представлення знань", 1974 г. Вчився в Нью-Йорку, ступені отримав в Гарварді А950) і Прінстоне (доктор філософії, 1954). Одним з викладачів Марвіна був Клод Шеннон – основоположник

теорії інформації і один з перших математиків, що працювали в області ІЇ. Шенон, наприклад, винайшов першу програму для гри в шахи (комп'ютер MANIAC, Los Alamos, 1956) і електронну мишу для вирішення завдання лабіринту. Мінськи і Шенон – легенди історії обчислювальної техніки.

На відміну від семантичних мереж, розроблених спочатку для вузького завдання аналізу текстів і тому охоплювали тільки окремі етапи психічної діяльності, теорія фреймів є достатньо повною. В ній розглядаються не тільки процеси мислення (їх також розглядають силогістика Аристотеля, семантичні мережі), але і сприйняття, розпізнавання, мислення і уява (як одна з високих форм мислення). Витоки теорії фреймів лежать, мабуть, в області гештальт – психології.

"Фрейм можна уявляти собі у вигляді мережі, що складається з вузлів і зв'язків між ними. Верхні рівні фрейма чітко визначені, оскільки утворені такими поняттями, які завжди справедливі по відношенню до передбачуваної ситуації. На нижчих рівнях є багато особливих вершин-терміналів або осередків, які мають бути заповнені характерними прикладами або даними. Кожним терміналом можуть встановлюватися умови, яким повинні задовольняти його завдання. Прості умови визначаються маркерами, наприклад, у вигляді вимоги, щоб завданням терміналу був який-небудь суб'єкт або предмет відповідних розмірів, або покажчик на субфрейм певного типу. Складнішими умовами задаються стосунки між поняттями, включеними в різні термінальні вершини. Групи семантично близьких один до одного фреймів об'єднані в систему фреймів" [Мінськи М. Фрейми для представлення знань. – М.: Енергія, 1979.].




Рисунок 4.42. Сфери застосування фреймів

Стосовно різних завдань людина в кожен момент часу має різні глобальні системи фреймів (рис. 4.42), що дозволяють їй орієнтуватися в просторі, рухатися по кар'єрних сходах, розуміти сенс поезії, шукати необхідну йому інформацію тощо.

Логічна модель не дозволяє аналізувати сенс, тому що в принципі відмовляється це робити. Теорія фреймів дозволяє моделювати як форму (синтаксис), так і сенс (семантику) пропозицій.

Для автоматичного перекладу використовують багаторівневі мережі фреймів, що складаються з поверхневих синтаксичних, семантичних, тематичних і оповідних фреймів.

Фундаментом ***фреймової моделі знань*** служить поняття ***фрейма*** – структури даних, що представляє деякий концептуальний об'єкт або типову ситуацію. Фрейм ідентифікується унікальним ім'ям і включає множину *слотів*. У свою чергу, кожному *слоту* відповідає певна структура даних. У *слотах* описується інформація про фрейм: його властивості, характеристики, факти, що відносяться до нього, і так далі. Крім того, слоти можуть містити посилання на інші фрейми або вказівки на асоційовані з ними приєднані процедури. Уявлення ПрО у вигляді ієрархічної системи фреймів добре відображає внутрішню і зовнішню структури об'єктів цієї ПрО.

У загальному виді фрейм може бути поданий рядком

$$\text{ІФ} < (\text{ІС}, \text{ЗС}, \text{ПП}),..., (\text{ІС}, \text{ЗС}, \text{ПП}) >, \qquad (4.7)$$

де ІФ – ім'я фрейма;
ІС – ім'я слота; ЗС – значення слота;
ПП – ім'я приєднаної процедури (необов'язковий параметр).

Розглянемо приклад фрейму в системі моделювання процесу видачі банківського кредиту під заставу (рис. 4.43). Ця система містить фрейм "Клієнт", який є класом позичальників кредиту. Він містить таку інформацію, що відноситься до людини-клієнта, як забезпечення клієнта, його відповідальність, майно і т.д.

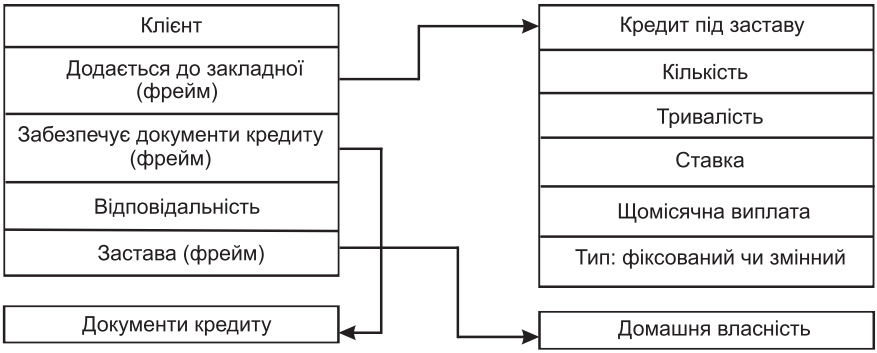



Рис 4.43. Приклад фрейма.

Фрейм має складну структуру даних, в якій атрибути фрейма самі є фреймами. Наприклад, слот "Додається до застави " сам є фреймом "Кредит під заставу" з властивими йому атрибутами.

Значення в слоті "Місячна платня" може бути результатом обчислення по формулі, в якій використовуються слоти "Кількість", "Ставка" і "Тривалість".

***Практична реалізація фреймової моделі***. На практиці найбільше розповсюдження модель, заснована на фреймах, отримала в об'єктно-

орієнтованому програмуванні і теорії об'єктних баз даних. Сучасне поняття об'єкту мови програмування досить точно відповідає класичному поняттю фрейма. Основними властивостями фреймів, використовуваними в сучасних мовах, є: інкапсуляція, спадкоємство і поліморфізм об'єктів.

*Інкапсуляція* – єдність даних і методів в рамках об'єкту.

*Спадкоємство* – здатність об'єкту користуватися методами і даними, визначеними в рамках одного з його предків.

*Поліморфізм* – здатність об'єкту в різні моменти часу поводитися по-різному, то як об'єкт свого типу, то як об'єкт типу будь-якого з предків.

Розрізняють *фрейми-зразки* (прототипи, абстрактні класи), що зберігаються в базі знань, і *фрейми-екземпляри* (об'єкти), які створюються для відображення реальних ситуацій на основі даних, що поступають. Для простоти пошуку потрібного фрейма їх типізують:

- ✧ *фрейми-структури* (матриця сканера, конструкція будинку);
- ✧ *фрейми-ролі* (операціоніст, покупець, директор);
- ✧ *фрейми-сценарії* (похід в ресторан, день народження).

Записують фрейми у вигляді таблиць.

Фреймова модель має ряд безумовних ***переваг***. Насамперед слід сказати, що це чи не єдина глибинна модель, добре обгрунтована теоретично. Її практична реалізація – побудова мікросвітів – дозволяє зосередитися на властивостях і поведінці учасників мікросвіту, надавши їм самим організувати свою взаємодію на основі правил, звичок, сценаріїв і просто випадковостей. Завдяки цим чудовим властивостям, фреймова модель дозволяє значно відсунути ту межу застосовності по розмірності системи, який має кожна модель (див., наприклад, правила продукцій).

Фреймова модель дозволяє добре систематизувати ієрархію об'єктів реального світу, справляючись навіть із завданнями чисельної таксономії, за рахунок введення абстрактних класів. Модель реалізована в багатьох мовах програмування і спеціальних мовах представлення знань – на мовах лінії LISP, FRL (Frame Representation Language), KRL, успішно використана у ряді відомих експертних систем: ANALYST, МОДІС і ін.

Переваги:

- ✧ наочність;
- ✧ гнучкість;
- ✧ однорідність;
- ✧ високий ступінь структуризації знань;
- ✧ відповідність принципам уявлення знанні людиною в довготривалій пам'яті;
- ✧ інтеграція декларативних і процедурних знань.

До ***недоліків*** моделі слід віднести, мабуть, деякі утруднення, що виникають при обміні великими порціями даних між двома об'єктами. Власне, фрейм в класичному вигляді, описаний М. Мінськи, взагалі не припускає ніякого обміну.

**Недоліки**:
складність управління виводом;
низька ефективність його процедур.

4.2.2.5 Семантичні мережі для представлення знань

Найбільш загальний спосіб представлення знань, при якому ПрО розглядається як сукупність об'єктів і відношень, що зв'язують їх, реалізований в ***мережевій моделі знань***. Як носій знань в цій моделі виступає ***семантична мережа***, вершини якої відповідають об'єктам (поняттям), а дуги – відношенням між поняттями. Крім того, і вершинам, і дугам привласнюються імена (ідентифікатори) і описи, що характеризують семантику об'єктів і стосунків ПрО.

**Семантичні мережі** – найбільш потужний клас математичних моделей для уявлення знань про ПрО, один з найважливіших напрямів ШІ. У загальному випадку під семантичною мережею розуміється структура

$$S = (0, R), \tag{4.8}$$

де $Про(Oi,\ i = \overline{1,n})$ – множина об'єктів ПРО; $R = (Rj, j = \overline{1,k})$ – множина стосунків між об'єктами.

У літературі виділено близько 200 типів семантичних стосунків. Всі вони представлені в природній мові і мають різні властивості як з математичної, так і з лінгвістичної точок зору. Розробка універсального представлення семантичної мережі при такій кількості типів відношень є складним і трудомістким завданням. До того ж оперувати подібною універсальною моделлю було б скрутно.

При вирішенні конкретних завдань значущими є не всі типи відношень, а їх підмножина. Тому моделі семантичних мереж зазвичай розробляються як проблемно-орієнтовані. У таких моделях використовують обмежене число типів відношень (зазвичай не більше 20–30).

Термін "семантична (смислова) мережа" (або "концепція слів") виник в 1968-69 рр. в роботах Р. Куїлліана[1]. Він має на увазі цілий клас загальних підходів, для яких характерне використання графічних схем з вузлами, сполученими дугами (графів). Особливість полягає в тому, що в теорії семантичних мереж розглядаються не тільки графи-дерева, але циклічні і повнозв'язні графи. Крім того окремо і особливо досліджуються можливі типи відношень (традиційні: АКО – a kind of, ISA, has part і ряд нових).

Спочатку семантичні мережі були розроблені для аналізу природних мов і побудови психологічних моделей людської пам'яті (завдання автоматичного перекладу – положення Аристотеля про те, що "людина мислить на мові", завдання підбору синонімів до заданого слова і ін.). На цьому етапі

[1] Ross Quillian – соціолог, працював в області ШІ, відкривач "семантичної пам'яті".

вважалося, що в пропозиції є якась "центральна тема", "розкрутивши" яку, машина може "зрозуміти" сенс (семантику) пропозиції (рис. 4.44).

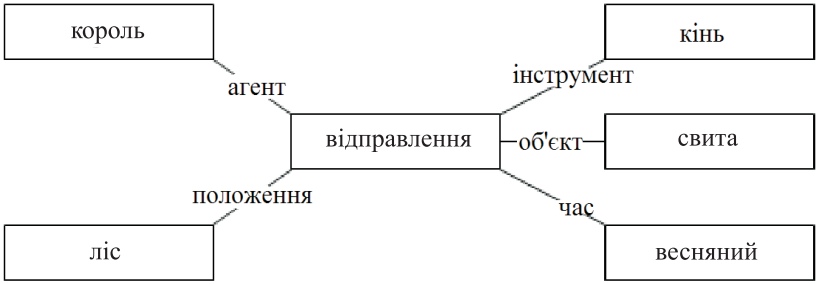



Рисунок 4.44. "Весняним ранком король узяв слуг і відправився в ліс на кінну прогулянку"

***Семантика*** – наука, що встановлює стосунки між символами і об'єктами, які вони позначають, тобто наука, що визначає сенс знаків.

***Семантична мережа*** – це орієнтований граф, вершини якого – поняття, а дуги – відношення між ними.

Експериментальну перевірку теоретичних положень про пристрій людської пам'яті і організації логічного виведення Р. Куїлліан провів спільно з психологом А. Колінз (Allan Collins) в 1969-70 рр. на основі ієрархічної мережі. Для такої мережі характерні стосунки вигляду клас – елемент класу; властивість – значення; приклад елементу класу (рис. 4.45 – 4.48).




Рисунок 4.45. Ієрархічна мережа з використанням відношення "є"

Подальший розвиток моделі пов'язаний з іменами Р. Симмонса (Robert Simmons), С. Шапіро (Stuart C. Shapiro, звернув увагу на наявність "відношень різного роду"), Хендрікса (Hendrix, ввів поняття "Розділеної

мережі/partitioned network") і Шенка (Roger C. Schank, ввів поняття "Концептуальної залежності" і "концептуальних примітивів").

У мережах Р. Симмонса немає строгої ієрархії, оскільки він вирішував дещо іншу задачу. Відповідно отримана і класифікація вузлів: Агент, Реципієнт (To receive (англ.) – отримувати.), Об'єкт.

Давати

Агент: { Яго, хлопчик, Отелло}

Реципієнт: { Дездемона, Родріго}

Об'єкт: { яблуко, квіти, хустка}

Вузли в семантичній мережі зазвичай відповідають об'єктам, концепціям, подіям або поняттям. Дуги можуть бути визначені різними методами, залежними від методу представлення знань. Будь-який фрагмент мережі, наприклад одна вершина, дві вершини і дуги, що сполучають їх, називають підмережею. Логічний вивід (пошук рішення) на семантичній мережі полягає якраз в тому, щоб знайти або сконструювати підмережу, що задовольняє деяким умовам. Для того, щоб формалізувати цей процес, вводять типізацію семантичних мереж і потім розробляють методи вирішення для мереж конкретного вигляду (на основі аналізу математичних властивостей відношень, що входять в мережу).

Мережі конкретного вигляду в сукупності з методами виводу на них іноді називають іменами першовідкривачів.

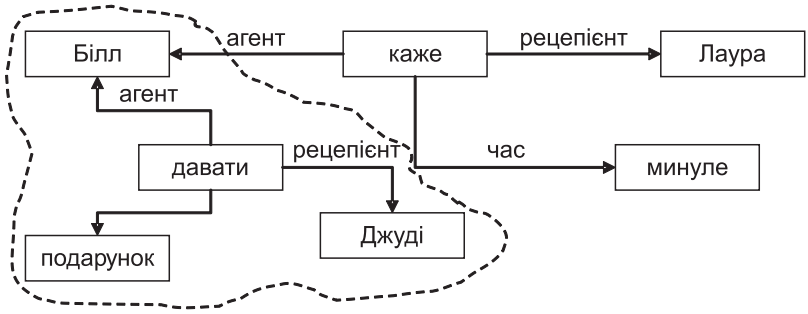



Рисунок 4.46. Уявлення семантичною мережею складної пропозиції: "Біл сказав Лаурі, що (він) дав Джуді подарунок"; підмережа "Біл дає Джуді подарунок".

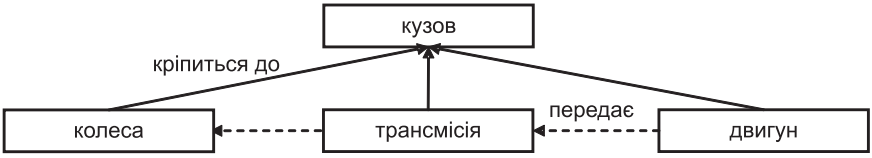



Рисунок 4.47. Семантична мережа "автомобіль" і підмережа "двигун передає зусилля колесам"

Мережі різного вигляду виходять і залежно від того, які обмеження накладають на вершини і дуги. Якщо вершини мережі не володіють внутрішньою структурою, то такі мережі називають *простими мережами*, якщо вершини володіють внутрішньою структурою, то такі мережі називають *ієрархічними*. *Динамічні семантичні мережі (сценарії)* – мережі з подіями. Одна з основних відмінностей ієрархічної мережі від простої мережі – в можливості розділяти мережу на підмережі і встановлювати відношення не тільки між вершинами, але і між підмережами. Різні підмережі, що існують в мережі, можуть бути впорядковані у вигляді дерева підмереж, вершини якого – підмережі, а дуги – відношення видимості. Поняття підмережі аналогічно поняттю дужок в математичній нотації, поняття видимості – поняттю змінних, що знаходяться усередині і зовні дужок.

Можна ввести декілька класифікацій семантичних мереж.

По кількості типів стосунків:

- ✧ однорідні (з єдиним типом відношень);
- ✧ неоднорідні (з різними типами відношень).

По типах стосунків:

- ✧ бінарні (у яких відношення зв'язують два об'єкти);
- ✧ n-арні (у яких є спеціальні відношення, що зв'язують більше двох понять, наприклад "трикутник").

Найчастіше в семантичних мережах використовуються такі відношення:

- ✧ зв'язки типу "частина – ціле" ("клас – підклас", "елемент – множина" і т. п.);
- ✧ функціональні зв'язки (визначувані зазвичай дієсловами "проводить", "впливає"...);
- ✧ кількісні (більше, менше, рівно ...);
- ✧ просторові (далеко від, близько від, за, під, над...);
- ✧ тимчасові (раніше, пізніше, в течію...);
- ✧ атрибутивні зв'язки ( мати властивість, мати значення ...);
- ✧ логічні зв'язки (і, або, не) і ін.

Семантичні мережі є одним з найвдаліших методів представлення знань про наочну область з добре встановленою таксономією[2] .

Як приклад може бути приведена частина семантичної мережі, що відноситься до поняття "фрукти" (рис. 4.48).

[2] Таксономія (від греч. taxis – розташування, лад, порядок і nomos – закон), теорія класифікації і систематизації складноорганізованих областей дійсності, що мають звичайна ієрархічна будова (органічний мир, об'єкти географії, геології, мовознавства, етнографії і т. д.)

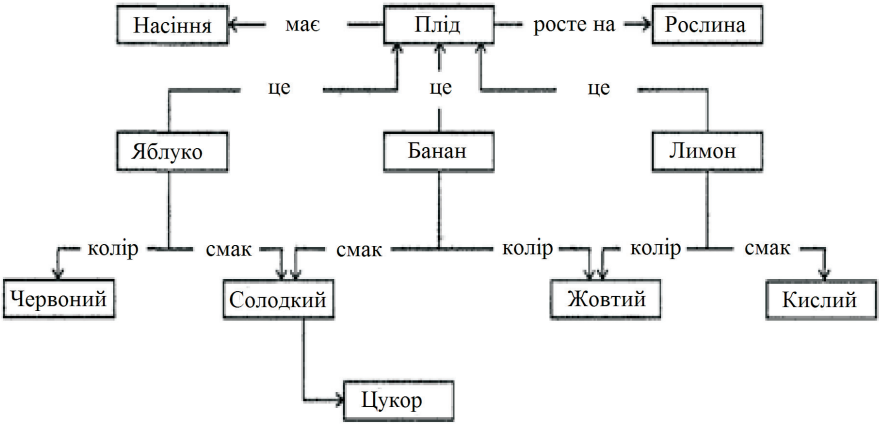



Рис. 4.48. Приклад семантичної мережі

Розглянемо практичні додатки моделі.

1. Джейм Карбонел (Jaime Carbonell) створив вельми специфічну експертно-повчальну систему SCHOLAR, призначену для вивчення географії Південної Америки. З подібних систем надалі виросли сучасні ГІС – географічні інформаційні системи.

2. Хайдерн створив PLNLP (Programming Language for Natural Language Processing) – мова програмування для обробки природної мови. Ця мова використовується для роботи з великими граматиками.

3. Створена спеціальна мережева мова NET.

4. Створені широко відомі експертні системи, що використовують семантичні мережі як мову представлення знань, – PROSPECTOR, CASNET, TORUS.

Основний ***недолік*** моделі – складність пошуку виводу; на семантичній мережі хороші математичні моделі розроблені тільки для декількох типів відношень.

Оскільки фактично мережева модель об'єднує множину методів уявлення ПрО за допомогою мереж, зіставлення даної моделі з іншими способами представлення знань майже неможливе. Очевидні ***переваги*** мережевої моделі полягають в її високій спільності, наочності відображення системи знань про ПрО, а також легкості розуміння подібного уявлення. В той же час в семантичній мережі має місце змішення груп знань, що відносяться до абсолютно різних ситуацій при призначенні дуг між вершинами, що ускладнює інтерпретацію знань. Інша проблема, властива мережевій моделі, полягає в трудності уніфікації процедур виводу і механізмів управління виводами на мережі.

Перспективним напрямом підвищення ефективності мережевого уявлення, що розвивається в даний час, є використання онтології і паралельних виводів на семантичних мережах.

Різновидом мережевої моделі є ***онтологія***. Швидкий розвиток онтологічного підходу останніми роками обумовлений розповсюдженням Intemet-технологій, що використовують онтологічні моделі.

**Онтологія** – це формально представлені на базі концептуалізації знання про ПРО.

Найпоширенішим на даний момент є визначення, згідно якому *онтологія є точна (що вимірюється формальними засобами) специфікація концептуалізації.* З цієї точки зору кожна БЗ, система, заснована на знаннях, або програмний агент явно або неявно базуються на деякій концептуалізації. Множина понять і відношень між ними відбивається в словнику. Таким чином, вважається, що основу онтології складає множина представлених в ній термінів.

На метарівні онтологія є різновидом мережевої моделі знань про ПРО. Ця модель може бути статичною або динамічною. У другому випадку говорять про онтологію як про модель світу, яка може представляти стани модельованою ПРО в часі.

Web-онтологією називають онтологію, яка або доступна на одному з web-узлов Internet, або використовується в рамках корпоративного порталу.

Визначення онтології як формального уявлення ПРО, побудованого на базі концептуалізації, припускає виділення її трьох взаємозв'язаних компонентів: таксономії термінів, описів сенсу термінів, а також правил їх використання і обробки. Таким чином, модель онтології ПрО задає трійка:

$$O = (X,r,ф), \tag{4.9}$$

де $X$ – кінцева множина концептів (понять, термінів) ПРО, яку представляє онтологія;

$R$ – кінцева множина відношень між концептами;

$Ф$ – кінцева множина функцій інтерпретації, заданих на концептах і (або) відношеннях.

Як бачимо, модель (4.5) є різновидом мережевої моделі знань.

До основних завдань, що вирішуються за допомогою онтології, відносяться створення і використання БЗ, організація семантичного пошуку в БД, БЗ і Internet, реалізація механізмів міркувань для різних класів НАС, представлення сенсу в метаданих про ІР і ін.

***Об'єктно-орієнтована модель знань*** отримала широке застосування в сучасних технологіях проектування різноманітних програмних і інформаційних систем.

Застосування на практиці того або іншого способу формалізації обумовлюється специфікою завдання, для вирішення якої планується використовувати БЗ. На думку фахівців найбільш перспективні ***змішані***

або ***комплексні моделі***, інтегруючі переваги розглянутих вище базових моделей уявлення ПРО. Прикладом комплексної моделі може служити уявлення, що використовує для опису знань ленемы.

4.2.2.6 Нові моделі представлення знань

Основні моделі представлення знань, перелік цінних завдань і методи їх рішення сформувалися, в основному, на початок 80-х рр. минулого століття. У розвитку теорії і практики ШІ наступив деякий застій, пов'язаний з тим, що подальше вдосконалення традиційних моделей не давало якісного стрибка.

На початку 90-х рр. філософське розуміння завдань і методів ШІ дещо змінилося, що відразу ж привело до появи ряду нових моделей: критерійної мови опису вибору, нейронних мереж, ряду імовірнісних моделей. Слід сказати, що далеко не всі фахівці з ШІ готові визнати їх повноцінними моделями для представлення знань, хоч би внаслідок того, що ці моделі не здатні обгрунтовувати свої висновки. Крім того, "новими" їх можна назвати з великою натяжкою; критерійній мові, наприклад, декілька сотень років.

Отже, новий погляд на поняття "Модель для представлення знань" полягає в тому, що мається на увазі деяка модель, що дозволяє отримувати "хороші рішення". При цьому не важливо, чи моделює вона спосіб мислення людини і чи здатна пояснювати свої виводи.

**Критерійні методи**

На практиці застосування інтелектуальних, нечітких (насамперед) моделей часто необхідне для оцінки альтернатив і вибору якнайкращої в деякому розумінні. Необхідність їх застосування обумовлена великою кількістю альтернатив, параметрів, що враховуються, вимогою "програти декілька сценаріїв розвитку подій"; наприклад, "сприятливий, звичайний і якнайгірший".

Завдання критерійного ухвалення рішень фактично є завданнями лінійного або нелінійного математичного програмування. Окремим випадком будь-яке з цих завдань може опинитися, наприклад, завдання цілочисельного програмування, коли потрібно знайти таку якнайкращу альтернативу, оцінки x1 і x2 якої – цілі числа.

Нижче приведена загальна схема (рис. 4.49) деяких найбільш часто вживаних методів.

**Імовірнісні методи**

Широкий клас підходів використовує як міру близькості альтернатив імовірнісні оцінки. В даний час методи, що дозволяють оцінити вірогідні результати того або іншого рішення, його правильність, досліджують в рамках так званої теорії ризиків. Теорія носить прикладний харак-

тер, тому зазвичай говорять про ризики в наочній області, наприклад в природокористуванні. У математичному плані це звичайна теорія вірогідності, причому в її найпростіших проявах. Суть теорії ризиків складають не математичні, а алгоритмічні процедури, пов'язані з особливостями вимірювання та імовірнісної обробки даних в конкретній наочній області, а також інтерпретації отриманих результатів.

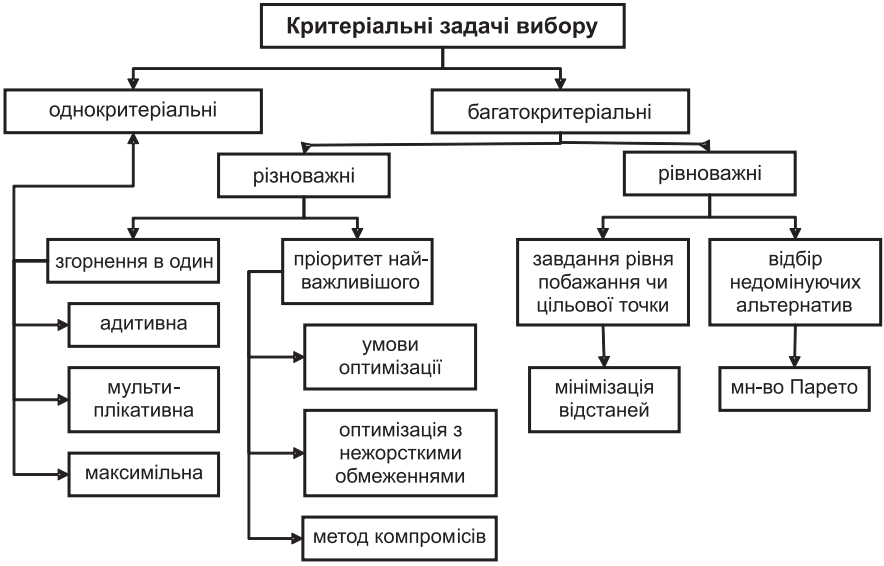



Рисунок 4.49. Деякі найбільш важливі критерійні методи.

**Нейронні мережі**

Одним з напрямів, що найактивніше розвивається сьогодні в ШІ є нейронні мережі – модель, що є в деякому розумінні гібридом критерійних, імовірнісних і логічних методів. (Нейронну мережу за великого бажання можна створити на основі правил продукцій. Інша справа, що продукційна система не дозволить створити достатньо велику мережу.).

ЕОМ VI покоління, що розроблялася в 80-х рр., японці назвали "нейрокомп'ютер" або "трансп'ютер", кажучи простіше – паралельна ЕОМ. До цього ж питання відносяться дослідження по "пороговій логіці", "обчислювальній геометрії" і ін.

Серед значущих результатів, отриманих СРСР, – алгоритм "Кора", що моделює діяльність людського мозку при розпізнаванні образів (М. Бонгард, 60-і рр.).

Розвиток теорії почався з робіт У. Макклоха (W. McCulloch) і У. Піттса (W. Pitts), що розглядають функціонування головного мозку в 1943 р.). У

1957 р. психолог Френк Розенблатт (Корнелл, США) створив "персептрон" – електромеханічний пристрій, що моделює око равлика і його взаємодію з мозком. Модель розрізняла букви алфавіту, але опинилася чутлива до написання букви А.

У 1969 р. вийшла класична робота "Персептрони", в якій М. Мінськи і С. Пейперт (М.І.Т.) довели ряд теорем про обмежені можливості розпізнавання з використанням персептрона. Фактично вони ввели в ШІ вимогу строгого математичного обгрунтування використовуваних методів. Далі почала розвиватися сучасна теорія нейромереж, персептрони в класичному вигляді вийшли із застосування.

Теорія нейронних мереж включають широкий круг питань з різних галузей науки: біофізики, математики, інформатики, схемотехніки і технології. Тому поняття "Нейронні мережі" детальне визначити складно.

*Штучні нейронні мережі (НС)* – сукупність моделей біологічних нейронних мереж – є мережею елементів – штучних нейронів – зв'язаних між собою синаптическими з'єднаннями. Мережа обробляє вхідну інформацію і в процесі зміни свого стану в часі формує сукупність вихідних сигналів.

Робота мережі полягає в перетворенні вхідних сигналів в часі, внаслідок чого міняється внутрішній стан мережі і формуються вихідні дії. Зазвичай НС оперує цифровими, а не символьними величинами.

Більшість моделей НС вимагають навчання. У загальному випадку, навчання – такий вибір параметрів мережі, при якому мережа краще всього справляється з поставленою проблемою. Навчання – це завдання багатовимірної оптимізації, і для її вирішення існує множина алгоритмів.

*Штучні нейронні мережі* – набір математичних і алгоритмічних методів для вирішення широкого круга завдань. Виділимо характерні риси штучних нейромереж як універсального інструменту для вирішення завдань:

1. НС дають можливість краще зрозуміти організацію нервової системи людини і тварин на середніх рівнях: пам'ять, обробка сенсорної інформації, моторика.

2. НС – засіб обробки інформації:

а) гнучка модель для нелінійної апроксимації багатовимірних функцій;
б) засіб прогнозування в часі для процесів, залежних від багатьох змінних;
в) класифікатор по багатьом ознакам, що дає розбиття вхідного простору на області;
г) засіб розпізнавання образів;
д) інструмент для пошуку по асоціаціях;
е) модель для пошуку закономірностей в масивах даних.

3. НС вільні від обмежень звичайних комп'ютерів завдяки паралельній обробці і сильній зв'язаності нейронів.

4. У перспективі НС повинні допомогти зрозуміти принципи, на яких побудовані вищі функції нервової системи: свідомість, емоції, мислення.

Істотну частину в теорії нейронних мереж займають біофізичні проблеми. Для побудови адекватної математичної моделі необхідно детально вивчити роботу біологічних нервових клітин і мереж з погляду хімії, фізики, теорії інформації і синергетики.

Швидкодія сучасних ЕОМ складає близько 100 Mflops (flops * операція з плаваючою комою в секунду).У мозку міститься приблизно $10^{11}$ нейронів. Час проходження одного нервового імпульсу близько 1 мс, і можна вважати, що продуктивність одного нейрона близько 10 flops. Еквівалентна швидкодія мозку складе $10^{11} * 10 = 10^{12}$ flops. Якщо розглянути завдання, що вирішуються мозком, і підрахувати необхідну кількість операцій для їх вирішення на звичайних ЕОМ, то отримаємо оцінку швидкодії до $10^{12}..10^{14}$ flops. Різниця в продуктивності між звичайною ЕОМ і мозком – 4..6 порядків!

