Б. В. Дурняк, М. С. Пасєка, Т. М. Майба

УПРАВЛІННЯ ЗАПИТАМИ В СИСТЕМАХ ДОКУМЕНТООБІГУ

Львів — 2016

УДК 004.01
ББК 76
Д 843

Затверджено до друку
вченою радою Української академії друкарства
(протокол № 5/666 від 31.03.2016 р.)

Рецензенти:
Машков О. А. – д-р техн. наук, професор
(Державна екологічна академія післядипломної освіти
та управління Мінприроди України, м. Київ)
Коростіль Ю. М. – д-р техн. наук, професор
(Щецинська морська академія, Польща)
Коробчинський М. В. – д-р техн. наук, професор
(Воєнно-дипломатична академія, м. Київ)

Дурняк Б. В.
Д 843 Управління запитами в системах документообігу / Б. В. Дурняк, М. С. Пасєка, Т. М. Майба. – Львів : Укр. акад. друкарства, 2016. – 192 с.
ISBN 978-966-322-419-0

У монографії розроблено нові методи вирішення задач прискорення запитів у базах даних і систем документообігу для прийняття рішень в соціоорієнтованих структурах. Запропоновано методику пошуку на основі нечітких мір для скорочення часу вибірки у великих і надвеликих базах даних. Удосконалено методи побудови автоматизованих інтегрованих мультимедійних комплексів для відображення динамічних ситуацій в об'єктах соціокомунальної структури.

Для студентів-магістрів та фахівців з управління системами документообігу.

УДК 004.01
ББК 76

ISBN 978-966-322-419-0

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АРМ — автоматизоване робоче місце
АСУ — автоматизована система управління
БД — база даних (банк даних)
БЗ — база знань
ВД — вітрина даних (Data Mart)
ЕС — експертна система
ЗВ — збережені відношення
ІТТ — інформаційні та телекомунікаційні технологій (англійською ІСТ)
ІС — інформаційна система
ЛПР — людина приймає рішення
ПЗ — програмне забезпечення
РБД — реляційна база даних
САПР — система автоматизованого проектування
СБЗ — система бази знань
СУК — система управління курсами (Course Management System (CMS)
СУН — система управління навчанням (Learning Management System (LMS)
СППР — система підтримки прийняття рішень
СУБД — системи управління базами даних
ТМЦ — товарно-матеріальна цінність
ADSI — Active Directory Service Interfaces
API — Application Program Interface (прикладний програмний інтерфейс)
DW — Data Warehause
LDAP — Lightweight Directory Access Protocol
LOLEPOP — Low-Level-Plan-Operator
RSS — Relational Storage System
SQL — мова запитів
STAR — Strategy Alternative Rules

ВСТУП

Рівень розвитку сучасних апаратних і програмних засобів та професійної й загальної підготовки управлінського апарату приватних і державних соціоорієнтованих та виробничих систем є індикатором стану держави і суспільства загалом.

Сучасні управлінські виробничі й адміністративні та суспільно-комунальні структури потребують для прийняття цілеорієнтованих рішень великих обсягів інформації — як структурованої, так і з різнорідних даних, швидкої їх вибірки з баз даних та техногенного й організаційного середовищ. Відповідно, для прийняття рішень необхідні вміння, засоби й методи їх опрацювання та інтерпретування змісту, щоб сформулювати стратегії розвитку сценаріїв подій і оцінити поточну ситуацію. Розв'язання таких прикладних задач потребує інтеграції інформаційних технологій, системного аналізу, методів теорії прийняття рішень і відповідних апаратних та програмних засобів, а також належної підготовки управлінського й адміністративного персоналу.

Проте задачі оперативного управління в низових структурах державного й організаційного управління (село, район, місто) в нинішніх умовах не були остаточно розв'язані, особливо задачі швидкого відбору даних і подання ситуацій, що, відповідно, зумовлює актуальність розвитку моделей та методів інтелектуальної обробки різнорідних даних від соціоорієнтованих об'єктів комунальних і господарських структур, надавши можливість забезпечити прийняття ефективних і своєчасних управлінських рішень.

У монографії досліджено підвищення ефективності та якості прийняття управлінських рішень на підставі розроблення інформа-

ційної технології опрацювання запитів у базах даних і системах документообігу в соціоорієнтованих структурах. Уперше розроблено метод пришвидшення процедур запитів і проведено їх оптимізацію на основі логіки (дає змогу підвищити ефективність опрацювання різнорідних і слабоформалізованих даних від об'єктів соціоорієнтованих структур).

Автори висловлюють щиру подяку рецензентам книги: д-ру техн. наук, професору Машкову О. А., д-ру техн. наук, професору Коростілю Ю. М., д-ру техн. наук, професору Коробчинському М. В. за поради та цінні вказівки при підготовці видання.

РОЗДІЛ 1
СТРУКТУРИЗАЦІЯ СОЦІООРІЄНТОВАНИХ СИСТЕМ ТА АНАЛІЗ ПРОБЛЕМИ УПРАВЛІННЯ

Інтегровані системи автоматизованого управління є новим якісним рівнем розвитку систем управління, які охоплюють всі головні функції управління виробництвом і забезпечують узгоджену поведінку елементів системи з огляду на основну мету її функціонування. Для більшості інтегрованих систем автоматизованого управління характерна значна централізація функцій обліку й планування, в результаті чого можна виділити три рівні управління:

— обробка даних для керівництва;

— оперативне управління господарськими процесами;

— управління технологічними процесами.

Усі системи будуються за ієрархічним принципом, відповідно до якого на кожному рівні здійснюється збір і первинне опрацювання даних, що характеризують стан господарських процесів, а на верхньому рівні вирішуються задачі управління виробництвом загалом.

Процес проектування інтегрованих систем автоматизованого управління головно базується на використанні двох підходів, один з яких — синтез, а другий — аналіз. Синтез потребує створення моделі об'єкта і системи управління, їх оптимізації й проектування оптимальної структури системи управління. Аналіз передбачає залишити існуючу методологію й організацію управління і тільки автоматизувати роботу процесу прийняття рішень, підвищити їх якість і швидкість.

Незважаючи на те, що сьогодні створено достатню кількість інтегрованих систем, проблема автоматизації залишається актуальною. Необхідна система, яка враховувала б вимоги конкретного підприємства і повноцінно виконувала свої функції. Щодо соціоорієнтованих систем, то проблемні задачі управління — як локального, так і корпоративного, мають регіональний стратегічний хара-

ктер; як процес управління ієрархічними комплексними структурами (виробничі, сільськогосподарські підприємства, організації, навчальні заклади, медицина) має координаційний, експертний характер. Це, відповідно, потребує індивідуального підходу до аналізу динаміки та створення стратегій цільового управління для кожної функціональної замкненої структури.

Кожне підприємство — як державної, так і приватної та корпоративної власності, повинно контролюватись державними та соціально орієнтовними регіональними структурами, водночас маючи власні стратегічні цілі, відповідні стратегії та ресурси для їх реалізації. Управління в них має виконуватись згідно з нормативними актами та законами держави.

Для розв'язання задач аналогічного типу потрібен розвиток як інформаційної інфраструктури, так і систем управління з використанням інформаційних технологій відбору, аналізу оцінки даних про динамічну ситуацію та формування на її основі стратегій управління, що супроводжується, відповідно, програмним і апаратним забезпеченням.

1.1. Характеристика соціоорієнтованого об'єкта управління та його структура і функція

В умовах ринкової економіки основною функцією будь-якого підприємства (організації) є випуск продукції (надання послуг) для одержання економічних результатів від її реалізації. Центральне місце серед завдань управління займає одержання прибутку від результатів господарської діяльності підприємства (організації). Придбання засобів і знарядь виробництва, виробничі процеси й організаційні заходи зазвичай передують прибуткам, одержуваним унаслідок господарської діяльності. Отож важливо зуміти зіставити матеріальні, трудові й фінансові потреби з наявними ресурсами.

Процес управління підприємством (організацією), метою якого є одержання прибутку, можна відобразити класичною схемою, поданою на рис. 1.1.

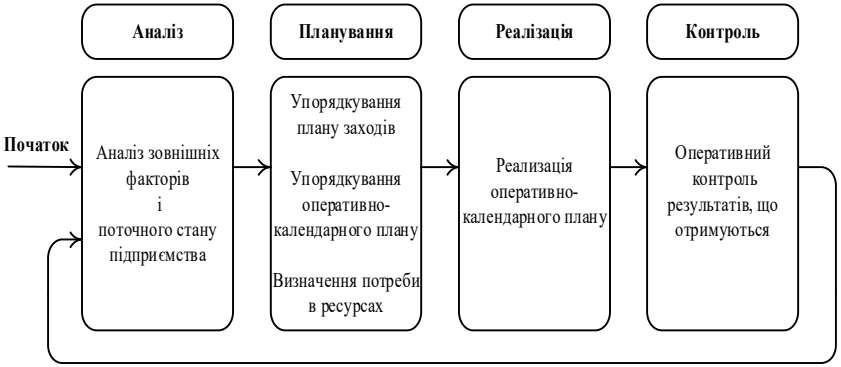


Рис. 1.1. Схема управління підприємством

Як випливає з наведеної схеми, процес прямування від поставлених цілей до результату є багатоступінчастим, він потребує оперативного коригування початкового плану дій залежно від досягнутих проміжних результатів. У загальному випадку кінцевий успіх підприємства залежить від багатьох чинників, частина з яких не піддається суворій формалізації. Склад таких чинників згруповано на рис. 1.2, де відображено напрям цільового руху.

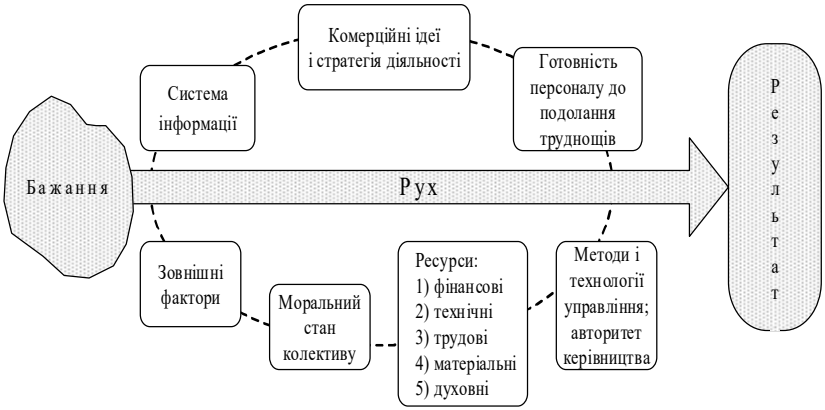


Рис. 1.2. Чинники комерційного успіху

При такій структурі власне сам об’єкт управління має приховану форму і взаємозв’язки матеріальних енергетичних та інформаційних ресурсів не простежуються, а потоки даних і управлінських команд не представлені в явному вигляді [1].

1.2. Вимоги до інформаційної системи оперативного управління

Система, що автоматизує збір, підготування й опрацювання інформації, виступає лише однією з необхідних складових, що визначають кінцевий успіх підприємства. Проте вже сьогодні очевидно, що найуспішнішими в діловому світі є ті фірми й корпорації, що спроможні якнайшвидше зібрати інформацію, опрацювати, проаналізувати її та на основі цього прийняти рішення, тобто сучасні інформаційні технології, які використовуються. Щораз більше керівників розуміють, що максимально ефективною автоматизованою системою є та, яка охоплює всі взаємозалежні багатогранні бізнес-процеси, всі аспекти внутрішньо- й зовнішньогосподарської діяльності, тобто комплексні автоматизовані системи. Безумовно, під час вибору потрібної системи варто враховувати, що жодна з них, хоч би якою відлагодженою вона була, не зможе функціонувати в умовах недостатньо чітко регламентованого технологічного процесу управління в рамках кожного бізнесу-процесу.

Найскладнішою задачею розробки інформаційної системи є узгодження стратегічної мети з ресурсними, фінансовими, технологгічними можливостями, які б забезпечили її досягнення при вибраних стратегіях управління з урахуванням відповідних поточних ситуацій. Для забезпечення розв'язання задач потрібно створити на основі інформаційних технологій відповідні структури збору, опрацювання даних, експертних систем, інтерфейсу управлінського діалогу, створення баз даних і предметно-орієнтованих знань зі швидким доступом і системою побудови запитів (логіко-лінгвістичного та інформаційного забезпечення).

1.3. Огляд літературних джерел щодо розробки оперативних систем управління соціоорієнтованими системами

Останнім часом практичні та теоретичні питання організації бухгалтерського обліку на базі ЕОМ досить широко були висвітлені в спеціальній літературі. Однак односторонньо викла-

ються окремі теоретичні та методологічні положення практичної організації бухгалтерського обліку в умовах АСУ. Наявність і функціонування великої кількості підприємств з різним рівнем автоматизації облікової інформації та забезпеченості комп'ютерною технікою вимагають розробки і застосування різних за змістом організаційних форм та методів автоматизації обліку.

Суттєвим недоліком деяких вирішень автоматизованих систем обробки облікових даних, на думку Г. Буча [4], є відсутність системного комплексного підходу до теоретичних та методологічних проблем обліку. Як правило, при проектуванні та впровадженні автоматизована система обробки облікової інформації розподіляється на комплекс задач, а останні, в свою чергу, — на задачі, підзадачі, функціональні модулі, які мають складну ієрархічну структуру, конкретні вхідні та вихідні інформаційні зв'язки та алгоритмічні рішення. Такий підхід Г. Буч вважає одностороннім, оскільки в багатьох випадках проектування і впровадження розпочинають з окремих задач та їх комплексів. У результаті комплекси задач і підсистеми згодом погано взаємодіють між собою, відповідно, різко знижується ефективність упроваджуваних систем.

Останнім часом на ринку програмних продуктів стали з'являтися нові розробки з автоматизації бухгалтерського обліку як закордонних, так і вітчизняних виробників. Звичайно, в наявній широкій гамі програм надзвичайно важко орієнтуватися та підібрати ту єдину, яка найповніше відповідає потребам конкретної ситуації.

Створено інфраструктуру, що забезпечує:

— розробку й удосконалення інструментальних засобів її підтримки, відповідаючи сучасним стандартам;
— глибоке опрацювання предметної галузі на етапі постановки задач проектування автоматизованих систем;
— якісну програмну реалізацію складних проектів у стислі старанно сплановані терміни;
— всебічну технічну й методичну підтримку;
— фахову підготовку користувачів до практичного використання придбаних програмних продуктів;

— системну інтеграцію, комплексне постачання обчислювальної техніки, розрахунок і монтаж мереж;
— налаштування й модернізацію комп'ютерного устаткування і засобів віддаленого доступу;
— консалтингові послуги в сфері удосконалення й перепроектування бізнес-процесів.

Це дало можливість розробити основні вимоги до комплексних систем управління організацією:

1. Адаптивність стосовно профілю діяльності підприємства (організації) будь-якої форми власності.

2. Максимально можлива кількість параметрів, що дозволяють налаштувати комплекс під специфічні особливості господарської, фінансової і виробничої діяльності організації-користувача.

3. Чітке розмежування оперативно-управлінських і бухгалтерсько-облікових задач при повноті їх інтеграції на рівні єдиної бази даних.

4. Охоплення всього спектра типових виробничо-економічних функцій.

5. Дотримання однакового для всіх розв'язуваних задач користувацького інтерфейсу.

6. Надання користувачам простого інструментарію для самостійного розвитку системи.

7. Підтримка розподілених баз даних для забезпечення інформаційної взаємодії багатоофісних корпорацій і територіально віддалених філій, підрозділів.

8. Використання рішень, які не потребують тривалої спеціальної підготовки системних адміністраторів, відповідальних за експлуатацію системи.

Розглянемо деякі програмні продукти, які стали найвідомішими. «1С: Бухгалтерія» — універсальна бухгалтерська програма, найрозповсюдженіша (за даними «Фінансової газети» та газети «Софт-маркет») бухгалтерська програма в СНД. «1С: Бухгалтерія» може бути налаштована безпосередньо бухгалтером на особливості бухобліку конкретного підприємства, будь-які зміни законодавства

та форм звітності. Засвоївши універсальні можливості програми, бухгалтер може автоматизувати різні розділи обліку [6, 65].

Може скластися враження, що потреби бухгалтерів в автоматизації обліку повністю задоволені. На ринку України не менше двадцяти спеціалізованих фірм вже не перший рік пропонують комплексні рішення для організацій різноманітної спеціалізації. Проте серйозні питання виникають у зв'язку з освоєнням різноманітних можливостей та подальшим супроводом комп'ютерних програм.

На думку авторів [91], хоч бухгалтерський облік є найбільш регламентованою управлінською функцією та вважається динамічною системою, яка постійно вдосконалюється, змінам зазвичай піддаються звітні показники, що супроводжується зміною методології розрахунку облікових показників, синтетичних та аналітичних рахунків. Усе це передбачає можливість постійної адаптації до змін, які відбуваються у зв'язку з вимогами управління й часу. Якщо ж на цей аспект обліку не звертати уваги при його автоматизації, помітно ускладнюються експлуатаційні можливості проектів.

Відрив теоретичної концепції обліку від практики є одним з недоліків наявних підходів до його автоматизації. Для вирішення цієї проблеми, на думку І. І. Яремко [91], є два підходи:

— перегляд методологічних аспектів обліку за схемою безпосередньо від вимог механізації й автоматизації обліку до його методології;
— комплексний перегляд теоретичних і методологічних основ бухгалтерського обліку.

У результаті реалізації першого підходу досягається обмежений ефект, що виражається в локальних змінах у методології системи обліку. Система обліку удосконалюється головно в напрямку задоволення потреб автоматизації. Другий підхід передбачає зміни в методології бухгалтерського обліку та реалізацію можливостей комплексної автоматизації облікових функцій на ПК.

Якщо проблемна задача бухобліку фінансових і матеріальних ресурсів вирішена на повному рівні, проблеми управління потребують свого розв'язання на вищому рівні, враховуючи комплекс-

ний характер задач управління, для розв'язання яких необхідно мати інформацію про:

— моделі динаміки й структури об'єкта управління;
— моделі потокових матеріальних ресурсів;
— моделі технологічних і виробничих процесів;
— моделі інформаційних потоків та їх опрацювання для управління;
— структуру програмного й апаратного забезпечення, інтерфейс діалогу;
— моделі предметно-орієнтованих баз даних і знань;
— геоінформаційні регіональні системи для візуалізації просторової структури регіону;
— бази даних, СППР і системи для експертних консультацій, що відповідно забезпечує теоретичні засади управління складними системами з ієрархічною структурою [5, 21, 25].

1.4. Інформаційні технології ситуативного прийняття рішень

Проблемна задача прийняття комплексних рішень щодо управління складними виробничими системами та адміністративно-організаційними структурами має тривалу історію. На різних етапах еволюції проблем, задач, способів їх розв'язання формувався математичний, системний інструментарій та теорія прийняття рішень як на рівні особи, так і колективів. У методологічному плані задачі такого класу повною мірою не розв'язані як у ринковій, так і в плановій та змішаній економіках, адже відповідні підходи базувалися на розгляді певних аспектів локальних задач (теорія цін і корисності, теорія програмно-цільового керування, теорія маркетингу і менеджменту і т. д.), не пов'язаних у цілеспрямовану структуру в явному вигляді [9, 26, 48, 56].

Сучасний рівень розвитку системного аналізу, теорії ігор і дослідження операцій, теорія сучасних систем керування на концептуальній основі системології цілеспрямованих структур дають змогу розробити методику аналізу і синтезу структур (народногосподарських) для досягнення заданих цілей при задовільних критеріях

якості, обґрунтувати їх стохастичну структуру [7, 28, 48, 56]. При такому підході чітко формулюється методика побудови функціональної структури регіону, технополісу, визначається цільовий простір параметрів станів як матеріальних, так і фінансових потоків. Граничні умови щодо структури й параметрів станів задаються через еталонну модель образу ситуації в цільовому просторі системи, який ґрунтується на базі знань законодавчих актів та на фізичних, енергетичних обмеженнях на наявні ресурси і рівень їх цін [7, 9, 26, 48, 61].

Функціональна структура регіональних систем складається з таких підсистем: цілезадавальна, цілевиконувальна, джерела ресурсів, споживачі й нагромаджувачі ресурсів. Утримання траєкторії параметрів потоків ресурсів у заданих межах базується на наборі стратегій керування в стратовій структурі. Показано, що стійкість таких структур забезпечується тільки в комплексі інженерних та економічних стратегій, при відповідній стійкості джерел ресурсів [26, 46–47, 61].

Для застосування сформульованої методики аналізу й синтезу до наявних структур мега- і технополісів необхідно, враховуючи концептуальну модель, цілеспрямовано здійснити ідентифікацію структури адміністративно-господарської системи регіону, визначити граничні умови для заданої організації цільового простору за потоками ресурсів, сформувати цільову тактику і стратегію управління ресурсами для досягнення визначених цілей, вибрати клас моделей функціоналів якості й метод оптимізації, які забезпечать відповідну поведінку траєкторії параметрів стану в цільовому просторі системи [46–47, 61].

Синтез цільового простору системи: цільовий простір системи формується з урахуванням проблемної ситуації й конкретної задачі в межах взаємодії ієрархічних державних і міждержавних структур на основі упорядкованої бази знань. Упорядкована база знань, відповідно, на концептуальних основах філософії, логіки, математики, які слугують мовою системного аналізу, теорії управління, статистики, теорії прийняття рішень, теорій інформаційних систем і технологій [7]. Основою бази знань є завдання моделей різної при-

роди в рамках стохастичної теорії систем, динаміка яких описується диференціально-інтегрованими рівняннями, що визначають баланс ресурсів у системі [8].

Методика розв'язання класу таких задач в умовах розвалу структури організації й джерел ресурсів, збоїв в інформаційних потоках, невизначеності під час прийняття рішень заводить траєкторію системи до границь параметричної й структурної стійкості або, у разі обмеженості ресурсів і нечіткості стратегій управління, — до катастроф (параметричних, структурних) [7, 48]. При цьому кризи зумовлені обмеженням ресурсів територіальних, енергетичних, інформаційних, фінансових, а найнебезпечніші — інформаційні кризи (незнання та некерованість).

Для розв'язання задач окресленого напрямку пропонується нетрадиційний підхід до аналізу й синтезу структур управління складними об'єктами типу адміністративних агроекотехнічних систем, які існують в умовах ресурсних обмежень та інформаційної невизначеності, з не чітко вираженою структурою.

Системи управління адміністративними, еко-, агро- і технологічними підструктурами належать до об'єктів керування з перемінною структурою й параметричною нестаціонарністю з чітко невизначеним цільовим простором. Це випливає із загальних принципів життєдіяльності — властивості росту, розвитку структури в умовах ресурсних і фінансових обмежень. Дослідження систем потребує адекватних методів аналізу — системології, статистики, теорії управління, теорії інформаційно-вимірювальних систем [7, 9, 26, 46–48, 61]. Окрім того, проектування таких систем пов'язане з обмеженістю енергетичних, матеріальних, інформаційних і фінансових ресурсів у часі та просторі за умови досягнення цілей.

При класичному підході до аналізу й синтезу систем керування проектувальник спирається на теорію управління зі зворотним зв'язком, а також на власну базу невпорядкованих знань. Враховуючи задані цілі, вибирається структура й алгоритми обробки даних без явного вибору стратегії прийняття рішень [18].

Системологія цілеспрямованих систем та інформаційно-ресурсна концепція побудови моделі об'єкта керування дозволяє розв'язати задачу синтезу стійких самокерованих систем з відповідним коректним інформаційним і програмним забезпеченням. На основі інформаційно-ресурсної концепції розроблена методика ідентифікації старих структур, аналіз і синтез нових з урахуванням інформаційно-ресурсної моделі об'єкта керування і концепції цілеспрямованої ієрархічної системи [7].

Враховуючи цілі, що формуються в цілеспрямовуючій системі вищого рівня, граничні ресурси і енергетичні, фінансові та законодавчі обмеження, на основі впорядкованої бази знань (експертні системи) будується цільовий простір всієї системи і визначаються критичні й аварійні режими, границі параметричної й структурної стійкості. З рівняння балансу потоків ресурсів визначаються моделі образів траєкторії динамічного стану системи, формуються стратегії керування для досягнення заданої цілі при вибраних функціоналах якості [7, 46].

Показано, що такі системи повинні мати відповідну структуру: модель бази знань, модель об'єкта керування, модель цілеспрямовуючої системи, модель спостерігача образів динамічної ситуації з програмним і алгоритмічним забезпеченням, модель інформаційного простору станів, модель стратегії прийняття рішень на керування, модель виконавчого механізму, модель оцінки фінансового стану системи і зовнішніх структур, модель ринкових ігрових ситуацій, пов'язаних з цілями і функціями корисності, моделі джерел і користувачів ресурсів, модель відновлення і функціонування людських ресурсів, модель кооперативної поведінки в зоні взаємодії підсистем та їх орієнтації, адаптації.

Згідно зі сформульованим підходом функціонування системи поділяється на етапи:

1. Оцінка зовнішньої ситуації, формування цілей.
2. Структуроутворення і вибір стратегій поведінки.
3. Оптимізація траєкторій стану за функціями ризику і корисності.

Адаптація стратегій поведінки і корекція структури проводиться за результатами розпізнавання образів динамічних ситуацій і за вибраними функціоналами якості. В таких системах цільовий простір глобально формується парламентськими органами, обмеження на правові норми прийняття рішень визначаються положеннями Конституції, а на ресурси і їх стохастичну структуру визначає ринок.

Компоненти структури ресурсів поділяються на: матеріальні, енергетичні, людські, інформаційні, фінансові; а за просторовою розподільністю — локальні, регіональні, державні; планетарні, космічно-галактичні. Відповідно до структур формуються моделі систем і динаміки ресурсів з урахуванням ієрархії перетворення ресурсів при виконанні умов керованості й спостережуваності, стохастичності процесів і структур, що їх породжують.

Робастна оцінка параметрів стану таких систем базується на сучасній статистиці й теорії фільтрації, траєкторія оцінки є базою для формування образу динамічної ситуації в цільовому просторі системи на заданому інтервалі спостереження. Відхилення образу ситуації від еталонної моделі є основою під час прийняття рішень за вибраними стратегіями і доступними ресурсами для досягнення заданих цілей. При цьому образ ситуації багатофакторний і багатовимірний, складається з компонентів [7]:

1) образ динамічної ситуації з матеріальними ресурсами;

2) образ динамічної ситуації з енергетичними ресурсами;

3) образ і структура інформаційних потоків.

Образ фінансової ситуації як відображення параметра стану вартості структури систем і потоків ресурсів, їх резерву. Важливою проблемою стає параметризація цільового простору й цільового стану системи. Адекватній структурі та динамічній ситуації вибраний параметр стану дозволяє зекономити об’єм програмного забезпечення, відобразити стан динамічної системи, синтезувати ефективну тактику і стратегію поведінки для досягнення заданих цілей у рамках вибраних політик.

1.5. Інформаційні технології та системологія управління регіональними структурами в умовах ризику

Науково-прикладні задачі управління господарськими і регіональними структурами в умовах обмеженості ресурсів і кризи на сучасному етапі остаточно не розв'язані. Важливо розкрити принципи формування стратегій прийняття рішень щодо управління та показати, що фінансові показники — це параметри стану в інформаційному просторі при вибраному показнику якості для заданого ступеня ризику. Стійкість до параметричних і структурних збурень визначається техніко-економічною стратегією прийняття рішень при виконанні цілеспрямованої програми.

На сучасному етапі розвитку адміністративно-технічних систем у народному господарстві регіонів актуальною задачею є оптимізація їх функціонування з урахуванням матеріальних, енергетичних, людських і фінансових потоків ресурсів при обмеженні та збуренні. Щодо теорії керування така задача адекватна вибору оптимальної стратегії поведінки цілеспрямованої системи та її стратегії адаптації при збуреннях у процесі досягнення цілі з забезпеченням робастності й стійкості, зокрема й соціальної [11–12, 17, 60].