Багато в чому цей виграш обумовлений паралельністю обробки інформації в мозку. Отже, для підвищення продуктивності ЕОМ необхідно перейти від принципів фон-Неймана до паралельної обробки інформації. Проте, паралельні комп'ютери поки не набули поширення з кількох причин:

1. Проблеми міжз'єднань. Кожен процесор в паралельній системі пов'язаний з великою кількістю інших. Кількість зв'язків займає набагато більший об'єм, чим самі процесори. Така щільність зв'язків не реалізується в звичайних інтегральних схемах.

2. Тривимірність структури зв'язків між процесорами. Існують різні типи зв'язності процесорів в паралельній системі. Зазвичай потрібні тривимірні зв'язки. Технологічно такі зв'язки теж поки нездійснимі.

3. Складність програмування. Поки не створено єдиних способів програмування паралельних ЕОМ і засобів для написання програм.

Не дивлячись на перспективність паралельних ЕОМ і, зокрема, нейронних мереж, для їх створення немає елементної бази. Тому, замість моделювання НС на паралельних машинах, велика частина досліджень проводиться двома способами:

1) моделювання НС на звичайних послідовних ЕОМ;
2) створення спеціалізованих нейроплат і нейропроцессоров для прискорення роботи ЕОМ з нейронними мережами.

Нейронні мережі перевершують послідовні машини у вирішенні тих же завдань, в яких машину перевершує людина. Завдання, що вимагають великого об'єму обчислень або високої точності краще виконують звичайні ЕОМ.

До завдань, успішно вирішуваним НС на даному етапі їх розвитку відносяться:

- ✧ розпізнавання зорових, слухових образів; величезна сфера застосування: від розпізнавання тексту і цілей на екрані радара до систем голосового управління;

- ✧ асоціативний пошук інформації і створення асоціативних моделей; синтез мови; формування природної мови;
- ✧ формування моделей і різних нелінійних і важко описуваних математично систем, прогнозування розвитку цих систем в часі:
- ✧ застосування на виробництві; прогнозування розвитку циклонів і інших природних процесів, прогнозування змін курсів валют і інших фінансових процесів;
- ✧ системи управління і регулювання з прогнозом; управління роботами, іншими складними пристроями;
- ✧ різноманітні кінцеві автомати: системи масового обслуговування і комутації, телекомунікаційні системи;
- ✧ ухвалення рішень і діагноста, що виключають логічний вивід; особливо в областях, де відсутні чіткі математичні моделі: у медицині, криміналістиці, фінансовій сфері.

Унікальна властивість нейромереж – універсальність.

**Біологічний нейрон**

Моделювання біологічних нейронних мереж обгрунтовано і перспективно. Але для дослідження НС необхідно мати математичну модель біологічного нейрона і біологічної нейронної мережі.

Центральна нервова система має клітинну будову. Одиниця – нервова клітина, нейрон. Нейрон має наступні основні властивості:

1. Бере участь в обміні речовин і розсіює енергію. Міняє внутрішній стан з часом, реагує на вхідні сигнали і формує вихідні дії і тому є активною динамічною системою.

2. Має множина синапсів – контактів для передачі інформації.

3. Нейрон взаємодіє шляхом обміну електрохімічними сигналами двох видів: електротонічними (із загасанням) і нервовими імпульсами (спайками), що розповсюджуються без загасання.

Існують два підходи до створення штучних нейронних мереж. *Інформаційний підхід*: байдуже, які механізми лежать в основі роботи штучних нейронних мереж, важливо лише, щоб при вирішенні завдань інформаційні процеси в НС були подібні біологічним. *Біологічний*: при моделюванні важлива повна біоподібність, і необхідно детально вивчати роботу біологічного нейрона.

Крупні роботи в дослідженнях біологічних нейронних мереж належать Ендрю Хакслі, Алану Ходжкину, Бернарду Катцу, Джону Екклзу, Стівену Куффлеру.

Біологічний нейрон містить наступні структурні одиниці (рис. 4.50):

*Тіло клітки (т) – сома:* містить ядро (я)**,** мітохондрії (забезпечують клітку енергією), іншу органелу, що підтримує життєдіяльність клітки.

*Дендріти (д)* – вхідні волокна, збирають інформацію від інших нейронів. Активність в дендритах міняється плавно. Довжина їх звичайне не більше 1 мм.

*Мембрана* – підтримує постійний склад цитоплазми усередині клітки, забезпечує проведення нервових імпульсів.

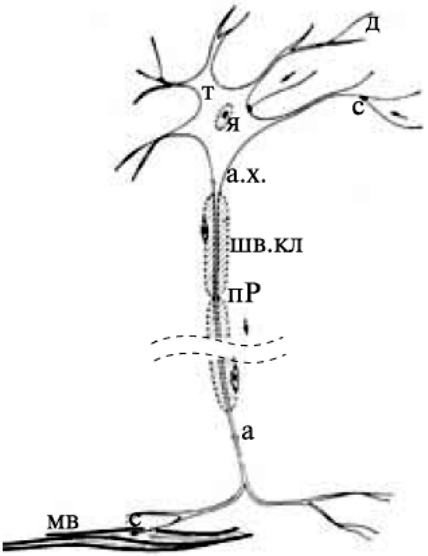



Рисунок 4.50. Біологічний нейрон.

*Цитоплазма* – внутрішнє середовище клітки. Відрізняється концентрацією іонів K+, Na+, Ca++ і інших речовин в порівнянні з позаклітинним середовищем.

*Аксон (а)*, один або жоден у кожної клітки, – довге, іноді більше метра, вихідного нервового волокна клітки. Імпульс генерується в аксонном горбику (**а.х.**). Аксон забезпечує проведення імпульсу і передачу дії на інші нейрони або м'язові волокна **(мв)**. Ближче до кінця аксон часто гілкується.

*Сі' напс (с)* – місце контакту нервових волокон – передає збудження від клітки до клітки. Передача через синапс майже завжди однонаправлена. Розрізняють пресинаптичні і постсинаптичні клітки – по напряму передачі імпульсу.

*Шванновські клітки* **(шв.кл).** Специфічні клітки, що майже цілком складаються з мієліну, органічної ізолюючої речовини. Щільно "обмотують" нервове волокно 250 шарами мієліну. Неізольовані місця нервового волокна між шванновскими клітками називаються *перехопленнями Ранвье* **(пР)**. За рахунок мієлінової ізоляції швидкість розповсюдження нервових імпульсів зростає в 5*10 разів і зменшуються витрати енергії на проведення імпульсів. Мієлінізіровані волокна зустрічаються тільки у вищих тварин.

У центральній нервовій системі людини налічується від 100 до 1000 типів нервових клітин, в залежності вибраного ступеня деталізації. Во-

ни відрізняються картиною дендритів, наявністю і довжиною аксона і розподілом синапсів біля клітки.

Клітки сильно зв'язані між собою. У нейрона може бути більше 1000 синапсів. Близькі по функціях клітки утворюють скупчення, кулясті або паралельні шаруваті. У мозку виділені сотні скупчень. Кора головного мозку – теж скупчення. Товщина кори – 2 мм, площа – біля квадратного фута.

У медицині нейроном (рис. 4.48) називається клітка кори головного мозку. Її відмітна особливість в тому, що, збуджуючись під впливом електричного сигналу, що приходить по спеціальних зв'язках (від інших нейронів), вона генерує власний сигнал, який передає таким, що оточує і так далі Через деякий час наступає процес гальмування – електрична активність нейронів, що недавно збудилися, утихає, тоді як "відпочилі" нейрони знов збуджуються.

Нейронну мережу слід розглядати, перш за все, як систему наближених математичних рівнянь. Вирішуючи конкретну задачу за допомогою мережі, або конструюючи нову мережу, слід думати саме про математичне обгрунтування отриманої системи, а вже ніяк не сподіватися на "здатність самоорганізації мережі".

З математичної точки зору нейрон – це нелінійний перетворювач (рис. 4.51), функція, на вхід якої подано деяке значення (збуджуючий сигнал), а з виходу знято інше значення (величина збудження), пов'язане з вхідним деяким складним чином або навіть імовірнісний:

$$Y(X,W) = w_0 + \sum f(w_i \times x_i), \quad i = 1,2,...,n \tag{4.10}$$

тут *xi* – значення інформаційних входів, *wi* – ваги, причому *w0* – зсув.

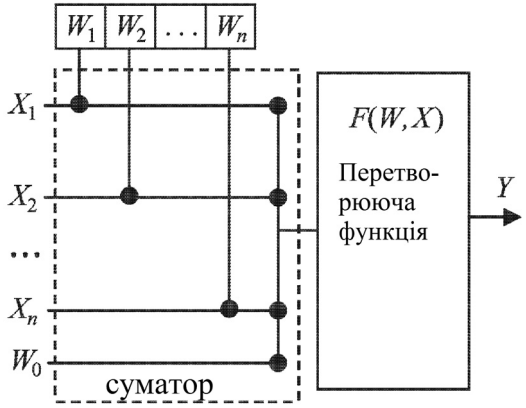



Рисунок 4.51. Нелінійний перетворювач.

Функціонує ця схема таким чином. У суматорі обчислюється збудження, отримане нейроном $\left(w_0 + \sum f\left(w_i \times x_i\right)\right)$. На основі цієї величини пере-

творююча функція обчислює збудження самого нейрона – *Y*. Конкретний вид перетворюючої функції підбирається з урахуванням специфіки вирішуваного завдання. Якщо вибрати таку функцію, що *Y* набуватиме тільки значень 0 або 1, то, очевидно, отримаємо правила продукцій, тільки в декілька незвичного запису.

**Штучні нейронні мережі**

Біологічний нейрон – складна система, математична модель якого до цих пір повністю не побудована. Введена множина моделей, що розрізняються обчислювальною складністю і схожістю з реальним нейроном. Одна з найважливіших – *формальний нейрон* (рис. 4.52.). Не дивлячись на простоту ФН, мережі, побудовані з таких нейронів, можуть сформувати довільну багатовимірну функцію на виході.

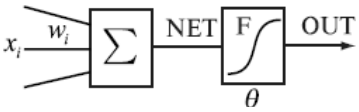



Рисунок 4.52. Формальний нейрон.

Нейрон складається із зваженого суматора і нелінійного елементу. Функціонування нейрона визначається формулами:

$$NET = \sum_i w_i x_i$$

$$OUT = F(NET - \theta) \qquad (4.11)$$

де $x_i$ – вхідні сигнали, сукупність всіх вхідних сигналів нейрона утворює вектор x;

$w_i$ – вагові коефіцієнти, сукупність вагових коефіцієнтів утворює вектор вагів w;

NET – зважена сума вхідних сигналів, значення NET передається на нелінійний елемент;

i – пороговий рівень даного нейрона;

F – нелінійна функція, звана функцією активації.

Нейрон має декілька вхідних сигналів **x** і один вихідний сигнал OUT. Параметрами нейрона, що визначають його роботу, є: вектор ваг **w**, пороговий рівень *i* і вид функції активації **F**.

Модель формального нейрона не є біоподібною і швидше схожа на математичну абстракцію, чим на живий нейрон. Тим дивовижніше виявляється різноманіття завдань, що вирішуються за допомогою таких нейронів і універсальність отримуваних алгоритмів.

**Багатошаровий перцептрон**

Практично корисні видів перетворюючої функції не так багато. При об'єднанні множини нейронів в мережу розрізняють "шари" (рис. 4.53), зв'язок між якими може бути однонаправленим або з "зворотним розповсюдженням". Очевидна аналогія з фронтом "готових продукцій" при управлінні перебором правил в продукційній моделі.

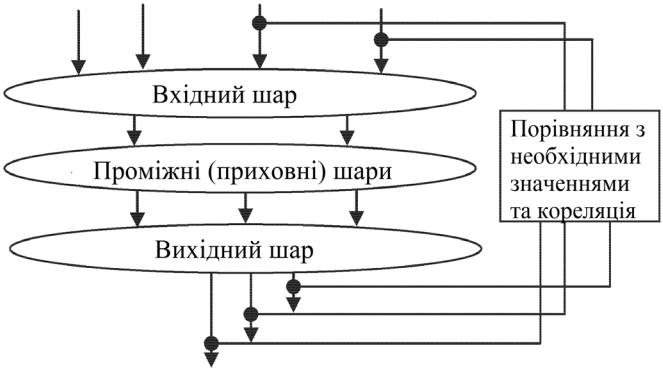



Рисунок 4.53. Багатошарова нейронна мережа.

Під навчанням мережі розуміється ітеративний процес обчислення вагів *wi*, що дозволяють набувати вихідних значень на мережі із заданою точністю.

Формальні нейрони можуть об'єднуватися в мережі різним чином. Найпоширенішим видом мережі став *багатошаровий перцептрон* (рис. 4.54).

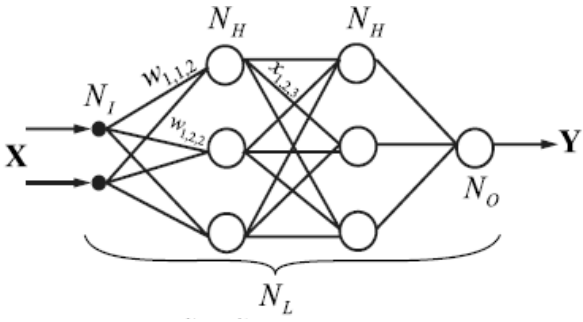



Рисунок 4.54. Багатошаровий перцептрон

Мережа складається з довільної кількості шарів нейронів. Нейрони кожного шару з'єднуються з нейронами попереднього і подальшого шарів за принципом "кожен з кожним". Перший шар (зліва) називається *сенсорним* або *вхідним*, внутрішні шари називаються *прихованими* або

*асоціативними*, останній (найправіший, на малюнку складається з одного нейрона) – *вихідним* або *результативним*. Кількість нейронів в шарах може бути довільною. Зазвичай у всіх прихованих шарах однакова кількість нейронів.

Позначимо кількість шарів і нейронів в шарі. Вхідний шар: ***N1*** нейронів; ***NH*** нейронів в кожному прихованому шарі; ***N0*** вихідних нейронів. **X –** вектор вхідних сигнали мережі, **Y –** вектор вихідних сигналів.

Існує плутанина з підрахунком кількості шарів в мережі. Вхідний шар не виконує ніяких обчислень, а лише розподіляє вхідні сигнали, тому іноді його рахують, іноді – ні. Позначимо через ***NL*** повну кількість шарів в мережі, рахуючи вхідний.

Робота багатошарового перцептрона (МСП) описується формулами:

$$\begin{gathered} NET_{jl} = \sum_i w_{0l} x_o; \\ OUT_{jl} = F(NET_{jl} - \theta_{jl}) \\ x_{ij\,(l+1)} = OUT_{il} \end{gathered} \tag{4.12}$$

де індексом і завжди позначатимемо номер входу;

j – номер нейрона в шарі;

l – номер шаруючи;

xijl – і-й вхідний сигнал j-го нейрона в шарі l;

wijl – ваговий коефіцієнт і-го входу нейрона номер j в шарі l;

NETjl – сигнал NET j-го нейрона в шарі l;

OUTjl – вихідний сигнал нейрона;

ијl – пороговий рівень нейрона j в шарі l;

Введемо позначення: ***wjl*** – вектор-стовпець вагів для всіх входів нейрона *j* в шарі *l*; ***Wl*** – матриця вагів всіх нейронів в шаруючи *l*. У стовпцях матриці розташовані вектора ***wjl***. Аналогічно ***xjl*** – вхідний вектор-стовпець шаруючи *l*.

У багатошаровому перцептроне немає зворотних зв'язків. Такі моделі називаються *мережами прямого розповсюдження*. Вони не володіють внутрішнім станом і не дозволяють без додаткових прийомів моделювати розвиток динамічних систем.

Практичне застосування мають також одношарові мережі, наприклад, повнозв'язна мережа Хопфілда, в якій всі нейрони зв'язані один з одним.

Дослідження по нейроінформатиці є сьогодні одними з фаворитів в області ШІ. Застосовуютб нейронні мережі у економічних, виробничих системах. Нейронні мережі "добре" реалізуються апаратний, у вигляді виконаних за спеціальним замовленням мікросхем. Такі мікросхеми є, наприклад, "мозком" всесвітньо відомих систем ППО, де вирішують задачу розпізнавання "непізнаного об'єкту на екрані радара" на відстанях зверху 300 км.

Одне з головний перешкод на шляху подальшого широкого впровадження нейронних мереж, реалізованих у вигляді спеціальних мікросхем, в реальні системи управління полягає в тому, що принципові можливості нейронних мереж математично не доведені. З цієї причини військові, наприклад, не можуть бути повністю упевнені в надійності такої системи наведення.

Фактично теорія, створена в нейроінформатике, об'єднує елементи всієї попередньої теорії ШІ.

Сьогодні існує множина технологій автоматизації роботи із знаннями: технологія концептуального програмування, технології автоматичного розпізнавання образів (детальніше OCR-системы вивчаються в курсі "Комп'ютерна графіка"), **гіпертекстова** інформаційна технологія і інші технології (детальніше ГІТ і інші технології вивчаються в курсах "WEB-програмирование", "WEB-дизайн, як основа INTERNET").

### 4.2.3 Характеристика й призначення інформаційної технології експертних систем

#### 4.2.3.1 Поняття та призначення експертних систем

Задумайтеся над таким питанням: "При виконанні яких умов комп'ютерну програму можна назвати експертом"?

Цілком логічно зажадати, щоб така програма мала знання. Просто здатність виконувати деякий алгоритм, наприклад робити аналіз списку елементів на наявність якої-небудь властивості, явно не відповідає цій вимозі. Це однаково, що дати першому випадковому перехожому список питань і відповідей, і очікувати від нього успішного виконання пошуку й усунення несправностей у системах певного типу. Раніше або пізніше, але він обов'язково зіштовхнеться із ситуацією, не передбаченою в цьому списку.

Знання, якими володіє програма, повинні бути сконцентровані на певну предметну область. Випадковий набір імен, дат і місць подій, сентенцій із класиків тощо – це аж ніяк не ті знання, які можуть послужити основою для програми, що претендує на здатність виконати експертний аналіз. Знання припускають певну організацію й інтеграцію – тобто окремі відомості повинні співвідноситися між собою і утворювати ланцюжок, у якому одна ланка "тягне" за собою наступне.

І, нарешті, із цих знань повинне безпосередньо випливати рішення проблем. Просто продемонструвати свої знання, що стосуються, наприклад технічного обслуговування комп'ютерів, – це далеко не те ж саме, що налагодити комп'ютер. Аналогічно одержати доступ до оперативної документації – це зовсім не те ж саме, що мати у своєму розпорядженні фахівця (або програму), здатного впоратися з виниклими проблемами.

Тепер спробуємо підсумувати ці міркування в наступному формальному визначенні експертної системи (ЕС).

У сучасному розумінні **експертна система** – це вузькоспеціалізований програмний комплекс, що дозволяє або дуже швидко приймати стандартні рішення (найчастіше по управлінню технічними об'єктами), або на основі тривалого діалогу з користувачем допомагати у виборі деякого рішення (шляхом оцінки варіантів, пропонованих користувачем, і їх корекції з урахуванням множинаі чинників, можливо невідомих користувачеві, а також пропозиції власних варіантів рішення і їх корекції відповідно до цілей користувача).

Перші такі програми з'явилися у кінці 60-х рр. минулого століття і призначалися для створення штучного "надрозуму" в деякій предметній області. Наприклад, програма MYCIN мислилася як "наддоктор" в області гематології, CASNET – в області хвороб очей; різні модифікації програми ELIZA претендували на роль "суперпсихолога" і т. д.

У 60-70-х рр. більшість ЕС мислилися як деякий програмний механізм, що дозволяє акумулювати досвід кращих людей-експертів (у деякій предметній області), а потім консультувати менш досвідчених користувачів. Класичні бази даних (основна теорія яких до цього моменту вже сформувалася) не підходили на цю роль, тому що їх мова запитів не була досить гнучкою, щоб описати усе різноманіття запитів реальної мови (світу); крім того мови запитів до БД не допускали рекурсивності, тобто. можливості виклику запитом самого себе. У рамках такого підходу сформувалися основні риси сучасної технології застосування і побудови ЕС.

Рішення прикладної задачі за допомогою ЕС – це завжди цілеспрямований ітеративний процес, що має на меті вироблення рішення, найкращого з деякої точки зору.

Рішення задачі без використання ЕС (іншими програмними засобами) найчастіше є разовим розрахунком навіть у разі, якщо для розрахунку використовується ітеративний алгоритм.

Найбільший прогрес серед комп'ютерних інформаційних систем спостерігається в області розробки ***експертних систем***, заснованих на використанні штучного інтелекту. Експертні системи дають можливість менеджерові або фахівцеві одержувати консультації експертів по будь-яких проблемах, про які цими системами накопичені знання.

Рішення спеціальних завдань вимагає спеціальних знань. Однак не кожна компанія може собі дозволити тримати у своєму штаті експертів по всім пов'язаним з її роботою проблемам або навіть запрошувати їх щораз, коли проблема виникла. Головна ідея використання технології експертних систем полягає в тому, щоб одержати від експерта його знання й, завантаживши їх в пам'ять комп'ютера, використовувати щораз, коли в цьому виникає необхідність. Будучи одним з основних додатків штучного

інтелекту, експертні системи являють собою комп'ютерні програми, що трансформують досвід експертів у якій-небудь області знань у форму евристичних правил (евристик). Евристики не гарантують одержання оптимального результату з такою же впевненістю, як звичайні алгоритми, які використовуються для рішення завдань у рамках технології підтримки прийняття рішень. Однак часто вони дають достатньо прийнятні рішення для їхнього практичного використання. Все це уможливлює використання технологію експертних систем як систем що радять.

Подібність інформаційних технологій, що використовуються в експертних системах і системах підтримки прийняття рішень, полягає в тому, що обидві вони забезпечують високий рівень підтримки прийняття рішень. Однак є три істотних розходження. Перше пов'язане з тим, що рішення проблеми в рамках системи підтримки прийняття рішень відбивало рівень її розуміння користувачем і його можливості одержати й осмислити рішення. Технологія експертних систем, навпаки, пропонує користувачеві прийняти рішення, що перевершує його можливості. Друга відмінність зазначених технологій виражається в здатності експертних систем пояснювати свої міркування в процесі одержання рішення. Дуже часто ці пояснення виявляються більше важливими для користувача, ніж саме рішення. Третя відмінність пов'язана з використанням нового компонента інформаційної технології – знань.

Сучасні експертні системи широко використаються для тиражування досвіду й знань провідних спеціалістів практично у всіх сферах економіки. Традиційно знання існують у двох видах – колективний досвід й особистий досвід.

Якщо більша частина знань у предметній області представлена у вигляді колективного досвіду (наприклад вища математика), ця предметна область не має потреби в експертних системах (рис.4.55).




Рисунок 4.55. Предметна область, не придатна для створення експертної системи.

Якщо в предметній області більша частина знань є особистим досвідом фахівців високого рівня (експертів), якщо ці знання за якимись причинами слабко структуровані, така предметна область швидше за все має потребу в експертній системі (рис. 4.56).




Рисунок 4.56. Предметна область, придатна
для створення експертної системи

В Україні сучасний стан розробок в області експертних систем можна охарактеризувати як стадію всі зростаючого інтересу серед широких шарів економістів, фінансистів, викладачів, інженерів, медиків, психологів, програмістів, лінгвістів.

ЕС може повністю взяти на себе функції, виконання яких звичайно вимагає залучення досвіду людини – фахівця, або відігравати роль асистента для людини, що приймає рішення. Інакше кажучи, система (технічна або соціальна), що вимагає ухвалення рішення, може одержати його безпосередньо від програми або через проміжну ланку – людини, що спілкується із програмою. Той, хто приймає рішення, може бути експертом зі своїми власними правами, і в цьому випадку програма може "виправдати" своє існування, підвищуючи ефективність його роботи. Альтернативний варіант – людина, що працює в співробітництві з такою програмою, може домогтися з її допомогою результатів більше високої якості. Загалом кажучи, правильний розподіл функцій між людиною й машиною – це одне із ключових умов високої ефективності впровадження ЕС.

Дослідження в цій області сконцентровані на розробці й впровадженні комп'ютерних програм, здатних емулювати (імітувати, відтворювати) ті області діяльності людини, які вимагають мислення, певного майстерності й накопиченого досвіду. До них ставляться завдання прийняття рішень, розпізнавання образів і розуміння людської мови. Ця технологія вже успішно застосовується в деяких областях техніки й життя суспільства – органічної хімії, пошуку корисних копалин, медичній діагностиці. Перелік типових завдань, розв'язуваних ЕС, включає:

- витяг інформації з первинних даних (таких як сигнали, що надходять від гідролокатора);
- діагностика несправностей (як у технічних системах, так і в людському організмі);
- структурний аналіз складних об'єктів (наприклад, хімічних сполук);
- вибір конфігурації складних багатокомпонентних систем (наприклад, розподілених комп'ютерних систем);

- планування послідовності виконання операцій, що приводять до заданої мети (наприклад, виконуваних промисловими роботами).

Хоча відомі й "звичайні" програми, що спеціалізуються на певних завданнях із представленого переліку (або аналогічних їм у суміжних областях), ЕС можна виділити в окремий, досить добре помітний клас програм.

4.2.3.2 Характеристики та базові функції експертних систем

**Характеристики експертних систем**

ЕС відрізняється від інших прикладних програм наявністю наступних ознак.

– Моделює не стільки фізичну (або іншу) природу певної проблемної області, скільки механізм мислення людини стосовно до рішення завдань у цій області. Це істотно відрізняє ЕС від систем математичного моделювання або комп'ютерної анімації. Звичайно, не можна сказати, що програма повністю відтворює психологічну модель фахівця в цій предметній області (експерта), але важливо, що основна увага все-таки приділяється відтворенню комп'ютерними засобами методики рішення проблем, що застосовується експертом, тобто виконанню деякої частини завдань так само (або навіть краще), як це робить він.

– Система, крім виконання обчислювальних операцій, формує певні міркування й висновки, ґрунтуючись на тих знаннях, якими вона розташовує. Знання в системі представлені, як правило, на деякій спеціальній мові й зберігаються окремо від властиво програмного коду, що і формує висновки й міркування. Цей компонент програми прийнятий називати базою знань.

– При рішенні завдань основними є евристичні й наближені методи, які, на відміну від алгоритмічних, не завжди гарантують успіх. Евристика, по суті, є правилом впливу, що у машинному виді представляє деяке знання, придбане людиною в міру нагромадження практичного досвіду рішення аналогічних проблем. Такі методи є приблизними:

- по-перше, вони не вимагають вичерпної вихідної інформації,
- по-друге, існує певний ступінь упевненості (або непевності) у тім, що пропоноване рішення є вірним.

ЕС відрізняються й від інших видів програм з області штучного інтелекту (ШІ).

– ЕС мають справа із предметами реального світу, операції з якими звичайно вимагають наявності значного досвіду, накопиченого людиною. Більшість програм з області ШІ – сугубо дослідницькі й основна увага в них приділяється абстрактним математичним проблемам або спрощеним варіантам реальних проблем (іноді їх називають "іграшковими" проблемами), а ціль виконання такої програми – "підвищення рівня інтуїції" або відпрацьовування методики. ЕС мають яскраво виражену практичну спрямованість у науковій або комерційній області.

– Одна з основних характеристик ЕС – її продуктивність, тобто швидкість одержання результату і його вірогідність (надійність). Дослідницькі програми штучного інтелекту можуть і не бути дуже швидкими, можна примиритися й з існуванням у них відмов в окремих ситуаціях, оскільки це інструмент дослідження, а не програмний продукт. А от ЕС повинна за прийнятний час знайти рішення, що було б не гірше, ніж те, що може запропонувати фахівець у цій предметній області.

– ЕС повинна мати здатність пояснити, чому запропоновано саме таке рішення, і довести його обґрунтованість. Користувач повинен одержати всю інформацію, необхідну йому для того, щоб бути впевненим, що рішення прийняте "не зі стелі". На відміну від цього дослідницькі програми "спілкуються" тільки зі своїм творцем, що і так (швидше за все) знає, на чому ґрунтується її результат. ЕС проектується розраховуючи на взаємодію з різними користувачами, для яких її робота повинна бути прозорої.

**Базові функції експертних систем**

***Придбання знань***

**Придбання знань** – це передача потенційного досвіду рішення проблеми від деякого джерела знань і перетворення його у вид, що дозволяє використовувати ці знання в програмі.

Передача знань виконується в процесі досить тривалих і великих співбесід між фахівцем із проектування ЕС (будемо надалі називати його інженером по знаннях) і експертом у певній предметній області, здатним досить чітко сформулювати наявний у нього досвід. За існуючими оцінками таким методом можна сформувати від двох до п'яти "елементів знання" (наприклад, правил впливу) у день. Звичайно, це дуже низька швидкість, а тому багато дослідників розглядають функцію придбання знань у якості одного з головних "вузьких місць" технології експертних систем.

Причин такої низької продуктивності предосить. Перелічимо тільки деякі з них.

– Фахівці у вузькій області, як правило, користуються власним жаргоном, що важко перевести на звичайний "людський" мову. Але зміст жаргонного "слівця" аж ніяк не очевидний, а тому потрібно досить багато додаткових питань для уточнення його логічного або математичного значення. Наприклад, фахівці з військової стратегії говорять про "агресивну демонстрацію" іноземної військової моці, але при цьому не можуть пояснити, чим така "агресивна" демонстрація відрізняється від демонстрації, що не несе погрози.

– Факти й принципи, що лежать в основі багатьох специфічних областей знання експерта, не можуть бути чітко сформульовані в термінах математичної теорії або детермінованої моделі, властивості якої добре зрозумілі. Так, експертові у фінансовій області може бути відомо, що певні події можуть стати причиною росту або зниження котирувань на фондовій

біржі, але він нічого вам не скаже точно про механізми, які приводять до такого ефекту, або про кількісну оцінку впливу цих факторів. Статистичні моделі можуть допомогти зробити загальний довгостроковий прогноз, але, як правило, такі методи не працюють відносно курсів конкретних акцій на коротких тимчасових інтервалах.

– Для того щоб вирішити проблему в певній області, експертові недостатньо просто мати суму знань про факти й принципи в цій області. Наприклад, досвідчений фахівець знає, якого роду інформацією потрібно розташовувати для формулювання того або іншого судження, наскільки надійні різні джерела інформації й, як можна розчленувати складну проблему на прості, які можна вирішувати більш-менш незалежно. Виявити в процесі співбесіди такого роду знання, засновані на особистому досвіді й погано піддаються формалізації, значно складніше, ніж одержати простий перелік якихось фактів або загальних принципів.

– Експертний аналіз навіть у дуже вузькій області, виконуваний людиною, дуже часто потрібно помістити в досить великий контекст, що включає й багато речей, що здаються експертові саме собою що розуміють, але для стороннього аж ніяк такими не є. Як приклад розглянемо роботу експерта – юриста, що бере участь у судовому процесі. Дуже важко окреслити кількість і природу знань загального роду, які виявляються залучені в розслідування тої або іншої справи.

Незадовільні результати подібних співбесід розбудили в деяких дослідників інтерес до автоматизації процесу передачі знань фахівцем машині. Один з напрямків досліджень у цій області – автоматизований витяг знань (automated knowledge elicitation) – з'явився як побічний продукт у розвитку систем людино-машинного діалогу. Інші дослідники думають, що "розшити" це вузьке місце можна, рухаючись по шляху машинного навчання (machine learning) Ідея полягає в тому, щоб машина вчилася вирішувати проблеми приблизно так, як учиться людина.

***Подання знань***

**Подання знань** – ще одна функція експертної системи. Теорія подання знань – це окрема область досліджень, тісно пов'язана з філософією формалізму й когнітивною психологією. Предмет дослідження в цій області – методи асоціативного зберігання інформації, подібні тим, які існують у мозку людини. При цьому основна увага приділяється логічній, а не біологічній стороні процесу, опускаючи подробиці фізичних перетворень.

В 70-х рр. дослідження в області подання знань розвивалися в напрямках розкриття принципів роботи пам'яті людини, створення теорій витягу відомостей з пам'яті, розпізнавання й відновлення. Деякі з досягнутих результатів привели до створення комп'ютерних програм, які моделювали різні способи зв'язування понять (концептів). З'явилися комп'ютерні до-

датки, які могли деяким чином відшукувати потрібні "елементи" знання на певному етапі рішення деякої проблеми. Згодом психологічна вірогідність цих теорій відійшла на другий план, а основне місце, з погляду проблематики ИИ, зайняла їхня здатність служити інструментом для роботи з новими інформаційними й керуючими структурами.

Таким чином, питання подання знання був суперечливим і, швидше за все, таким і залишиться. Філософи й психологи найчастіше бувають шоковані безцеремонністю фахівців з ИИ, які жваво бовтають про людське знання на жаргоні, що представляє дику суміш термінології, узятої з логіки, логістики, філософії, психології й інформатики. З іншого боку, комп'ютерний формалізм виявився новаторським засобом постановки, а іноді й пошуку відповідей на важкі запитання, над якими сторіччями билися метафізики.

В області експертних систем подання знань цікавить нас в основному як засіб відшукання методів формального опису більших масивів корисної інформації з метою їхньої наступної обробки за допомогою символічних обчислень. Формальний опис означає впорядкування в рамках якої-небудь мови, що володіє досить чітко формалізованим синтаксисом побудови виражень і такого ж рівня семантикою, що погоджує зміст вираження з його формою.

Символічні обчислення означають виконання нечислових операцій, у яких можуть бути сконструйовані символи й символьні структури для подання різних концептів і відносин між ними.

В області штучного інтелекту ведеться інтенсивна робота зі створення мов подання. Під цим терміном розуміються комп'ютерні мови, орієнтовані на організацію описів об'єктів і ідей, на противагу статичним послідовностям інструкцій або зберіганню простих елементів даних. Основними критеріями доступу до подання знань є логічна адекватність, евристична потужність і природність, органічність нотації. Ці терміни, мають потребу в поясненнях.

– Логічна адекватність означає, що подання повинне мати здатність розпізнавати всі відмінності, які ви закладаєте у вихідну сутність. Наприклад, неможливо представити ідею, що кожні ліки мають якийнебудь побічний, небажаний ефект, якщо тільки не можна буде провести відмінність між призначенням конкретного лікарського препарату і його побічним ефектом (наприклад, аспірин збільшує виразкову хворобу). У більше загальному виді вираження, що передає цей ефект, звучать так: "кожні ліки мають небажаний, побічним ефект, специфічним для цього препарату".

– Евристична потужність означає, що поряд з наявністю виразної мови подання повинне існувати деякий засіб використання подань, сконструйованих і інтерпретіруемих таким чином, щоб з їхньою допомогою можна було вирішити проблему. Часто виявляється, що мова, що володіє більшою

виразною здатністю в термінах кількості семантичних відмінностей, більше складна в керуванні описом взаємозв'язків у процесі рішення проблеми. Здатність до вираження в багатьох зі знайдених формалізмів може виявитися досить обмеженої в порівнянні з англійською мовою або навіть стандартною логікою. Часто рівень евристичної потужності розглядається по результаті, тобто по тому, наскільки легко виявляється витягти потрібне знання стосовно до конкретної ситуації. Бути впевненим, які знання найбільше підходять для рішення конкретної проблеми, – це одне з якостей, що відрізняє дійсно фахівця, експерта в певній області, від новачка або просто начитаної людини.

– Природність нотації варто розглядати як якусь чесноту системи, оскільки більшість додатків, побудованих на базі ЭС, має потребу в нагромадженні великого обсягу знань, а вирішити таке завдання досить важко, якщо угоди в мові подання занадто складні. Будь-який фахівець скаже вам, що при інших рівних характеристиках краще та система, з якої простіше працювати. Вираження, якими формально описуються знання, повинні бути по можливості простими для написання, а їхній зміст – зрозумілий навіть тому, хто не знає, як комп'ютер інтерпретує ці вираження. Прикладом може служити декларативний програмний код, що сам по собі дає досить чітке подання про процес його виконання навіть тому, хто не має подання про деталі реалізації окремих інструкцій комп'ютером.

За минулі роки було запропоновано чимало угод, придатних для кодування знань на язиковому рівні. Серед них відзначимо правила, що породжують (production rules), структуровані об'єкти (structured objects) і логічні програми (logic programs). У більшості ЕС використовується один або трохи з перерахованих формалізмів, а доводи на користь і проти кожного з них дотепер являють собою тему для жвавих дискусій серед теоретиків.

***Управління процесом пошуку рішення***

При проектуванні ЕС серйозна увага повинне бути приділене й тому, як здійснюється доступ до знань, і як вони використовуються при пошуку рішення. Знання про те, які знання потрібні в тої або іншій конкретній ситуації, і вміння ними розпорядитися – важлива частина процесу функціонування ЕС. Такі знання одержали найменування **метазнання** – тобто знань про знання. Рішення нетривіальних проблем вимагає й певного рівня планування й управління при виборі, який питання потрібно задати, який тест виконати й т. ін.

Використання різних стратегій перебору наявних знань, робить досить істотний вплив на характеристики ефективності програми. Ці стратегії визначають, яким способом програма відшукує рішення проблеми в деякому просторі альтернатив. Як правило, не буває так, щоб дані, якими розташовує програма роботи із БЗ, дозволяли точно "вийти" на область у цьому просторі, у якій має сенс шукати відповідь.

Більшість формалізмів подань знань може бути використане в різних режимах управління, і розроблювачі ЕС продовжують експериментувати в цій області.

***Роз'яснення ухваленого рішення***

Питання про те, як допомогти користувачеві зрозуміти структуру й функції деякого складного компонента програми, зв'язаний з порівняно новою областю взаємодії людини й машини, що з'явилася на перетинанні таких областей, як штучний інтелект, промислова технологія, фізіологія й ергономіка. На сьогодні внесок у цю область дослідників, що займаються ЕС, складається в розробці методів подання інформації про поводження програми в процесі формування ланцюжка логічних висновків при пошуку рішення.

Подання інформації про поводження ЕС важливо з багатьох причин.

Користувачі, що працюють із системою, мають потребу в підтвердженні того, що в кожному конкретному випадку висновок, до якого прийшла програма, в основному коректно.

Інженери, що мають справу з формуванням БЗ, повинні переконатися, що сформульовані ними знання застосовані правильно, у тому числі й у випадку, коли існує прототип.

Експертам у предметній області бажано простежити хід міркувань і спосіб використання тих відомостей, які з їхніх слів були уведені в БЗ. Це дозволить судити, наскільки коректно вони застосовуються в даній ситуації.

Програмістам, які супроводжують, налагоджують і модернізують систему, потрібно мати у своєму розпорядженні інструмент, що дозволяє заглянути в "її нутро" на рівні більше високому, чим виклик окремих язикових процедур.

Менеджер системи, що використовує експертну технологію, що відповідає за наслідки рішення, прийнятого програмою, також має потребу в підтвердженні, що ці рішення досить обґрунтовані.

Здатність системи пояснити методику ухвалення рішення іноді називають прозорістю системи. Під цим розуміється, наскільки просто персоналу з'ясувати, що робить програма й чому. Відсутність достатньої прозорості поводження системи не дозволить експертові вплинути на її продуктивність або дати пораду, як можна неї підвищити. Простежування й оцінка поводження системи – завдання досить складна й для її рішення необхідні спільні зусилля експерта й фахівця з інформатики.

Правильне логічне подання найчастіше значно спрощує рішення завдання й робить його більше зрозумілим. Але сформувати таке подання – це значною мірою мистецтво. Пояснення не завжди має форму доказу.

4.2.3.3 Класифікація експертних систем

Загальноприйнята класифікація експертних систем відсутня, однак найбільш часто експертні системи розрізняють за призначенням, предметної області, методам подання знань, динамічності і складності (http://www.aiportal.ru/downloads/):

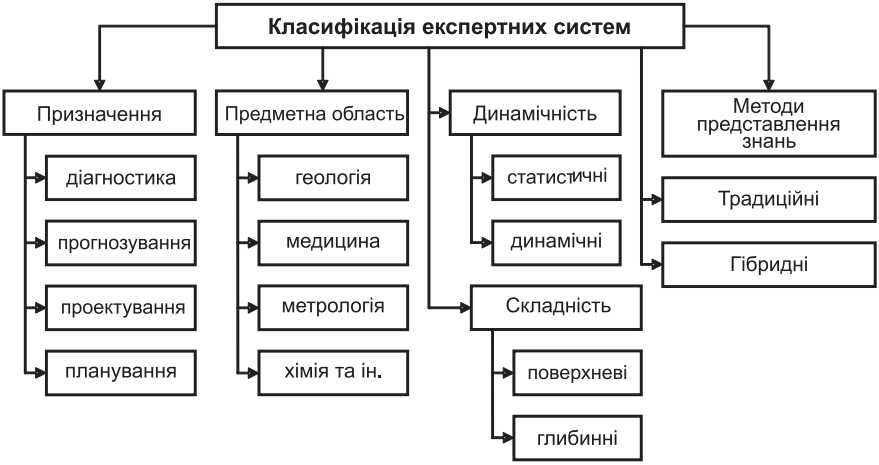



Рисунок 4.57. Класифікація експертних систем

За призначенням класифікацію експертних систем можна провести таким чином (рис. 4.57):

- діагностика стану систем, у тому числі моніторинг (безперервне відстеження поточного стану);
- прогнозування розвитку систем на основі моделювання минулого і сьогодення;
- планування та розробка заходів в організаційному та технологічному управлінні;
- проектування або вироблення чітких приписів щодо побудови об'єктів, що задовольняють поставлених вимогам;
- автоматичне керування (регулювання);
- навчання користувачів і ін.

За предметної області найбільша кількість експертних систем використовується у військовій справі, геології, інженерній справі, інформатики, космічній техніці, математиці, медицині, метеорології, промисловості, сільському господарстві, управлінні процесами, фізики, філології, хімії, електроніці, юриспруденції.

Класифікація експертних систем за методами подання знань ділить їх на традиційні та гібридні. Традиційні експертні системи використовують, в

основному, емпіричні моделі подання знань і числення предикатів першого порядку. Гібридні експертні системи використовують всі доступні методи, в тому числі оптимізаційні алгоритми і концепції баз даних.

За ступенем складності експертні системи діляться на поверхневі і глибинні. Поверхневі експертні системи представляють знання у вигляді правил "ЯКЩО-ТО". Умовою виводимості рішення є безобривность ланцюжка правил. Глибинні експертні системи мають здатність при обриві ланцюжка правил визначати (на основі метазнаній) які дії слід зробити для продовження вирішення завдання. Крім того, до складних належать предметні області в яких текст запису одного правила природною мовою займає більше 1/3 сторінки.

Класифікація експертних систем по динамічності ділить експертні системи на статичні і динамічні. Предметна область називається статичною, якщо описують її вихідні дані не змінюються в часі. Статичність області означає незмінність описують її вихідних даних. При цьому похідні дані (виводяться з вихідних) можуть і з'являтися заново, і змінюватися (не зраджуючи, проте, вихідних даних).

Якщо вихідні дані, що описують предметну область, змінюються за час вирішення завдання, то предметну область називають динамічною. В архітектуру динамічної експертної системи, у порівнянні зі статичної, вводяться два компоненти:

- підсистема моделювання зовнішнього світу;
- підсистема зв'язку із зовнішнім оточенням.

Остання здійснює зв'язок із зовнішнім світом через систему датчиків і контролерів. Крім того, традиційні компоненти статичної експертної системи (база знань і механізм логічного висновку) зазнають суттєвих змін, щоб відобразити тимчасову логіку відбуваються в реальному світі подій.

Розглянемо ще декілька варіантів класифікації. Для цього виберемо такі параметри, які задовольняють двом умовам.

**По-перше**, вибираючи значення цих параметрів, користувач, що не є фахівцем в ЕС, повинен бути здатний характеризувати особливості свого додатка. Це дозволить розроблювачеві ЕС вибрати інструментальний засіб (ІЗ), адекватне даному додатку.

**По-друге**, параметри і їхні різні значення повинні забезпечувати розроблювача ЕС інформацією, достатньої для відповіді на стратегічні питання, що виникають у користувача на різних етапах існування додатка.

Приклади питань, які ставляться перед користувачем, наступні: "Чи зможе створювана ЕС використовувати створені раніше програми?", "Чи буде ЕС працювати з різнорідним програмно-технічним середовищем користувача?", "Наскільки створювана ЕС буде критична до передбачуваної зміни платформ (ЕС із операційною системою)?", "Чи зможе ЕС вирішувати всі завдання даного додатка або частина залишиться,

наприклад за експертом?, "Які строки окупності ЕС?", "Чи адекватні обрані розроблювачем ІЗ завданням користувача?", "Коли ЕС буде використовуватися для рішення практичних завдань користувача, а не просто буде здана користувачеві?", "Яка вартість розробки, використання й супроводу (модифікації) ЕС?"

Будемо класифікувати додатка з ЕС по наступних параметрах:

- тип додатка;
- стадія існування;
- масштаб;
- тип проблемного середовища.

***Тип додатка***

Тип додатка характеризують наступні набори параметрів.

1. Можливість взаємодії додатка з іншими програмними засобами:

*ізольований додаток*, що складається з ЕС, не здатної взаємодіяти з іншими програмними системами, використовуваним кінцевим користувачем (наприклад, з базою даних, електронними таблицями, пакетами прикладних програм, контролерами, системою датчиків і т. ін.);

*інтегрований додаток*, що складається з ЕС та інших програмних систем, з якими ЕС взаємодіє в ході роботи.

Підкреслимо, що більшість сучасних (особливо динамічних) ЕС, використовуваних для рішення практично значимих завдань, є інтегрованими.

2. Можливість виконувати додаток на різнорідній апаратурі й переносити його на різні платформи:

*закриті додатки*, які виконуються тільки в програмному середовищі даної фірми й можуть бути перенесені на інші платформи тільки шляхом перепрограмування додатка;

*відкриті додатки*, які орієнтовані на виконання в різнорідному програмно-апаратному оточенні й в ідеалі можуть бути перенесені на інші платформи без перепрограмування.

3. Архітектура додатка:

додаток реалізується як *централізований*, на базі центральної ЕОМ, з якої зв'язані термінали;

децентралізований *розподілений додаток*, у цей час звичайно використовується архітектура клієнт – сервер.

***Стадія існування***

Стадія існування характеризує ступінь пропрацьованості й налагодженості ЕС. Звичайно виділяють наступні стадії:

- дослідницький прототип;
- діючий прототип;
- промислова система;
- комерційна система.

**Дослідницьким прототипом** називають систему, що вирішує представницький клас завдань додатка, але може бути нестійка в роботі й не повністю перевірена. При наявності розвинених ІС для розробки дослідницького прототипу потрібно приблизно 2-4 місяці. Дослідницький прототип звичайно має в базі знань не більше 50 загальних тверджень, що виконуються; при використанні тільки приватних тверджень їхня кількість зростає від 3 до 10 разів.

**Діючий прототип** – це система, що надійно вирішує всі завдання, але для рішення складних завдань може вимагати надмірно багато часу й (або) пам'яті. Доведення системи від початку розробки до стадії діючого прототипу вимагає приблизно 6-9 місяців, при цьому кількість тверджень, що виконуються, у БЗ збільшується до 100.

ЕС, що досягла стадії **промислової системи**, забезпечує висока якість рішень всіх завдань при мінімумі часу й пам'яті. Звичайно процес перетворення діючого прототипу в промислову систему складається в розширенні БЗ (до 150 тверджень, що виконуються,) і її ретельному налагодженню. Доведення ЕС від початку розробки до стадії промислової системи на розвиненому ІС вимагає приблизно 12-18 місяців.

Узагальнення завдань, розв'язуваних ЕС на стадії промислової системи, дозволяє перейти до стадії **комерційної системи**, тобто до системи, придатної не тільки для власного використання, але й для продажу різним споживачам. Доведення системи до комерційної стадії вимагає приблизно 1,5-2 роки. Наведені вище строки справедливі для ЕС середньої складності.

***Масштаб ЕС***

Багато фахівців класифікують ЕС за їхньою складністю (типу використовуваної ЕОМ) на

- малі,
- середні,
- великі,
- символьні.

**Малі ЕС** реалізуються на ПК типу PC або Macintosh, часто будучи ізольованими. Малі ЭС звичайно використовуються з метою первинного навчання або для дослідження можливості використання технології ЭС у даній області.

**Середні ЕС** реалізуються на робочих станціях. Вони бувають ізольованими й інтегрованими з базою даних (БД) і електронними таблицями. Дані додатки охоплюють весь спектр використання ЭС.

**Великі ЕС** реалізуються на робочих станціях або ЕОМ загального призначення (mainframe). Вони, як правило, мають доступ до величезних БД.

**Символьні ЕС** реалізуються на символьних ЕОМ або з використанням ІС типу Lisp і Prolog. Ці ЕС є дослідницькими й не використовуються для рішення реальних завдань.

***Тип проблемного середовища***

Поняття "проблемне середовище" включає предметну область (множина сутностей, що описують область експертизи, тобто множина об'єктів, значень їхніх характеристик і єднальних їхніх відносин) і розв'язуваних у предметній області завдань. Інакше кажучи, проблемне середовище включає сутності (структури даних) і розв'язувані над ними завдання, що представляються у вигляді тверджень, що виконуються, (у вигляді правил, процедур, формул і т. ін.). У зв'язку із цим проблемне середовище визначається характеристиками відповідної предметної області й характеристиками типів розв'язуваних у ній завдань. Помітимо, що поряд з поняттям "проблемне середовище" використовується сінонимичний йому термін "проблемна область".

***Характеристики предметної області*** визначаються наступним набором параметрів:

1) тип предметної області:
   - ✧ *статичний*, тобто вхідні дані не змінюються за час сеансу роботи додатка, значення інших (не вхідних) даних змінюються тільки ЕС;
   - ✧ *динамічний*, тобто вхідні дані, що надходять від зовнішніх джерел, змінюються в часі, значення інших даних змінюються ЕС або підсистемою моделювання зовнішнього оточення;
2) спосіб опису сутностей предметної області:
   - ✧ *сукупність атрибутів і їхніх значень* (фіксований склад сутностей);
   - ✧ *сукупність класів (об'єктів) і їхніх екземплярів* (змінюваний склад сутностей);
3) спосіб організації сутностей у БЗ:
   - ✧ *неструктурована БЗ*;
   - ✧ *структурування сутностей БЗ по різних ієрархіях* (найпоширеніші ієрархії: "загальне/частка", "частина/ціле"), що забезпечує спадкування властивостей сутностей, що представляються в БЗ.

***Характеристиками типів завдань є:***

1) тип розв'язуваних завдань:
   - ✧ *завдання аналізу й (або) синтезу*;
   - ✧ *статичні або динамічні завдання*;
2) частковість (спільність) тверджень, що виконуються, (правил, процедур, формул і т. ін.):
   - ✧ *частки (спеціалізовані, конкретні) твердження, що виконуються*;
   - ✧ *загальні твердження, що виконуються*.

Найбільш природним для людини способом *опису сутностей предметної області* є співвіднесення з ними в пам'яті ЕОМ об'єктів, що

складаються з атрибутів зі значеннями. Звичайно вводиться опис об'єкта деякого типу, відповідно до якого створюються конкретні екземпляри об'єктів цього типу. При цьому кількість екземплярів об'єкта ніяк не обмежується, тобто склад сутностей, що представляються, при такому поданні проблемної області змінюємо.

Однак для простих додатків при малій кількості об'єктів (від 1 до 5) іноді застосовують спрощене подання у вигляді атрибут/значення без згадування об'єкта, якому ці атрибути належать. Наслідком цього спрощення з'явилося те, що реально існуючі об'єкти (сутності) предметної області стали представлятися у вигляді фіксованої кількості розмножених імен відповідних атрибутів.

***Недоліки*** в загальному випадку досить значні. Вони полягають у наступному:

1) неможливо використовувати "загальні" виконуються твердження, що (правила, процедури, функції й т. ін.), що посилаються на довільну кількість сутностей; доводиться використовувати частки (спеціалізовані) твердження, що збільшує їхню кількість. Крім того, при зміні складу сутностей у БЗ доводиться вводити нові відповідні їм спеціалізовані твердження;

2) усунення об'єктів не дозволяє явно визначити природні взаємозв'язки між атрибутами одного об'єкта (всі атрибути є як би незалежними);

3) з усуненням об'єктів зникає можливість визначити ієрархію класів об'єктів.

Наступним параметром, що характеризує предметну область, є ***наявність (відсутність) структурування БЗ***. Зміст структурування БЗ полягає в наступному:

1) обмежити коло сутностей, які повинні розглядатися механізмом висновку, і в такий спосіб скоротити перебір при виборі рішення;

2) забезпечити спадкування властивостей сутностей, тобто передачу властивостей вищестоящих в ієрархії сутностей нижчестоящим, що значно спрощує процес придбання й використання знань. Наприклад, загальні властивості класу "автомобіль" автоматично успадковуються всіма підкласами автомобілів і конкретних екземплярів цих підкласів. У подібній ієрархії успадковується відношення "бути підмножиною (екземпляром)". Крім того, широко застосовується й інша ієрархія – "є частиною".

***По типу розв'язуваних завдань*** у першу чергу всі завдання доцільно розділити на завдання аналізу й синтезу.

У *завданні аналізу* задана модель сутності (об'єкта), потрібно в результаті аналізу цієї моделі визначити деякі невідомі її характеристики (функції). У *завданні синтезу* задаються умови, яким повинні задовольняти характеристики (функції) деякої "невідомої" моделі сутності, потрібно побудувати модель цієї сутності. Рішення завдання синтезу являє собою ітераційний процес, що складається з наступних кроків:

1) створення дослідницької моделі сутності;
2) аналіз цієї моделі (тобто рішення завдання аналізу);
3) порівняння результатів аналізу з умовами завдання.

Таким чином, завдання синтезу містить у собі аналіз. Відзначимо, що в процесі створення конкретної ЕС, що вирішує завдання аналізу, розроблювач, створюючи БЗ (модель області експертизи), вирішує завдання синтезу, а побудована ЕС буде вирішувати завдання аналізу.

Завдання синтезу й аналізу можуть вирішуватися в статичних і динамічних областях. Якщо ЕС базується на припущенні, що вихідна інформація про предметну область (про навколишній світ), на основі якої вирішується завдання, не змінюється за час рішення завдань, то говорять про *статичну предметну область* (точніше про статичне подання області в ЕС); якщо інформація про предметну область змінюється за час рішення завдань, то говорять про *динамічну предметну область*. При поданні динамічної області виникає завдання моделювання навколишнього світу, зокрема моделювання активних агентів.

Якщо завдання, розв'язувані ЕС, явно не враховують фактор часу й (або) не змінюють у процесі свого рішення дані (знання) про навколишній світ, то говорять, що ЕС вирішує *статичні завдання*; якщо завдання враховують фактор часу й (або) змінюють у процесі рішення дані про навколишній світ, то говорять про рішення *динамічних завдань*. Таким чином, ЕС працює в статичному проблемному середовищі, якщо вона використовує статичне подання й вирішує статичні завдання. ЕС працює в динамічному проблемному середовищі, якщо вона використовує динамічне подання й (або) вирішує динамічні завдання.

З огляду на значимість часу в динамічних проблемних середовищах, багато фахівців називають їхніми додатками ЕС, що працюють у реальному часі. Звичайно виділяють наступні системи реального часу: *"псевдореального"*, *"м'якого"* і *"твердого"* реального часу. Системи псевдореального часу, як треба з назви, не є системами реального часу, однак вони (на відміну від статичних систем) одержують і обробляють дані, що надходять від зовнішніх джерел. Системи псевдореального часу вирішують завдання швидше, ніж відбуваються значимі зміни інформації про навколишній світ.

Системи "м'якого" реального часу працюють у тих додатках, де припустимий час реакції на події більше 0,1-1 с. До цього діапазону ставляться майже всі існуючі ЕС реального часу. Системи "твердого" реального часу повинні забезпечувати час реакції швидше 0,1- 0,5 с. Для досягнення такої швидкодії вони використовують не стандартні операційні системи (ОС) типу Unix і Windows NT, а спеціалізовані ОС і спеціалізовані бортові ЕОМ, що забезпечують швидкий час реакції. У цей час ЕС, що працюють в "твердому" реальному часі, нам не відомі.

Завдання, розв'язувані ЕС, розрізняють тим, як представляються твердження, що виконуються. Використовуються як *частки (спеціалізовані)*

*твердження*, т е. твердження, що містять посилання на конкретні сутності (об'єкти), так і *загальні твердження*, що ставляться до будь-яких сутностей заданого типу (поза залежністю від їхнього числа й ім'я). Використання загальних тверджень дозволяє значно лаконічніше представляти знання. Тому що загальні твердження не містять явних посилань на конкретні сутності, то для їхнього використання потрібно затратити значну роботу з визначення тих сутностей, до яких вони повинні застосовуватися, тобто виконати, як говорять фахівці, операцію зіставлення.

4.2.3.4 Структура експертної системи

На рис. 5.56 приведена дуже загальна структура абстрактної експертної системи. Швидше за все не існує реальних експертних систем, які повністю відповідають наведеній структурі. Реальні ЕС містять велику кількість додаткових блоків, специфічних для кожної предметної області. Проте загальна технологія роботи і призначення основних структурних елементів ЕС, виходячи з наведеної схеми, повинні стати зрозумілі.

Типова ЕС передбачає наявність експерта (або групи експертів), групи користувачів і функціональних модулів, що підтримують її стан і роботу (рис. 4.58).

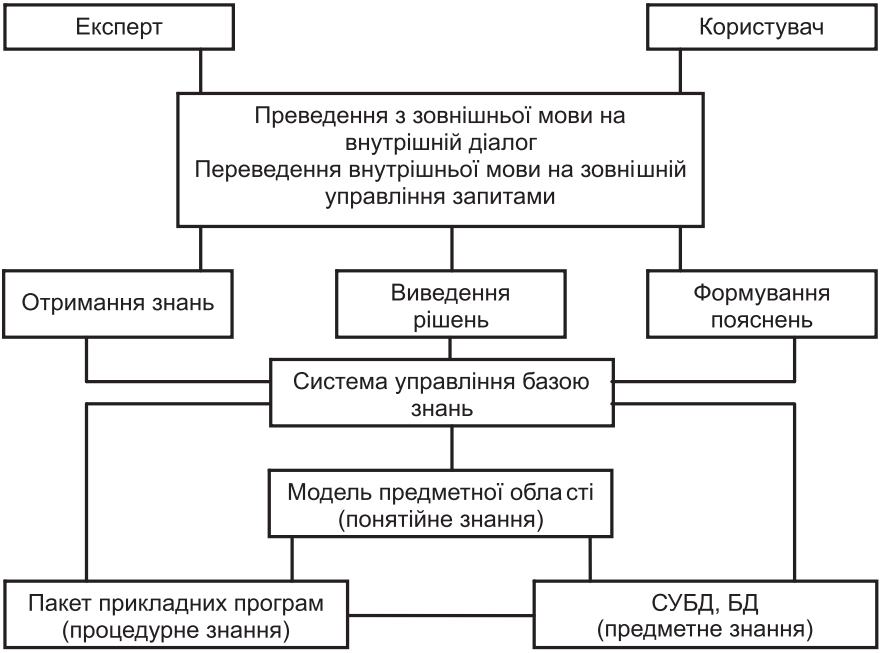



Рисунок 4.58. Структура експертної системи

Головною структурною відмінністю ЕС від усіх інших типів програм є наявність бази знань і, як наслідок, здатність до навчання і самонавчання. Її конкретний вид сильно залежить від обраної моделі представлення, але в найбільш загальному вигляді вона завжди буде містити фактуальную і алгоритмічну частини.

Так для системи правил продукцій ми будемо говорити про факти і правила (логічного, найчастіше дедуктивного виводу).

***Приклад.***

Вислів: крокодил (лапи, зуби, хвіст, колір).

Правило: крокодил (X, У, Z, U), якщо X = "короткі", У = "великі", Z = "довгий", U = "зелений".

Для системи ШІ, заснованої на фреймовой моделі подання знань, йдеться про факти і методи (приєднані процедури).

***Приклад***. Фрейм "Лінія на екрані":

координати початку 100, 100;
координати кінця 200, 200;
координати сусідній лінії невідомі.
Методи:
плавно обертатися навколо своєї середини;
зникати;
з'являтися.

Для мережевої моделі (мережевих моделей безліч) БЗ в загальному випадку містить: стан системи (наприклад, значення деякого параметра у вузлах мережі), обрану активну підмережу і правила виводу (які включають в себе і опис відносин, що є в мережі).

Експерт – досвідчений фахівець в деякій предметній області, який грає важливу роль при створенні ЕС. Предметною або проблемною областю називають сукупність взаємопов'язаних відомостей, необхідних і достатніх для вирішення деякого класу задач. Знання про предметну область включають описи об'єктів, явищ, фактів, а також відносин між ними. У процесі створення ЕС і підтримки її в працездатному стані використовуються знання і досвід експерта. Він пропонує структуру та можливості ЕС, мову для експерта, модель подання знань, форми одержуваних результатів і пояснень.

При налагодженні ЕС і її функціонуванні експерт вносить зміни в БЗ (додає, видаляє і модифікує деякі факти і правила), поглиблює й уточнює знання, виходячи з отриманих результатів.

Користувачами ЕС, наприклад в області медицини, можуть бути досвідчені або початківці лікарі, студенти-медики і, нарешті, самі хворі, які хочуть знати суть своєї хвороби, розвиток, наявний досвід її лікування, ліки, їх позитивні і негативні впливи.

Розглянемо функціональні модулі ЕС.

Діалог (інтелектуальний інтерфейс користувача) – це інтерфейс, який забезпечує спілкування між експертом, користувачем на звичній для них термінології з іншими компонентами системи. Він виконує переклад запитів, сформульованих на зовнішній мові, тобто природній або спеціальній мові, у внутрішню мову системи, а також перетворення відповідей і пояснень з внутрішньої мови в зовнішню, зрозумілу користувачеві. Він також управляє запитами, тобто ув'язує їх у контексті – усіх обмінів, які раніше мали місце.

Інтелектуальний інтерфейс користувача – це програма, відповідальна за організацію такого діалогу з користувачем, який оптимальним чином призводить до досягнення мети спілкування людини з ЕС. Так само як і у випадку з редактором БЗ, визначення "інтелектуальний" стало активно застосовувати в кінці XX ст. "Інтелектуальність" інтерфейсу полягає в тому, що в ході діалогу система допомагає користувачеві уточнити або зовсім переформулювати мету його консультації з експертною системою. Така необхідність дійсно має місце, оскільки сучасні об'єкти, процеси, явища, що моделюються за допомогою ЕС, настільки великі і складні, що кінцевий користувач не може пам'ятати (а іноді і знати взагалі) навіть їхні імена (назви) і, отже, просто не здатний грамотно сформулювати запит і отримати відповідь.

Висновок рішень включає: механізм виведення (інакше, машина виводу), який шляхом умовиводів визначає одне або декілька рішень; механізм оцінки пропонованих рішень; блок управління механізмом виведення, що підвищує ефективність його роботи.

Машина виведення – це програма, відповідальна за порядок і спосіб актуалізації алгоритмічної частини БЗ. Для різних МПЗ вона, природно, різна.

Для логічної моделі, де кожен раз відбувається повний перебір всіх логічних рівнянь, незалежно від уже наявних даних вона має найпростіший вид послідовного автомата.

Для системи продукцій (машина логічного висновку), де вибір правила, яке слід актуалізувати наступним, неоднозначний, вона має систему переваг, засновану на деякій детермінованою або ймовірнісної оцінкою, що залежить від ситуації, що склалася до цього моменту.