У методологічному плані задачі такого класу повною мірою не розв'язані як у ринковій, так і в соціалістичній та змішаній економічних теоріях, оскільки методологічно такі підходи до розв'язання задач аналізу динаміки й синтезу структур базувались на розгляді певних аспектів і проблем (теорія цін і користі, теорія програмно-цільового керування, теорія маркетингу і менеджменту тощо), без урахування можливих екологічних криз і катастроф, соціальних збурень, інформаційного насичення та ускладнення процесів прийняття рішень [62]. Сучасний рівень розвитку системного аналізу, теорії ігор і дослідження операцій, теорії сучасних систем керування на концептуальній основі системології цілеспрямованих систем і філософії наукового пізнання дає можливість розробити методику синтезу структур за заданими цілями і критеріями якості, обґрунтувати методи аналізу динаміки потоків ресурсів залежно від їх стохастичної структури, прогнозувати аварійні ситуації в складних

соціально-технічних системах [23, 45]. При такому підході чітко формується методика побудови функціональної структури регіону, визначається цільовий простір параметрів станів — як матеріальних, так і фінансових потоків ресурсів; граничні умови на структуру й параметри стану з урахуванням правових норм задаються через еталонну модель образу динамічної ситуації в параметризованому цільовому просторі системи, представлення якої базується на фундаментальних фізико-математичних моделях законів природи, законодавчих актах і енергетично-ресурсних обмеженнях.

Функціональна структура регіональних систем, побудована на методології цілеспрямованої системи, складається з певних підсистем: цілезадавальна, цілевиконувальна, упорядкована база знань, джерела ресурсів, технологічні перетворювачі ресурсів (виробництво), споживачі й накопичувачі ресурсів, системи внутрішнього моніторингу. Динаміка поведінки системи характеризується траєкторією розбалансування потоків ресурсів, параметри їх визначають динамічну ситуацію в інформаційному цільовому просторі станів. Відповідно, цільовою задачею буде розроблення стратегії управління, яка утримала б систему в цільовій області Vci (рис. 1.3.) Утримання (керування) траєкторії параметрів потоків ресурсів у заданих межах базується на наборі стратегій керування в багатоярусній ієрархічній структурі. Показано, що стійкість таких структур при зовнішніх і внутрішніх збуреннях забезпечується комплексом інженерно-економічних стратегій при виконанні цільових програм з умовою відповідного забезпечення стійкості джерел ресурсів.

Використовуючи цю методику щодо аналізу й синтезу структури організаційно-технологічної системи, необхідно виходити з концептуальної моделі цілеспрямованої системи, бази знань, провести ідентифікацію адміністративно-господарської будови регіону, впорядкувати скелетну схему, визначити параметри потоків у внутрішньому і зовнішньому середовищах, встановити на основі фізичних законів обмеження енергетичних і матеріальних ресурсів, а також рівень фінансових ресурсів і границі їх обмеження. Формування тактики і стратегії поведінки у процесі управління системою

проводиться на основі системи заданих цілей, за моделями стратегій залежно від ситуації й виду збурювальних факторів, ступенем їх інтенсивності при постійній оцінці функціоналу якості з урахуванням правових норм. Такий підхід дозволяє вивчити оптимальну стратегію поведінки системи, яка забезпечить відповідну траєкторію параметрів стану у цільовому просторі системи за умови виконання умов спостережуваності й керованості об'єктом управління, що входить в організаційно-технологічну структуру регіону.

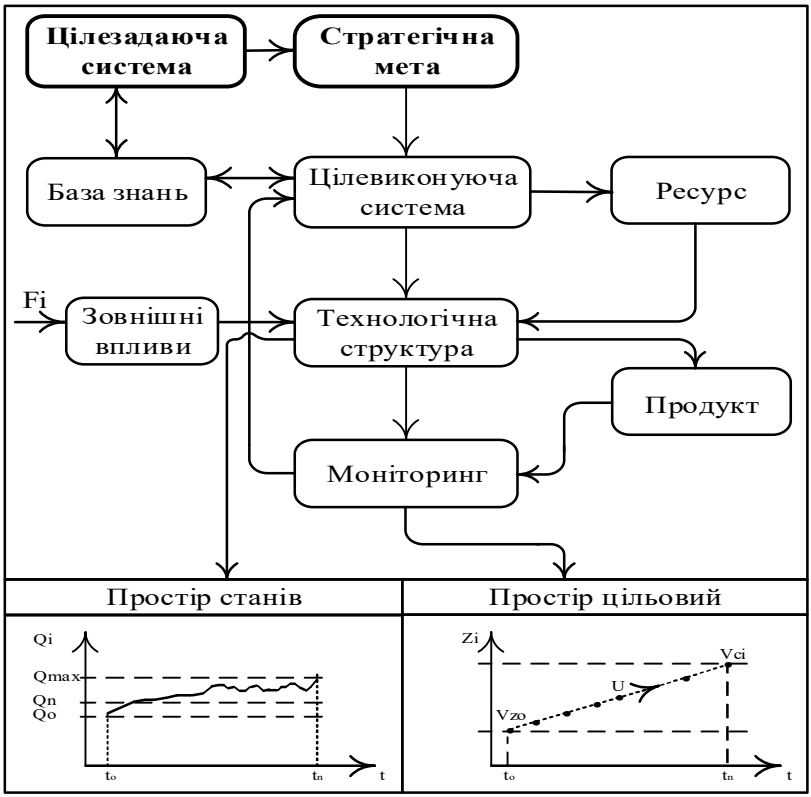


Рис. 1.3. Компонентна схема виробничої регіональної структури:
Qi — параметр стану; Qmax — максимальний параметр стану; Qn — нормальний параметр стану; Q0 — початковий параметр стану; Zi — вісь інформаційного цільового простору; U — стратегії управління; t є tn — термінальна вісь часу; Vzo — початкова область стану; Vci — цільова область; Fi — фактор

Для аналізу відповідних структур необхідно створити (або мати в своєму розпорядженні) впорядковану базу знань у таких предметних галузях:

1) логіка, методологія, філософія науки управління;

2) системний аналіз (методологія) і теорія управління;

3) стохастична динаміка, моделі сигналів, систем;

4) теорія і методологія технологічними організаційними структурами, принципи формування мети;

5) статистика й теорія прийняття рішень для цільового управління;

6) системні методи в техніці, фізиці, хімії, технології;

7) системологія математики, мови опису динаміки і структури стратегій досягнення цілей;

8) моделювання динамічних систем і інформацій технології; комп'ютерні технології для задач управління;

9) САПР і експертні інтелектуальні системи підтримки прийняття рішень;

10) системна економіка, динаміка ресурсів; розробка моделей;

11) системний аналіз людських ресурсів на основі оцінки інтелекту;

12) моделі екобіотехнічних систем і середовища їх існування.

Для оцінки фінансового і технологічно-ресурсного стану таких структур при взаємодії з людським фактором господарських механізмів необхідне вивчення і розв'язання таких проблемних ситуацій і задач:

1) побудова інформаційно-ресурсної моделі системи управління об'єктом;

2) побудова цільового простору системи як основи синтезу стратегій;

3) формування методології генерації цілей як на локальному, так і вищому рівнях;

4) генерація цілей для проблемних задач; прийняття управлінських рішень;

5) синтез локальних і глобальних стратегій досягнення цілей.

Потреба розв'язання цих проблем ставить вимоги до управлінського персоналу, що повинен володіти потрібними знаннями, мати належний професійний і когнітивний інтелектуальний рівень.

I. Системологія цілеспрямованих систем:

1) принцип формування знань у цілеспрямованих системах;

2) моделі об'єктів дослідження та їх динамічного стану;

3) структура цілеспрямованої системи, методи аналізу.

II. Людина як цілеспрямована система (ЛПР-ЦС):

1) формування цілей ЛПР для проблемних задач;

2) спостереження динамічних ситуацій і методи їх оцінки;

3) принципи прийняття рішень і стратегій керування.

III. Модель підприємства як цілеспрямованої системи:

1) людина — ресурси — підприємство як елементи моделі;

2) кооперативні й конфліктні ситуації в системах;

3) модель підприємство — ринок.

IV. Моделі народного господарства:

1) моделі адміністративної структури;

2) цільові простори — закон — конституція;

3) кооперація, конфлікт, криза як основні класи ігрових стратегій поведінки складних систем;

4) господарсько-технічні системи, принципи узгодження стратегій.

V. Стійкість системи при обмеженні ресурсів:

1) збурення стратегій, зміщення цілей, глобальні цілі й стратегії їх досягнення при обмеженнях на ресурси;

2) динаміка кооперативних систем, компроміси, конфлікти;

3) інформаційне забезпечення та ситуаційна динаміка систем;

4) системний підхід до синтезу адміністративних і економічних структур, баланс і зростання, цикли, стійкість.

В основі означеного підходу міститься концептуальна модель прийняття рішень щодо управління для досягнення цілей (рис. 1.4).

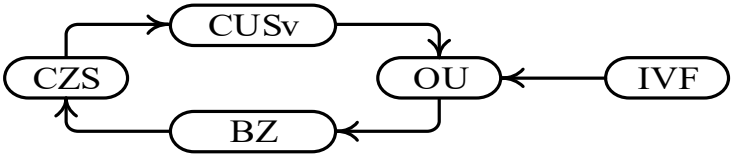


Рис. 1.4. Концептуальна модель прийняття рішень:
CZS — цілезадавальна система; OU — об'єкт; BZ — база знань;
IVF — збурювальні системи; CUSv — цілевиконувальна система

Керуючись цією концепцією, розглянемо базові управлінські структури та професійні й когнітивні вимоги до персоналу виконавчої та управлінської структур. Для реалізації процедури використовуються спеціальні психологічні та інтелектуальні тести й проводиться відповідне тренування персоналу для роботи в нормальних та аварійних ситуаціях [111]. Розглянемо структурні схеми управління складними об'єктами та вимоги до управлінського персоналу.

Вимоги до керівника: здатність витримувати велике навантаження; високий рівень професійної підготовки; спроможність приймати рішення в екстремальних умовах; ерудиція, креативність; високий рівень знань; цілеспрямованість, рішучість; психологічна і функціональна стійкість до стресу.

Представлення лінійної ієрархічної структури управління.

1. Лінійно-штабна ієрархічна структура управління (рис. 1.5). Вимоги до керівника: уміння працювати з колективом штабів при виробленні стратегій прийняття рішень; функції керівника — стратегічне мислення; формування цільових рішень, планів, завдань, команд управління; здатність оцінювати ситуацію на всіх рівнях ієрархії; стійкість до атак.

2. Характеризується функціональний тип ієрархічної структури управління спеціалізацією функцій, які виконуються при оперативному управлінні об'єктом (рис. 1.6).

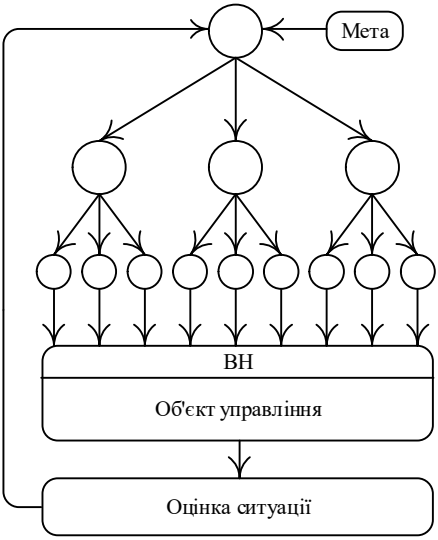


Рис. 1.5. Лінійна ієрархічна структура управління

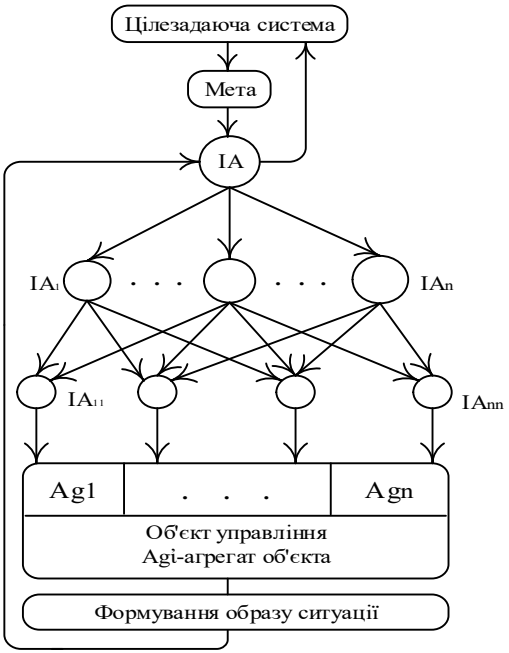


Рис. 1.6. Функціональний тип ієрархічної структури управління:
ІА — інтелектуальний агент; Ag1 — агрегат об’єкта

Вимоги до керівника: уміння координувати дії на нижніх рівнях ієрархії та психологічна стійкість; можливість розв'язувати кризові ситуації при формуванні некоректних команд; здатність до професійного зростання і навчання особи (ІА) — інтелектуального агента; стратегічне мислення; здатність до адаптації.

3. Матрична структура об'єднує цільовий і функціональний підходи до управління складними об'єктами з агрегатною організацією з відповідним комплексом вимог до керівника (рис. 1.7).

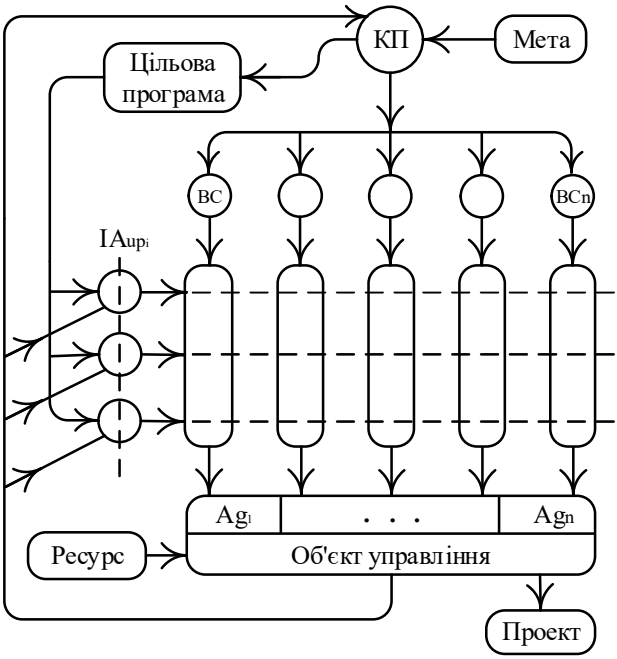


Рис. 1.7. Матрична структура:
КП — координатор проекту; ВС — виконавчі служби;
ІА — інтелектуальний агент; Ag1 — агрегат об'єкта

Вона включає: цільову систему; координатор проекту (КП); ВС — виконавчі служби; IAupi — інтелектуальні системи управління проектом; Agi — агрегатні структури виробництва.