Для фреймової моделі машина виведення складається ніби з двох частин. Єдина загальна частина відповідає за те, щоб події, що відбуваються в системі фреймів, ставали відомі всім фреймам одночасно або в деякій послідовності. Друга частина – це обробники подій самих фреймів, які вирішують, який з методів слід актуалізувати у відповідь на отримане повідомлення про подію.

Придбання знань (інтелектуальний редактор БЗ) отримує нові знання з одного боку, від експерта, а з іншого, від модуля виводу рішень і передає їх в БЗ. Цей блок може також змінювати вже існуючі знання в БЗ.

Інтелектуальний редактор БЗ – це програма, що дозволяє додавати, видаляти, модифікувати факти і правила, що містяться в БЗ. Визначення "інтелектуальний" стосовно до редактору БЗ стало активно використовуватися в кінці XX ст. Під "інтелектом" стосовно до даного класу програм розуміється здатність забезпечити цілісність, коректність і несуперечність БЗ. Аналогічна проблема в теорії баз даних вирішується за допомогою ключів і нормалізації, однак для концепції знань таке рішення не підходить. Основна причина полягає в тому, що знання являють собою щось єдине, ціле і багатозначне. Окремо вирваний з контексту елемент знань не має сенсу, тому, щоб здійснити будь-яку операцію над БЗ (додавання, видалення, модифікація записи), необхідно перевірити дотримання цілісності, коректності та несуперечності всій БЗ, причому на деяку задану глибину.

Формування пояснень ініціюється користувачем, коли потрібно обгрунтування отриманого результату. Система може включати пояснення різних рівнів для фахівців, а також для непосвячених в даній області користувачів, наприклад студентів або пацієнтів.

Підсистема пояснень – це один з найважливіших елементів ЕС, який значною мірою визначає ефективність її практичного використання. Практична реалізація підсистеми пояснень часто використовує ті ж процедури, що і вирішувач, тільки в іншому порядку. Тому часто можна вважати, що машина виведення складається з вирішувача і підсистеми пояснень. Специфіка завдань, що вирішуються інтелектуальними системами, така, що дуже часто цінність має не саме рішення, отримане людиною чи машиною, а процес – хід рішення, отриманого людиною в діалозі з машиною. Підсистема пояснень – це програмний механізм отримання відповіді на питання "А чому так?" Або "А чому так, а не так?" Щодо вирішення в цілому і його елементів (кроків) окремо.

Не всі інтелектуальні системи володіють такою здатністю (мають підсистему пояснень). Її відсутність може бути пов'язано з труднощами програмування, але частіше з вадами самої моделі ПЗ. У класичному розумінні підсистема пояснень існує тільки у систем, заснованих на правилах продукцій і семантичних мережах.

Управління знаннями дозволяє ефективно використовувати інформацію, розташовану в БЗ і БД. Найбільш істотним в цьому модулі є метод відбору підмножин даних з БД, необхідних для вирішення поставленого питання (рис. 4.59.) При відборі підмножин даних виникає дві проблеми:

1) семантики – як визначити в БЗ підмножина корисної інформації;
2) ефективності – якого рівня деталізації інформації в БД потрібно досягти, враховуючи можливість декомпозиції задачі на підзадачі.

Модуль управління знаннями може сам розглядатися як ЕС, яка генерує запити до системи управління бази даних (СУБД).

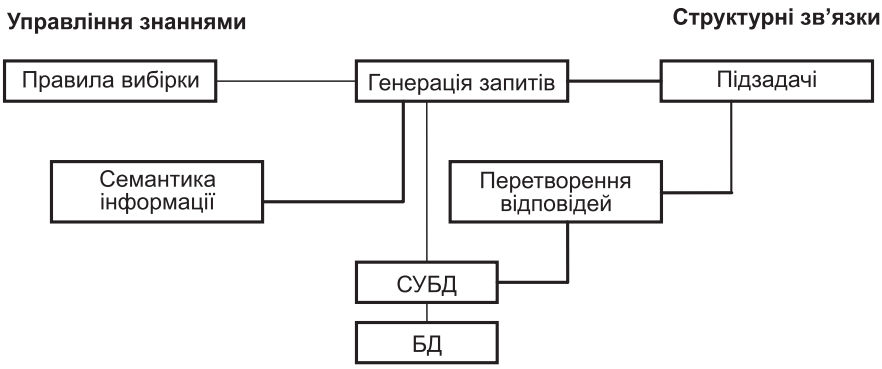



Рисунок 4.59. Схема модуля управління знаннями

Така ЕС повинна розглядати кілька типів правил, які назвемо правилами вибірки.

Перший тип правил пов'язує задачу зі складовими її подзадачами, що можна висловити так:

1) тип завдання: – тип завдання 1, тип завдання 2, ...

Другий тип правила визначає інформацію, корисну для вирішення задачі:

2) тип завдання: – інформація.

Третій тип правил з'єднує тип інформації із запитами користувача:

3) тип інформації: – питання.

Нарешті, ЕС повинна розташовувати даними про структуруванні різних типів інформації в такі структури, як тезаурус (словник), документальний файл, файл з ключами.

Використовуючи правила вибірки 1-3, блок генерації запитів створює їх безліч для даної підзадачі. Відповіді на поставлені запити, отримані з БД, представляються у формі відносин, які перетворюються в мову, використовуваний для представлення знань першого і другого типів.

Система управління базою знань (СУБЗ) управляє базою правил і через СУБД – БД. У функції СУБД входить відбір підмножини правил для вирішення підзадачі, а також отримання нових фактів на основі бази правил і бази фактів шляхом використання машини виводу.

У розглянутим ЕС однією з важливих проблем є вибір мови, який дозволяє описати тип завдань та інформацію. Відомо, що схема БД включає описи імен об'єктів, атрибутів, відносин і областей. Але цього недостатньо, оскільки потрібно ще уявити семантичні зв'язки між елементами. Слід зауважити, що схема БД – це статичне опис, оскільки вона не змінюється. Навпаки, для кожної знову з'являється підзадачі потрібно динамічно генерувати її опис у формі "тип завдання". А цей блок реалізовувати не просто.

Раніше ми відзначили, що головна відмінність ЕС від інших програмних засобів – це наявність бази знань, в якій знання зберігаються у вигляді сукупності записів на деякій мові представлення знань (ЯПЗ), який дозволяє легко змінювати і доповнювати базу знань у формі, зрозумілій фахівцям – розробникам експертної системи.

Розробку ЕС здійснюють такі фахівці:

експерт в тій проблемної області, на яку орієнтована ЕС;

Вибір тієї чи іншої моделі визначається структурою знань у конкретної предметної області. Виявлення структури передує вибору ЯПЗ та розробці бази знань. Обгрунтування і конкретизація елементів знань та їх взаємозв'язків відбуваються найчастіше в безпосередньому контакті з фахівцями предметної області – експертами. Цей процес називається витяганням знань. Розробники експертних систем, що займаються витяганням і структуруванням знань, називаються інженерами по знаннях.

- ✧ інженер по знаннях – фахівець з розробки ЕС;
- ✧ програміст – фахівець з розробки інструментальних засобів.

Експерт визначає знання (дані і правила), що характеризують проблемну область, забезпечує повноту та правильність введених в ЕС знань, оцінює отримані системою рішення в процесі розробки системи.

Інженер по знаннях допомагає експерту виявити і формалізувати знання, вибирає інструментальні засоби, найбільш відповідні для подання знань та реалізації ЕС, виділяє і програмує стандартні (типові) функції, які використовуються експертом при формулюванні знань.

Програміст розробляє компоненти ЕС. Його діяльність відноситься до так званого, інтелектуального програмування, яке становить:

Мови штучного інтелекту:

- Об'єктно-орієнтовані мови;
- Мови для представлення знань.

Автоматичний синтез програм:

- Дедуктивні методи;
- Індуктивні методи.

Інструментальні системи:

- "Порожні";
- Системи (оболонки).

Системи когнітивної графіки.

4.2.3.5 Приклади реалізації експертних систем

Перерахуємо деякі з предметних областей, в яких застосовуються ЕС в даний час: військова справа, геологія, інженерна справа, інформатика, комп'ютерні системи, космічна техніка, математика, медицина, метеорологія, промисловість, сільське господарство, управління процесами, фізика, хімія, електроніка, юриспруденція. Медицина найбільш

популярна; саме в цій області було розроблено більше ЕС, ніж у всякій іншій, хоча хімія ненабагато відстає від неї, і розрив швидко скорочується.

У медицині відповідні дослідження почалися з системи MYCIN, однією з перших і найбільш відомих ЕС. MYCIN – це ЕС, розроблена для медичної діагностики та прийняття рішення. Зокрема, вона призначена для роботи в області діагностики і лікування зараження крові та медичних інфекцій. Система ставить відповідний діагноз, виходячи з представлених їй симптомів, і рекомендує курс медикаментозного лікування будь-якої з діагностованих інфекцій. Вона полягає в загальній складності з 450 правил, розроблених за допомогою групи з інфекційних захворювань Стенфордського університету. Її основним моментом є використання імовірнісного підходу.

Система MYCIN справляється із завданням прийняття рішення шляхом призначення показника визначеності кожному зі своїх 450 правил. Тому можна представляти MYCIN як систему, що містить набір правил виду "ЯКЩО ... ТО" з певністю, яку надали люди – експерти і яку виклали в правилах, вказавши свій ступінь довіри до кожного правила за шкалою від 1 до 10.

Встановивши ці правила і пов'язані з ними показники визначеності, MYCIN йде по ланцюжку назад від можливого результату, щоб переконатися, чи можна вірити такому результату. Встановивши всі необхідні вихідні передумови, MYCIN формує судження з даного результату, розраховане на основі показників визначеності, пов'язаних з усіма правилами, які потрібно використовувати.

Припустимо, щоб отримати результат Z, потрібно визначити передумови X і Y, що дають можливість вивести Z. Але правила для визначення X і Y можуть мати пов'язані з ними показники визначеності P і Q. Якщо значення P і Q були рівні 1,0, то результат Z не викликає сумніву. Якщо P і Q менше 1,0 (як це буває), то результат Z не послідує напевно. Він може вийти лише з деякою мірою визначеності.

MYCIN не видає діагноз і не розкриває його точний показник невизначеності. Система видає цілий список діагнозів, називаючи показник визначеності для кожного з них. Всі діагнози з показниками вище певного специфічного для кожного діагнозу рівня приймаються як в тій чи іншій мірі вірогідні, і користувачеві вручається список можливих результатів.

Стандартні фрази і граматичні форми були без праці пристосовані до програми, і в результаті вийшов істотно вироджений діалект англійської мови, що легко піддається програмуванню.

Лікарі залишилися дуже задоволені таким результатом, тому що вони говорили, використовуючи дуже невеликий набір слів англійської мови (принаймні, коли повідомляли про свою роботу).

У хімії роботи по ЕС почалися з новаторського проекту DENDRAL, розпочатого в Стенфордському університеті в середині 60-х рр. XX століття

і присвяченого розробці методів ШІ для визначення топологічних структур органічних молекул. Сучасні роботи по ЕС в галузі хімії включають виведення структури молекул, синтез органічних молекул і планування експериментів в молекулярній біології.

У найзагальніших рисах процес прийняття рішення наступний. Користувач дає системі DENDRAL деяку інформацію про речовину, а також дані спектрометрії (інфрачервоної, ядерного магнітного резонансу і мас-спектрометрії), і та, в свою чергу, видає діагноз у вигляді відповідної хімічної структури.

Можна для простоти представити систему DENDRAL складається з двох частин, як якщо б в одній ЕС були дві самостійні системи. Перша з них містить набір правил для вироблення можливих хімічних структур. Введена інформація складається з ряду висновків, зроблених хіміком, і дозволяє судити, які структури вірогідні в тому чи іншому випадку.

На виході першої системи є не один простий відповідь. Зазвичай це ціла серія можливих структур – програма не в змозі точно сказати, яка з них вірна. Потім DENDRAL "бере" кожну з цих структур по черзі і використовує другий ЕС, щоб визначити для кожної з них, які були б результати спектрального аналізу, якби ця речовина існувало і було насправді досліджено по спектрограмі.

Процес, часто іменований "генерація і перевірка", дозволяє постійно скорочувати число можливих розглянутих варіантів, щоб в будь-який момент воно було якомога менше. На відміну від деяких експертних систем DENDRAL задумана не як "іграшка". Вона не використовується лише для перевірки теоретичних основ ЕС, а реально застосовується для визначення хімічних структур.

У геології першою була ЕС PROSPECTOR, розроблена в Стенфордському дослідницькому інституті в середині 70-х рр. XX століття PROSPECTOR – це ЕС, підтримуюча рішення при пошуку комерційно виправданих родовищ корисних копалин.

Система PROSPECTOR, за аналогією з MYCIN, містить велику кількість правил, що відносяться до різних об'єктів, а також можливих результатів, виведених на їх основі. У цій системі використовується також "рух по ланцюжку назад" та ймовірності. Методи цієї системи є одними з кращих серед всіх розроблених методів для будь-якої з існуючих нині систем.

Найпростіший випадок – правила, що виражають логічні відносини. Це правила типу "ЯКЩО x, ТО z", де подія "z" безпосередньо випливає з "x". Вони залишаються настільки ж простими, якщо зіставити "x" ймовірність.

Якщо у "x" всього один аргумент, то це правило істотно спрощується. Зазвичай замість "x" ми представляємо більш складне логічне вираження, наприклад: (x І y) або (x АБО y).

Якщо елементи відносини пов'язані з допомогою логічного І та окремих елементах цього відношення зіставлені певні ймовірності, то система PROSPECTOR вибирає мінімальне з цих значень і привласнює цю мінімальну ймовірність розглянутого можливого результату. Тому, коли ймовірність x = 0,1, ймовірність y = 0,2, ймовірність результату z = 0,1. Легко бачити, чому обрано такий метод: щоб "z" було істинним, і "x", і "y" повинні бути істинними. Це є "жорстким" обмеженням, тому слід брати мінімальне значення.

Система PROSPECTOR користується методом, заснованим на застосуванні формули Байєса з метою оцінки апріорної і апостеріорної вірогідності якої-небудь події.

В цілому правила в системі PROSPECTOR записуються у вигляді ЯКЩО ..., ТО (LS, LN), причому кожне правило встановлюється з відношенням правдоподібності як для позитивного, так і для негативної відповіді.

Система PROSPECTOR пропонує користувачеві шкалу відповідей в діапазоні від -5 до +5. Нижня межа – це безперечно "Так", верхній – виразно "Ні". Зазвичай відповідь користувача знаходиться десь між крайніми значеннями, і PROSPECTOR коригує його, враховуючи LS і LN за допомогою лінійної інтерполяції. Це легко уявити собі у вигляді лінійної шкали, де LN – крайнє ліве, а LS – крайнє праве значення.

В області комп'ютерних систем типовим зразком може служити ECXCON, одна з перших і найбільш успішно застосовуються розробок цього роду. Вона була розпочата корпорацією DEC та Університетом Карнегі-Меллон в кінці 70-х рр. як дослідницький проект, а зараз XCON досягла рівня комерційної системи і використовується для проектування конфігурацій комп'ютерів. Сучасні розробки ЕС в цій області пов'язані з діагностикою несправностей, проектуванням комп'ютерних конфігурацій і управлінням процесом виробництва комп'ютерів.

В електроніці переважають дослідження і розробки, пов'язані з діагностикою несправностей і проектуванням НВІС. Система AGE, розроблена на початку 80-х рр. фірмою Белл, типова діагностична система в даній області. Вона використовується для виявлення несправностей в телефонній мережі та визначення їх характеру. В даний час роботи по ЕС в галузі електроніки пов'язані з навчальними системами, що допомагають знаходити відмови в електричних ланцюгах і проектувати електронно-цифрові схеми.

В інженерній справі типовий зразок ЕС – це DELTA, система діагностики несправностей, розроблена компанією "Дженерал Електрик" в середині 80-х рр. "Дженерал Електрик" планує використовувати DELTA на комерційній основі для допомоги обслуговуючому персоналу при пошуку несправностей в дизель-електричних локомотивах. В даний час розробля-

ються інші системи діагностування несправностей і навчання операторів складних систем управління (рис. 4.60).

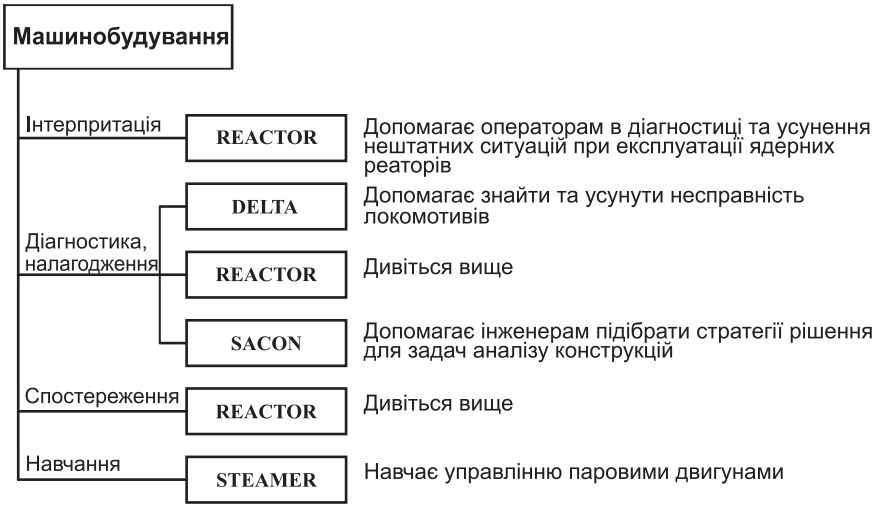



Рисунок 4.60. ЕС в інженерній області

У військовій справі зусилля були сконцентровані на інтерпретації, прогнозуванні та плануванні. Одна з перших військових ЕС, названа HASP / SIAP, розроблялася спільно з Стенфордським університетом та System Control Technology на початку 70-х рр. Ця система визначає типи кораблів, інтерпретуючи дані від мережі гідрофонів, прослуховуючих деякі акваторії океану. В даний час потенційні військові додатки ЕС включають інтерпретацію інформації від датчиків, прогноз бойових дій та тактичне планування.

Прикладами найбільш досконалих ЕС є:

- ЄС діагностики обладнання літаків F-15 и F-18 (шифр AFTA);
- ESIAS – система ОЦІНКИ тактічної обстановки, яка склалось;
- EXTRAS – система планування режімів роботи головного двигуна корабля Space Shuttle та ін.

## 4.3 Перспективи розвитку інформаційних технологій

### 4.3.1 Сучасний стан і напрямки розвитку ІТ

Розвиток ІТ надзвичайно динамічний. При цьому і апаратні і програмні технології розвиваються шляхом, який робить можливим обробку великої кількості інформації в режимі реального часу і доступ до цієї інформації майже з любої точки за допомогою розподілених баз даних. Більш того, збільшення комфортності для користувачів і зростання можливостей удо-

сконалення персональних комп'ютерів і робочих станцій дозволяють працювати з цими системами людям, які формально мають невелику підготовку. Удосконалення перш за все включають в себе взаємозв'язок комп'ютерів в локальних, регіональних, національних і міжнародних мережах. Крім цього, все важливішою стає робота на персональних комп'ютерах, робочих станціях або універсальних комп'ютерах. Цей розвиток основується на тенденції до відкритих систем, які дозволяють переносити сімейства програмних засобів загального використання з однієї моделі комп'ютера на іншу.

Посилюються спроби встановлення більш ефективних і тісно взаємозв'язаних ділових відносин між підприємствами і організаціями. Акцент робиться на збільшення продуктивності, скорочення витрат і удосконалення товарів і послуг за допомогою наскрізного електронного зв'язку з високим рівнем автоматизації. З іншого боку, існує тенденція до більшої взаємодії всередині організації для підвищення ефективності функціонування і продуктивності. В якості другорядного продукту цих вимог в організаціях і між організаціями також необхідно добавити сили, які можна визначити як оточення, яке "постійно змінюється". Організаціям (як і всій економіці в цілому) знадобляться суттєві корективи, щоб пристосуватися до прискорюючих змін буквально у всьому: виробництві, організації послуг тощо, які потребують більшої адаптивності виробничих організаційних структур і розробки засобів їх підтримки.

Вже сьогоднішній економічних розвиток включає в себе проблему інформаційного перевантаження бізнес-менеджерів. Що потрібно менеджерам – так це допомога в оцінці і виборі даних, які містять важливу інформацію для конкретної проблемної ситуації, і перетворення цих даних в знання для покращення процесу прийняття рішення. В цьому контексті інформаційна технологія (ІТ) – це тільки початок розв'язування цих проблем.

Сьогодні при створенні комп'ютерів пріоритет віддається надпотужним комп'ютерам (суперкомп'ютерам), а також мініатюрним і надмініатюрним персональним комп'ютерам. Основною тенденцією розвитку електроно – обчислювальної машин є розширення їхньої сфери застосування й перехід від експлуатації окремих комп'ютерів до роботи їх у складі обчислювальних мереж або систем.

Здійснюється орієнтація територіальних обчислювальних мереж на надання комунікаційно-інформаційні послуг: E-mail, телеконференцій, IP-Телефонії, надання різної інформації.

Одержувана суспільством інформація з комп'ютерних мереж до середини 21 століття зрівняється з обсягом інформації отриманий по традиційних каналах ЗМІ (радіо, телебачення, преса), а до кінця сторіччя територіальні комп'ютерні мережі повинні стати основним інформаційним середовищем, а Інтернет стане основним засобом зв'язку.

Останнім часом ведуться активні роботи з передачі нових видів інформації через Інтернет. Конвергенція телекомунікаційних мереж (радіо, телефонних, телевізійних і обчислювальних мереж) відкриває нові можливості для передачі даних, голоси й зображення.

Саме Інтернет претендує на роль глобальної універсальної мультисервісної мережі нового покоління для якісної передачі даних, голоси й зображення. Використання мультиканальних широкосмугових радіо й оптичних каналів зв'язку забезпечать практично необмежену пропускну здатність. Не тільки комп'ютери, але й телефони, телевізори, відеокамери й інші пристрої будуть підключатися прямо до Інтернету.

Що стосується комунікаційних послуг, то територіальні обчислювальні мережі повинні стати не просто засобом комунікації, а інтерактивним каналом взаємодії людей (четвертим каналом спілкування людей), що дозволяють їм спілкуватися в режимі онлайн.

Уже сьогодні в Інтернеті є величезна кількість інформації практично на будь-яку тему, але для того, щоб знайти потрібну інформацію й використовувати її, необхідно вміти працювати з ресурсами цієї глобальної мережі.

World Wide Web (WWW) – один з найважливіших видів сервісу Інтернет. WWW надає можливість роботи з документами, у яких об'єднані текст, графічні ілюстрації, звукові фрагменти й навіть анімація, що робить ці документи надзвичайно виразними й полегшує сприйняття інформації.

У документах системи WWW реалізований принцип гіпертексту, тобто документи містять посилання на інші, зв'язані за змістом, документи, причому останні можуть перебувати на вилучених комп'ютерах. Тому при наявності досить високошвидкісного підключення до Інтернету ви можете переглядати WWW-Документи, що зберігаються на серверах у різних країнах миру також швидко й легко, як якби вони перебували на вашім комп'ютері.

У цей час тривають роботи зі створення комп'ютерів нового покоління – нейрокомп'ютерів на базі розподіленої нейронної архітектури. Вони якісно відрізняються від комп'ютерів попередніх поколінь відсутністю заздалегідь створених комп'ютерних програм і здатністю до самоорганізації й навчання. Основу нейрокомп'ютерів будуть становити нейронні мережі, які забезпечують взаємодія з об'єктами реального світу так само, як і біологічна нервова система.

Широке впровадження аудіо- і відеозасобів уведення й виводу інформації дозволять користувачам спілкуватися з комп'ютерами природною мовою. При роботі на комп'ютері користувач буде бачити віртуального співрозмовника й активно обмінюватися з ним природною мовою, одержувати ради, підказки, роз'яснення.

Розвиток таких інтернет-технологій як засобу оперативного обміну й поширення інформації, а також засобів створення й зберігання й підтримки

інформаційних ресурсів у глобальній мережі забезпечать ефективне середовище для мережної економіки.

4.3.1.1 Введення в нові інформаційні технології

Процес витягу (одержання) інформації будується на основі впорядкованих послідовних дій по зборі, нагромадженню, відбиттю, перетворенню, актуалізації даних; такі процеси в інформатиці називаються **інформаційними технологіями**, і їхніми основними елементами є технічні засоби й пристрої, наприклад, у телеграфі – телетайпний пристрій, у телебаченні – телевізор і т. інш.

***Нові інформаційні технології – це** інформаційні технології**, що базуються на нових, інфологічних і комп'ютерних засобах одержання, зберігання, актуалізації інформації, знань.***

***Високі технології – це технології якісної зміни складу, характеру, методів розв'язуваних завдань, технології еволюції, а не функціонування.***

***Приклад***. Звичайна ("стара") технологія обчислень ставить *основну мету* – знайти рішення завдання за прийнятний час і вартість. Нова технологія використання математичних комп'ютерних пакетів ставить *нову мету* – знайти рішення досить швидко, точно й економічно. **Висока технологія** розподілених, квантових обчислень *ставить мету* – знайти рішення завдання, не розв'язуваної (складнорозв'язуваних) звичайними технологіями.

У вузькому розумінні, *нова інформаційна технологія* – використання обчислювальної техніки й систем зв'язку для створення, збору, передачі, зберігання, обробки інформації; вона – частина інформаційного бізнесу.

Будь-яка технологія базується на науково-теоретичному, інженерно-технічному, програмному забезпеченні. Саме по собі це ядро ще не утворить технологію. Для цього воно повинне бути інтегроване й підтримуване мережними просторово-тимчасовими, організаційно-людськими зв'язками й відносинами. Повинна бути система, мережа підтримки технологічних відносин (TSN).

***Приклад***. TSN системи дистанційного навчання складається з інфраструктури – комп'ютерних мереж, протоколів їхньої взаємодії й т.д. Хаб (маршрутизатор) – елемент цієї системи, але він управляється своїм програмним забезпеченням (наприклад, програма перемикання), своїм електричним забезпеченням. Хаб сам по собі – не технологія. Програма Word – сама по собі не технологія (хоча її часто називають технологією підготовки й редагування документів), а елемент технології, обумовленої як MS Office – технології автоматизованого, комп'ютеризованого діловодства, автоматизації робіт в офісі.

Традиційна (класична) **інформаційна технологія**, як правило, будується на базі добре формалізуємих, структуруємих інтелектуальних

процедур. **Нова інформаційна технологія**, як правило, будується на основі погано формалізованих і структурованих інтелектуальних процедур.

Цивілізація можлива тільки при наявності інформації, інформаційних потоків і обмінів у суспільстві.

Інформація робить народи людством.

Спочатку **інформаційні технології** використовувалися, через їхню складність, унікальності й дорожнечі, тільки в наукових центрах і великих промислових компаніях. У міру вдосконалювання, поширення й здешевлення, *інформаційні технології* проникнули в різні галузі й стали розвивати їх і розвиватися самі, що привело до розвитку потреб суспільства.

***Нові інформаційні технології*** бувають наступних базових типів:

- ***когнітивні технології***, спрямовані в основному на отримання, зберігання та актуалізацію знань, прийняття інтелектуальних рішень;
- ***інструментальні технології,*** спрямовані здебільшого на використання як інструментарій, середовища для побудови інших технологій і для обслуговування їх;
- ***прикладні технології,*** спрямовані здебільшого на рішення проблем деякої проблемної області (або областей);
- ***комунікативні технології,*** спрямовані здебільшого на рішення проблем зв'язку, комунікацій, спілкування.

Відзначимо, що такий розподіл – досить умовний – і технологія може з успіхом бути й *прикладною*, і *когнітивною*, і *інструментальною*, і *комунікативною*.

***Приклад***. Така, наприклад, технологія комп'ютерного моделювання, гіпермедіа.

Можливий розподіл (також умовне) *інформаційних технологій* і по сфері використання, наприклад:

- інформаційні технології в науці;
- інформаційні технології в утворенні;
- інформаційні технології в проектуванні й виробництві;
- інформаційні технології в керуванні;
- інформаційні технології в сфері послуг;
- інформаційні технології в сфері побуту.

Можна також умовно розбити всі нові технології на дві групи:

- технології корпоративної роботи;
- технології індивідуальної роботи.

### 4.3.1.2 Нові технології проектування і аналізу систем, тенденції розвитку

Розглянемо нові *інформаційні технології*, обмежуючись змістовним простим їхнім оглядом, з обліком того, що найбільш важливі *інформаційні технології* аналізу й синтезу систем – математичне й комп'ютерне, іміта-

ційне моделювання – нами розглядатися не будуть (розглядаються при вивченні інших дисциплін). Відзначимо лише, що математичне моделювання – "стара" *інформаційна технологія*, на відміну від комп'ютерного моделювання, що є новою технологією.

***Технологія баз даних (БД) і систем управління БД (СУБД).***

***БД*** – досить більші набори структурованих даних деякої предметної області, представлені на машинних носіях і имеющие загальну й зручну структуру, єдині організаційно-методичні, програмно-технічні і язикові засоби забезпечення використання даних різними програмами користувачів.

Останнім часом поширюється **технологія розподідених БД**. Вона базується на колективному доступі користувачів до інформаційних ресурсів, зосередженим на єдиному комп'ютері, або хост-комп'ютері, у діалоговому режимі по мережах передачі даних. Інформаційними продуктами тут виступають БД різних предметних областей, а також різні директорії, рубрикаторы й інші дані, що полегшують користувачеві пошук по БД. Інформаційні послуги надаються завдяки наявності різноманітних засобів пошуку, обробки й видачі інформації. Інформаційні продукти й програмні засоби служать головними елементами банків даних або автоматизованих банків даних (АБД) – основної організаційної форми, у якій розвиваються сучасні технології комерційного поширення інформації. Основними особливостями даної технології, що визначають її достоїнства і її недоліків, є:

- ✧ надання користувачеві тільки інформаційних послуг, а не **безпосередньо інформаційних продуктів, у результаті чого він одержує** (оплачує) тільки дійсно потрібну інформацію;
- ✧ повнота інформації, пов'язана із завантаженням на потужні хост-комп'ютери більших масивів даних;
- ✧ висока швидкість відновлення, модифікації й переміщення інформації;
- ✧ розвинене програмне забезпечення, що дозволяє не тільки знаходити й одержувати інформацію, але й при необхідності здійснювати її графічну, наукометричну й економетричну обробку.

Інтерактивні послуги АБД можуть надаватися в режимах:

- ✧ локальному, коли робота користувача здійснюється з термінала, підключеного до хост-комп'ютера;
- ✧ вилученому, коли робота користувача здійснюється з фізично вилученого від хост-комп'ютера термінала по мережах зв'язку.

***Приклад***. У локальному режимі працюють читачі бібліотеки, що здійснюють пошук в АБД, що розташований на її обчислювальному центрі, з терміналів по всьому приміщенню бібліотеки. У вилученому режимі можна працювати, наприклад, з бібліотекою Конгресу США.

***СУБД (DBMS – DataBase Management System)*** – програмна система, що забезпечує спілкування (інтерфейс) програм користувача й даних із БД. Це спілкування відбувається на спеціальній непроцедурній мові логічного подання даних і структур даних; самі дані описуються засобами також спеціальної мови подання даних, програми користувача при цьому можуть бути написані мовою програмування.

**Технології сховищ даних і інтелектуального аналізу даних.**

**Сховище даних –** спеціалізована БД і програмна система, призначена для витягу, корекції (чищення, виправлення) і завантаження даних із джерел у БД із багатомірною структурою, включаючи засобу спрощення доступу, аналізу з метою ухвалення рішення.

**Інтелектуальний аналіз даних (Data Mining)** – автоматичний пошук у базах даних схованих ("не поверхневих") залежностей і зв'язків за допомогою математичного й інфологічного аналізу, виділення трендів, кластерізації (кластерного аналізу), класифікації й розпізнавання (таксономії), шкалювання тощо. Спеціальні моделі й алгоритми аналізу витягають із великих баз даних (або з інших сховищ даних, наприклад, електронних таблиць) знання, що дозволяють агрегувати, інтегрувати й деталізувати ці дані й, саме головне, приймати на їхній основі рішення. Це, по суті, ідентифікація схованих у них залежностей.

***Приклад***. Сховища даних збирають і централізують поточну інформацію про стан справ корпорації, про її послуги, клієнтів, постачальниках, і надають аналітичні й звітні інструменти. За допомогою аналізу фінансових звітів фірм, можна розбити їх на класи по фінансовій стійкості, по ймовірності банкрутства, що допоможе банку-кредиторові здійснювати політику кредитування більш ефективно. Інтелектуальний аналіз даних у геоінформаційних системах може допомогти виявити й візуалізувати ділянки земної кори з покладами нафти, газу, сейсмонебезпечні. У бізнесі такий аналіз може здійснюватися для оцінки надійності клієнтів, виявлення шахрайства, інтерактивного маркетингу, аналіз трендів і ін. тобто для Business Intellgence.

**Технологія баз знань (БЗ) і експертних систем (ЕС).**

***БЗ*** – нагромадження, структурування й зберігання за допомогою ЕОМ знань, відомостей з різних областей таким організованим способом, що можна мати доступ до цих знань, розширювати їх, одержувати, виводити нові знання тощо.

***Приклад.*** БЗ по хірургічних операціях черевної порожнини, з якої молодий і недосвідчений хірург в екстреній хірургічній ситуації може витягти необхідну інформацію про операцію; сама ж БЗ розроблена на основі знань високопрофесійних і досвідчених хірургів.

***ЕС –*** нагромадження досвіду, знань, умінь, навичок високого рівня професіоналів-експертів, структурування й зберігання, актуалізація за допомогою ЕОМ з метою одержання експертних суджень по різних проблемах даної області.

***Приклад***. Прикладом ЕС "Хірург" може бути експертна система, побудована на основі наведеного вище приклада БЗ. БЗ і ЕС тісно зв'язані. Прикладом інший ЕС може бути система "Митниця", що дає можливість аналізувати документацію про фінансові угоди, знаходити й видавати підозрілі факти, досліджувати їхнього зв'язку й давати рекомендації фінансовим інспекторам.

**Технологія електронної пошти й телекомунікаційного доступу до вилученого від користувача інформації, носію інформації, співрозмовникові – людині або комп'ютеру.**

***Електронна пошта –*** система передачі повідомлень за допомогою комп'ютера відправника й прийому їх за допомогою комп'ютера одержувача. При цьому повідомлення відправника перетвориться із цифрових кодів, наприклад, за допомогою модему, у коди електромагнітних коливань, переданих по телефонних каналах, а ЕОМ адресата робить зворотне перетворення. Розвиток мереж зв'язку – віртуальні локальні обчислювальні мережі, що поєднують користувачів не по територіальному принципі, а по професійних інтересах.

***Телеконференція*** – обмін повідомленнями (доповідями) між учасниками (передплатниками) конференції, анонсованої на спеціальній дошці оголошень у мережі, зокрема , на електронній дошці оголошень. Телеконференція являє собою технологію на базі програмних засобів інтерактивного доступу до ресурсів мережі й призначена для обговорення якої-небудь тематики. За допомогою *телеконференцій* можна проводити консалтинг, навчання, нараду, автоматизацію офісу й ін. Базова система проведення відеоконференцій звичайно включає: потужну робочу мультимедійну станцію; відеокамеру й спеціальну плату для стиску відеоінформації; мікрофон і відеомагнітофон; засобу сполучення з використовуваної для проведення конференції мережею. *Телеконференції* можуть проводитися як у режимі обміну листами по електронній пошті (режим поштового підключення), так і в режимі термінального інтерактивного підключення через телекомунікаційні мережі. У режимі термінального підключення користувач може мати доступ (підписку) до цілої системи *телеконференцій*, але, на відміну від режиму поштового підключення, можна підключатися до заявленої конференції безпосередньо в мережі, за допомогою спеціальних програм, керуючих роботою користувачів з *телеконференціями*. Ці програми дозволяють виконувати наступні маніпуляції: знайти конференцію; підписатися на конференцію (зареєструватися); перейти в конференцію;

послати відгук (доповідь); одержати відгук (доповідь); закрити підписку й інші.

***Приклад***. Розглянемо медичні *відеоконференції* (один з найбільш переконливих і яскравих соціально-економічних прикладів використання *телеконференції*). У великих лікарнях і клініках зараз є сучасне медичне встаткування – томографи, ехокардіографи й ін., а також досить висококваліфікований медичний персонал, за допомогою яких у режимі відеодіалогу (конференції) лікарі з регіональних (аж до районних) медичних установ можуть обговорити результати діагностики хворого, діагнозу, методів і стратегій лікування. Проблема "наближення" цих засобів і кадрів особливо актуальна для нашої країни. Основні напрямки використання медичних *телеконференцій*:

- ✧ первинне консультування доопераційних хворих і уточнення попереднього діагнозу, аналіз обстеження;
- ✧ післяопераційні консультації й спостереження хворих;
- ✧ термінові невідкладні консультації хворих у критичних ситуаціях;
- ✧ консиліуми й консультації, обмін думками лікарів.

Технологія проведення медичної відеоконференції:

- ✧ узгодження часу проведення відеоконференції (сеансу зв'язку);
- ✧ підготовка інформації про пацієнта (файлів записів з історій хвороби, статичних даних, наприклад, рентгенограми, ехокардіограми, і динамічних, наприклад, відеозапису операцій і результатів аналізу);
- ✧ попереднє пересилання даних по електронній пошті;
- ✧ обговорення в режимі відеодіалогу інформації про хворого й діагноз;
- ✧ ухвалення рішення, а також документування результатів обговорення.

Економічна й соціально-медична вигода від таких відеоконференцій в 6-10 разів вище, ніж від класичної технології проведення консультацій з виїздом у клініку (що іноді неможливо).

**Технологія (використання) автоматизованих систем (*АС*) і автоматизованих робочих місць (*АРМ*).**

***АС*** **–** це людино-машинна система для виконання щоденних, часто рутинних, професійно виконуваних на робочому місці співробітника робіт – з метою зменшення витрат часу, скорочення числа помилок і забезпечення оперативного зв'язку з іншими співробітниками; інтелектуальні системи мають також здібність до перебудови технологічного ланцюжка, вони здатні і до навчання.

Можливі різні системні цілі автоматизації (залежно від типу організації, структури): ефективне управління потоками матеріальних, тру-

дових, інформаційних, енергетичних ресурсів, підвищення соціальних, економічних і технічних показників системи і інших систем (для систем, що управляють); мінімізація риски невиконання планів і максимізація якості ухвалених рішень, підвищення конкурентоспроможності, рентабельності (для виробничих систем); отримання нових знань, підвищення престижу в області інновацій, розширення сфери використання результатів дослідження, створення "ноу-хау", забезпечення економічної ефективності (для науково-дослідних організацій); мінімізація риски невиконання замовлення або послуг, підвищення конкурентоспроможності, підвищення економічної ефективності функціонування (для обслуговуючих організацій); підвищення престижу, вдосконалення учбового процесу, перехід до нових форм навчання, до дистанційної освіти (для освітніх організацій).

Останніми роками розповсюдилася концепція корпоративних і розподілених систем в народному господарстві, в яких широко використовуються локальні інформаційні системи. Для реалізації ідеї розподіленого управління необхідне створення автоматизованих робочих місць на базі професійних комп'ютерів (робочих станцій).

***АРМ –*** наочно-орієнтована інструментальна *АС*, встановлювана безпосередньо на робочому місці фахівця і призначена для автоматизації професійної діяльності (співробітника, що сидить за цим робочим столом). Можна їх визначити як автоматизовані системи локального характеру, відповідні деякому функціональному призначенню. Не дивлячись на різний характер завдань, що вирішуються за допомогою *АРМ*, принципи створення будь-яких *АРМ* мають бути загальними:

- системність, по-перше, підходу до проектування і вирішення завдань і, по-друге, можливість роботи у складі мережі, системи;
- гнучкість, пристосовність, адаптується до змін завдань;
- стійкість, надійність в роботі, відновлюваність при збоях;
- ефективність (за витратами, підвищенням продуктивності праці);
- швидкість відгуку – мінімум часу на кожен крок діалогу з користувачем;
- повнота виконуваних функцій, вирішуваних професійних завдань;
- інтерактивність – можливість втручатися в діалог, вибирати наступний крок діалогу, наприклад, у формі команд на спеціальній командній мові, у формі вибору об'єктів, у формі "меню", в змішаній формі;
- функціональність, дружність, ергономічні характеристики і зручність використання, зокрема, орієнтація на непрофесіонала в області комп'ютерної підготовки і ін.

Призначений для користувача інтерфейс *АРМ* часто організовується за допомогою поняття робочого столу на екрані. Екран ділиться на три частини (три об'єкти):

- ***перша*** (зазвичай верхня частина) – рядок меню, з її допомогою здійснюється доступ до інших об'єктів;
- ***друга частина*** (зазвичай нижня частина) називається рядком стану, з її допомогою швидко викликаються найбільш використовувані об'єкти або відображається важлива поточна інформація;
- ***третя частина*** (основна, середня частина екрану) називається робочою поверхнею (поверхнею столу), з її допомогою відображаються всі об'єкти, що викликаються з меню або з рядка стану.

Така форма організації діалогу людини і машини найбільш зручна, і багато програм використовують саме її. Програмні засоби *АРМ* – частина інструментального програмного забезпечення.

***Приклад***. *АРМ* секретаря-референта повинен включати редактор текстів, електронну таблицю, перекладачі, органайзер і ін. *АРМ* студента-економіста повинен мати електронні підручники по дисциплінах, що вивчаються, повчальні програми і середовища, електронні довідники і енциклопедії, перекладачі, органайзер і ін. *АРМ* адміністратора бази даних має бути оснащений *СУБД*, електронним журналом адміністратора і ін. *АРМ* керівника повинен володіти засобами опису управлінської діяльності у вигляді мережевого графіка, системою контролю виконання, системою узгодження документів, системою електронного підпису, системою ведення наради і ін. *АРМ* статистика (працівника статистичної служби) повинен включати інформаційно-довідкові бази і матеріали, засобу його обслуговування і підтримки, засобу аналізу статистичних даних для оперативного складання доповідей, довідок і звітів, зокрема, для аналізу динамічних рядів і перевірки статистичних гіпотез. *АРМ* банківського службовця і банківські системи, – системи, що найбільш розвиваються. Вони містять програмне і технічне забезпечення як спеціального призначення (наприклад, для банківських розрахунків і операцій з банкоматами), так і для забезпечення безпеки таких систем. Банківські системи використовують локальні обчислювальні мережі, спеціалізовані бізнес-сервери, ЕОМ загального призначення, технологію *"клієнт-сервер"* і, часто, ОС UNIX, об'єднання декілька локальних мереж, міжмережевий обмін і видалений доступ філій банку до ресурсів центрального офісу банку для виконання електронних платежів, транзакцій. Банківські системи повинні мати засоби адаптації до конкретних умов експлуатації. Для підтримки оперативної роботи банку, банківська система повинна функціонувати в режимі реального часу OLTP (On-Line Transaction Processing, Онлайнова Обробка Стикувань). Основні їх функції:

- автоматизація всіх щоденних внутрішньобанківських операцій, ведення бухгалтерії і складання звідних звітів;
- забезпечення зв'язків з філіями і відділеннями;
- автоматизована взаємодія з клієнтами (система "банк-клієнт");
- аналіз діяльності банку і вибір оптимальних в даній ситуації рішень;
- автоматизація роздрібних операцій – застосування банкоматів і кредитних карток;
- міжбанківські розрахунки;
- автоматизація діяльності банку на ринку цінних паперів (моніторинг курсів, об'ємів операцій і так далі);
- оперативна інформаційна підтримка фінансової і кредитної політики банку.

Необхідно відзначити, що автоматизація установи, організації полягає не тільки в забезпеченні працівників *АРМ* локальною мережею, *електронною поштою* і тому подібне, але і в створенні нової технології спільної роботи і управління з метою ефективної роботи всієї установи. Це свого роду інформаційна, економічна (корпоративна) культура спільної роботи, використання (актуалізації) професійних знань, умінь і інновацій. При цьому знання кожного співробітника можуть бути актуалізовані іншими співробітниками, обговорені (проголосовані) і застосовані ними.

**Технології *комп'ютерного* (комп'ютеризованого) *офісу* колективної роботи в офісі.**

***Комп'ютерний офіс*** **–** офіс, в якому є високий рівень комп'ютеризації, впровадження *АРМ*, систем діловодства, так, що вся професійна діяльність офісу може бути успішно автоматизована.

***Приклад***. *Комп'ютерний офіс* – це, наприклад, офіс, де робота здійснюється з використанням локальних мереж зв'язку і інтегрованого програмного середовища Microsoft Office, яке включає всі основні програмні пакети для виконання типових і регулярно виконуваних операцій, робіт в офісі, зокрема, ведення діловодства, контроль виконання і ін. Microsoft Office має вбудована проста мова програмування – Visual Basic for Applications (VBA). Ця мова дозволяє створювати нові застосування або коректувати і зв'язувати старі, виконувані в середовищі Microsoft Office, а також розширювати можливості офісу, його використовуваних застосувань. Стандартне ядро Microsoft Office включає:

- редактор текстів Microsoft Word (функції редактора – набір, іменування і збереження тексту, модифікація, перейменування і переміщення тексту або його окремих фрагментів, вставка різних формул, графіків, таблиць, діаграм і ін.);

- електронну таблицю Excel (функції – обробка, зберігання і модифікація в довільних таблицях чисел, рядків, стовпців, формул, по яких динамічно змінюються числа, рядки і стовпці);
- систему для презентацій (презентаційний пакет) PowerPoint (функції – створення і проектування на великому екрані електронних презентацій, слайд-шоу, яскравих плівок для проектора, роздавальних друкарських матеріалів);
- систему управління базами даних Access (реляційна *СУБД*, доступна будь-якому користувачеві і що дозволяє швидко і ефективно організовувати, аналізувати, переміщати, вести пошук і так далі для великих масивів інформації, без дублювання інформації в них), наприклад, по шаблонах створення бази даних: Адресна книга – створює базу даних типу адресної книги, Бібліотека – створює базу даних типу бібліотеки, Контакти – створює базу даних типу контактних зв'язків і ін.

Досконаліші версії Microsoft Office-97 -2000, -2003, -2007 окрім приведених стандартних застосувань, мають і наступні застосування:

- Office Assistant – помічник для підказок;
- HTML- і Web-підтримку (Internet Assistants);
- різні програми-помічники (Graph – графічне представлення даних, Organization Chart – творець штатного розклад офісу, Equation Editor – редактор формул, WordArt – творець логотипів, заголовків, ClipArt Gallery – для проглядання малюнків) і ін.

Так до складу Microsoft Office-2000 (і в її подальші модифікації) входять можливості одночасного показу презентацій по локальній мережі, використання можливостей системи розпізнавання мови, візуальні середовища розробки різних офісних застосувань (наприклад, заповнення платіжних доручень), сайти *робочих груп*, системи візуалізації даних, система сканування і введення даних і ін.

Технологія "***Робоча група"*** **–** технологія спільної роботи декілька зв'язаних між собою загальними інформаційними ресурсами комп'ютерів (*"робочої групи"*), об'єднаних для вирішення якого-небудь загального завдання.

***Приклад***. Типи *робочих груп*: "Дирекція", "Бухгалтерія", "Канцелярія". Комп'ютерна мережа організації може об'єднувати декілька *робочих груп*. У кожного комп'ютера *робочої групи* є ідентифікатор, ім'я в групі, наприклад, по ФІО людини, на нім що працює. У *робочій групі "*Бухгалтерія" може існувати комп'ютер (робоче місце) "Головбух" або "Іванов Сергій Миколайович".

*Робоча група* може бути і тимчасовою – для роботи над конкретним проектом в межах певного проміжку часу.

***Приклад***. Можна організувати *робочу групу "*Презентація фірми", яка складається з комп'ютерів співробітників фірми, що готують презентацію своєї фірми, або "Річний звіт", – для підготовки річного фінансового звіту фірми. Всі ці люди можуть працювати в різних відділах, але вони складають тимчасову *робочу групу*, щоб було легко обмінюватися інформацією загального доступу при роботі над звітом.

Обмін інформацією може відбуватися і між *робочими групами*. Для цього не потрібно фізично переміщати комп'ютери: щоб сформувати *робочу групу*, досить привласнити всім комп'ютерам, що входять до складу групи, її ім'я.

***Приклад***. Операційна система Windows for Workgroups дозволяє виділення комп'ютерів в *робочі групи* при її інсталяції. Змінювати склад і структуру *робочої групи* потім можна з "Панелі управління", запустивши прикладну програму Network (мережа). При цьому всі комп'ютери однієї мережі, незалежно від їх об'єднання в *робочі групи*, мають доступ до загальних принтерів і загальних файлів, а такі застосування як Mail (*Електронна пошта*), Shedule+ (Щоденник), працюють тільки в межах однієї *робочої групи*. Передача пошти через Mail можлива тільки в межах однієї *робочої групи*. Як правило, в невеликих фірмах є одна *робоча група*.

Технологія (модель взаємодії) ***"Клієнт-сервер"*** **–** це технологія взаємодії комп'ютерів в мережі, в якій кожен з комп'ютерів має своє робоче призначення. Один потужний комп'ютер (сервер) в мережі володіє і розпоряджається інформаційними і апаратними ресурсами (процесор, файлова система, поштова служба, база даних і ін.), інший, менш могутній ("клієнт"), має доступ до цих ресурсів лише через сервер.

Цей принцип розповсюджується і на взаємодію програм і інформаційних середовищ. Програма (середовище), що виконує надання відповідного набору послуг, – "сервер", а програма (середовище), що користується цими послугами, – "клієнт". Технологія традиційної моделі *"клієнт-сервер"* модернізується і удосконалюється.

***Приклад***. Зараз говорять вже про принципово іншій концепції взаємодії між елементами мережі peer-to-peer (P2P), що дозволяє окремим комп'ютерам працювати один з одним безпосередньо.

**Технології використання інтегрованих пакетів прикладних програм.**

(***ППП***) – технології на базі спеціальним чином організованих комплексів програм для вирішення різних класів однотипних завдань, що часто зустрічаються, з різного типу наочних областей. Сучасні *ППП* мають діалоговий, інтерактивний зворотний зв'язок з користувачем в процесі постановки завдання, рішення і аналізу результатів. При вирішенні завдань застосовують зазвичай використовуваний в наочній області інтерфейс.

"Інтелектуальність" ***ППП*** – можливість постановки завдання змістовно, не указуючи алгоритму її рішення. Побудова алгоритму рішення і збірка цільової програми проводяться автоматично і приховано від користувача. Наочне забезпечення ***ППП*** – база знань про методи, алгоритмах рішення задачі і про самі завдання. Програмування здійснюється в термінах наочної області, ЕОМ використовується вже на етапі постановки завдання, вирішення завдань – за допомогою автоматичної побудови ланцюжка програм, по ходу накопичення знань про вирішуване завдання, тобто можливе поповнення бази знань ***ППП.*** Використовуються інструментальні ***ППП***, які прискорюють і спрощують процес створення ***ППП*** і знижують його вартість. Для цього методу характерна висока інтеграція: наповнення ***ППП*** само складається з ***ППП*** різного призначення.

***Приклад***. Як приклад інтегрованого ***ППП*** приведемо пакет MATHCAD, призначений як для складних математичних обчислень, так і для нескладних (у режимі інженерного калькулятора).

**Технології машинної графіки і візуалізації.**

***Технології машинної графіки і візуалізації*** – технології, що базуються на системах малювання і креслення різних графічних об'єктів і образів за допомогою ЕОМ і пристроїв малювання (наприклад, плоттерів), а також їх візуального, наочного уявлення. Особливо слід зазначити засоби ***анімації*** – "пожвавлення" зображень на екрані, тобто методи і засоби створення динамічних зображень, інакше кажучи – комп'ютерних мультфільмів.

***Приклад***. Прикладом засобів машинної графіки може служити програмний комплекс зображення просторових об'єктів і їх динамічної актуалізації – пакет "3D-Studio". Цей пакет дозволяє не тільки створювати тривимірні сцени, але і використовувати їх при реалізації комп'ютерних анімаційних ситуацій (мультиплікації) з використанням різних графічних файлів різних форматів, що дає можливість застосовувати при розробці мультфільмів відомі графічні пакети: Coreldraw, PhotoPaint і ін. 3D-Studio має модульну структуру, що складається з п'яти модулів, за кожним з яких закріплені завдання конкретного типу, що вирішуються в строгій послідовності. Перший модуль (2D-Shaper) є основним інструментом створення і редагування плоских фігур, а також постачання інших модулів особливими геометричними структурами, формами і траєкторіями. Для перетворення плоских фігур в тривимірні каркасні об'єкти є модуль 3D-Lofter, в який включені могутні засоби генерації складних просторових форм і структур. Підготовлені двовимірні плани моделей відображаються ("видавлюються") в третє вимірювання по спеціально заданих траєкторіях. Модуль 3D-Lofter забезпечений засобами деформації, наприклад, по осях, що дозволяє створювати тривимірні об'єкти складніших форм. Можна побудувати 3D-фигуру трьом проекціям на координатну плоскість.

Сучасні технології 3D-графического моделювання дозволяють будувати повні тривимірні об'єкти по їх ескізах.

***Приклад***. Програмне середовище LightWave Modeler, дозволяє, використовуючи відповідні графічні примітиви, формувати графічний персонаж, який легко анімується за бажанням (масштаб, напрям, колір і тому подібне) користувача.

**Гіпертекстові технології.**

***Гіпертекст*** (Hypertext – "надтекстова") – ця технологія на базі засобів обробки великих, глибоко вкладених, структурованих, зв'язаних семантично і понятійний текстів, інформації, які організовані у вигляді фрагментів (тексту), які відносяться до однієї і тієї ж системи об'єктів, що розташованих у вершинах деякої мережі і виділяються зазвичай кольором; вони дають можливість при машинній реалізації швидко, натисненням декількох клавіш, викликати і поміщати в потрібне місце нового тексту, що проглядається або організовуваного, задані фрагменти *гіпертексту*, тобто тексти, "прив'язані" до виділених за кольором ключових слів або словосполук; гіпертекстова технологія дозволяє визначати, вибирати варіант актуалізації інформації *гіпертексту* залежно від інформаційних потреб користувача і його можливостей, рівня підготовки, тобто жорстко і заздалегідь не визначає сценарії діалогу. При роботі з гіпертекстовою системою користувач може проглядати документи (сторінки тексту) в тому порядку, в якому йому це більше подобається, а не послідовно, як це прийнято при читанні книг, тобто *гіпертекст* – нелінійна структура. Досягається це шляхом створення спеціального механізму зв'язку різних сторінок тексту за допомогою гіпертекстових посилань, тобто окрім лінійних посилань звичайного тексту типу "текст-попередник – текст-наступник", у *гіпертексту* можна побудувати ще скільки завгодно багато інших динамічних посилань, що асоціюються з документом в цілому або тільки з окремими його фрагментами, тобто контекстні посилання.

***Приклад***. Прикладами *гіпертекстів* можуть бути електронні журнали.

**Засоби і системи *мультимедіа* (multimedia) і *гіпермедіа* (hypermedia).**

***Медіа –*** "середовище або носій інформації". ***Мультімедійність***, багатосередовищність – актуалізація різних середовищ і відчуттів сприйняття інформації: засоби озвучування, пожвавлення – мультиплікація, графічного і наочного представлення вхідних і вихідних даних завдання і сценаріїв рішення або навіть самого рішення.

***Приклад***. Прикладами засобів *мультимедіа* можуть служити звукові карти (Sound Blaster) для генерування на ЕОМ широкого діапазону звуків, активні звукові колонки для їх передачі і пристрою прочитування

інформації з компакт-дисків – CD-ROM, що дозволяють прочитувати великі об'єми інформації, наприклад, деяку складну і тривалу музичну композицію, а потім відтворювати з використанням попередніх двох засобів *мультимедіа*.

Засоби ***гіпермедіа*** **–** засоби на основі синтезу концепції *гіпертексту* і *мультимедіа*, тобто в гіпертекстові фрагменти можуть бути "вбудовані" мультимедійний супровід, мультимедійні застосування: hypermedia=hyp ertext+multymedia.

***Приклад***. Глобальною гіпермедійною системою є WWW (Word Wide Web – "Усесвітня Павутина") – система навігації, пошуку і доступу до гіпертекстових і мультимедійних ресурсів Інтернет в реальному масштабі часу. Глобальною її можна рахувати тому, що, на відміну від звичайного (локального) *гіпертексту*, посилання на документ в нім (здійснювана одним або декількома клацаннями миші) може привести не тільки до іншого документа (як в локальному *гіпертексті*), але і до іншого комп'ютера (WWW-серверу), можливо, в іншій півкулі. Робота ведеться за допомогою універсальної програми-клієнта, яка дозволяє об'єднати в єдине ціле клієнта і сервер. Для доступу до WWW-серверу (інформації на нім) необхідно знати адресу сервера, наприклад, адреса http://www.mark-itt. ru – сервер із списком російських WWW-серверів, http (HyperText Transfer Protocol) – протокол роботи з гіпертекстом. Є система автоматичного пошуку по певних ключах (запитам, розділам). Інформація в WWW представлена у вигляді гіпертекстового документа, що включає різні типи даних (текст, графіка, відео, аудіо, посилання на інші гіпертекстові документи і так далі). Такі документи називають WWW-сторінками (WWW-pages). Ці сторінки є видимими за допомогою броузеров, спеціальних програм для навігації по мережі. Сторінки зберігаються на комп'ютерах-вузлах мережі, які називають сайтами (site). Кожен комп'ютер має своя унікальна IP-адрес URL (Uniform Resource Locator – універсальний локатор ресурсів), за допомогою якої броузер знає, де знаходиться інформація і що треба з нею робити. Сторінка – основний елемент WWW. На них знаходиться та інформація, яку ми шукаємо в мережі, або посилання на цю інформацію. Сторінки, гіпертекст – це легка і швидка у використанні, надзвичайно могутня система зв'язаних ключових слів і фраз (посилань), що дозволяє посилатися на інші ключові слова і фрази інших сторінок. Ці посилання зазвичай виділені іншим кольором, і досить просто клацнути мишкою по виділеному посиланню, щоб перейти до інформації, на яку посилає це посилання. Для створення гіпертекстових застосувань (наприклад, особистої WWW-сторінки) використовується спеціальна мова HTML (HyperText Markup Language), що дозволяє створювати гіпертекстовий документ в будь-якому текстовому редакторові формату ASCII, з підключенням графічних файлів двох основних форматів GIF, JPEG.

На думку ряду дослідників, наступною після Web формою колективного співіснування комп'ютерів буде Grid, яка дасть користувачам більше можливостей для роботи з видаленими машинами. Якщо World Wide Web можна порівняти з аналоговою телефонною мережею, здатною передавати тексти, аудіо і відео, то Grid подібна до сучасної системи електропостачання, що надає споживачам стільки ресурсів, скільки їм необхідно. В даний час в Церне ведуться роботи за визначенням стандартів для Grid. Так само, як і WWW, нова концепція, насамперед, буде затребувана в дослідницьких кругах. Учені за допомогою Grid діставатимуть доступ до ресурсів, необхідних для вирішення їх завдань. Архітектура Grid трехслойна: інтерфейс, шар додатків і операційна система Grid, що дозволяє підключити користувачів до розподілених ресурсів.

**Нейро-математичні і нейро-інформаційні технології і мережі.**

***Нейротехнології –*** технології на базі моделей, методів, алгоритмів, програм, що моделюють, імітують нейронні мережі і процеси вирішення завдань *штучного інтелекту*; дозволяють ефективно реалізовувати паралелізм, самонавчання, розпізнавання і класифікацію, адаптивність, перебудову структури, топології.

***Приклад***. Ідентифікація особи в кримінології; вибір дій, що управляють, в складних системах; геологорозвідка; діагностика в сейсмології. Нейросистема VNS-736, наприклад, дозволяє обробляти (розпізнавати) зображення розміром 512 512 елементи.

***Приклад***. Одним з поширених зарубіжних нейромереж є пакет Brain Maker. Хай необхідно вирішити задачу прогнозу ціни закриття на сьогоднішніх торгах по валютних тримісячним ф'ючерсним контрактам. Хай нас влаштовує точність прогнозу, при якій правильно указується ціновий тренд (під'їм, спад) і зміна ціни з точністю не нижче 90% від останнього стрибка. Застосування нейронної мережі починається з підготовки вхідних даних: курс долара, індекс інфляції, ставка міжбанка, біржові індекси, об'єм торгів, кількість операцій, максимальні і мінімальні ціни і ін. Після попереднього налаштування мережі починається ітераційний процес навчання, в результаті якої нейромережа настроює свою логічну структуру для точної реакції ринку на ті або інші дії. Для цього в пакеті Brain Maker передбачений могутній аналітичний блок, який дозволяє побачити, які параметри роблять позитивний вплив на ситуацію, а які – негативне. Потім мережа знову навчається і далі тестується на якість і адекватність, і після вдалого тестування використовується для прогнозів. За десять біржових днів мережа жодного разу не помилилася в знаку відхилення ф'ючерсних котирувань, а дев'ять днів з десяти відхилення прогнозу від реальної ціни склало менше 10 рублів. BrainMaker – це програма, з якою почалася історія застосування нейронних мереж в Росії. У

цьому пакеті на професійному рівні реалізована класична багатошарова нейронна мережа. Це єдина програма, в якій є можливість налаштування всіх параметрів нейронних мереж і алгоритмів навчання. Останнім часом BrainMaker найчастіше використовується не як самостійна програма, а як надбудова до програми TradeStation для аналізу в режимі реального часу. NeuroShell, хоча і є універсальною програмою, але завдяки тому, що вона була першою русифікованою нейромережевою програмою із зручним інтерфейсом, їй вдалося завоювати широке розповсюдження на російському ринку. Для вирішення фінансових завдань NeuroShell має модуль ринкових індикаторів, що дозволяє використовувати більше 20 індикаторів технічного аналізу при роботі з нейромережею.

***Приклад***. Відоме сімейство російських програм NeuroScalp побудоване за модульним принципом. Базовим модулем є модуль класичного технічного аналізу, в який інтегруються додаткові модулі, що реалізовують різні методи аналізу фінансових ринків. В даний час доступні наступні додаткові модулі: "Експертний модуль, російський ринок акцій" – модуль, що містить готові нейромережі для трьох російських акцій: РАО Газпром, РАО ЄЕС Росії, НК Лукойл; "Модуль Нейронних мереж" – емулятор класичних багатошарових нейронних мереж з використанням генетичних алгоритмів; "Модуль Карти Кохонена" – модуль, що реалізовує карти Кохонена в додатку до фінансових ринків; "Модуль Статистика" – модуль статистичної обробки фінансової інформації і аналізу ринку. NeuroScalp має зручний інтерфейс і реалізує необхідну множина методів, потрібну для реалізації різних ідей користувача. Демо-версию нейропакета можна знайти за адресою http://www.tora-centre.ru/nscalp.

**Технологія віртуальної реальності, віртуальна реальність.**

***Технологія віртуальної реальності, віртуальна реальність –*** технології актуалізації різних гіпотетичних середовищ і ситуації, що не існують реально і можливих як варіанти розвитку реальних аналогів систем реального миру; ці технології і системи дозволяють управляти віртуальним об'єктом, системою шляхом моделювання законів простору, часу, взаємодії, інерції і ін.