4. Програмно-цільова структура управління охоплює: великі проекти і регіональні системи (рис. 1.8):

— CZs — цілезадавальну систему стратегічного управління;
— ILK — локальний ієрархічний координатор;
— SK — стратегічний координатор;
— PKi — програмний координатор;
— Sui — системи агрегатного управління;
— BMi — виконавчі механізми;
— SSm — системи моніторингу стану об’єкта управління як компоненти регіональної структури;
— BZk — бази знань ієрархічного координатора;
— BDz — база даних про стан об’єкта управління.

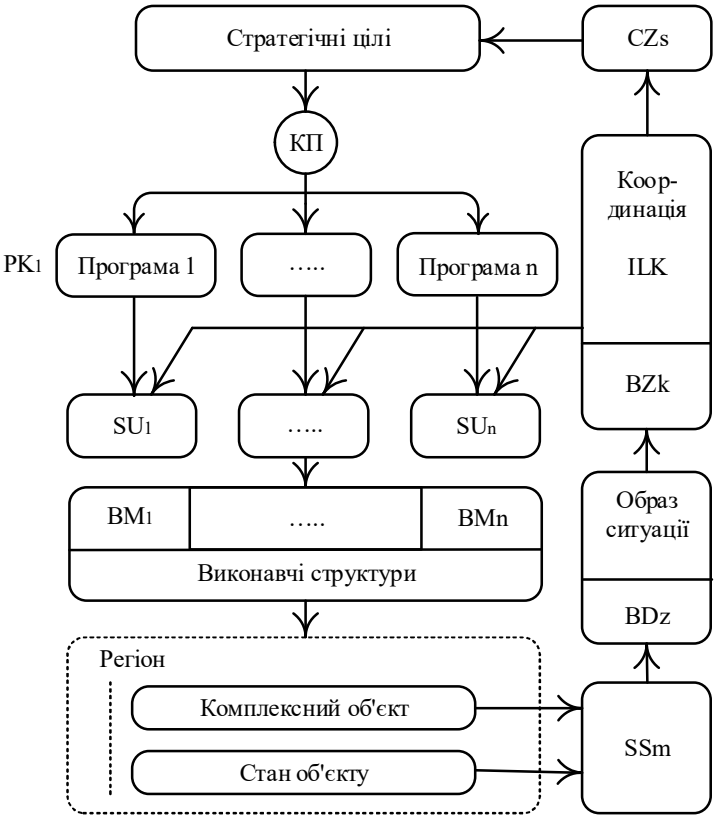


Рис. 1.8. Програмно-цільова структура управління

Для ефективного управління соціальною сферою або виробничою структурою регіону необхідно встановити стратегічні цілі та наявні ресурси і на основі системного аналізу вибрати необхідну структуру.

1.6. Інформаційні та системні технології розробки автоматизованого робочого місця в системі підтримки прийняття рішень

У процесі оперативного управління інтелектуально-складним виробництвом іноді дуже важко врахувати всі фактори, що впливають на виробництво, тож проаналізувати і прийняти адекватні рішення не завжди можливо. Сьогодні потрібний спеціальний інструментарій, який виконує роль помічника. Таким інструментарієм може бути система підтримки прийняття рішень (СППР) [27].

Із практики виробничої діяльності відомо, що фахівці, які керують процесом, не завжди здатні врахувати всі фактори, що впливають на виробництво. Проте важливо не тільки враховувати їх, а й проаналізувати та прийняти адекватні рішення. Останнім часом таких факторів стає дедалі більше, а реакція системи на їх вплив відчутніша. Отже, обсяг витрат унаслідок допущених помилок під час прийняття управлінських рішень істотно збільшився. Потреба в якісному управлінні технологіями зростає й займає важливе місце в складній системі виробництва. Допустимих помилок стане значно менше, якщо людина, яка виробляє й приймає рішення (ЛПР), володіє спеціальним інструментарієм, що дає змогу нагромаджувати потрібну інформацію, упорядковувати, аналізувати її і, як результат, видавати кваліфіковані поради.

Інструментарієм, що виконує роль помічника фахівця-управлінця, може бути розроблена нами на базі комп'ютера СППР, яка створюється з урахуванням можливості її розвитку і функціонування, починаючи від рівня проектування технологій управління виробництвом і завершуючи оперативним управлінням та експертними системами [27]. Розв'язання такої складної задачі, як створення автоматизованих систем, потребує насамперед чіткої, строго

послідовної реалізації зумовленого набору певних обов'язкових принципів: системний підхід до розробки; ієрархічна структура розподілу функцій; орієнтація на доступну користувачеві фактичну інформацію; орієнтація на користувачів різного рівня підготовки; орієнтація здебільшого на математичне моделювання внаслідок ненадійності засобів оперативної інформації, засобів верифікації даних, їх автоматичного відновлення; можливість удосконалення системи в майбутньому; використання в складі системи автоматизованих робочих місць фахівців (АРМ) і вузькопрофільних прикладних задач [22].

Системний підхід до вирішення потребує аналізу об'єкта, створення моделі його поведінки в умовах ситуації, що змінюється, постановки задач і розробки алгоритмів її розв'язання, застосування методів математичного моделювання. Кінцевим компонентом при цьому має бути формування порад і рекомендації щодо здійснення заходів і видача їх у тій чи іншій формі користувачам. Залежно від інформаційної бази, складності моделей, що використовуються в конкретному випадку, типи СППР поділяються на: розрахункові, які дають змогу оцінити наслідки прийнятих рішень; імітаційні, що використовують математичні моделі, які описують фізичну суть задачі; оптимізаційні, які враховують обмеження; рекомендаційні (радницькі), що описують і розв'язують погано або неструктуризовані задачі.

Зауважимо, що об'єктом вищого рівня, на який орієнтована запропонована розробка, є галузь управління, а нижчого — виробничий процес із комплексом матеріальних, енергетичних і фінансових ресурсів.

1.6.1. Аналіз задачі створення системи підтримки прийняття рішень

Аналіз об'єкта, зокрема його інформаційної бази, показує, що до встановленої мети (кваліфікаційні поради до часу) найшвидше приведе орієнтація на створення комбінованого типу СППР. Отже, якщо кваліфікувати СППР, що розробляється нами, найдоцільніше

віднести її до оптимізаційно рекомендаційних. Тобто за своєю цільовою функцією в загальному вигляді система рекомендаційна, коли деякі поради ґрунтуються на результатах розв'язання оптимізаційних задач.

Якщо СППР управління розглянути у функціональному плані, система становить набір індивідуальних підсистем, зорієнтованих на розв'язання конкретних спеціалізованих задач виробництва. Блок-схема системи (рис. 1.9) дає уявлення про структуру побудови. Головні частини становлять підсистеми (блоки) першого рівня — підсистема бази знань, АРМ фахівців і комплекс координуючих програм (управляючий програмний комплекс). Програмна організація структури СППР і кожної підсистеми складаються з інформаційної бази, блока висновку, діалогового інтерфейсу, блока верифікації БЗ, блока пояснень [5, 21].

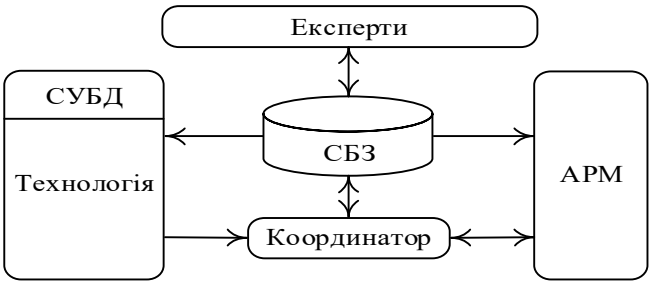


Рис. 1.9. Структура СППР

Зазначимо, що підсистеми, які входять до складу СППР, здатні функціонувати індивідуально. Система регіональної бази знань (СБЗ) є сховищем великих масивів інформації, організація якої дає можливість використовувати її як для формування поради, так і у вигляді довідкової.

1.6.2. Проблемна задача розробки автоматизованих систем управління

Відомо, що головною проблемою при розробці автоматизованих систем є створення зручної бази знань. Проблема виникає че-

рез те, що й досі немає чітко сформованих методів подання знань, особливо це стосується галузі, про яку йдеться. До того ж знання експертів часто бувають невизначені й неточні. Отож у більшості випадків застосовувалися прийоми, пов'язані з логічними висновками [55]. Знання експертів при цьому подаються у вигляді великої кількості простих правил, застосовуваних при організації діалогу між комп'ютером і користувачем.

Загальна схема інформаційного фонду СБЗ зображена на рис. 1.10. Серед інформаційних блоків СБЗ головне місце належить базі знань — містить інформацію загального характеру, концептуальну і системну.

На особливу увагу заслуговує вибір моделі представлення знань. Існують логічні моделі, що ґрунтуються на формалізмі мови математичної логіки, системі продукції, де знання подаються у вигляді набору правил, які зв'язують конкретний стан об'єкта з певною дією; реляційні моделі (табличні) — якщо співвідношення між поняттями задаються в явній формі, а математичним апаратом є реляційна алгебра, фреймові моделі.

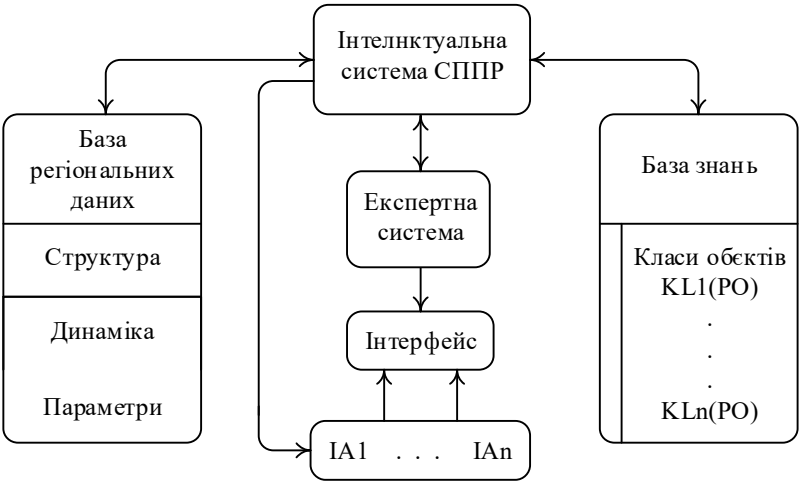


Рис. 1.10. Структура інформаційного фонду СБЗ

При створенні СБЗ представлення знань організовано у вигляді спеціальних інформаційних структур — фреймів. Структури мають сукупність вічок-слотів, що поєднані з різними аспектами об'єкта та його ієрархічними зв'язками. Із фреймами пов'язана різна інформація, а відношення між ними визначаються таким чином, що можна використовувати ім'я одного фрейму в значенні слова в іншому фреймі. Оскільки слот може мати кілька значень, можлива побудова дерева відношення фреймів.

У зв'язку з визначальним впливом технологічних рішень на ефективність виробництва головне місце серед функціональних задач СППР належить підсистемі «Управління». Призначення такої задачі полягає в створенні оптимальних або раціональних оперативних планів в умовах, що змінюються [3, 45].

Блок «БЗ» у системі СППР функціонує на підставі технологічних знань, а окремі задачі з ієрархічної структури, крім того, мають індивідуальне інформаційне забезпечення. Тобто всі задачі у вигляді підсистем другого рівня, що входить до складу «БЗ», мають індивідуальну нормативно-довідкову інформацію відповідно до мети та специфіки.

Блок «Експерт», в який входить база знань експертів, охоплює:

— знання нормативних актів;
— блок обробки експериментальних даних;
— блок ідентифікації структури, динаміки, параметрів об'єктів;
— блок верифікації та класифікації ситуації в конкретний термін часу;
— забезпечує інформаційну підставу для формування управління.

Тобто об'єктом управління в цьому випадку є агрегована виробнича система.

Інформаційний фонд, на відміну від БЗ, складається з конкретніших показників, прив'язаних до параметрів об'єкта (рис. 1.11). Це дає змогу формувати поради відповідно до ситуацій, що виникають у процесі виробництва.

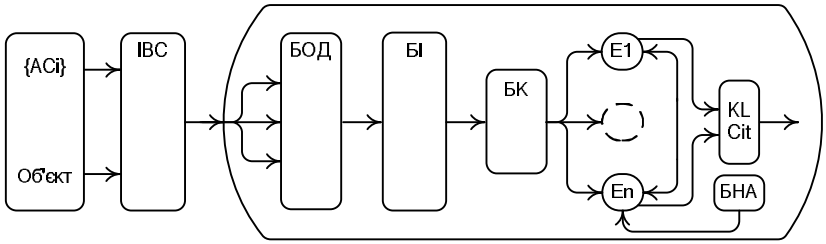


Рис. 1.11. Блок-схема експертної системи СППР:
АСі — автоматизована система; БОД — блок обробки даних; БК — блок керування; БІ — блок інформації; БНА — блок навчання адміністративного персоналу; ІВС — інформаційна вимірювальна система; Еі — експерти

Головні труднощі під час розробки системи пов'язані зі специфічністю інформаційного забезпечення, що потребує, окрім іншого, спеціальної мови алгоритмізації. Специфічність інформаційного забезпечення системи «СППР» полягає в тому, що вона погано або взагалі не структуризована; не дає можливості реалізувати загальновідомі методи формалізації. Як наслідок, мова написання алгоритму потребує певних умов і повинна забезпечити: уніфікацію засобів спілкування між експертами і програмістами; легкість постановки інформації на ПЕОМ і їх наступної модифікації; комфортність інтерфейсу між фахівцями та ПЕОМ.

Одним із варіантів мови формалізації технологічних знань може бути мова, побудована на принципах об'єктно-орієнтованого програмування. Це специфічна мова, що дає змогу описувати технологію в компактній, зручній для реалізації у машинних програмах формі. Мова має три рівні: слабкоконструктиризована, що зберігає всі виразні ознаки професійної мови експерта; базова, яка є формалізованішою і краще пристосована для описування моделі та алгоритмів прийняття рішення; вихідна, що відповідає задачам програмування й надання вихідної інформації у необхідній формі. Модель технології на цій мові подається у вигляді ієрархічної сукупності моделей елементів нижчого рівня — технологічних процесів.