Вища форма розвитку *комп'ютерного офісу* – ***віртуальний офіс*** і ***віртуальна корпорація –*** офіси і корпорації, що не існують в звичайному, класичному вигляді ("що мають вивіску, штат, будівлю"), а створений уявний, розподілений – як в просторі, так і в часі (відділи і співробітники можуть знаходитися навіть на різних континентах, спілкуючись по роботі за допомогою ЕОМ і мереж зв'язку). Вони є вищим ступенем ділової співпраці і в корені міняють організацію робіт і систему інформаційного забезпечення співробітників.

Віртуалізація властивостей і атрибутів корпорації, динамічне (а іноді і віртуальне!) виділення їх загальних фундаментальних (родових, кла-

сових) властивостей, їх опис і використання в рамках єдиної технології, дозволяє скоротити проміжок між прогнозованим (або імітованим) станом корпорації і його реальним станом. У корпорації загальні інтереси з'являються, актуалізуються, віртуалізуються на той період, коли вони служать загальній меті. Успіх *віртуальної корпорації* (її розробки, впровадження і супроводи) залежить від повноти і якості інформаційних потоків між об'єктами корпорації. При цьому нові співробітники корпорації (або співробітники нової корпорації) мають справу, насамперед, з інформаційними моделями, наприклад, з моделями складу сировини, виробу, менеджера, постачальника, банку (банківських розрахунків). Таким чином, *віртуальні* комп'ютерні *корпорації* підтримують широкий спектр робіт і послуг – починаючи від з'єднання підприємств по ресурсах, виробництві, збуті, постачанні, управлінні, інформаційному забезпеченню і так далі, забезпечуючи базову компетентність – сукупність знань і умінь по організації, координації і узгодженню загальних організаційних, матеріальних, інформаційних ресурсів елементів корпорації, підтримки і розвитку її інфраструктури (архітектури, інновацій, активів, реклами і ін.) і закінчуючи забезпеченням корпоративних функцій системи з меншими витратами і якісніше, зменшуючи бюрократизм і рівні ієрархії в системі, а також час реакції на зміни на ринку.

Основні характеристики *віртуальної корпорації*: наявність основного виду бізнесу (діяльності) для всіх підсистем; концентрація користувачів навколо цього виду діяльності і загальні взаємозв'язані цілі, планування і ресурсозабезпечення, загальні (інтегровані) стратегії поведінки і актуалізації ресурсів, загальна (інтегрована) технологія актуалізації ресурсів.

У *віртуальній корпорації "*Нові робочі або службовці" в основному матимуть справу з інфологічними моделями робочих місць, інструментів, сировини, постачальників, ринків збуту і послуг, "нові менеджери" прийматимуть оперативні і більш інтелектуальні рішення по набагато ширшому кругу питань, а "нове керівництво" буде зайнято багатокритерійними стратегічними проблемами. Отже, від них вимагається якісно новий рівень професіоналізму, відповідальності і прагнення до самоосвіти.

***Приклад***. *Віртуальна* маркетингова *корпорація "*Да Вінчі" об'єднує ряд гірничорудних родовищ, виробничі (машинобудівні і будівельні), транспортні, інвестиційні, екологічні системи. Всі підсистеми "Да Вінчі" поставляються без доопрацювань під конкретний об'єкт (як дитячі конструктори збірно-розбірного типу). Один з сценаріїв, пропонованих в проекті (Venture Managment Model), моделює нижченаведену ситуацію. Гірничодобувна компанія веде розробки в Новій Гвінеї. Побудований в цій місцевості готель може бути розширений для обслуговування зростаючого потоку ділових клієнтів цієї компанії, а також туристів. Консорціуму, що має бізнес у сфері комунікацій і готельних послуг, пропонується пайова

участь в розвитку цієї місцевості і експлуатації готелю. Для зниження накладних витрат на розширення готелю і інфраструктури туризму притягуються крупні будівельні компанії (на умовах пайової участі в прибутках). Відзначимо при цьому, що критерії ефективності бізнесу в такому складі – різні, а процес ухвалення стратегічних рішень зв'язаний з конфліктними інтересами партнерів, їх картиною, що динамічно змінюється. Для реалізації цієї корпорації є *електронна* (мультимедійна) *пошта* для підтримки процесів ухвалення рішень першими особами, засоби *телеконференцій* для функціональних підрозділів і аналітиків, геоінформаційна система, САПР, що взаємодіє з *СУБД* через структуру даних з просторовою прив'язкою, система комп'ютерного діловодства на всіх етапах. Використовуються сучасні технології типу *"клієнт-сервер"* і об'єктно-орієнтовані під Windows NT, Windows-95 (робочі місця), Unix (сервер), повні версії MS Office і комп'ютерний документообіг. У системі електронного документообігу використовуються: повнотекстовий пошук, доступ до проектної документації на всіх етапах життєвого циклу проекту, підготовка інтерактивної технічної документації. Документ може містити текст, наприклад, HTML-документ, ілюстрації в одному або декількох шарах, редакторські правки і коментарі учасників різних *робочих груп*, що беруть участь в проекті, тривимірні об'єкти з програм САПР, що підключаються до документа відео- і аудіофайли.

***Приклад***. Технології віртуальної реальності широко використовують різні тренажери для навчання пілотів літаків, водіїв автомобілів, капітанів судів, які дозволяють поміщати навчаного у відповідні уявні ситуації (включаючи і аварійні), у тому числі і що ніколи не існували в реальності і не "укладаються" в рамки законів класичної механіки, фізики; ефекти віртуальної реальності створюються часто за рахунок одночасної дії на різні органи чуття, включаючи підсвідомість, сенсомоторику. Цікаві проекти створення хірургічних тренажерів з використанням методів і засобів віртуальної реальності. Важливою формою віртуальної реальності (віртуального поняття) є ринок. Якщо раніше під ринком розумілося реальне місце зустрічі продавців і покупців, то тепер це поняття складається з економічних, комерційних, виробничих і комунікаційних стосунків і систем; вони тепер можуть зустрічатися і реалізовувати свої функції в комп'ютерних системах.

**Когнітивні технології**

***Когнітивні технології –*** методи, засоби і прийоми, що забезпечують візуальне, гіпермедійне представлення умов завдань і/або наочної області, яке допомагає знаходити або стратегію рішення (або само рішення), або дозволяє оцінювати і порівнювати шляхи рішення, прийняти той або інший адекватний вибір.

***Приклад***. Когнітивна графіка, що дозволяє геометрично, образно представляти наочне середовище і побудувати, виходячи з цього, необхідний графічний об'єкт, зокрема, просторове представлення цього об'єкту. Є і засоби, і методи візуального програмування (проектування програм), зокрема, середовище VISUAL-C. Когнітивні методи вибору рішень в області бізнесу дозволяють ухвалювати рішення і визначати стратегії поведінки на основі якісних даних, особових думок (ефективно для ліквідації невизначеностей). Наприклад, модель ухвалення рішень Інституту США і Канади РАН, використовуючи блоки, подмоделі типу "Мир", "Цінність", "Засіб", "Інтерес", "Стереотип", "Мета", "Сценарій", "Проблема", дозволяє змінювати змістовне наповнення цих блоків, генерувати нові цілі і сценарії (використовуючи старі).

**Технології інформаційного реінжиніринга**

***Технології інформаційного реінжиніринга –*** методи і засоби корінного перегляду, перепроектували інформаційних мереж і процесів з метою досягнення різких, наприклад, "порядкових" поліпшень в ключових показниках інформаційних мереж і систем, зокрема, за показниками типу "продуктивність-вартість", "час-об'єм інформації", "функціонування-документація", "технологія-зручність" і ін. Рєінжинірингові заходи змінюють роботу (з монопланової вона стає багатоплановою), роль працівника (від підконтрольного виконавця – до ухвалення самостійних рішень), оцінку ефективності роботи і оплати праці (від оцінки трудовитрат – до оцінки результату), роль менеджера (від контролюючої функції – до тренінгової), і, найголовніше, організаційну структуру (від ієрархічної – до матричної і мережевої).

***Приклад***. Скорочення часу ухвалення рішення і циклу підготовки і підписання документів, наприклад, засобами комп'ютерного діловодства, мережами зв'язку і експертними системами, які забезпечують доступ керівника, що ухвалює рішення, до всіх етапів, вузлів і інструментаріїв підготовки рішення; перенесення акценту з проблеми "Як робити?" на проблему "Що робити?". Відзначимо, що популярну в інформаційних системах, мережах технологію "видалений сервер даних і клієнтський доступ" можна рахувати реінжирінговою. Рєінжінірінг системи підготовки доповідей, звітів, наприклад, може звести цикл підготовки звіту з 20-30 операцій до 5-10. Просте удосконалення не може дати таких результатів.

**Об'єктно-орієнтовані технології, технології об'єктно-орієнтованого аналізу.**

***Об'єктно-орієнтовані технології, технології об'єктно-орієнтованого аналізу*** (технології уявлення і актуалізації інформації, інформаційних процесів, систем як сукупність об'єктів і класів з використанням наступних понять: об'єкт, екземпляр класу – все те, що може бути повно описане деякими атрибутами стану; клас – сукупність

об'єктів з однаковими атрибутами; інкапсуляція – утаєння внутрішньої інформації, можливість відділення об'єктів і класів від зовнішнього світу; спадкоємство – можливість створювати з класів-батьків нові класи-нащадки, що зберігають атрибути і властивості батьків; поліморфізм – здатність об'єктів вибирати метод уявлення на основі типів даних, повідомлень, що актуалізуються).

Інструменти об'єктно-орієнтованого аналізу: атрибути (опису об'єктів, класів); операції (процеси, вживані до класів об'єктів); потоки даних (групи елементів даних, що реалізовують зв'язки між об'єктами); спадкоємство (агрегація і узагальнення).

***Приклад***. Об'єктно-орієнтовані середовища програмування, наприклад, C++, Smalltalk; об'єктно-орієнтований інжиніринг або набір прийомів і методів проектування бізнесу, що найефективніше забезпечують задані цілі і прибуток; об'єктно-орієнтований призначений для користувача інтерфейс, що використовує, наприклад, поняття "Клас описів", "клас мов", "клас операційних середовищ" і ін. При об'єктно-орієнтованому програмуванні в середовищі мови Apl, наприклад, процедури виконуються відповідно до логіки і інструкцій деякої програми, яка визначає послідовність і зміст дій; виконання цієї програми ініціюється за допомогою повідомлення, що посилається заданому об'єкту користувачем, іншою програмою або об'єктом. Одержувач повідомлення вирішує, яка програма буде виконана.

***Приклад***. HTML – статичний засіб. Щоб "пожвавити" вміст Web, зробити інтерактивні HTML-сторінки, використовується середовище JavaScript. Однією з важливих для інформатики об'єктно-орієнтованих систем є Java-система, середовищ – Java-інтерпретаторна машина, технологій – Java-технологія. Розглянемо їх коротко. У основі всіх їх лежить мова програмування Java, орієнтована на мережу Internet і сервери WWW. Мова Java відбулася від мови програмування Oak, з синтаксисом, близьким до синтаксису мови C++. Засобами мови Java можна розробляти додатки для різних платформ: Intel Pentium, Macintosh, Sun і ін. Java-програми бувають автономного використання (виконувані в режимі інтерпретації на конкретній комп'ютерній платформі) і аплети, (applets), виконувані в режимі інтерпретації віртуальною Java-машиною, яка вбудована практично у всі сучасні браузери. Аплети Java вбудовуються в документи HTML, що зберігаються на сервері WWW. За допомогою аплетов можна зробити сторінки сервера Web динамічними і інтерактивними. Всі дані для обробки аплеты можуть отримати тільки від сервера Web. Мова Java є об'єктно-орієнтованою і має об'ємну бібліотеку класів, застосувань, що значно спрощують розробку, оскільки програміст більше уваги може приділити функціональній частині додатку, а не організації інтерфейсу,

динамічних масивів і тому подібне У широкому сенсі, Java – це технологія, спочатку розрахована на інтеграцію з мережевим Web-середовищем, повністю незалежним від платформи. Віртуальна Java-машина – машина, на якій початкові Java-програми інтерпретуються в коди цієї машини. Це робить Java-середовище могутнім і зручним засобом розробки клієнтських компонентів Web-систем. У Java-середовищі користувач може здійснювати динамічне завантаження об'єктів з мережі, тобто йому не потрібні дорогі роботи по наладці, адмініструванню клієнтських Java-систем, оскільки для забезпечення роботи клієнта на новій версії досить завантажити її на сервері. Є інструментальні середовища, наприклад, Java Studio, що дозволяють проектувати додатки взагалі без програмування, з готових компонент, встановлюючи між ними зв'язку і відношення відповідно до внутрішньої логіки додатку. Для підвищення продуктивності Java-приложений в браузерах використовується компіляція Just-In-Time compilation ("на льоту"). При першому завантаженні аплета його код транслюється в звичайну виконувану програму, яка зберігається на диску і запускається. В результаті загальна швидкість виконання аплета збільшується у декілька разів.

**Середовищо-орієнтовані технології**

***Середовищо-орієнтовані технології*** (інтерактивні технології проектування, розробки, актуалізації інформаційних систем, в яких спершу будується потрібне середовище, інструментарій, а потім відбувається їх автоматизоване налаштування за допомогою виконання процедур типу: перемістити, вставити, видалити, вказати, активізувати і др.; готові середовища, "як кубики", об'єднуються в потрібні структури, а потім настроюються на конкретні класи проблем або користувачів, причому зміни одні з них можуть змінювати та інші).

***Приклад***. Середо-орієнтовані системи програмування, в яких часто використовується "віконний інтерфейс", "віконне середовище". Вони грунтуються на поняттях "вікно", "рамка", "фрейм", "малюнок на екрані", "віконне меню" і ін., кожен з яких асоціюється з найбільш відповідним інструментальним середовищем: тексти – з текстовим процесором, таблиці – з електронною таблицею, графіки – з середовищем ділової графіки і так далі До цих типових середовищ можуть бути додані також і розроблені самим програмістом середовища, а також бібліотеки середовищ. Відзначимо, що інтерактивне планування дозволяє знаходити оптимальні структури і набір ресурсів для досягнення поставленої мети.

Останніми роками виріс інтерес до розподілених систем – програмних комплексів, складові частини яких функціонують на різних комп'ютерах в мережі, використовуючи при взаємодії технології різного рівня, від безпосереднього використання пакетів TCP/IP до технологій з високим рівнем абстракції, таких, наприклад, як CORBA. У цих системах забезпечені

наступні можливості, нездійсненні при використанні традиційних технологій:

- ✧ масштабованість, тобто ефективне обслуговування різного числа клієнтів одночасне;
- ✧ надійність створюваних застосувань, тобто стійкість не тільки до помилок користувачів, але і до збоїв в системі комунікацій;
- ✧ безперервна тривала робота (режим 24 7, тобто 24 години протягом 7 днів тижня);
- ✧ високий рівень безпеки системи, тобто захисту і відстежування, протоколювання інформації на всіх етапах функціонування;
- ✧ висока швидкість розробки додатків і простота їх супроводу і модифікації (достатній середній рівень програміста).

Технологія CORBA створювалася некомерційною організацією – співдружністю розробників і користувачів програмного забезпечення OMG як універсальна технологія створення розподілених систем з використанням мов Java, 3, Ada, Smalltalk, Delphi, Perl, Python і ін. Клієнтська частина може бути написана на будь-якій мові програмування, підтримуючим CORBA.

Оскільки CORBA – стандартна інфраструктура розробки і використання різних платформ, ОС і додатків, то всі специфікації CORBA є повністю відкритими. CORBA реалізує високий рівень абстракції – всі проблеми і описи взаємодії з операційною системою або мережевими засобами здійснюються на низькому рівні і приховано від прикладного програміста. Високий рівень абстракції досягається за рахунок відображення інструкцій на мові специфікацій – на конкретну мову програмування. CORBA може передавати дані різних типів: структури, об'єднання і ін. Передбачена система опису і контролю типів. Для кожної мови використовується своє відображення даних на мові специфікацій. CORBA підтримує статичний і динамічний спосіб організації видалених викликів і має розвинені засоби отримання інформації про сервери. CORBA володіє високим рівнем стійкості до збоїв за рахунок більшої ізоляції клієнтів і серверів, автоматичного збереження стану об'єктів, могутнішої і продуманої схеми управління транзакціями. Управління транзакціями бере на себе так званий Сервіс Управління Транзакціями CORBA (Object Transaction Service). CORBA забезпечує високий рівень безпеки. Передбачена ідентифікація користувача, списки прав доступу до ресурсів, система аудиту і багато що інше. Інтеграція CORBA і мережі Інтернет виконується за рахунок використання протоколу, побудованого поверх TCP/IP, що дозволяє використовувати URL-имена як імена для Служби Іменування CORBA.

**CASE-технології**

***CASE-технології*** (Computer-Aided System Engineering автоматизоване проектування інформаційних систем, або технології, що дозволяють авто-

матизувати основні етапи і процедури життєвого циклу інформаційних систем: від аналізу початкового стану і цілей – до проектування інтерфейсів, звичних проектувальникові, користувачеві і основних процедур функціонування системи; чим більше етапів і процедур автоматизується, тим краще і швидше виходить інформаційна система, тим ширше за її застосування).

***Приклад***. Технологія STRADIS (STRategic Architecture for the Deployment of Information Systems – стратегічна архітектура для розгортання інформаційних систем) визначає і підтримує основні етапи життєвого циклу системи: цілі, їх пріоритети, вимоги до ресурсів, розподілу робіт, складу і змісту проектної документації, методика виконання процедур проектування і програмування, тестування і управління. Включає наступний інструментарій для цього: графічний редактор (графічне середовище), *СУБД*, засобу опису сценаріїв діалогу з системою, вихідних документів і ін.

**Технології і системи комп'ютерної алгебри**

*Технології* і системи *комп'ютерної алгебри*, системи символьних перетворень, аналітичних обчислень (системи, що дозволяють проводити автоматичні перетворення формул і виразів алгебри, зокрема, приведення подібних членів у виразі алгебри, знаходження первісної заданої аналітично функції, її диференціювання і так далі).

***Приклад***. Система Reduce для формульних перетворень, яка дозволяє як знаходити еквівалентні вирази алгебри, так і обчислювати їх чисельні значення (у тому числі і комплекснозначні), підсумовувати кінцеві і нескінченні ряди (сума – як функція!), проводити операції алгебри з поліномами, матрицями, інтегрувати і диференціювати.

**Нечіткі технології**

***Нечіткі технології*** (технології обробки даних і виведення знань, ухвалення рішень на основі опису систем апаратом нечітких множин і нечіткої логіки).

***Приклад.*** Медичний діагноз часто заснований на нечітких, невизначених чітко зв'язках симптомів і хвороб, їх нечіткій залежності, тому для комп'ютерної постановки діагнозу, побудови експертної системи постановки діагнозу ця технологія особливо ефективна, оскільки дозволяє робити нечіткі виводи, які потім можуть бути перевірені. Проблеми дактилоскопії також можуть бути вирішені ефективно за допомогою нечітких систем розпізнавання відбитків.

Все нові *інформаційні технології*, так або інакше, використовують методи і проблематику *штучного інтелекту*, *інженерії знань*, часто переплітаються і інтегруються.

Проблематику ***штучного інтелекту*** складають знання, інформація про дану область, які поки об'єктивно незрозумілі, неточні, не формалізуються, не структуруються, не актуалізуються доступними засобами (і можуть стати такими в процесі функціонування системи, придбання знань).

***Інженерія знань*** – наука, що вивчає проблеми виявлення, структуризації, формалізації і актуалізації знань для розробки різного типу інтелектуальних систем, технологій.

Все нові *інформаційні технологій* повинні забезпечувати цілеспрямованість, інформативність, адекватність, точність, повноту, сприйнятність і структурованість повідомлень, а також гнучкість, комфортність, своєчасність і простоту їх актуалізації в часі, в просторі і інформаційно.

Всі *інформаційні технології* – основа багато інших технології, а також спосіб актуалізації інформації, основа мислення.

**Основні тенденції розвитку** нових *інформаційних технологій*, незалежно від сфери їх використання:

- ✧ зростання ролі і активності (актуальності) інформаційного ресурсу, тобто якість і оперативність ухвалюваних інтелектуальних рішень в суспільстві у все більшого ступеня залежить від змісту, точності і своєчасності отримуваної інформації, її просторово-часових характеристик;
- ✧ розвиток здібності до активної технічної, програмної і технологічної взаємодії (стандартизації і сумісності таких взаємодій), тобто поява досконаліших стандартів взаємодії, все частіше – вже на рівні проектних робіт, на рівні розробки специфікацій;
- ✧ зміна структури інфологічних і структурних взаємодій, ліквідація проміжних ланок (безпосередність), тобто усунення етапів і функцій посередників інформаційного обміну і послуг, ліквідація проміжних функцій усередині компаній і між ними, ширше розповсюдження, спрощення доступу, зниження цін і т.д.;
- ✧ глобалізація або використання просторових, тимчасових і організаційних можливостей і ємкості інформаційного ринку (практично безмежного);
- ✧ конвергенція або формування ринку нових *інформаційних технологій*, що складається з основних сегментів, – приватне споживання (розваги, побутові послуги і тому подібне), забезпечення бізнесу (виробництво, продаж, маркетинг і тому подібне), інтелектуальна професійна робота (автоформалізація професійних знань і ін.).

***Приклад***. У 1990 р. близько 40% інтелектуальних працівників в США використовували на своїх робочих місцях нові *інформаційні технології*, зокрема, концерн Microsoft інвестує в нові медіапроекти до 20% свого на-

укового бюджету, випускаючи енциклопедії і довідники на CD, працюючи паралельно з нормальним телебаченням в мережі інтернет, відкриваючи в Інтернеті свої мультимедійні журнали, наприклад, Slate (http://www.slate.com) і ін. Виконання японської програми створення комп'ютерів п'ятого покоління стримується тим, що нова архітектура програмного забезпечення поки не поєднується з існуючими центрами *штучного інтелекту*, нові протоколи не можуть бути використані в старих системах зв'язку, а нові машинні мови не підходять для старих систем і так далі У банківській сфері зменшується роль дрібних банків, оскільки на впровадження *інформаційних технологій* потрібні значні ресурси. Що мають доступ до терміналів загального користування ("електронним кіоскам") замовляють товари і отримують електронні купони. Прикладом глобальної програми є програма ESPRIT (Європейська стратегічна програма досліджень в області *інформаційних технологій*). Прикладами апаратно-програмної конвергенції можуть бути принтери з функціями ксерокса і факсу.

Закінчуючи свій неповний огляд (повний огляд, мабуть, зробити неможливо) нових *інформаційних технологій*, найбільш важливих для системного аналізу, відзначимо, що з'являються все нові їх різновиди і додатки, а вони стають основним чинником (інструментарієм) глобалізації, чинником, що змінює традиційні критерії ухвалення рішення і можливості світового бізнесу (ціноутворення, витрати, місцеположення і так далі).

## КОНТРОЛЬНІ ПИТАННЯ

1. Як ви розумієте термін “управління знаннями”?
2. Які процеси включаються в управління знаннями?
3. Чим відрізняються один від одного поняття “дані”, “інформація” і “знання”?
4. Які способи класифікації знань ви знаєте?
5. Які моделі представлення знань вам відомі?
6. Наведіть приклад продукційної моделі представлення знань.
7. Дайте визначення семантичній мережі.
8. Що таке “фрейм”? Що таке “слот”?
9. Перерахуєте ознаки інтелектуальної поведінки. Які типи систем, заснованих на знаннях, вам відомі?
10. Дайте визначення експертної системи.
11. З яких основних елементів складається експертна система?
12. Які достоїнства і недоліки експертних систем?
13. У чому суть методології побудови баз даних?
14. Що таке “штучна нейронна мережа”?
15. Перерахуєте характеристики нейронної мережі?
16. Які типи нейронних мереж вам відомі?
17. Який алгоритм роботи системи, заснованої на прецедентах?
18. У якому виді знання представлене в штучній нейронній мережі?
19. У експертній системі? У системі, заснованій на прецедентах?
20. Що таке “предметна область”?
21. Які визначення поняття “База даних” вам відомі? Приведіть мінімум два.
22. Що таке “СУБД”?
23. З яких компонентів складається СУБД?
24. Які основні групи користувачів СУБД?
25. Що таке “модель даних”?
26. Які види класифікацій СУБД вам відомі?
27. Як СУБД класифікуються залежно від прийнятої моделі даних?
28. Що таке “розподілена СУБД”?
29. Чим клієнт-серверна СУБД відрізняється від вбудованої?
30. Які функції виконує СУБД?
31. Які моделі даних вам відомі?
32. Що таке “елемент даних”?
33. Що таке “запис”? Що таке “схема запису”?
34. Навіщо потрібне ключове поле?
35. Що таке “суть” в моделі “суть-зв’язок”? Наведіть приклад.
36. Які типи зв’язків вам відомі?

37. Чим відрізняється зв'язок один до багато чим від зв'язку багато до багато чим? Наведіть приклади.
38. Що таке "реляційна модель даних"? Ким вона була запропонована?
39. Яким умовам повинні задовольняти дані в реляційних таблицях? Що таке "первинний ключ"?
40. Що таке "зовнішній ключ", яка його роль у встановленні зв'язку між таблицями?
41. У чому полягає перша, друга і третя нормальні форми?
42. Перерахуєте переваги реляційної моделі.
43. Які причини виникнення постреляційних моделей СУБД?

## ТЕСТИ ДЛЯ САМОКОНТРОЛЮ

1. Що таке база даних?
   - Будь-який текстовий файл
   - Організована структура для зберігання інформації
   - Будь-яка інформація, представлена в табличній формі
   - Будь-яка електронна таблиця

2. Що таке SQL?
   - Мова розмітки бази даних
   - Структурована мова запитів
   - Мова програмування низького рівня
   - Мова програмування високого рівня
   - Мова розмітки гіпертексту

3. Яка база даних будується на основі таблиць і тільки таблиць?
   - Мережева
   - Ієрархічна
   - Реляційна

4. В якій моделі баз даних існують горизонтальні та вертикальні зв'язки між елементами
   - Мережній
   - Ієрархічній
   - Реляційній
   - Об'єктно-орієнтованій

5. Який з нижче перерахованих елементів не є об'єктом MS Access?
   - Таблиця
   - Книга
   - Запит
   - Макрос
   - Звіт

6. Штучний інтелект – це
- розділ інформатики, що займається формалізацією задач, які нагадують задачі, виконувані людиною
- розділ інформатики, що займається рішенням задач, які не здатна виконати людина
- розділ інформатики, що займається створенням та навчанням нейронних мереж

7. Нисхідний (семіотичний) підхід до проектування систем штучного інтелекту передбачає:
- створення символьних систем, моделюючих високорівневі психічні процеси: мислення, судження, мова, емоції, творчість і т.д.
- вивчення нейронних мереж і еволюційні обчислення, моделюючі інтелектуальну поведінку на основі більш менших "не інтелектуальних" елементів

8. Висхідний (біологічний) підхід до проектування систем штучного інтелекту передбачає:
- створення символьних систем, моделюючих високорівневі психічні процеси: мислення, судження, мова, емоції, творчість і т.д.
- вивчення нейронних мереж і еволюційні обчислення, моделюючі інтелектуальну поведінку на основі більш менших "не інтелектуальних" елементів

9. В яких роках почалися дослідження в області штучного інтелекту?
1940
1950
1960
1970
1980

10. Еволюційний підхід до створення систем ШІ передбачае
- здатність копіювати те, що роблять інші, без поділу на елементарні операції і формального опису дій
- спроби побудови ШІ шляхом моделювання структури людського мозку
- побудову початкової моделі, і правил, за якими вона може змінюватися

11. До загальних парадигм навчання нейромереж відносять:
- навчання "з вчителем"
- навчання "без вчителя"
- самонавчання
- змішане навчання (з вчителем та без вчителя)
- всі відповіді вірні

12. Експертна система – це:
- інтелектуальна комп'ютерна програма, що містить знання та аналітичні здібності одного або кількох експертів по відношенню до деякої галузі застосування і здатна робити логічні висновки на основі цих знань, тим самим забезпечуючи вирішення специфічних завдань без присутності експерта
- складні програмні комплекси, що займаються формалізацією задач, які нагадують задачі, виконувані людиною
- особливого роду база даних, розроблена для управління знаннями, тобто збором, зберіганням, пошуком і видачею знань

13. За якими ознаками класифікують експертні системи?
- за задачею
- за зв'язком з реальним часом
- за ступенем інтеграції з іншими програмами
- за кількістю користувачів
- за ціною

14. Мінімальний склад колективу розробників експертних систем
- користувач, експерт, інженер по знаннях, програміст
- експерт, інженер по знаннях, програміст
- експерт, програміст
- програміст

15. Відношенням можна вважати:
- будь-яку таблицю
- таблицю, яка відповідає певним умовам
- таблицю, в якій зберігаються тільки числові значення
- таблицю, в якій зберігаються тільки літерні значення

16. Зв'язування таблиць здійснюється за допомогою:
- ключа зв'язку
- індекса
- імен зв'язуваних таблиць
- пароля

17. Атрибут відношення, який однозначно ідентифікує кожен запис, називається:
- первинним ключем
- ключовим атрибутом
- зовнішнім ключем
- полем зв`язку
- тілом відношення

18. Концепція сховищ даних вперше було сформульована у
    - 1992 р
    - 1990 р
    - 2000 р
    - 2002 р

19. Сховище даних включає до свого складу такі компоненти:
    - система парольного захисту
    - реєстр облікових записів користувачів
    - драйвер ODBC
    - менеджер завантаження
    - менеджер сховища
    - детальні (оперативні) дані
    - агреговані дані
    - репозитарій метаданих

# ДОДАТКИ

## Додаток А
## Логічні основи цифрових ЕОМ

**Поняття та визначення алгебри логіки**

Вивчення роботи електронно-обчислювальної машини потребує математичного опису процесу функціонування її вузлів та пристроїв. Щоб вибрати відповідний математичний апарат, слід ще раз підкреслити, що ЕОМ обробляє двійково-кодовану інформацію, тобто інформацію, що подана за допомогою двох символів: 0 та 1. Це означає, що будь який вузол чи пристрій ЕОМ на вході та виході має деякі сукупності одиниць та нулів. Вхідна сукупність обробляється пристроєм у відповідності із законом його функціонування та видається на вихід також у вигляді сукупності 1 та 0.

Якщо позначити вхідний сигнал літерою X, а вихідний – літерою Y, у загальному випадку процес функціонування будь-якої частини ЕОМ можна записати у вигляді системи рівнянь таким чином:

$X = x_1, x_2, \ldots x_{12}$; $Y = y_1, y_2, \ldots y_p$, де x та y можуть приймати значення 1 або 0;

$$Y_i^S = f(x_1^S, x_2^S, \ldots x_n^S; a_1^S, a_2^S, \ldots a_p^S) \qquad (A.1)$$

де $a_1^S$ – змінні, що визначають внутрішній стан частини ЕОМ, що розглядається;

S – індекс, що позначає s-ий такт роботи схеми.

Процес перетворення однієї сукупності 1 та 0 в іншу вивчається спеціальною математичною дисципліною, що називається алгеброю логіки. Поява алгебри логіки як розділу математики пов'язана з роботами англійського вченого Дж.Буля, що опублікував у 1848р. її основи. Його роботи були спрямовані на формалізацію логічних висловлень, тобто на правила утворення складених висловлень, що є єдиними незалежно від конкретного змісту вихідних висловлень.

Під висловленням розуміють будь-яку фразу, судження, зауваження, запис і т.д., що можуть відповідати або не відповідати дійсності. У першому випадку висловлення називається істинним, у другому – хибним. Істинне висловлення можна позначити символом 0, хибне – символом 1. Таким чином, незважаючи на різноманітність висловлень, усі вони в алгебрі логіки можуть набувати тільки двох значень: 1 або 0. Існують висловлення, що можуть бути тільки істинними чи тільки хибним. Ці висловлення будемо називати простими. Значення простого висловлення не залежить від істинності чи хибності інших висловлень. Приклади: сніг – білий, Земля

– планета, кисень – хімічний елемент. З простих висловлень, які можна назвати логічними аргументами, за допомогою зв'язок І, АБО, ЯКЩО ТО та ін.. утворюють складені висловлення, які називають логічними функціями. Логічні функції будуть істиннми чи хибними в залежності від значень їх аргументів, тобто можуть приймати значення 0 чи 1. Прикладами складених висловлень (логічних функцій) можуть бути: "курсанти вивчають технічні і гуманітарні науки", "якщо електричне коло замкнене, то в колі протікає струм".

Загальні правила утворення логічних функцій розглядає алгебра логіки. В алгебрі логіки, як і у звичайній алгебрі, введено поняття змінної. Але, як випливає з характеристики висловлень, логічна змінна може мати лише два значення: 0 чи 1. Формалізуючи викладене про висловлення, розподілимо змінні на незалежні (аргументи) та залежні (функції). Логічна функція зберігає суть залежної змінної, але на відміну від функцій, що розглядаються у класичному математичному аналізі (алгебраїчні, тригонометричні, логарифмічні та ін..) може приймати лише два значення: 0 чи 1. Як прийнято у математиці, логічні функції можна задавати табличним і аналітичним способами. Графічний спосіб подання в алгебрі логіки не використовують.

В залежності від кількості аргументів логічні функції розподіляються на функції одного, двох, трьох та більше аргументів. Обмеження значень аргументів та функцій лише двома (0 та 1) призводить до обмеження кількості можливих логічних функцій. Кількість N логічних функцій n аргументів, що може бути утворена, визначається співвідношенням:

$$N = 2^{2^n} \qquad \text{(А.2)}$$

Наприклад, існує лише чотири логічні функції однієї змінної. Якщо n=2 (два аргументи), N=16.

Повертаючись до рівняння (А.1), можна зробити висновок, що воно повністю відповідає поняттю логічної функції. З цього випливає, що роботу будь-якого пристрою чи вузла ЕОМ можна математично описати в термінах алгебри логіки.

**Елементарні логічні функції**

Логічною функцією $P(x_1, x_2,\ldots,x_n)$ будемо називати будь-яку функцію n змінних $x_1, x_2,\ldots,x_n$, що як і змінні $x_i$ (аргументи) може приймати лише два значення: 0 чи 1.

Будь-яка логічна функція може бути подана в аналітичній чи табличній формі. Таблична форма має назву таблиці істинності. У цій таблиці враховують всі можливі комбінації значень логічних аргументів і подають значення функції для кожної комбінації.

Розглянемо логічні функції однієї змінної.

Вони подані у табл. А.1. Тут показані їх найменування та умовні позначення.

Таблиця А.1

**Логічні функції однієї змінної**

| Аргумент x | Функція P(x) | | | |
|---|---|---|---|---|
| | $P_0(x)=0$ Константа 0 | $P_1(x)=x$ повторення x | $P_2(x)=x$ інверсія x | $P_3(x)=1$ константа 1 |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |

Функція $P_0(x)=0$ завжди дорівнює нулю, тобто вона завжди хибна.

Функція $P_1(x)=x$ повторює значення аргумента x.

Функція $P_2(x)=x$ виявляє заперечення аргументу x, тому що має протилежні йому значення. Вона називається логічним запереченням, чи інверсією x, а коротко читається "не x". Тому її ще називають операцією "НЕ". Її позначають рискую над аргументами.

Функція $P_3(x)=1$, тобто завжди дорівнює одиниці, тому вона називається константою 1.

З розглянутого випливає, що найбільш цікавою функцією однієї змінної є функція логічного заперечення «НЕ» (інверсія). Зрозуміло, що $1=\overline{0}\ та\ 0=\overline{1}$. Якщо виконується подвійне заперечення, маємо:

$$\overline{\overline{1}}=1,\ \overline{\overline{0}}=0 \qquad (A.3)$$

Розглянемо більш детально логічні функції двох змінних.

**Кон'юнкція**

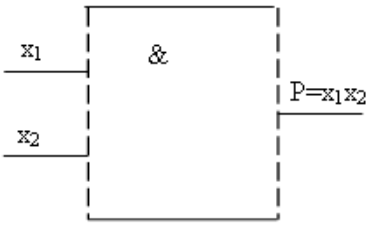
Логічна функція кон'юнкція є складним висловленням, що істинне лише тоді, коли істинними є прості висловлення, які її утворюють. Операція кон'юнкція позначається в алгебрі спеціальним символом &:

$$P=x_1\,\&\,x_2 \qquad (A.4)$$

Значення функції подані у табл. А.2.

Таблиця А.2.

| $x_1$ | $x_2$ | $x_3$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |




З таблиці випливає, що вони повністю співпадають з арифметичною функцією множення. Тому кон'юнкцію ще називають логічним множенням

і застосовують у її запису крапку:

$$P = x_1 \cdot x_2 \qquad \text{(A.5)}$$

Читається функція Р є $x_1$ і $x_2$, а тому скороченою назвою її є функція "І". На логічних Операціях кон'юнкція може бути поширена на будь-яке число позначення аргументів:

$$P = x_1 \cdot x_2 ... x_n \qquad \text{(A.6)}$$

Вона дорівнює 1 (є істинною) лише тоді, коли всі аргументи х дорівнюють одиниці (є істинними).

**Диз'юнкція**

Логічна функція диз'юнкція є складеним висловленням, що істинне лише тоді, коли істинним є хоча б одне просте висловлення з тих, що її утворюють. Вона позначається в алгебрі логіки символом V:

$$P = x \vee x \qquad \text{(A.7)}$$

Таблиця істинності для неї має вигляд (табл. А.3)

Таблиця А.3

| x | x | P |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

З таблиці випливає, що ця логічна операція схожа на арифметичне додавання. Тому її називають логічним додаванням. Читається функція так: "Р є х або х", тому скороченою назвою диз'юнкції є така функція "АБО". На логічних схемах диз'юнкція позначається:

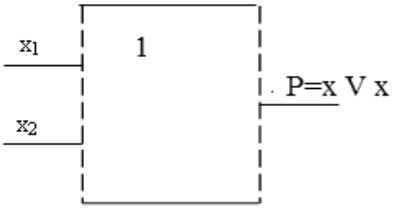



Рисунок А.1.

Операція диз'юнкція може бути виконана над будь-яким числом аргументів:

$$F = x \vee x \vee x ... \vee x \qquad \text{(A.8)}$$

Вона дорівнює нулю (є хибною) тільки тоді, коли всі аргументи х дорівнюють нулю.

**Функція Шеффера (штрих Шеффера)**

Ця логічна функція є хибною лише у тому випадку, коли істотними є обидва прості висловлення. Тому зона фактично є запереченням кон'юнкції і читається так: "Р є заперечення х і х". Формульний запис функції Шеффера має вигляд: Р=х\х. Відповідно таблиця істинності показана у табл. А.4.

Таблиця А.4

| $x_1$ | $x_2$ | P |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Умовне графічне позначення функції має два варіанти:

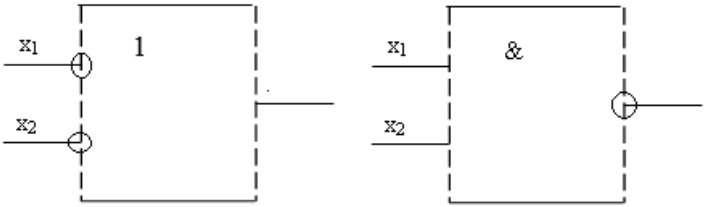



Рисунок А.2.

Обидва вони відображають факт логічного заперечення кон'юнкції, тому функцію Шеффера називають ще функцією "І-НЕ".

**Функція Пірса (стрілка Пірса)**

Функція Пірса істинна лише тоді, коли хибним є обидва прості висловлення. Вона записується так $P = x_1 \downarrow x_2$ таблиця істинності має вигляд (табл. А.5).

Таблиця А.5

| $x_1$ | $x_2$ | P |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

З таблиці випливає, що функція Пірса є запереченням диз'юнкції, тому вона коротко називається функцією "АБО-НЕ". Графічно функція Пірса позначається так:

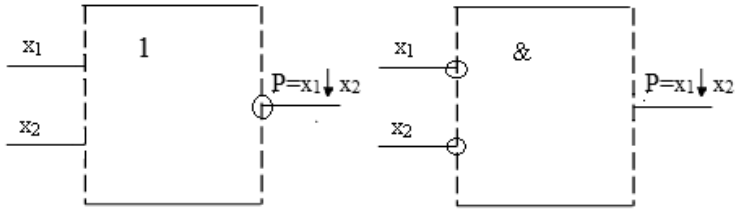


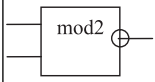
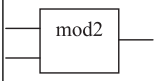

Рисунок А.3.

Розглянуті чотири функції двох аргументів дозволяють зробити певні висновки, що допоможуть вивчити інші логічні функції:

- Повне уявлення про логічну функцію дає її таблиця істинності.
- Для позначення логічних функцій використовуються нові незвичні символи.
- Між логічними функціями існують зв'язки, тобто одні логічні функції можна подати сукупністю інших логічних функцій.
- Будь-яка логічна функція графічно позначається прямокутником. Аргументи і функцію показують лініями без стрілок. Функція заперечення показується колом.
- Вивчити логічну функцію це значить: знати її назву, таблицю істинності, аналітичний запис (формулу) та графічне позначення.

Тому ще декілька функцій двох змінних подано у вигляді таблиці А.6.

Таблиця А.6.

| Назва логічної функції | Таблиці істинності | | | Читання функції | Аналітичний запис | Графічне позначення |
|---|---|---|---|---|---|---|
| | x | x | P | | | |
| Нерівнозначність двох висловлювань (додавання за моделем 2) | 0<br>0<br>1<br>1 | 0<br>1<br>0<br>1 | 0<br>1<br>1<br>0 | $x_1$ нерівнозначне $x_2$ | $P = x_1 + x_2$ |  |
| Рівнозначність двох висловлювань (еквівалентність) | 0<br>1<br>1 | 1<br>0<br>1 | 0<br>0<br>1 | $x_1$ рівнозначне $x_2$ | $P = x_1 \approx x_2$ |  |
| Імплікація двох висловлень | 0<br>0<br>1<br>1 | 0<br>1<br>0<br>1 | 1<br>1<br>0<br>1 | з $x_1$ випливає $x_2$ | $P = x_1 \to x_2$ | |
| Зворотня імплікація двох висловлювань | 0<br>0<br>1<br>1 | 0<br>1<br>0<br>1 | | з $x_2$ випливає $x_1$ | $P = x_1 \leftarrow x_2$ | |

**Форми подання логічних функцій**

Розглянуті логічні функції двох змінних показують, що між ними існує певний взаємозв'язок, тобто одні логічні функції можна утворити з інших. З цього випливає, що серед усіх функцій можна виділити деяку сукупність, за допомогою якої можна утворити всі інші логічні функції. Така сукупність називається функціонально повною системою логічних функцій. Існують різні функціонально повні системи логічних функцій. Прикладами можуть бути:

- Кон'юнкція, диз'юнкція, інверсія.
- Функція Пірса
- Функція Шеффера

Для кожної функціонально певних системи розроблені відповідні правила утворення складених логічних функцій та способи перетворення функцій. Ці правила складають суть алгебри логіки. Розглянемо закони алгебри логіки на прикладі історично першої функціонально-повної системи (І, АБО, НЕ), що були сформульовані Дж.Буллем (тому ця алгебра носить назву мулевої алгебри). Закони алгебри логіки частково співпадають із законами звичайної алгебри. Але є і відмінності, що з'являються внаслідок обмеженості аргументів та функцій лише двома значеннями: 0 та 1.

1. Закон комутативності.

Він стверджує, що зміна місць змінних не впливає на результат:

$$\begin{aligned} x_1 \vee x_2 &= x_2 \vee x_1 \\ x_1 \cdot x_2 &= x_2 \cdot x_1 \end{aligned} \qquad \text{(A.9)}$$

2. Закон асоціативності.

Він показує, що порядок виконання операцій не впливає на результат:

$$\begin{aligned} x_1 \vee (x_2 \vee x_3) &= (x_1 \vee x_2) \vee x_3 \\ x_1 \cdot (x_2 \cdot x_3) &= (x_1 \cdot x_2) \cdot x_3 \end{aligned} \qquad \text{(A.10)}$$

3. Перший закон дистрибутивності:

$$x_1 \cdot (x_2 \vee x_3) = (x_1 \cdot x_2) \vee (x_1 \cdot x_3) \qquad \text{(A.11)}$$

Ці закони є справедливими і для звичайної алгебри, тому вони відомі ще з середньої школи.

4. Другий закон дистрибутивності

$$x_1 \vee x_2 \cdot x_3 = (x_1 \vee x_2) \cdot (x_1 \vee x_3) \qquad \text{(A.12)}$$

Цей закон не має аналога в алгебрі, він є специфічним саме для алгебри логіки. Щоб перевірити його справедливість, розглянемо його дію на всіх можливих сукупностях аргументів x. Для цього зробимо таблицю істинності, де покажемо послідовно результати виконання операцій ліворуч та праворуч від знака рівняння (табл. А.7).

Таблиця А.7.

| $x_1$ | $x_2$ | $x_3$ | $x_2\ x_3$ | $x_1 \vee x_2x_3$ | $x_1 \vee x_2$ | $x_1 \vee x_3$ | $(x_1 \vee x_2)\cdot(x_1 \vee x_3)$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

З таблиці випливає, що дійсно $x_1 \vee x_2x_3=(x_1 \vee x_2)(x_1 \vee x_3)$

Обраний спосіб доведення за допомогою таблиці істинності можна використовувати під час вивчення інших законів алгебри логіки.

5. Закон інверсії (правило де-Моргана)

$$\overline{x_1 \vee x_2} = \overline{x_1} \cdot \overline{x_2}$$
$$\overline{x_1}\ x_2 = \overline{x_1} \cdot \vee\overline{x_2} \qquad \text{(А.13)}$$

6. Закон склеювання

$$x_1 x_2 \vee x_1 \overline{x_2} = x_1$$
$$(x_1 \vee x_2) \cdot (x_1 \vee \overline{x_2}) = x_1 \qquad \text{(А.14)}$$

Закон поглинання

$$x_1 \vee x_1 \cdot x_2 = x_1$$
$$x_1 \cdot (x_1 \vee x_2) = x_1 \qquad \text{(А.15)}$$

8. Правила операції з константами

$$\overline{0} = 1; \quad x \cdot 1 = x; \qquad x \vee 1 = 1$$
$$\overline{1} = 0; \quad x \cdot 0 = 0; \qquad x \vee 0 = x \qquad \text{(А.16)}$$

9. Правила операцій із замінною та її інверсією

$$x \vee \overline{x} = 1; \qquad x \cdot \overline{x} = 0 \qquad \text{(А.17)}$$

Перелічені закони алгебри логіки використовуються для еквівалентних перетворень складних логічних функцій і приведення їх до більш простого вигляду. У зв'язку з цим слід відкреслити, що відсутність у алгебрі логіки таких операцій, як віднімання та ділення, призводить до деяких особливостей:

- ✧ не можна переносити логічні доданки з однієї частини рівняння в іншу;
- ✧ не можна скорочувати однакові множними, оскільки при цьому результат може стати не еквівалентним вихідному запису.

З розглянутого випливає, що одна і та ж логічна функція може мати різне аналітичне подання. Тому постає задача знаходження такої аналітичної форми подання логічної функції, коли кожній функції буде відповідати одна і лише одна формула стандартного типу. Такі форми подання логічних функцій мають назву канонічних.

Перед тим, як розглядати утворення канонічних форм логічних функцій, нагадаємо, що одним із способів подання логічних функцій є табличний. У таблиці істинності записують всі можливі набори аргументів та значення логічної функції, що їх відповідають. Табличний спосіб подання логічної функції є наочним, але не дає змоги реалізувати її у вигляді логічної схеми. Тому виникає задача переходу до аналітичного запису логічної функції, поданої у вигляді таблиці істинності. Це й виконується за допомогою канонічних форм. На практиці застосовують дві канонічні форми: досконалу диз'юнктивну нормальну форму (ДДНФ) та досконалу кон'юнктивну нормальну форму (ДКНФ). Термін "досконала" підкреслює, що всі компоненти логічної функції утворюються із всіх її аргументів. Термін "нормальна" означає, що форма утворюється сполученням всіх змінних за допомогою лише двох операцій: кон'юнкції та диз'юнкції.

Складові частини канонічних форм мають назви конотитуенти одиниці та нуля. Конституентою одиниці є кон'юнкція (логічний добуток) усіх змінних, що беруться без інверсії, якщо вони істинні, чи з інверсією, якщо вони хибні. Тому конституанта одиниці на своєму наборі дорівнює одиниці та обертається в 0 на всіх інших наборах.

Конституентою нуля називається диз'юнкція (логічна сума) всіх змінних, що беруться без інверсії, якщо вони хибні, чи з інверсією, якщо вони істинні. Таким чином, конституанта нуля на своєму наборі дорівнює 0 та обертається в 1 на всіх інших наборах.

Позначаючи конституанту одиниці символом К (1), а конституанту нуля – К(0), наведемо приклади їх визначення на всіх наборах трьох змінних (табл. А.8).

Використовуючи введені поняття конституант одиниці та нуля, визначимо згадані вище досконалі нормальні форми подання логічних функцій.

Досконалою диз'юнктивною нормальною формою (ДДНФ) логічної функції будемо називати диз'юнкцію конституант одиниці на тих наборах змінних, де логічна функція дорівнює одиниці. Таким чином ДДНФ зовнішньою логічною операцією має диз'юнкцію, а операція кон'юнкція використовується під час утворення конституант одиниці.

Досконалою кон'юнктивною нормальною формою (ДКНФ) логічної функції будемо називати кон'юнкцію ноституент нуля на тих наборах змінних, де логічна функція дорівнює нулю. Це означає, що зовнішньою логічною операцією є кон'юнкція, а диз'юнкція використовується під час утворення коституент нуля.

Таблиця А.8.

| № набору | аргументи | | | Конституента одиниці К (1) | Конституента нуля К (0) |
|---|---|---|---|---|---|
| | x | x | x | | |
| 0 | 0 | 0 | 0 | $\overline{x_1}\,\overline{x_2}\,\overline{x_3}$ | $x_1 \vee x_2 \vee x_3$ |
| 1 | 0 | 0 | 1 | $\overline{x_1}\,\overline{x_2}\,x_3$ | $x_1 \vee x_2 \vee \overline{x_3}$ |
| 2 | 0 | 1 | 1 | $\overline{x_1}\,x_2\,\overline{x_3}$ | $x_1 \vee \overline{x_2} \vee x_3$ |
| 3 | 0 | 1 | 0 | $\overline{x_1}\,x_2\,x_3$ | $x_1 \vee \overline{x_2} \vee \overline{x_3}$ |
| 4 | 1 | 0 | 0 | $x_1\,\overline{x_2}\,\overline{x_3}$ | $\overline{x_1} \vee x_2 \vee x_3$ |
| 5 | 1 | 0 | 0 | $x_1\,\overline{x_2}\,x_3$ | $\overline{x_1} \vee x_2 \vee \overline{x_3}$ |
| 6 | 1 | 1 | 1 | $x_1\,x_2\,\overline{x_3}$ | $\overline{x_1} \vee \overline{x_2} \vee x_3$ |
| 7 | 1 | 1 | 1 | $x_1\,x_2\,x_3$ | $\overline{x_1} \vee \overline{x_2} \vee x_3$ |

Розглянемо утворення досконалих нормальних форм логічної функції на прикладі, коли функція задана своєю таблицею істинності (табл.А.9).

Таблиця А.9.

| № набору | аргументи | | | Функція $P(x_1;x_2;x_3)$ |
|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 1 |
| 3 | 0 | 1 | 1 | 0 |
| 4 | 1 | 0 | 0 | 0 |
| 5 | 1 | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |

Щоб утворити досконалу диз'юнктивну нормальну форму заданої логічної функції $P(x_1x_2x_3)$, здобудемо спочатку конституанти одиниці. З таблиці 1.15 випливає, що вони відповідають 1,2,6 та 7 наборами змінних. Відповідно до табл..1.14 конституенти одиниці на цих наборах мають вигляд:

$$K_1(1) = \overline{x_1}\,\overline{x_2}\,\overline{x_3}; \qquad K_2(1) = \overline{x_1}\,x_2\,\overline{x_3};$$
$$K_6(1) = x_1\,x_2\,\overline{x_3}; \qquad K_7(1) = x_1\,x_2\,x_3; \qquad \text{(А.18)}$$

Щоб краще зрозуміти ДДНФ, треба просто прочитати задану логічну функцію: вона дорівнює одиниці (є істинною) на 1 або на 2 або на 6 або на

7 наборі змінних. З цього безпосередньо випливає, що всі конституанти одиниці К (1) мають бути зв'язані операцією диз'юнкція:

$$P(x_1, x_2, x_3) = K_1(1) \vee K_2(1) \vee K_6(1) \vee K_7(1) = \overline{x_1}\,\overline{x_2}x_3 \vee \overline{x_1}x_2\overline{x_3} \vee x_1x_2\overline{x_3} \vee x_1x_2x_3 \quad \text{(A.19)}$$

Щоб визначити досконалу кон'юнктивну нормальну форму заданої логічної функції $P(x_1x_2x_3)$, утворимо спочатку її конституанти нуля. Вони відповідають 0, 3, 4 та 5 набору змінних (див.табл.9). Тому

$$\begin{aligned} K_0(0) &= x_1 \vee x_2 \vee x_3; \qquad & K_3(0) &= x_1 \vee \overline{x_2} \vee \overline{x_3}; \\ K_4(0) &= \overline{x_1} \vee x_2 \vee x_3; \qquad & K_5(0) &= \overline{x_1} \vee x_2 \vee \overline{x_3} \end{aligned} \quad \text{(A.20)}$$

Виконуючи над цими конституантами операцію кон'юнкція, здобудемо ДКНФ заданої логічної функції:

$$\begin{aligned} P(x_1x_2x_3) &= K_0(0) \cdot K_3(0) \cdot K_4(0) \cdot K_5(0) = \\ &= (x_1 \vee x_2 \vee x_3) \cdot (x_1 \vee \overline{x_2} \vee \overline{x_3}) \cdot (\overline{x_1} \vee x_2 \vee x_3) \cdot (\overline{x_1} \vee x_2 \vee \overline{x_3}) \end{aligned} \quad \text{(A.21)}$$

Таким чином, задана логічна функція трьох змінних $P(x_1x_2x_3)$ може бути подана двома еквівалентними формами:

ДДНФ (А.20) та ДКНФ (А.21)

Зрозуміло, що методика утворення досконалих нормальних форм логічної функції може бути поширена на функції будь-якої кількості аргументів.

**Структурний синтез логічних схем**

Апарат алгебри логіки використовують для математичного опису роботи вузлів та пристроїв ЕОМ. Логічні функції задані таблицею істинності або подані у ДДНФ чи ДКНФ, відображають процес функціонування певних схем ЕОМ. Тому треба розглянути процес переходу від логічних функцій до логічних схем, що їх реалізують. Будь-яка логічна схема складається з логічних елементів, що об'єднуються певними зв'язками. Логічним елементом будемо називати електронну схему, що реалізує конкретну логічну функцію. Оскільки вище розглядалася булева алгебра, основу якої складають три логічні функції: кон'юнкція, диз'юнкція та інверсія, надалі будемо вивчати логічні схеми, що будуються з елементів лише трьох типів кон'юнктор, диз'юнктор та інвертор ("І", "АБО", "НЕ").

Якщо логічна функція задана аналітично, то щоб побудувати логічну схему, що її реалізує, треба визначити потрібні логічні елементи та з'єднати їх зв'язками відповідно до вихідної логічної функції.

Наприклад, треба побудувати логічну схему, що реалізує логічну функцію

$$P(x_1x_2) = x_1\overline{x_2} \vee \overline{x_1}x_2 \quad \text{(A.22)}$$

З формули (А.22) випливає, що до логічної схеми входять два інвертори ("НЕ"), два кон'юнктори ("І") та один диз'юнктор ("АБО"). Формула

(1.16) визначає також порядок виконання логічних операцій: спочатку потрібно утворити інверсії обох змінних, потім виконати дві операції кон'юнкції, останньою буде диз'юнкція здобутих кон'юнкцій. Цей порядок треба відтворити і логічною схемою. Тому вона має вигляд, показаний на рис 1.4.

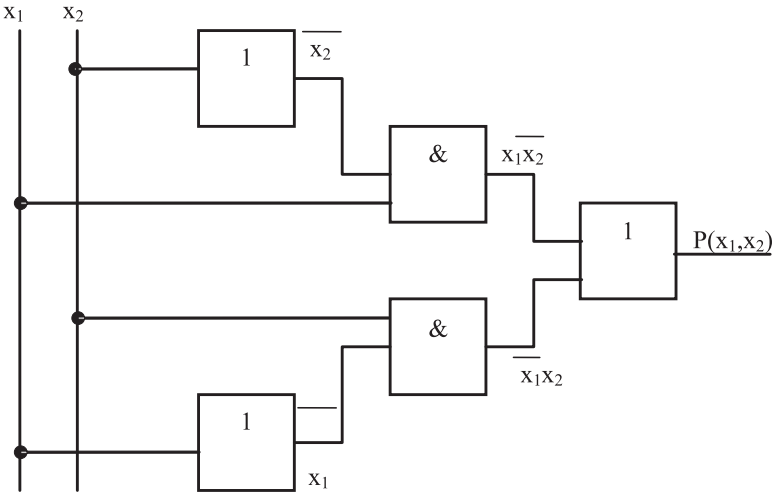



Рисунок А.4.

Таким чином, побудування логічної схеми за заданої логічної функції не викликає труднощів, якщо вірно визначити послідовність виконання логічних операцій. Ця послідовність у загальному випадку визначається, як це було і у звичайній алгебрі, дужками. Зрозуміло, що треба чітко знати графічні позначення логічних операцій. Але перед тим, як звертатися до логічної схеми, треба вибрати найпростішу з еквівалентних формул логічної функції.

Наприклад, для розглянутої логічної функції (А.22) існує декілька еквівалентних аналітичних формул:

$$P(x_1x_2) = x_1\overline{(x_1 \vee x_2)} \vee \overline{x_1}x_2 = x_1(\overline{x_2}x_1) \vee \overline{x_1}x_2 = x_1\overline{x_2} \vee x_1x_2 \qquad (A.23)$$

Зрозуміло, що доцільно реалізувати логічною схемою ту аналітичну форму логічної функції, яка потребує мінімальної кількості логічних елементів.

Процес визначення логічної функції у вигляді, що призводить до мінімуму потрібних логічних елементів, називається мінімізацією заданої логічної функції.

Задачу мінімізації логічної функції можна сформулювати так: задана певна логічна функція (таблицею істинності або аналітично). Треба

визначити найпростішу логічну формулу, еквівалентну логічній функції, що потребує для своєї реалізації мінімальну кількість змінних та логічних операцій над ними.

Всі методи мінімізації логічних функцій ґрунтуються на законах алгебри логіки. Вони можуть бути розподілені на аналітичний (або метод безпосередніх перетворень) та табличний (або метод площинних діаграм Вейча).

**Аналітичний метод мінімізації логічних функцій**

Аналітичний метод чи метод безпосередніх перетворень спрямований на таке перетворення аналітичної форми логічної функції, що веде до виділення та виключення зайвих змінних та їх інверсій. Зайвими змінними та зайвими елементарними операціями є такі, що не впливають на значення логічної функції. Виключення зайвих змінних та елементарних операцій виконується застосуванням законів склеювання та поглинання.

$$x_1 x_2 \vee x_1 \overline{x_2} = x_1(x_2 \vee \overline{x_2}) = x_1 \qquad (A.24)$$
$$x_1 \vee x_1 x_2 = x_1(1 \vee x_2) = x_1$$

Розглянемо конкретний приклад аналітичних перетворень, посилаючись на конкретні закони алгебри логіки, (закони будемо зазначати за їх номерами так, як вони викладені у 1.3.3.).

***Приклад 1***. Мінімізувати логічну функцію

$$P(x_1, x_2, x_3) = x_1 \overline{x_2} x_3 \vee x_1 \overline{x_2}\,\overline{x_3} \qquad (A.25)$$

1. Відповідно до закону 3 маємо:

$$P(x_1 x_2 x_3) = x_1 \overline{x_2}(x_3 \vee \overline{x_3}) \qquad (A.26)$$

2. За правилом 9 $x_3 \vee x_3 = 1$. Тому $P(x_1 x_2 x_3) = x_1 \overline{x_2}$

***Приклад 2***. Мінімізувати логічну функцію

$$P(x_1 x_2) = x_1 x_2 \vee x_1 \overline{x_2} \vee x_1 \qquad (A.27)$$

$$P(x_1 x_2) = x_1(x_2 \vee x_2 \vee 1) = x_1(1 \vee 1) = x_1 \qquad (A.28)$$

Закон 3

***Приклад 3***. Мінімізувати логічну функцію

$$P(x_1 x_2 x_3) = x_1 x_2 x_3 \vee x_1 x_2 \overline{x_3} \vee x_1 \overline{x_2} x_3 \vee x_1 \overline{x_2}\,\overline{x_3} \vee \overline{x_1} x_2 \overline{x_3} \qquad (A.29)$$

Функцію утворюють шість елементарних кон'юнкцій. Згрупуємо їх так: першу з другою, третю з четвертою, п'яту з шостою і винесемо за

дужки спільні змінні:

$$P(x_1x_2x_3) = x_1x_2(x_3 \vee x_3) \vee x_1\overline{x_2}(x_3 \vee \overline{x_3}) \vee \overline{x_1}x_2(x_3 \vee \overline{x_3}) \qquad \text{(A.30)}$$

оскільки $x_3 \vee x_3 = 1$ (правило 9), маємо:

$$P(x_1x_2x_3) = x_1x_2 \vee x_1\overline{x_2} \vee \overline{x_1}x_2 = x_1(x_2 \vee \overline{x_2}) \vee \overline{x_1}x_2 = x_1 \vee \overline{x_1}x_2 \qquad \text{(A.31)}$$

Звертаючись до закону 4 (дистрибутивності), одержимо:

$$P(x_1x_2x_3) = x_1 \vee \overline{x_1}x_2 = (x_1 \vee \overline{x_1})(x_1 \vee x_2) \qquad \text{(A.32)}$$

Тому остаточна форма заданої функції буде мати вигляд:

$$P(x_1x_2x_3) = x_1 \vee x_2 \qquad \text{(A.33)}$$

Останній приклад показує, що безпосередні перетворення потребують певних навиків. Шлях перетворень суттєво залежить від вибору елементарних кон'юнкцій, що групуються. Щоб вибрати найкоротший шлях, треба мати певний досвід роботи а логічними функціями та добре знання алгебри логіки. Тому на практиці більш поширеним є табличний метод.

**Табличний метод мінімізації логічних функцій**

Табличний метод чи метод площинних діаграм Вейча вигідно відрізняється своєю зручністю та наочністю. Він є найбільш ефективним методом мінімізації логічних функцій, якщо кількість змінних не перевищує чотирьох.