Маючи модель мовою високого рівня, можна перейти до формального представлення технологічних знань. Як і в поданні інформації до БЗ, застосовуються фреймові моделі, основою яких є формальна система, побудована на ієрархічному способі представлення неструктуризованих знань. Тобто модель будь-якого рівня (технологія загалом, технологічний процес) будується як ієрархія зв'язаних підмоделей за таким типом: модель (ідентифікатор) — об'ява підмоделей — об'ява підмоделей і механізмів зв'язку — рівняння та оператори зв'язків. Кожна така модель представляється відповідними фреймами.

Прийняття більш-менш обґрунтованого рішення в багатьох випадках потребує від фахівця додаткових розрахунків. Щоб полегшити цей процес, за останні роки в сфері організації інформаційних потоків з'явилося чимало прикладних комп'ютерних програм, що автоматизують розрахунки, методи розробки АРМ і ЕС (рис. 1.12) [18]. Такі програмні комплекси найчастіше належать до автоматизованих робочих місць (АРМ) фахівців. Проте є й складні системи, які, розв'язуючи суто розрахункові задачі, надають інші інформаційні послуги за фахом користувача. Такі системи мають іншу від попередніх організацію програмного комплексу та змістової частини. Вони також належать до АРМ.

Проведений аналіз структури системи управління виробництвом показав, що наявне інформаційне забезпечення недостатнє для управління соціоорієнтованими системами. Таким чином, потрібно визначити поняття «АРМ» і його структурної побудови. На наш погляд, АРМ — це насамперед система, а не окрема розрахункова задача або комплекс функціонально не пов'язаних задач.

1.6.3. Аналіз інформаційної структури експертної системи

Для створення системи $\{АРМі|_{і}^{м}\}$ робочих місць у структурі управління складною організацією або виробництвом необхідно розробити експертну систему та підібрати кваліфіковану групу експертів з відповідною професійною орієнтацією та знаннями.

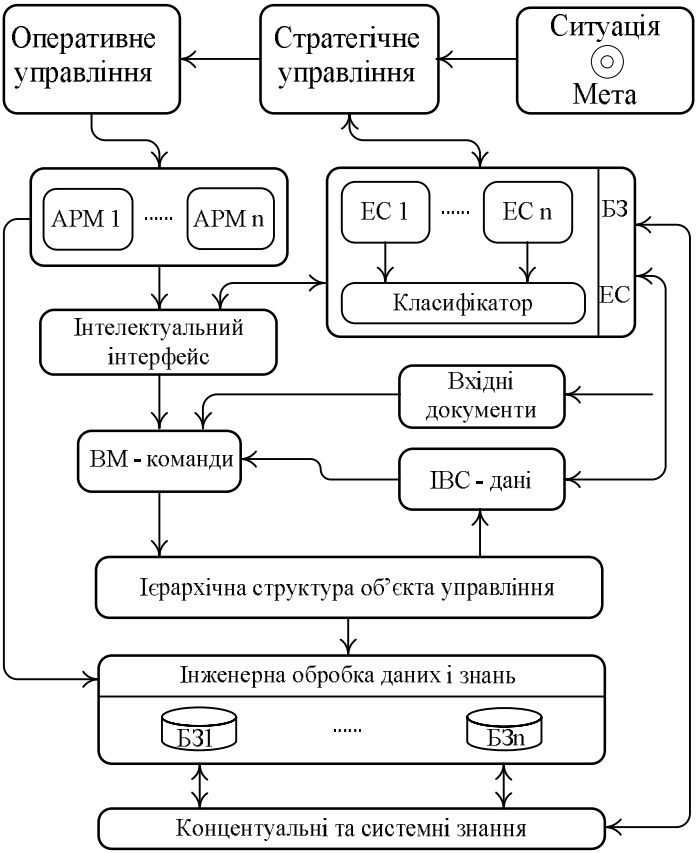


Рис. 1.12. Функціональна система АРМ оперативного управління: АРМі — автоматизовані робочі місця; ЕСі — експертні системи; ВМ — виконуючі модулі; БЗ — база знань

Типи експертних систем згідно з задачами розв'язуваних СППР:

— інтерпретація подій і визначення їх змісту, сценарії розвитку подій;

— прогнозування сценаріїв майбутніх подій у складних системах;

— планування поточної діяльності;

— проектування нових об'єктів і САПР;

— навчання й набуття нових знань у певних предметних галузях;

— контроль стану в реальному часі виробничих та організаційних систем;
— стратегічне управління корпоративними структурами виробництва;
— управління державними, місцевими та регіональними структурами.

Відповідно до вимог СППР розробляється блок-схема інтелектуального інтерфейсу (рис. 1.13). Наповнення експертних систем ЕС (Sn) передбачає:

1. Евристику для формування й прийнятя рішень.
2. Логічні правила виводу LPv.
3. Інформаційну модель системи Sn

$$Sn \rightarrow Gr : M \rightarrow K : Obi \rightarrow Sitij(tj) \quad (1.1)$$

(Система граф. автом. категорія стан)

4. Прийняття рішень в умовах невизначеностей.

$$\begin{cases} Zi(XiU) \rightarrow R\big(\tau(U)\big), \\ M^{*} - Mk\{Mi\} \end{cases} \quad (1.2)$$

де Zi — завдання; U — управління; R — рівень досяжності цілі; Mk — корінь ієрархічного дерева; Mi — модель.

Властивості M характеризуються топологією Z, якщо Vz міститься на вершині комутативного графа системи прийняття рішень z€Z.

$$Z \leftrightarrow Im(Ker\delta_{\eta}),\ Im - образ\ ситуації,\ Ker - Ядро.$$

Таблиця Z представлена триступеневою ієрархією в багатовимірному просторі цільових постановок задач. Відповідно, логічний і лінгвістичний процесори є основними компонентами в структурі n-місних АРМ-ів.

Насамперед АРМ має відповідати таким вимогам: забезпечити роботу з інформаційними масивами, тобто надається можливість нагромадження, обробки й подання інформації; наявність діалогового режиму. Діалог «ПЕОМ — користувач» відбувається у вигляді гри — «що буде…, якщо…»; можливість оптимізації за пріоритетами інформаційно-довідкової та інформаційно-пошукової системи; можливість друку й редагування тексту; видача статистичного звіту.

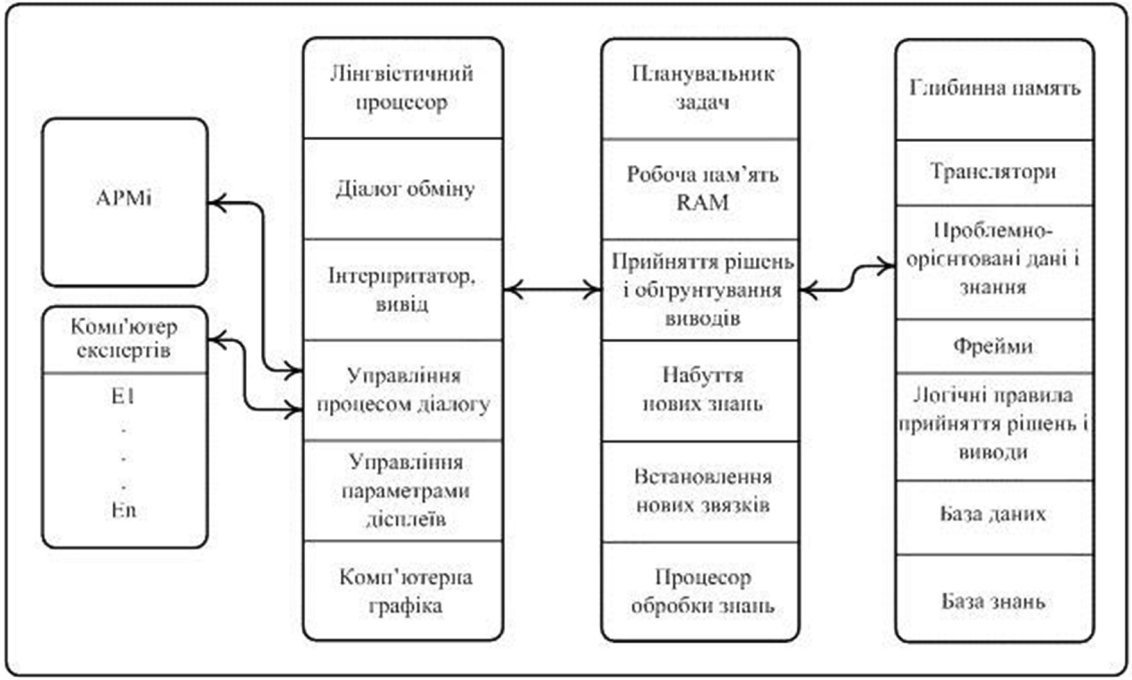


Рис. 1.13. Інтелектуальний інтерфейс експертної системи

За наявності можливостей професійно орієнтовані системи можна віднести до АРМ-фахівця.

Перелік задач і структури АРМ: важливо, що, крім єдиної інформаційної бази (інформація загального рівня), кожна задача має власний спеціалізований інформаційний фонд. Функціонування СППР як складової системи забезпечує комплекс програм, які об'єднуються в окрему підсистему «Координатор». У ширшому значенні ця підсистема охоплює функціональні й службові алгоритми, засоби автоматизації програмування та налагодження програмних комплексів.

Головні задачі, що розв'язуються функціональними алгоритмами, різноманітні й тісно пов'язані з конкретними задачами, які входять до складу СППР. Варто зазначити, що автономна функціональна ефективність управління не може бути реалізованою, якщо структура алгоритму не забезпечує найповільнішого використання ресурсів ПЕОМ. Отже, при створенні керуючої підсистеми постає проблема архітектури, яка передбачає: оптимізацію структури алгоритму з погляду як максимального використання ресурсів ПЕОМ, так і проектування алгоритмів (ієрархічність структур, поділ на підпрограми, уніфікація підпрограм для розв'язання типових задач); організацію послідовного розв'язання функціональних задач та організацію оперативної взаємодії з користувачем; задачі оперативного контролю обчислюваного процесу й забезпечення надійності; задачі контролю достовірності рішення, задачі забезпечення можливості коригування керуючих алгоритмів.

Головні функції підсистеми — координаційні, які треба розуміти як: установлення раціональної черги використання рішень (або розподіл часу між підсистемами, окремими задачами); аналіз результатів; недопущення помилок у розрахунках. Вимоги, поставлені до підсистеми «Координатор», передбачають: можливість ефективної підтримки інформації; забезпечення уявлення інформації до системи без додаткового кодування, здатність діалогу; забезпечення різних режимів розрахунків; видання результатів у вигляді машино- та відеограм; контроль за інформацією.

Програмна частина містить комплекс програмних моделей, що комплектуються за ієрархічним принципом із невеликих програмних блоків. Кожний із таких блоків виконує окрему логічну функцію. Коли окремі підпрограми невеликі за розміром, а їх правила побудови і зв'язки уніфіковані, посилюється контроль, підвищується ефективність експлуатації. Крім того, стає доцільним багаторазове використання блоків, що істотно зменшує обсяг усього програмного комплексу. Такий принцип побудови також забезпечує можливість модернізації комплексу програм за допомогою формалізованих зв'язків між програмами. Отже, залучення підпрограм із новими характеристиками не потребує змін в інших.

З організаційних моментів важливими є: розподіл, ієрархічна декомпозиція, що розв'язується, упорядкування зусиль розробників системи представлення знань, програмістів відповідно до призначення, поділ процесів розробки і відлагоджування програм, що входять до комплексу.

На основі проведеного аналізу сформулюємо задачі, які необхідно розв'язати:

1) здійснити аналіз характеристик соціоорієнтованих об'єктів для побудови ефективних стратегій і систем управління;

2) обґрунтувати використання інформаційних технологій як підстави створення систем підтримки прийняття управлінських рішень;

3) проаналізувати структури управління щодо ефективності управління соціоорієнтованими об'єктами та сформулювати вимоги до керівного персоналу;

4) обґрунтувати використання інформаційних технологій для створення базису моделей управління адміністративно-господарськими структурами;

5) проаналізувати вимоги до системи підтримки прийняття рішень та структури експертної системи, що входить до її складу;

6) розглянути структуру баз даних та проаналізувати ефективність методів формування запитів для швидкого відбору даних;

7) провести оптимізацію SQL-запитів у спеціальних базах даних про стан і ситуацію соціоорієнтованих систем;

8) розробити методи оптимізації процесу генерації планів запитів у реляційних СУБД із гнучкою структурою;

9) проаналізувати методи логічної оптимізації запитів у СУБД на швидкість відбору даних;

10) провести аналіз СУБД для побудови сховищ різнотипних і мультимедійних та картографічних даних;

11) розробити метод і стратегію виконання процедури об'єднань банків даних у розподілених базах інтелектуальних управляючих систем;

12) розробити метод побудови архітектури інформаційно-керівних систем, орієнтованих на аналіз різнорідних даних;

13) розробити методи й алгоритми оптимізації роботи пошукової машини СУБД у системі підтримки прийняття рішень на основі прискорення пошуку даних з символьним представленням;

14) розробити моделі активного навчання оперативного персоналу на підставі інформаційних та веб-технологій.

РОЗДІЛ 2
ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ СТВОРЕННЯ БАЗИСУ МОДЕЛЕЙ ДЛЯ УПРАВЛІННЯ АДМІНІСТРАТИВНО-ГОСПОДАРСЬКИМИ СТРУКТУРАМИ

2.1. Інформаційний базис систем підтримки прийняття рішень

Проблемна задача створення інформаційного базису для СППР на управління складними адміністративними й господарськими структурами з використанням комплексу ієрархічних моделей компонент системи повною мірою не розв'язана. Управління складними системами ґрунтується на інтеграції потенціалу знань як людини, колективу, так і використанні інформаційних та комп'ютерних технологій, експертних систем у структурі СППР [18, 46–47, 66–67].

Проблемна задача розроблення інформаційного забезпечення процесу управління полягає в наступному. Процес управління складними організаційними, адміністративними та корпоративними виробничими структурами ґрунтується на інформаційних технологіях та системному аналізі для відображення цільових задач та розроблення стратегії досягнення за наявних обмежень на ресурси при розв'язанні динамічних ситуацій. Відповідно, управлінський процес може бути реалізований особою з підтримкою СППР або автоматично.

Запропонована концепція синтезу програмного забезпечення має ієрархічну структуру, що описується системою моделей, які входять в інформаційну структуру ЕС — СППР: моделі об'єкта контролю й управління; моделі траєкторії контрольного стану; моделі взаємодії вимірювальна система — параметр стану об'єкта; моделі алгоритмів робастних систем відбору й цільової обробки даних; моделі відображення даних у цільовому просторі системи як образу динамічної ситуації; моделі класифікації та ситуації і стратегій прийняття цільових рішень; моделі забезпечення робастності каналу вимірювання.