При використанні табличного метода мінімізації таблиць істинності заданої логічної функції трансформується у іншу таблицю, що має назву діаграми Вейча. Ця таблиця складається з $N = 2^n$ клітинок, де $n$ – кількість змінних логічної функції, тобто кількість клітинок дорівнює кількості наборів змінних. Клітинки розташовані таким чином, що кожні дві суміжні відрізняються одна від іншої значенням лише одної змінної. Кожна клітинка відповідає одній конституенті одиниці. Наприклад, для логічної функції двох змінних діаграма вейча має $N = 2^n = 4$ клітинки. Діаграма виглядає так:

Таблиця А.10

| | $\overline{x_2}$ | $x_2$ |
|---|---|---|
| $\overline{x_1}$ | | |
| $x_1$ | | |

Для функції трьох змінних діаграма складається з N=2=8 клітинок. Щоб виконати умову про суміжні клітинки (повинні відрізнятися значенням лише одної змінної) та залишити діаграму площиною, клітинки розміщуються відповідно до значень змінних, таким чином:

Таблиця А.11

| | $\overline{x_2}\,\overline{x_3}$ | $\overline{x_2}x_3$ | $x_2x_3$ | $x_2\overline{x_3}$ |
|---|---|---|---|---|
| $\overline{x_1}$ | | | 1 | 1 |
| $x_1$ | 1 | 1 | | |

тобто по горизонталі кожній клітинці відповідає кон'юнкція двох змінних. Треба звернути увагу на те, що крайні горизонтальні клітинки ($x_2x_3$ та $x_2x_3$) теж є суміжними.

Суть табличного методу мінімізації полягає в тому, що на діаграмі показують всі конституанти одиниці заданої логічної функції, відмічають суміжні по вертикалі чи горизонталі клітинки, в яких стоять конституанти, та об'єднують їх за такими правилами.

**Правило 1.** Якщо в двох суміжних клітинках стоять конституанти одиниці, вони замінюються одною конституантою, що має ранг на одиницю менший та включає змінні з однаковими показниками інверсії.

Це правило можна пояснити, звернувшись до аналітичного запису функції. Двом суміжним клітинкам відповідають дві конституанти, що відрізняються значеннями лише однієї змінної. Наприклад, для логічної функції двох змінних ними можуть бути: $x_1x_2$ та  Для кожної пари можна використовувати закон дистрибутивності (винести змінну за дужки) та правило 3: . Тому кожна пара конституант, об'єднана диз'юнкцією, замінюється змінною, що виноситься за дужки:

$$\begin{aligned} x_1x_2 \vee x_1\overline{x_2} &= x_1(x_2 \vee \overline{x_2}) = x_1 \\ x_1x_2 \vee \overline{x_1}x_2 &= x_2(x_1 \vee \overline{x_1}) = x_1 \\ \overline{x_1}x_2 \vee x_1\overline{x_2} &= x_2(\overline{x_1} \vee x_1) = \overline{x_2} \end{aligned} \qquad \text{(А.34)}$$

Цей факт відображений у сформульованому вище правилі 1.

**Правило 2.** Якщо конституанти одиниці стоять у чотирьох (восьми) суміжних клітинках по горизонталі чи вертикалі або утворюють квадрат (прямокутник), то відповідна четвірка (вісімка) конституант замінюється конституантою, що має ранг на дві (три) одиниці нижчий та включає змінні з однаковими показниками інверсії.

Ще правило також можна пояснити, розглядаючи аналітичні перетворення диз'юнкцій елементарних кон'юнкцій за законами алгебри логіки.

Розглянемо приклади мінімізації логічних функцій за допомогою діаграм Вейча.

***Приклад 1***. Логічна функція задана своєю таблицею істинності (табл.А.12). Треба її мінімізувати.

Таблиця А.12

| аргументи | | | Функція $P(x_1;x_2;x_3)$ |
|---|---|---|---|
| $x_1$ | $x_2$ | $x_3$ | |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Заданій логічній функції відповідає формула:

$$P(x_1,x_2,x_3) = \overline{x_1}x_2\overline{x_3} \vee \overline{x_1}x_2x_3 \vee x_1\overline{x_2}x_3 \vee x_1\overline{x_2}x_3 \qquad \text{(А.35)}$$

Запишемо конституанти одиниці функції у діаграму Вейча (табл.А.13). Це можна зробити, звертаючись до аналітичного запису, чи безпосередньо до таблиці істинності.

Таблиця А.13

| | $\overline{x_2}\,\overline{x_3}$ | $\overline{x_2}x_3$ | $x_2x_3$ | $x_2\overline{x_3}$ |
|---|---|---|---|---|
| $\overline{x}$ | | | 1 | 1 |
| $x$ | 1 | 1 | | |

З діаграми випливає, що є дві пари елементарних конституант одиниці, що розташовані у суміжних клітинках: Застосовуючи до них правило 1 здобудемо

$$P(x_1,x_2,x_3) = \overline{x_1}x_2 \vee x_1\overline{x_2} \qquad \text{(А.36)}$$

Щоб впевнитися у результаті, продублюємо його аналітичними перетвореннями:

$$P(x_1,x_2,x_3) = \overline{x_1}x_2\overline{x_3} \vee \overline{x_1}x_2x_3 \vee x_1\overline{x_2}\,\overline{x_3} \vee x_1\overline{x_2}x_3 = \overline{x_1}x_2(\overline{x_3} \vee x_3) \vee x_1\overline{x_2}(\overline{x_3} \vee x_3) = \overline{x_1}x_2 \vee x_1\overline{x_2} \qquad \text{(А.37)}$$

Обидва результати співпадають. Це означає, що мінімізація виконана вірно.

***Приклад 2***. Мінімізувати логічну функцію трьох змінних, що задана аналітично:

$$P(x_1, x_2, x_3) = \overline{x}_1 x_2 x_3 \vee \overline{x_1 x_2 x_3} \vee x_1 x_2 x_3 \vee x_1 x_2 \overline{x}_3 \vee x_1 \overline{x_2 x_3} \vee \overline{x_1 x_2 x_3} \qquad \text{(A.38)}$$

Запишемо конституанти одиниці функції у діаграму Вейча.

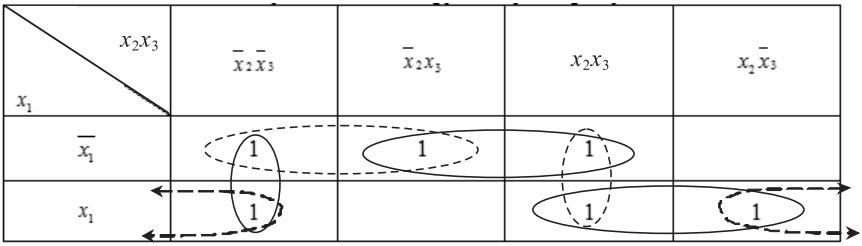



Рисунок А.5.

З діаграми випливає, що є декілька варіантів групування суміжних клітинок. Два з них (суцільною та пунктирною лініями) показані на діаграмі. Тому здобудемо:

$$P_1(x_1 x_2 x_3) = \overline{x_2 x_3} \vee x_1 x_3 \vee x_1 x_2$$
$$P_2(x_1 x_2 x_3) = x_1 \overline{x_3} \vee \overline{x_1 x_2} \vee x_2 x_3 \qquad \text{(A.39)}$$

Таким чином, у загальному випадку можливо одержання декількох мінімальних форм логічної функції в залежності від того, як були подані у групи суміжні клітинки діаграми.

***Приклад 3***. Мінімізувати логічну функцію трьох змінних, що задана аналітично.

$$P(x_1 x_2 x_3) = \overline{x_1 x_2} x_3 \vee x_1 \overline{x_2} x_3 \vee x_1 x_2 x_3 \vee x_1 x_2 x_3 \qquad \text{(A.40)}$$

Запишемо конституанти одиниці у діаграму Вейча

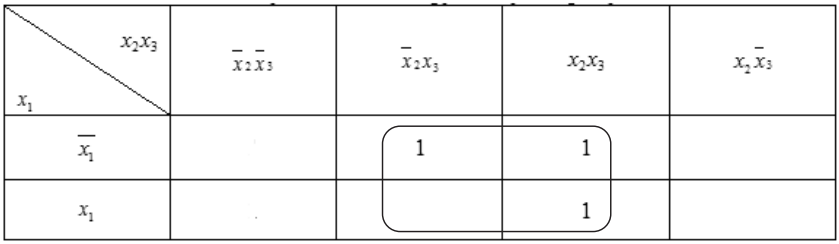



Рисунок А.6.

Використовуючи правило 2. здобудемо: $P(x_1x_2x_3) = x_3$.

Нагадаємо, що відповідно до цього правила у функції залишаються лише ті змінні, що мають однаковий показник інверсії. Такою зміною у наведеній функції є х.

Підбиваючи підсумки викладеному, можна перелічити основні етапи синтезу логічних схем:

- Завдання умов функціонування логічної схеми у вигляді таблиці істинності.
- Здобуття аналітичного запису логічної функції у вигляді досконалих диз'юнктивної чи кон'юнктивної нормальних форм.
- Виконання мінімізації логічної функції.
- Побудова логічної схеми, що реалізує мінімізовану логічну функцію.

Зрозуміло, що обсяг навчального посібника не дозволяє розглянути всі деталі перелічених етапів.

# Додаток Б
# Основні вбудовані статистичні функції MS Excel

| *Функція* | | | *Опис* |
|---|---|---|---|
| *MS Office 2003* | *MS Office 2007* | *MS Office 2010* | |
| **СРОТКЛ** | | | Повертає середнє абсолютне значення відхилень точок даних від середньої |
| **СРЗНАЧ** | | | Повертає середнє арифметичне аргументів |
| --- | **СРЗНАЧЕСЛИ** | **СРЗНАЧЕСЛИ** | Повертає середнє значення (середнє арифметичне) всіх чарунок в діапазоні, що задовольняє задану умову |
| --- | **СРЗНАЧЕСЛИМН** | **СРЗНАЧЕСЛИМН** | Повертає середнє значення (середнє арифметичне) всіх чарунок в діапазоні, що задовольняє декілька умов |
| **ДОВЕРИТ** | **ДОВЕРИТ** | **ДОВЕРИТ.НОРМ** | Повертає довірчій інтервал для середнього значення генеральної сукупності |
| **КОРРЕЛ** | | | Повертає коефіцієнт кореляції між двома сукупностями даних |
| **СЧЁТ** | | | Підраховує кількість чисел в переліку аргументів |
| **СЧИТАТЬПУСТОТЫ** | | | Підраховує кількість порожніх чарунок в діапазоні |
| **СЧЁТЕСЛИ** | | | Підраховує кількість чарунок в діапазоні, що задовольняє задану умову |
| --- | **СЧЁТЕСЛИМН** | **СЧЁТЕСЛИМН** | Підраховує кількість чарунок в діапазоні, що задовольняє декілька умов |
| **КВАДРОТКЛ** | | | Повертає суму квадратного відхилення |
| **ПРЕДСКАЗ** | | | Повертає значення лінійного тренду |

| *Функція* | | | *Опис* |
|---|---|---|---|
| *MS Office 2003* | *MS Office 2007* | *MS Office 2010* | |
| **ЧАСТОТА** | | | Повертає розподіл частот у вигляді вертикального масиву |
| **СРГЕОМ** | | | Повертає середнє геометричне |
| **РОСТ** | | | Повертає значення у відповідності до експоненціального тренду |
| **СРГАРМ** | | | Повертає середнє гармонійне |
| **ЭКСЦЕСС** | | | Повертає ексцес сукупності даних |
| **НАИБОЛЬШИЙ** | | | Повертає k-те найбільше значення в сукупності даних |
| **ЛИНЕЙН** | | | Повертає параметри лінійного тренду |
| **ЛГРФПРИБЛ** | | | Повертає параметри експоненціального тренду |
| **МАКС** | | | Повертає найбільше значення в списку аргументів |
| **МЕДИАНА** | | | Повертає медіану заданих чисел |
| **МИН** | | | Повертає найменше значення в сукупності аргументів |
| **МОДА** | **МОДА** | **МОДА. ОДН** | Повертає значення моди сукупності даних |
| **ПИРСОН** | | | Повертає коефіцієнт кореляції Пірсона |
| **ПЕРСЕН-ТИЛЬ** | **ПЕРСЕН-ТИЛЬ** | **ПРОЦЕНТ ИЛЬ.ВКЛ** | Повертає k-ту персентиль для значення діапазону |
| **КВАРТИЛЬ** | **КВАРТИЛЬ** | **КВАРТИ-ЛЬ.ВКЛ** | Повертає квартиль сукупності даних |
| **РАНГ** | **РАНГ** | **РАНГ.СР РАНГ.РВ** | Повертає ранг числа в списку чисел |
| **КВПИРСОН** | | | Повертає квадрат коефіцієнта кореляції Пірсона |
| **СКОС** | | | Повертає асиметрію розподілу |
| **НАКЛОН** | | | Повертає нахил лінії лінійної регресії |
| **НАИМЕНЬШИЙ** | | | Повертає k-те найменше значення в сукупності даних |

| *Функція* | | | *Опис* |
|---|---|---|---|
| *MS Office 2003* | *MS Office 2007* | *MS Office 2010* | |
| СТАНДОТКЛОН | СТАНДОТКЛОН | СТАНДОТКЛОН.В | Оцінює стандартне відхилення за вибіркою |
| СТАНДОТКЛОНП | СТАНДОТКЛОНП | СТАНДОТКЛОН.Г | Розраховує стандартне відхилення в генеральній сукупності |
| СТОШҮХ | | | Повертає стандартну похибку прогнозних значень Y для кожного значения X в регресії |
| СТЬЮДРАСП | СТЬЮДРАСП | СТЬЮДЕНТРАСП.ПХ | Повертає t-розподіл Стьюдента. |
| ТЕНДЕНЦИЯ | | | Повертає значення у відповідності до лінійного тренду |
| ДИСП | ДИСП | ДИСП.В | Оцінює дисперсію за вибіркою |
| ДИСПР | ДИСПР | ДИСП.Г | Розраховує дисперсію для генеральної сукупності |

# Додаток В
# Список фінансових функцій Excel 2003

| Назва функції | Стислий опис функції |
|---|---|
| **АМОРУВ** | Повертає пропорційну амортизацію майна для кожного звітного періоду |
| **АПЛ** | Повертає величину амортизації активу за один період, розраховану лінійним методом |
| **АСЧ** | Повертає величину амортизації активу за даний період, розраховану методом суми річних чисел |
| **БЗРАСПІС** | Повертає майбутнє значення основного капіталу після нарахування складних відсотків |
| **БС** | Повертає майбутню вартість інвестиції на основі періодичних постійних (рівних по величині сум) платежів і постійної процентної ставки |
| **ВСД** | Повертає внутрішню ставку прибутковості для ряду потоків грошових коштів, представлених чисельними значеннями |
| **ДАТАКУПОНДО** | Повертає попередню дату купона перед датою угоди |
| **ДАТАКУПОНПОСЛЕ** | Повертає наступну дату купона після дати угоди |
| **ДДОБ** | Повертає значення амортизації активу за даний період, використовую метод подвійного зменшення залишку або інший явно вказаний метод |
| **ДЛІТ** | Повертає щорічну тривалість дії цінних паперів з періодичними виплатами по відсотках |
| **ДНЕЙКУПОН** | Повертає число днів в періоді купона, який містить дату угоди |
| **ДНЕЙКУПОНДО** | Повертає кількість днів від початку періоду купона до дати угоди |
| **ДНЕЙКУПОНПОСЛЕ** | Повертає число днів від дати угоди до терміну наступного купона |
| **ДОХІД** | Повертає дохід від цінних паперів, який складає періодичні процентні виплати |
| **ДОХОДКЧЕК** | Повертає дохід по казначейському чеку |
| **ДОХОДПОГАШ** | Повертає річний дохід від цінних паперів, який складає дохід в строк набирання чинності |

| Назва функції | Стислий опис функції |
|---|---|
| **ДОХОДПОСЛЕРЕГ** | Повертає дохід по цінних паперах з останньою процентною виплатою |
| **ДОХОДСЬКИДКА** | Повертає щорічний дохід по знижених в ціні цінних паперах. Наприклад, по казначейському чеку |
| **ІНОРМА** | Повертає процентну ставку для повністю інвестованих цінних паперів |
| **КПЕР** | Повертає загальну кількість періодів виплати для інвестиції на основі періодичних постійних виплат і постійної процентної ставки |
| **МВСД** | Повертає внутрішню ставку прибутковості для ряду періодичних грошових потоків, враховуючи як витрати на залучення інвестиції, так і відсоток, що отримується від реінвестування грошових коштів |
| **МДЛІТ** | Повертає модифіковану тривалість Макалея для цінних паперів з передбачуваною номінальною вартістю 100грн. |
| **НАКОПДОХОД** | Повертає накопичених дохід по цінних паперах з періодичною виплатою відсотків |
| **НАКОПДОХОДПОГАШ** | Повертає накопичений дохід по цінних паперах, відсоток по яких виплачується в строк набирання чинності |
| **НОМІНАЛ** | Повертає номінальну річну процентну ставку |
| **ОБЩДОХОД** | Повертає загальну виплату по позиці між двома періодами |
| **ОБЩПЛАТ** | Повертає загальну виплату, проведену між двома періодичними виплатами |
| **ОСПЛТ** | Повертає величину платежу в погашення основної суми по інвестиції за даний період на основі постійності періодичних платежів і постійності процентної ставки |
| **ПЛТ** | Повертає суму періодичного платежу для аннуитета на основі постійність сум платежів і постійності процентної ставки |
| **ОТРИМАНО** | Повертає суму, отриману в строк набиранні чинності повністю забезпечених цінних паперів |
| **ПРОЦПЛАТ** | Обчислює відсотки, що виплачуються за певний інвестиційний період |

| Назва функції | Стислий опис функції |
|---|---|
| **ПРПЛТ** | Повертає суму платежів відсотків по інвестиції за даний період на основі постійності сум періодичних платежів і постійності процентної ставки |
| **ПС** | Повертає приведену (до теперішнього моменту) вартість інвестиції – загальну суму, яка на справжній момент рівноцінна ряду майбутніх виплат |
| **ПУО** | Повертає величину амортизації активу для будь-якого вибраного періоду, зокрема для часткових періодів, з використанням методу подвійного зменшення залишку або іншого вказаного методу |
| **РАВНОКЧЕК** | Повертає еквівалентний облігації дохід по казначейському чеку |
| **РУБЛЬ.ДЕС** | Перетворить ціну в рублях, виражену у вигляді дробу, в ціну в рублях виражену десятковим числом |
| **РУБЛЬ.ДРОБЬ** | Перетворить ціну в рублях, виражену десятковим числом, в ціну в рублях, виражену у вигляді дробу |
| **ЗНИЖКА** | Повертає норму знижки для цінних паперів |
| **СТАВКА** | Повертає процентну ставку по аннуитету за один період. Наприклад при річній процентній ставці в 6% для квартальної ставки використовується значення 6%/4 |
| **ФУО** | Повертає величину амортизації активу для заданого періоду розраховану методом фіксованого зменшення залишку |
| **ЦІНА** | Повертає ціну за 100 грн. номінальної вартості цінних паперів, по яким виплачується періодичний дохід |
| **ЦЕНАКЧЕК** | Повертає ціну за 100 грн. номінальної вартості для казначейського чека |
| **ЦЕНАПЕРВНЕРЕГ** | Повертає ціну за 100 грн. номінальної вартості цінних паперів з нерегулярним першим періодом |
| **ЦЕНАПОГАШ** | Повертає ціну за 100 грн. номінальної вартості цінних паперів, дохід по яких виплачується в строк набирання чинності |
| **ЦЕНАПОСЛНЕ-РЕГ** | Повертає ціну за 100 грн. номінальної вартості цінних паперів з нерегулярним останнім періодом |

| Назва функції | Стислий опис функції |
|---|---|
| **ЦЕНАСЬКИДКА** | Повертає ціну за 100 грн. номінальної вартості цінних паперів, на які зроблена знижка |
| **ЧИСЛКУПОН** | Повертає кількість купонів, які можуть бути сплачені між датою угоди і терміном набирання чинності |
| **ЧИСТВНДОХ** | Повертає внутрішню суму доходу для розкладу грошових надходжень |
| **ЧИСТНЗ** | Повертає чисту поточну вартість інвестиції, що обчислюється на основі ряду періодичних надходжень готівки і норми амортизації |
| **ЧПС** | Повертає величину чистої приведеної вартості інвестиції, використовуючи ставку дисконтування і вартості майбутніх виплат (негативні значення) і надходжень (позитивні значення). |
| **ЕФЕКТ** | Повертає щорічні поточні процентні ставки |

## Список джерел


1. Кримінальний кодекс України від 05 квітня 2001 р. № 2341-III, ст. № 361 [Електронний ресурс] – Режим доступу: http://zakon1.rada.gov.ua/laws/show/2341-14.
2. Про захист інформації в автоматизованих системах: закон України від 05 липня 1994 р. № 80 // Відомості Верховної Ради України. – 1994. – № 31, С. 287.
3. Про інформацію: закон України від 02 жовтня 1992 р. № 2567 // Відомості Верховної Ради України. – 1992. – № 48.
4. Про Концепцію Національної програми інформатизації: закон України від 4 лютого 1998 р. № 75/98 // Відомості Верховної Ради України. – 1998. – № 27-28, С. 114.
5. Про науково-технічну інформацію: закон України від 25 червня 1993 р. № 3322 – XII // Відомості Верховної Ради України. – 1993. – № 33, С. 345.
6. Указ Президента України від 31 липня 2000 р. № 928/2000 "Про заходи щодо розвитку національної складової глобальної інформаційної мережі Інтернет та забезпечення широкого доступу до цієї мережі в Україні" [Електронній ресурс] – Режим доступу: http://zakon4.rada.gov.ua/laws/show/928/2000.
7. Указ Президента України від 31 травня 1993 р. № 186/93 "Про державну політику інформатизації України" [Електронний ресурс] – Режим доступу: http://zakon1.rada.gov.ua/laws/show/186/93/.
8. Ахрамович В.М. Інформаційна безпека: [навч. посіб] / В.М. Ахрамович. – К.: ДП "Інформ. – аналіт. Агенство", – 2009. с.
9. Башмаков А.И. Интеллектуальные информационные технологи / А.И. Баш-маков, И.А. Башмаков. – М.: Вид-во МГТУ им. Н.Э. Баумана, 2005. – 304 с.
10. Брукшир Дж. Информатика и вычислительная техника / Дж. Брукшир; 7-е изд.– СПб. : Питер, 2004.– 624 с.
11. Введение в криптографию / [под общей ред. В.В. Ященко]. – Питер: СПб., 2001. – 288 с.
12. Вєртузаев М.С., Юрченко О.М. Захист інформації в комп'ютерних системах від несанкціонованого доступу: [навч. посібник / за ред. С.Г. Лаптева]. – К.: Видавництво Європейського університету, 2001. – 201 с.
13. Воробьев А.В. Курс: методы и средства защиты компьютерной информации. Юнита 1. Основы построения систем защиты информации. Дистанционное образование [рабочий учебник] / А.В. Воробьев, Т.В. Воробьева. – М.: "Современный Гуманитарный Институт", 2003. – 96 с.

14. Воробьев А.В. Системы искусственного интеллекта Юнита 1. Искусственный интеллект и экспертные системы [рабочий учебник] / А.В. Воробьев, Т.В. Воробьева. – М.: Вид-во СГУ, 2003 – 131 с.
15. Гаврилова Т.А. Базы знаний интеллектуальных систем / Т.А. Гаврилова, В.Ф. Хорошевський. – Спб.: Питер, 2001. – 384 с.
16. Гайдамакин Н.А. Автоматизированные информационные системы, базы и банки данных. Вводный курс: [учебное пособие] / Н.А. Гайдамакин. – М.: Гелиос АРВ, 2002. – 368 с.
17. Дейт К.Дж. Введение в системы баз данных / К.Дж. Дейт; пер с англ. – К.; М.; СПб.: Издательский дом "Вильямс", 1999. – 844 с.
18. Дейтел Г. Введение в операционные системы / Г. Дейтел. – М.: Мир, 1987.
19. Джексон П. Введение в экспертные системы [учеб. пособ.] / П. Джексон; пер. з англ. – М.: Издательский дом "Вільямс", 2001. – 624 с.
20. Дженнифер Н.Р. Web-дизайн: [учебно-справочное издание] / Н.Р Дженнифер; пер. с англ. – М.: КУДИЦ-ПРЕСС, 2008. – 816 с.
21. Домарев В.В. Безопасность информационных технологий. Системный поход / В.В. Домарев. – К.: ТОВ "ТВД "ДС", 2004. – 992 с.
22. Дунаев В.В. Основы WEB-дизайна [самоучитель] / В.В. Дунаев. – СПб.: БХВ-Петербург, 2007. – 512 с.
23. Економічна інформатика / [Стельмах В.С., Єпіфанов А.О., Сало І.В. та ін.]. – Суми. Видавництво "Слобожанщина", 2000. – 260 с.
24. Есипов А.С. Информатика и информационные технологи для учащихся школ и колледжей / А.С. Есипов. – СПб. : Питер, 2004. – 470 с.
25. Зацеркляний М.М. Основи комп'ютерних технологій для економістів: [навч. посіб.] / М.М. Зацеркляний, О.Ф. Мельников, В.М. Струков. – К.: ВД "Професіонал", 2006. – 672с.
26. Информатика. Базовий курс [под ред. С.В. Симоновича]. – 2-е изд. – СПб.: Питер, 2005. – 640 с.
27. Информатика: [Учебник для вузов] / Макарова Н.В., Волков В.Б. – СПб.: Питер, 2011. – 576 с: ил.
28. Інформаційні системи в менеджменті: [підручник / В.О. Новак, Ю.Г. Симоненко, В.П. Бондар, В.В. Матвеев]. – К.: Каравела; Піча Ю.В, 2008. – 616с.
29. Кейпингерт П. Элементы операционных систем / П. Кейпингерт. – М.: Мир, 1985.
30. Куроуз Дж. Компьютерные сети / Дж. Куроуз, К. Росс. – 4-е изд. – Питер: СПб., 2004.
31. Мамченко С.Д. Економічна інформатика: [практикум, навч. посіб.] / С.Д. Мамченко, В.А. Одинець. – К.: Знання, 2008. – 710с.
32. Методы и средства защиты информации. В 2-х томах / [С.В. Ленков, Д.А. Перегудов, В.А. Хорошко]; под ред. В.А. Хорошко. – К.: Арий, 2008. – 464 с. – Том I. Несанкционированное получение информации.

33. Нельсен Я. Web-дизайн: удобство использования Web-сайтов. / Я. Нельсен, Х. Лоранжер; пер. с англ. – М.: ООО "И.Д.Вильямс", 2007. – 368с.
34. Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы: [учебник для вузов] / В.Г. Олифер, Н.А. Олифер. – 4-е изд. – Питер: СПб., 2006. – 958 с.
35. Основи інформатики та обчислювальної техніки [навч. посіб. / за загал. ред. В.Г. Іванова]. – К.: Юрінком Інтер, 2004.
36. Основы современных компьютерных технологий [учебник / под ред. А.Д. Хоменко]. – СПб.: Корона-принт, 2005.
37. Пауэлл Т. Web-дизайн / Т. Пауэлл; 2-е изд., перераб. и доп; пер. с англ. – СПб.: БХВ-Петербург, 2005. – 1072 с.
38. Симонович С. Практическая информатика. Универсальный курс / С. Симонович, Г. Евсеев. – М.: АСТ-Пресс, Инфорком-Пресс, 2002. – 480 с.
39. Смолин Д. В. Введение в искусственный интеллект: [конспект лекций] / Д.В. Смолин. – М.: ФИЗМАТЛИТ, 2004. – 208 с.
40. Соловьева Л.Ф. Сетевые технологи [учебник-практикум] / Л.Ф. Сольвьева. – СПб.: БХВ-Петербург, 2004. – 416 с.
41. Столингс В. Передача данных / В. Столингс. – 4-е изд. – Питер: СПб., 2004.
42. Таненбаум Э. Компьютерные сети / Э. Таненбаум. – 4-е изд. – Питер: СПб., 2002.
43. Хорошко В.О. Методы и средства защиты информации / В.О. Хорошко, А.А. Чекатков. – К.: Издательство Юниор, 2003.– 504 с.

# Інформаційні джерела INTERNET


44. Антивірусний відділ НТЦ "КАМИ" [Електронний ресурс] – Режим доступу: http://www.avp.ru.
45. Аудіо-засоби [Електронний ресурс] – Режим доступу: http://www.aes.org/www-links.
46. Бібліотека комп'ютерних технологій [Електронний ресурс] – Режим доступу: http://www.fas.sfu.ca/cs/librarv.
47. Бібліотека посилань з питань, що мають відношення до PC [Електро-нний ресурс] – Режим доступу: http://www.mtp.semi.harris.com/pc info.html.
48. Відео-засоби [Електронний ресурс] – Режим доступу: http://www.heartlab.rri.uwo.ca/vidfag/videofag.html.
49. Дисководи, інтерфейси [Електронний ресурс] – Режим доступу: http://therf.c3d.rl.af/mil.
50. Жорсткі диски, CD [Електронний ресурс] – Режим доступу: http://www.cs.vorku.ca/People/frank.
51. Комп'ютерна графіка [Електронний ресурс] – Режим доступу: http://mambo.ucsc.edu/psl/cg.html.
52. Модеми [Електронний ресурс] – Режим доступу: http://www.rosenet.net/~costmo.
53. Новини PC [Електронний ресурс] – Режим доступу: http://home.zcinet.eom/home/filters/news.html.
54. Офіційний сайт Packard Bell [Електронний ресурс] – Режим доступу: http://www.packardbell.com.
55. Офіційний сайт компанії Hewlett-Packard [Електронний ресурс]. – Режим доступу: http://www.hp.com.
56. Офіційний сайт компанії IBM [Електронний ресурс] – Режим доступу: http://www.ibm.com.
57. Офіційний сайт компанії Inte [Електронний ресурс] – Режим доступу: http://www. intel.com.
58. Офіційний сайт компанії Microsoft (сторінка російською мовою) [Електронний ресурс] – Режим доступу: http://www.microsoft.com/rus.
59. Офіційний сайт компанії Microsoft [Електронний ресурс] – Режим доступу: http://www.microsoft.com.
60. Предметний покажчик з питань комп'ютерної безпеки [Електронний ресурс] – Режим доступу: http://www.telstra.com.au/info/security.html.
61. Принтери [Електронний ресурс] – Режим доступу: http://www.printgrc.com/pdrivers.html.
62. Сервер інформаційних технологій [Електронний ресурс] – Режим доступу: http://www.citforum.ru.
63. Ткаченко В. Экономическая інформатика [Електронний ресурс] / В. Ткаченко. – Режим доступу: http://lessons-tva.info.
64. Web-вузол Compag [Електронний ресурс] – Режим доступу: http://www.cpi.nrjad.com.
65. Web-сервер "Диалог Наука" [Електронний ресурс] – Режим доступу: http://www.dials.ccas.ru

**Навчальне видання**

**Козлов Валерій Володимирович,**
**Сбітнєв Анатолій Іванович, Пашковська Анна Юріевна,**
**Томашевська Тетяна Володимирівна**

# ІНФОРМАТИКА

**Навчальний посібник**

Авторська редакція

Комп'ютерна верстка – Грищенко І. О.

Підписано до друку 28.01.2016. Формат 60х84/16.
Папір офсетний. Гарнітура Times New Roman.
Ум.-друк. арк. 42,31. Обл.-вид.арк. 58,45.
Тираж 300 прим. Зам. № 24.

ДП «Інформаційно-аналітичне агентство»
вул. Еспланадна 4–6, м. Київ, 01601, Україна
тел/факс 287-03-79, 289-77-62
E-mail: iaa@dstati.kiev.ua

Свідоцтво
про внесення суб'єкта видавничої справи
до державного реєстру видавництв, виготівників і
розповсюджувачів видавничої продукції
ДК № 2881 від 21.06.2007 р.