На основі ієрархії моделей, які описують процеси формування, відбору, обробки даних, синтезуються відповідні алгоритми, що реалізовуються в рамках відповідного програмного й оперативного забезпечення. Наявні операційні системи і мови програмування володіють арсеналом засобів, який дає можливість створювати в середовищі Windows XP, Windows Vista, Windows 7, Windows 8 системи автоматизованого проектування та системи моделювання динаміки процесів у межах понять і засобів класичних теорій автоматичного регулювання, системного аналізу — без урахування збурень і завад, що діють на структуру і параметри об'єктів моделювання, а також стратегію прийняття рішень при управлінні об'єктами і блоками енергосистем [57, 63].

Системологія цілеспрямованих систем з єдиних позицій дає змогу впорядкувати і розвинути понятійний алгоритм, створити концептуальні моделі виробничої чи організаційної структури, на їх основі параметричні операторні й графові моделі вузлів і блоків, за допомогою яких можна описати моделі та синтезувати алгоритми обробки даних, що будуть базовими для синтезу програмного забезпечення систем автоматизованого моделювання, проектування і обробки даних для експертних діагностичних оцінок у процесі експлуатації [18, 46–47, 60, 68].

Особливої ваги набуває питання забезпечення стійкості й робастності програмного забезпечення на рівні алгоритмів і процесів для їх реалізації, що обгрунтовує відповідні вимоги до методології синтезу моделей блоків і моделей сигналів, які відображають динаміку процесів в агрегатах і блоках структури. При цьому проблемна ситуація задає два класи цільових задач: забезпечення робастності систем управління на наявних виробничих структурах; забезпечення робастності моделей об'єктів і алгоритмів обробки при ідентифікованих джерелах збурень у процесі моделювання й проектування.

Системологія цілеспрямованих структур дозволяє розв'язувати сформульовані проблемні задачі поетапно, відповідно, виконуючи [60, 67–68]: оцінку проблемної ситуації та формування цілей; синтез цільових задач у рамках моделей цільового простору; синтез

стратегій досягнення цілей; синтез концептуальних моделей систем для реалізації цілей; синтез моделей системи контролю і спостереження на основі робастних алгоритмів відбору і обробки даних; синтез моделі структури процесора прийняття рішень; синтез програмного забезпечення в інтегрованому операційному середовищі для реалізації числових методів і алгоритмів моделювання; моделювання динаміки на фоні збурень; оцінку якості концептуальних і числових моделей за заданими критеріями якості, стійкості, робастності; проектування систем та їх інформаційного і комп'ютерного забезпечення; монтаж, накладку, випробування, верифікацію режимів.

Методологія розв'язання проблемних задач у рамках заданих цілей базується на знанні проблемних областей: теорії сигналів і систем зі стохастичною структурою, системного аналізу й теорії інформації, стохастичної динаміки, статики й теорії випадкових процесів, теорії алгоритмів, понятійний апарат яких повинен бути узгодженим з використанням концепції інтегрованості людини, колективу у виробниче чи організаційне середовище. При цьому особа виконує функції технолога, управлінця, оперативного адміністратора, експерта [18, 66–67].

2.2. Експертна система в структурі системи підтримки прийняття рішень

Експертна система (ЕС) — прикладна система штучного інтелекту, що використовує формалізовані емпіричні знання фахівців з певної вузькоспеціалізованої предметної галузі, спроможна в межах цієї галузі приймати рішення на рівні експерта-професіонала, котрий допомагає оцінювати ситуацію в системі й формувати рішення. Експертними системами зазвичай замінюють експертів у небезпечних чи шкідливих умовах (наприклад, радіоактивного зараження) або для оперативної оцінки ситуації та ухвалення рішень, коли особиста участь експерта утруднена або неможлива (особливо в умовах аварійних, кризових ситуацій) [47].

Приклади сфер застосування ЕС у виробничих системах: інтерпретація даних експериментів; виявлення хімічних і біологічних структур; прогнозування подій після природних або техногенних катастроф; діагностика несправностей техніки або захворювань людини; планування цільових експериментів для ідентифікації структури об'єктів; пошук корисних копалин та створення геоінформаційних систем; керування наземним транспортом; в СППР для управління організаціями; в системах автоматизованого проектування.

Метою досліджень в галузі експертних систем є: розробка проблем (пристроїв), які під час вирішення важких для експерта-людини завдань одержують не гірші за якістю та ефективністю результати порівняно з експертними результатами. Зазвичай ЕС розв'язують важкоформалізовувані завдання або такі, що не мають алгоритмічного розв'язання; створення ефективних баз знань і баз даних реального часу на основі предметно-орієнтованої інженерії знань. Функції класифікації експертних систем представлені на рис. 2.1.

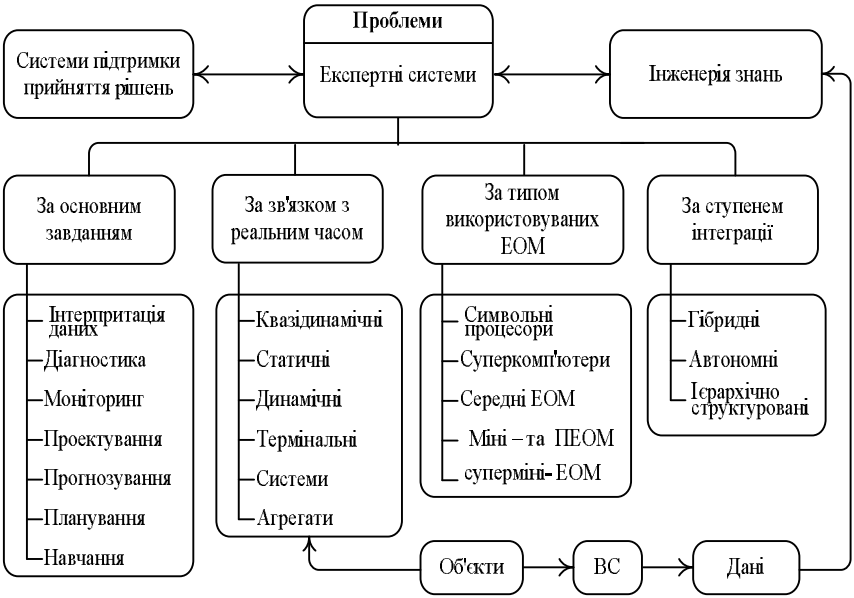


Рис. 2.1. Класифікація експертних систем

Основні задачі, що постають перед ЕС:

1. Інтерпретація — аналіз спостережуваних даних чи ситуацій для визначення їх змісту чи різних типів океанських суден.

2. Діагностика — класифікація та пошук несправностей у живих чи неживих системах, що базуються на результатах інтерпретації. Прикладом діагностичної ЕС є ANGY, що допомагає здійснювати діагностику та терапію звуження коронарних судин. Для діагностики помилок в апаратурі та математичному забезпеченні ЕОМ використовується ЕС GRIP.

3. Моніторинг — порівняння спостережуваних величин чи ситуацій з опорними (критичними) точками плану та видання повідомлень при відхиленні від плану; інший вид моніторингу — неперервний процес інтерпретації сигналів і видання повідомлень у ситуаціях, що потребують втручання системи вищого рівня чи людини. Приклади: допомогу диспетчерам атомного реактора забезпечує ЕС REACTOR; контроль аварійних датчиків на хімічному заводі — FALCON.

4. Проектування — знаходження такої конфігурації компонентів системи, що задовольняє цільові умови та множину проектних обмежень. Прикладом є ЕС SYN для синтезу електричних ланцюжків.

5. Прогнозування — проектування можливих наслідків конкретної ситуації. Прикладами таких ЕС є: WILLARD для передбачення погоди, ECON для здійснення прогнозів в економіці тощо.

6. Планування — розробка послідовності дій для досягнення множини поставлених цілей при заданих початкових умовах і часових обмеженнях. Прикладами ЕС такого типу є система ISIS для планування промислових замовлень, MOLGEN для планування експериментів тощо.

7. Інструктування (навчання) — допомога в освітньому процесі для вивчення певної дисципліни. Системи навчання з допомогою ЕОМ діагностують помилки під час вивчення певного предмета та підказують правильні рішення, а також планують процес спілкування вчителя з учнем залежно від успіхів учня для передання

знань. Приклад: система PROUST для вивчення мови програмування Паскаль.

8. Керування — керування поведінкою складного середовища чи системи.

9. Тестування — перевірка якості роботи з допомогою спеціальних тестів.

10. Ремонт — виконання плану організації виправлення деякого виявленого дефекту.

Класифікація ЕС за зв'язком з реальним часом:

— квазідинамічні ЕС інтерпретують ситуацію, що змінюється з деяким фіксованим інтервалом часу. Приклад: мікробіологічні ЕС, що знімають лабораторні виміри та покази щодо певного процесу один раз на 4–5 годин (наприклад, при виробництві лізину) та аналізують динаміку одержаних показників порівняно з попередніми результатами;
— статичні ЕС розробляються для предметних галузей, для яких база знань та інтерпретовані дані не змінюються в часі, є повністю стабільними. Приклад: діагностика несправностей в автомобілях;
— динамічні ЕС працюють разом із датчиками об'єктів у режимі реального часу з неперервною інтерпретацією даних, що надходять. Прикладами є керування гнучкими промисловими комплексами, моніторинг у палатах реанімації тощо.

Згідно з класифікацією за типом обчислювальних систем та інформаційних технологій маємо:

— ЕС для унікальних стратегічно важливих задач на суперкомп'ютерах (Ельбрус, CRAY, CONVEX та інші);
— ЕС на ЕОМ середньої потужності;
— ЕС на символьних процесорах та робочих станціях (SUN, APOLLO);
— ЕС на міні- та суперміні-ЕОМ (VAX, micro — VAX і т. д.);
— ЕС на персональних комп'ютерах (IBM PC, MAC II тощо).

Класифікація за ступенем інтеграції з іншими програмами:

— автономні ЕС працюють безпосередньо в режимі консультації з користувачем для вирішення специфічних «експертних» за-

вдань, причому немає потреби залучати традиційні методи опрацювання даних, моделювання тощо;

— гібридні ЕС — програмні комплекси, що мають у собі стандартні пакети прикладних програм (наприклад, математичну статистику, лінійне програмування або системи керування базами даних) та засоби маніпулювання знаннями.

Означення продукційної експертної системи в структурі СППР: розглянемо елементи теорії рішень у рамках ЕС. Продукційна система — це спосіб представлення знань у вигляді:

1) невпорядкованого набору продукційних правил;

2) робочої пам'яті;

3) механізму логічного виведення типу «розпізнавання — дія».

Правила формування рішень: продукційні правила часто називають ще продукціями. Продукція — це пара типу «умова–дія», «ситуація–дія», «причина–наслідок», «умова–висновок» і т. д., що визначає одну порцію знань, необхідних для розв'язання завдання. В умовній частині правила, закономірно, містяться умови, за яких виконується інша частина правила — частина дії.

$$\exists \text{Strat(U/Ci)} : \forall z, \exists \text{(Ui/Sitti)}; \pi \text{Rj [Ui:Zi} \rightarrow \text{Zi(Ci)]} \qquad (2.1)$$

Узагальнений запис правила-продукції такий:

$$\pi \text{Rj:}[Rnj:(Pr, Bc, A \Rightarrow B, Ac),] \,, \qquad (2.2)$$

де Rnj — індефікатор j–продукції в n–наборі продукцій; Pr – пріоритет правила продукції; Bc — передумова застосованості ядра продукції, що є предикатом, при виконанні якого активізується ядро продукції; $A \Rightarrow B$ — ядро продукції; Ac — дії та процедури, що повинні бути виконані після виконання ядра продукції; Zi — стан; U — управління; Ci — ціль; Strat — стратегія; ∀z — збурення.

Ядро продукції може бути детермінованим або недетермінованим: коли воно звучить, як «якщо A, то B» — детерміноване; коли «якщо A, то, *можливо, B*» — недетерміноване. Тобто в певних випадках при виконанні умови друга частина виконується обов'язково, а в деяких — з певною ймовірністю, причому величина такої ймовірності може бути також визначеною. Тоді ядро правила буде звучати: «якщо A, то з ймовірністю p реалізувати B».

Детерміновані продукції можуть бути також альтернативними. При цьому в правій частині ядра вказуються альтернативні можливості вибору, що оцінюються спеціальними вагами — коефіцієнтами вибору. Такими вагами можуть бути ймовірнісні, лінгвістичні, експертні оцінки тощо. Наприклад: «якщо *A*, то найчастіше реалізовується *B1,* а рідше *B2*».

У системі, що базується на продукційних правилах, їх кількість визначає величина бази знань. Досить складні системи можуть використовувати бази знань, що містять понад 5000 продукційних правил. Тож при складанні правил необхідно:

1. Використовувати мінімально достатній набір умов для визначення продукційного правила.
2. Уникати суперечливих продукційних правил.
3. Конструювати правила, базуючись на структурі відповідної предметної галузі.

Робоча пам'ять містить опис поточного стану об'єкта в процесі міркувань, який зіставляється з умовними частинами продукційних правил для вибору відповідних дій під час вирішення завдання. Якщо умова деякого правила відповідає змісту робочої пам'яті, може виконуватися дія, пов'язана з цією умовою. Дії продукційних правил призначені для зміни вмісту робочої пам'яті.

Механізм «розпізнавання — дія» (пошук за зразком). Поточний стан досліджуваної предметної галузі відображений у робочій пам'яті у вигляді сукупності образів, кожен з яких подається з допомогою фактів. Факти порівнюються з умовними частинами продукції та вибираються відповідні правила. Якщо факти збіжні з умовами більш ніж одного правила, такі правила утворюють так звану конфліктну множину і називаються допустимими. Відповідно до вибраного механізму вирішення конфлікту вибирається та активізується одна з продукцій конфліктної множини. Активація правила означає виконання його дії. При цьому змінюється вміст робочої пам'яті і далі все повторюється. Процес завершується, якщо вміст робочої пам'яті не відповідає жодній з умов наявних продукційних правил.

Таким чином, процес логічного висновку, що базується на пошуку за зразком, складається з чотирьох кроків:

1. Вибір образу.

2. Порівняння образу з умовами продукцій та формування конфліктної множини правил.

3. Вирішення конфлікту.

4. Виконання правила, яке забезпечує коректний висновок при прийнятті рішень на основі класифікації ситуації.

На основі проведеного аналізу процесу управління з використанням СППР важливою задачею є розроблення моделей предметно-орієнтованих баз даних і знань та виявлення ієрархії компонентів, блоків, агрегатів, побудова на цій основі модельного базису системи взаємозв'язків між ними. Необхідно встановити всі інформаційні канали передачі й прийому даних, які відображають стан об'єктів та структуру управляючих команд згідно зі стратегією досягнення мети функціонування ($A - \mathcal{T}$) структури.

2.3. Семантична оптимізація запитів у процесі формування та відбору даних

Перетворення запитів до баз даних (БД), розглянуті в [29, 112], базувалися на семантиці мови запитів. Однак у них зовсім не використовувалася семантика БД, до якої адресується запит. Кожне зі згадуваних перетворень можна здійснювати незалежно від того, яка конкретна БД використовується. Насправді ж при кожній правдиво реляційній базі даних (РБД) зберігається і деякий набір знань, що визначають, наприклад, цілісність бази даних. Беручи до уваги, скажімо, бази даних для *System R*, такі знання зберігаються в системних каталогах БД у вигляді попередньо сформульованих обмежень цілісності. Оскільки СУБД гарантує цілісність БД відповідно до цих обмежень, їх можна розглядати як аксіоми, навколо яких формулюються запити до бази даних [81, 106].

2.3.1. Модель семантичної оптимізації

Як початкові приклади перетворень запитів на основі семантичної інформації можна розглянути підходи, які використовуються у відомих *СУБД System R* і *INGRES* до забезпечення роботи з базами даних через представлення (подання) [74, 122]. Такі перетворення

не були орієнтовані на оптимізацію запитів, але безпосередньо її стосуються.

Нагадаємо, що представлення бази даних у термінах мов *SQL* і *QUEL* — це іменований каталогізований запит, який з погляду користувачів є таким самим об'єктом бази даних, як і відношення. Зокрема, поля представлення (елементи списку вибірки відповідного запиту) можуть мати окремі імена, оскільки поле представлення не обов'язково відповідає полю збереженої таблиці (елемент списку вибірки може бути заданий у вигляді арифметичного виразу). Подання можуть використовуватися в запитах так само, як і збережені відношення (ЗВ), що зберігаються в самій базі. Оскільки таким відношенням властиві деякі обмеження на використання їх в операторах модифікації БД, для нас їх розгляд не становить інтересу [29, 79, 82–83, 89].

Семантика представлень потребує лише їх точності. Це означає, що в момент виконання запиту над представленням одержувана інформація повинна точно відповідати поточному стану збереженої частини БД. Вважається, що виконання запиту над представленням потребує його матеріалізації, тобто обчислення відношення, визначеного запитом, пов'язаним із представленням. Загалом можливі два підходи до матеріалізації представлень.

Перший підхід допускає постійне збереження матеріалізованого представлення з підтримкою його узгодженості з поточним станом БД. При такому підході існують певні недоліки і переваги. До недоліків можна віднести непередбачувані накладні витрати на модифікацію матеріалізованих представлень при модифікації збережених відношень. Очевидна перевага — відсутність у потребі матеріалізації при кожному використанні представлення. Варто зазначити, що цей підхід не є поширеним у централізованих базах даних, проте він досить інтенсивно починає застосовуватись стосовно РБД, де матеріалізація представлення може бути затратною.

Другий підхід (його наводимо як приклад семантичних перетворень запитів) полягає в тому, що матеріалізація представлення з одержанням реального відношення, як правило, не здійснюється взагалі. Пояснимо, як слід розуміти тезис «як правило». При засто-

суванні другого підходу представлення зберігаються в каталогах БД у внутрішній формі, яка одержується після виконання граматичного аналізу запиту, що визначає конкретне представлення. Загалом може виявитися доцільним додаткове проведення логічної оптимізації цього запиту, зберігаючи вже перетворену внутрішню форму [112].

При обробці запиту над представленням до виконання фази логічної оптимізації здійснюється злиття-об'єднання внутрішніх форм запиту і представлення. При цьому утворюється певна нова перетворена внутрішня форма, а вже над нею здійснюється вся подальша обробка запиту, включно з логічною оптимізацією і вибором оптимального плану виконання запиту. Тепер можливо пояснити, чому такий спосіб обробки запиту над представленням допустимий не завжди. Одержане після злиття внутрішнє представлення запиту не повинно істотно відрізнятися від тих внутрішніх представлень, які можуть виникнути при обробці запитів на ЗВ. Інакше ускладнюються наступні фази. Наприклад, якщо представлення визначене з використанням SQL як:

DEFINE VIEW V (*C1*, *C2*) AS SELECT *C3*, AVG(*C4*) (2.3)
FROM *R* GROUP BY *C3*,

а запит має вигляд:

SELECT *C2*, AVG(*C1*) FROM *V* GROUP BY *C2*, (2.4)

у разі допущення злиття внутрішньої форми запиту з внутрішньою формою представлення ми б одержали внутрішню форму, що не відповідає жодному запиту на *SQL* над ЗВ. Отже, подальша обробка перетвореного запиту повинна бути дуже специфічною. Тож у *System R*, наприклад, якщо розпізнається така чи подібна ситуація, запит компілюється так, ніби *V* було ЗВ, а при виконанні запиту здійснюється явна матеріалізація представлення зі створенням тимчасового відношення.

Як уже було зазначено, такий підхід застосовується в *System R* і *INGRES* головно для того, щоб уникнути потреби в явній матеріалізації представлення під час виконання запиту. Однак насправді тут існує безпосередній зв'язок з оптимізацією. При злитті запиту з

представленням оптимізатор одержує більше інформації про конкретний запит, а тому може приймати правильніші рішення. Наведемо два прості приклади.

Нехай представлення визначене як:

DEFINE VIEW *V*(*C2*) AS SELECT *C1* FROM *R* WHERE *C1* > 6. (2.5)

Запит визначається:

SELECT *C2* FROM *V* WHERE *C2* < 6. (2.6)

Якби запит компілювався в розрахунку на явну матеріалізацію представлення, було б обрано такий спосіб його виконання, який базується на послідовному скануванні *V* з вибором кортежів, що задовольняють умову. Навіть більше, запит так і виконувався б, аби зрештою видати порожній результат [119]. Якщо ж використовувати підхід зі злиттям внутрішніх форм запиту і представлення, після злиття одержимо внутрішню форму, еквівалентну запиту:

SELECT *C1* FROM *R* WHERE *C1* < 6 AND *C1* >6. (2.7)

Уже на етапі логічної оптимізації можна було б встановити, що він еквівалентний запиту:

SELECT *C1* FROM *R* WHERE FALSE, (2.8)

з чого випливає, що результат запиту є порожнім.

Дещо складніший приклад.

Нехай представлення визначається як:

DEFINE VIEW *V*(*C3*) AS SELECT *C2* FROM *R* WHERE *C1* = 6. (2.9)

Задається запит

SELECT *C3* FROM *V* WHERE *C3* < 16. (2.10)

Допустимо, що відношення *R* кластеризовано за значеннями поля *C1*. На цей момент важливо, що може бути зроблений ефективний доступ до відношення *R* за умови, коли відомо умову рівності значення *C1* константі.

Після злиття запиту з представленням одержуємо внутрішню форму, еквівалентну запиту:

SELECT *C2* FROM *R* WHERE *C2* < 16 AND *C1* = 6. (2.11)

Для такого запиту існує спосіб виконання, не менш ефективний від того, який був би вироблений для матеріалізованого представлення *V*. Отже, явна матеріалізація представлення спричинила б лише додаткові накладні витрати.

З наведених прикладів випливає, що в ряді випадків означений спосіб обробки запитів над представленнями БД дозволяє досягти ефективнішого виконання запиту за рахунок надання оптимізатору більше інформації [121, 124, 126]. Концептуально та сама ідея міститься і в основі семантичної оптимізації запитів: використовувати при оптимізації запиту знання, які зберігаються в БД, незалежно від конкретного запиту [86, 94, 102].

Іншим прикладом аналогічних перетворень запитів, знову ж таки вироблених не для оптимізації, а які мають до неї безпосереднє відношення, є перетворення, здійснені над запитами на модифікацію БД для виконання існуючих у базі даних обмежень цілісності. Такий підхід уперше був використаний у СУБД *INGRES*, сьогодні він використовується й в інших системах, наприклад, у *System R* для обліку певних обмежень цілісності.

Нагадаємо, що в термінах мов *SQL* і *QUEL* обмеженням цілісності називається логічний вираз, який зберігається в каталогах бази, і складений з допустимих у мові предикатів над об'єктами бази даних. Бази даних перебувають у цілісному стані, якщо задовольняються всі сформульовані раніше обмеження цілісності. Транзакцією називається неподільна з точки зору цілісності БД послідовність дій над об'єктами БД, яка зберігає базу даних у цілісному стані. Проте загалом допускаються порушення цілісності БД усередині транзакції [64]. Пояснимо це на прикладі.

Нехай БД, що містить інформацію про структуру деякої організації, складається з двох відношень — *EMP* і *DEPT*. У відношенні *EMP* реєструються всі службовці цієї організації. Відповідно, схема цього відношення *EMP* (*EMP#*, *EMPNAME*, *DEPT#*), де поле *EMP#* має унікальний номер працівника, поле *EMPNANE* — ім'я працівника, а *DEPT#* — номер відділу організації, в якій працює конкретний працівник. У відношенні *DEPT* зберігається вся інформація про відділи. Воно має схему *DEPT*(*DEPT#*, *EMPCOUNT*), де в полі

EMPCOUNT зберігається загальна кількість співробітників цього відділу.

Очевидним обмеженням цілісності для такої бази даних у синтаксисі SQL є наступне:

ASSERT *A* ON *DEPT*: *EMPCOUNT* = SELECT COUNT (*) (2.12)
FROM *EMP* WHERE *EMP.DEPT#* = *DEPT.DEPT#*.

За своїм змістом це обмеження цілісності означає, що значення поля *EMPCOUNT* будь-якого кортежу відносин *DEPT* повинно бути узгоджене з реальною кількістю співробітників, зареєстрованих у відношенні *EMP* як працюючих у конкретному відділі.

Допустимо, що транзакція полягає в реєстрації нового співробітника і містить два оператори:

INSERT INTO *EMP*: <223, *Smith*, 3>; (2.13)
UPDATE *DEPT* SET *EMPCOUNT* = *EMPCOUNT* + 1 WHERE *DEPT#* = 3.

Тоді після виконання першого оператора сформульоване вище обмеження цілісності не виконується, проте другий оператор приведе базу даних до цілісного стану, тобто умову атомарності транзакції з погляду цілісності БД збережено. Для коректності зазначимо, що, по-перше, для забезпечення неподільності транзакції необхідними є деякі допоміжні синхронізуючі дії з боку системи. По-друге, в мові *SQL* існують спеціальні засоби автоматичної підтримки цілісності, які називаються тригерами (*triggers*). У роботі не розглядаємо подібних питань, оскільки вони не пов'язані з оптимізацією [93, 99–100].

Оскільки, з одного боку, подібні обмеження цілісності стосуються стану БД, а не елементарних дій з модифікації її об'єктів, а з другого — можуть порушуватися в межах транзакції, їх перевірка не може бути прив'язана до виконання оператора модифікації, а повинна виконуватись по закінченні транзакції (чи при виконанні явного оператора перевірки цілісності, що існує, наприклад, у *SQL*). Існують й інші обмеження цілісності, які необхідно перевіряти при кожному виконанні операторів модифікації, що потенційно можуть впливати на обмеження.

Зокрема, для тієї ж БД, що складається з відношень *EMP* і *DEPT*, можна адміністративно накласти обмеження, що прийняття і звільнення службовців може здійснюватися тільки в одному відділі, наприклад, у відділі № 6. Таке обмеження неможливо сформулювати як обмеження на стан бази даних. Зазвичай його формулюють у вигляді вимоги незмінності чисельності співробітників у всіх відділах, окрім шостого. Цього, однак, недостатньо, оскільки в одній транзакції можуть бути послідовно виконані оператори видалення й вставки кортежу у відношенні *EMP*. Таким чином можна поміняти співробітників інших відділів, не порушуючи зазначеного обмеження, але й не виконуючи вимог адміністрації.

Для формулювання обмежень цілісності такого типу в мові *SQL* існують спеціальні засоби. Наприклад, розглянуте вище обмеження можна сформулювати у вигляді:

ASSERT *A* IMMIDIATE ON *EMP*: *EMP.DEPT#* = 6. (2.14)

Семантично обмеження означає заборону занесення, видалення й модифікації кортежів у відношенні *EMP*, у яких значення поля *DEPT#* відмінне від 6. Допустимо тепер, що обробляється запит на видалення кортежу з відношення *EMP:*

DELETE *EMP* WHERE *EMPNAME* = '*Brown*'. (2.15)

Якщо при компіляції запиту не враховувати наявність сформульованого вище обмеження цілісності, єдино коректним способом виконання такого запиту буде наступний. Деяким способом послідовно вибрати кортежі, у яких значенням поля *EMPNAME* є текстовий рядок '*Brown*'; після чого викликати перевірку цього кортежу стосовно виконання умов обмеження цілісності, і якщо така перевірка виявиться позитивною, вилучити кортеж.

Якщо ж обмеження цілісності враховуються при компіляції, можна, як і у випадку обробки запитів над представленнями, зробити злиття внутрішніх форм запиту і відповідних обмежень цілісності, а потім здійснити спільну оптимізацію. У нашому випадку після злиття утвориться внутрішня форма, еквівалентна запиту:

DELETE *EMP* WHERE *EMPNAME* = '*Brown*' AND *DEPT#* = 6. (2.16)

При виконанні такого запиту стають зайвими додаткові виклики перевірок обмежень цілісності іншого типу, оскільки вони усі вже входять у логічну умову виконання операції видалення. Окрім того, оптимізатор отримує достатньо свободи у виборі способу виконання запиту [105, 107]. Наприклад, якщо відділи малочисленні й для відношення EMP підтримується індекс на поле *DEPT#*, можливо, оптимальним способом виконання запиту буде сканування відношення через індекс по *DEPT#* з умовою *DEPT#* = 6 разом із видаленням всякого вибраного таким чином кортежу зі значенням поля *EMPNAME*, рівним '*Brown*'. Таким чином, і в цьому випадку дії стосовно перетворення запиту, не спрямовані спеціально на оптимізацію, можуть сприяти ефективнішому виконанню запиту. Тут також, як і у випадку обробки запитів над представленнями, вдається підвищити ефективність виконання запиту за рахунок використання інформації, яка незалежно зберігається в базі даних.

Отже, розглянуто два приклади, в яких можна одержати ефективніший план виконання запиту за рахунок його перетворення з використанням додаткової семантичної інформації, щоб показати, що ідея семантичної оптимізації в принципі не є новою, хоча, звичайно, відповідні перетворення, які проводяться над запитами в *System R* і *INGRES*, ніколи не називалися семантичною оптимізацією. Водночас існує принципова різниця між розглянутими вище перетвореннями запитів і тими, які здійснюються при семантичній оптимізації. Основна відмінність полягає у тому, що, коли здійснюється злиття внутрішньої форми запиту з внутрішньою формою представлення чи внутрішніми формами обмежень цілісності, потрібно повністю й однозначно використовувати зовнішню інформацію.

При обробці запиту над представленням, якщо запит, який визначає представлення, містить умови вибірки, всі вони повністю повинні бути додані через AND до умови вибірки запиту, який обробляється. Інакше не буде гарантованою точність представлення. У випадку обробки запиту на модифікацію бази даних до логічної умови модифікації повинні бути додані через AND усі логічні умови відповідних обмежень цілісності, бо не буде гарантоване їх до-

тримання при виконанні запиту. Тобто у таких випадках перетворення здійснюються однозначно і незалежно від того, чи будуть вони сприяти оптимізації запиту [70, 109, 115, 116].

Сутність ідеї семантичної оптимізації полягає у використанні набору знань, які не обов'язково застосовувати при обробці запиту, але використання яких у певній комбінації може призвести до оптимальнішого виконання запиту. Якщо вважати, наприклад, що семантична оптимізація має справу тільки зі знаннями, представленими у вигляді обмежень цілісності бази даних, концептуально дії при семантичній оптимізації можна розуміти наступним чином. Виробляється безліч перетворених внутрішніх представлень запиту, причому кожне перетворення використовує деякий піднабір обмежень цілісності. Якщо, наприклад, у БД визначені два обмеження цілісності *A* і *B* з логічними умовами *F1* і *F2* відповідно, і обробляється запит з логічною умовою вибірки *F*, то в процесі семантичної оптимізації будуть отримані внутрішні представлення, еквівалентні запитам з умовами вибірки:

$$F,\ F \text{ AND } F1,\ F \text{ AND } F2 \text{ і } F \text{ AND } F1 \text{ AND } F2. \qquad (2.17)$$

Далі кожне з отриманих внутрішніх представлень піддається повній подальшій обробці, включаючи логічну оптимізацію і вибір оптимального плану виконання; і з усіх отриманих у такий спосіб планів (усі вони семантично еквівалентні, тобто виробляють той самий результат) вибирається найдешевший, який і стає реальним планом виконання вихідного запиту.

Зазначимо, що умови цілісності можна використовувати для розширення умови запиту тільки тоді, коли вони гарантовано є істинними. Як уже зазначалося, в *System R*, наприклад, обмеження цілісності першого типу (тобто обмеження на стани бази даних) можуть загалом порушуватися всередині транзакції. Тож у загальному випадку під час обробки запитів мовою *SQL* у СУБД класу *System R* використання семантичної оптимізації запитів на основі обмежень цілісності може виявитися некоректним. Проте це загальна властивість підходу *System R*: у транзакціях, які змінюють стан БД, запити на вибірку можуть видавати результати, що суперечать обмеженням цілісності БД [84, 123].

Наприклад, нехай у транзакції виконується така послідовність операторів SQL:

INSERT INTO *EMP*: <223, *Smith*, 3>;

SELECT *EMPCOUNT* FROM *DEPT* WHERE *DEPT*# = 3; (2.18)

UPDATE *DEPT* SET *EMPCOUNT* = *EMPCOUNT* + 1WHERE *DEPT*# = 3.

Транзакція коректна зі сторони цілісності БД, але другий оператор видасть семантично неправильний результат. Аналогічного ефекту можна досягти і при спробі семантичної оптимізації. Нехай у транзакції виконуються оператори:

INSERT INTO *EMP*: <223, *Smith*, 3>;

SELECT COUNT (*) FROM *EMP* WHERE *EMP.DEPT*# = 3; (2.19)

UPDATE *DEPT* SET *EMPCOUNT* = *EMPCOUNT* + 1 WHERE *DEPT*# = 3.

Якщо в базі даних, як і раніше, існує природне обмеження цілісності *A*, після семантичної й логічної оптимізації другий запит може бути перетворений до внутрішнього представлення як еквівалентний запит:

SELECT *EMPCOUNT* FROM *DEPT* WHERE *DEPT*# = 3. (2.20)

Тоді, як і в попередньому прикладі, результат запиту буде відповідати обмеженню цілісності, але існуватиме розбіжність із реальним станом БД до моменту виконання запиту.

Однак якщо транзакція є «лише для читання», тобто в ній виконуються тільки запити на вибірку, і це відомо при компіляції окремих вхідних у транзакцію операторів, семантична оптимізація на основі обмежень цілісності буде завжди коректною, оскільки кожна транзакція на початку має гарантовано узгоджений стан бази даних.

Очевидно, властивість *System R* допускати порушення цілісності бази даних усередині транзакції (на практиці дуже корисна), а також підхід до автономної компіляції запитів поза їх зв'язком із властивостями транзакції, в якій вони будуть виконуватися, не дозволили застосувати в цій системі механізму, подібного до семантичної оптимізації.

Відходячи від особливостей *SQL* і *System R*, зауважимо, що для розумного застосування семантичної оптимізації зручно мати семантичну інформацію про обмеження цілісності бази даних у вигляді правил, представлених в імплікативній формі. Тоді можна домогтися більш осмисленого порядку перетворень. Наприклад, у нашій базі даних про структуру організації відношення *EPM* розширене ще двома полями — *STATUS* і *SALARY*. Значення поля *STATUS* характеризують посаду працівника, а поле *SALARY* — його платню. На БД може бути накладено обмеження цілісності, яке полягає в тому, що платня, яка перевищує 40000, може бути призначена тільки начальникам відділів. З використанням *SQL* це обмеження може бути сформульоване як:

ASSERT *A* ON *EMP*: IF *SALARY* > 40000 THEN STATUS = '*Manager*'. (2.21)

Припустимо, що обробляється запит:

SELECT *EMPNAME*, (2.22)

STATUS FROM *EMP* WHERE *SALARY* = 20000.

Якщо не враховувати імплікативного характеру обмеження цілісності, за загальними правилами буде здійснено злиття внутрішніх представлень запиту та обмеження цілісності, що, очевидно, переваг не дасть. Якщо ж розглядати обмеження цілісності як правило перетворення, а ліву частину імплікації — як умову застосування правила, злиття навіть не здійснюватимуться, що в загальному випадку скоротить час обробки запиту.

Проте для запиту

SELECT *EMPNAME* STATUS FROM *EMP* WHERE *SALARY* > 40000 (2.23)

правило перетворення застосовне і приводить до семантично еквівалентного запиту:

SELECT EMPNAME, (2.24)

STATUS FROM EMP WHERE STATUS = 'Manager',

виконання якого може бути ефективнішим.

2.4. Оптимізації SQL-запитів у спеціалізованих базах даних

Зазвичай, кажучи про оптимізацію в реляційних СУБД, розуміють аспект оптимізації запитів, тобто такий спосіб виконання запитів, коли за початковим представленням запиту шляхом його синтаксичних і семантичних перетворень виробляється процедурний план виконання запиту, найоптимальніший при існуючих у базі даних керуючих структурах. Відповідні перетворення початкового представлення запиту виконуються спеціальним компонентом СУБД — оптимізатором, а оптимальність виробленого ним плану запиту має досить умовний характер: план оптимальний відповідно до критеріїв, закладених в оптимізатор; при цьому цілком реально існують можливі відхилення від фактичної оптимальності [36].

У зв'язку з оптимізацією запитів існує достатня кількість проблем: перетворень запиту до ефективнішого непроцедурного представлення (логічна оптимізація), вибору набору альтернативних процедурних планів виконання запиту, оцінок вартості виконання запиту за обраним планом і т. д. Для кожного класу проблем зафіксовано декілька підходів їх розв'язання. Наприклад, проблеми, пов'язані з логічною оптимізацією запитів, породили напрям, названий семантичною оптимізацією [88, 94, 114]. Водночас зауважимо, що чимало дослідників працюють над проблемами оцінок вартості процедурних планів виконання запитів [73, 77, 84, 86, 103, 123], проте питання вірогідності оцінок дотепер остаточно не вирішено.

Можна розглядати оптимізацію й у ширшому контексті. Оптимізатор запитів вибирає найоптимальніший спосіб виконання запиту на основі відомих в оптимізаторі стратегій виконання елементарних складових запиту і способів композиції складніших стратегій з урахуванням елементарних. Тому простір пошуку оптимального плану виконання запиту обмежено заздалегідь фіксованими елементарними стратегіями. Отож значеннєвим напрямом досліджень, безпосередньо пов'язаних із питаннями оптимізації, є пошук нових, ефективніших елементарних стратегій [72, 85, 108, 120]. У контексті реляційних СУБД сказане найбільше стосується розробки ефективних алгоритмів виконання реляційної операції з'єднання найбільш

накладної реляційної операції [76, 78]. При цьому досліджуються як можливості вибору більш адекватних для ефективного виконання операції керуючих структур бази даних, так і шляхи підвищення ефективності внаслідок розпаралелювання виконання операції на спеціалізованій апаратурі (тут напрям наших досліджень дотичний з тематикою машин баз даних).

2.4.1. Запити в реляційних СУБД

Останнім часом особливо багато праць стосується оптимізації запитів і вибору ефективних способів виконання реляційних операцій у розподілених реляційних системах керування базами даних [80, 87, 102, 103, 104, 127]. Звичайно, існує чимало варіантів і фізичної організації розподілених баз даних (з підтримкою копій зв'язків у декількох вузлах мережі, з горизонтальним чи вертикальним поділом відносин у декількох вузлах, з підтримкою миттєвих знімків бази даних тощо), і алгоритмів виконання реляційних операцій при кожній такій організації. Хоч сьогодні реально існує і функціонує декілька розподілених реляційних СУБД (наприклад, System R і розподілена INGRES, MSSQL, MySQL, Oracle), не можна вважати, що остаточно знайдено адекватні розв'язання аналізованих проблем.

Зрештою, порівняно новою, ще недостатньо дослідженою, але, безумовно, надзвичайно важливою є задача глобальної оптимізації запитів у системах баз даних. Глобальну оптимізацію розуміємо як спільну оптимізацію заздалегідь відомого набору запитів. Це найактуальніше в системах логічного програмування (і подібних системах, пов'язаних з обробкою правил), реалізованих на основі реляційних баз даних. При такому підході виконання логічної програми в кінцевому варіанті зводиться до виконання великої кількості запитів до бази даних, причому, як правило, запити містять об'єднання (тобто є складеними). Спільна оптимізація таких запитів може різко скоротити загальний час виконання. Тобто глобальна оптимізація систем запитів зводиться до вияву спільних підвиразів у цих запитах, а потім однократного обчислення підвиразів зі збереженням результатів у тимчасових буферах-зв'язках. У цьому

аспекті в роботі опрацьовано також певні рекомендації, зокрема зі створення тимчасових керівних структур баз даних для оптимізації виконання системи запитів [32–33]. З наведеного огляду напрямів досліджень і розробок в галузі оптимізації виконання запитів у реляційних СУБД можна зробити висновок про практичну неосяжність цієї тематики. Отож у науковому дослідженні ми не прагнемо докладно описати кожний напрям з характеристиками усіх найважливіших задач та методів їх розв'язання, обмежимося лише точним формулюванням проблем і прикладами рішень на основі наявних джерел. Вибір джерела при цьому, звичайно, цілком суб'єктивний; з ним можна і не погоджуватися. Проаналізуємо ті роботи, які, на наш погляд, є найважливішими під час організації реляційних систем, орієнтованих на використання мови SQL (це зумовлено щораз більшою сферою поширення цієї мови), а також розглянемо інші напрацювання, які (знову-таки ж на наш погляд) є найцікавішими й найперспективнішими. Зауважимо, що оптимізації запитів у реляційних СУБД присвячені декілька оглядових праць. Російською мовою сьогодні доступні книги Дейта [8], Ульмана [64] і Мейєра [25], де проблем оптимізації стосуються окремі розділи. Матеріал у [64, 25] викладено на більш формальному рівні й стосується він меншої кількості проблем. Сучасніші проблеми розглянуто в останньому виданні книги Дейта. Найповніший, на наш погляд, огляд підходів і методів оптимізації запитів подано в статті [71]. Проте, зважаючи на широкий обсяг тематики, вважаємо доцільним написання ще однієї праці, оскільки останнім часом з'явилося ряд нових і цікавих робіт, пов'язаних з тематикою оптимізації в реляційних СУБД. Крім того, як уже відзначалося, у роботі розглядається ширший клас проблем.

Перш ніж перейти до опису конкретних проблем і рішень в галузі оптимізації запитів, розглянемо типовий для сучасних реляційних СУБД шлях обробки запиту, що надійшов у СУБД мовою запитів, наприклад, SQL чи QUEL.

2.4.2. Алгоритм виконання обробки запиту в реляційних СУБД

Згідно зі [106] обробку запиту, що надійшов у систему, можна представити як таку, що складається з етапів чи фаз, поданих на рис. 2.2.

На першому етапі запит, представлений мовою запитів, піддається лексичному і синтаксичному аналізу. При цьому виробляється його внутрішнє представлення, що відображає структуру запиту і містить інформацію, яка характеризує об'єкти бази даних, згадані в запиті (відношення, поля і константи). Інформація при збереженні в базі даних об'єктів вибирається з каталогів бази даних. Внутрішнє представлення запиту використовується і перетворюється на наступних стадіях обробки запиту. У пропонованій праці нас не будуть цікавити питання синтаксичного аналізу запиту, оскільки вони не пов'язані з оптимізацією. Зауважимо лише, що важливим є вибір внутрішнього представлення, яке повинно бути достатньо зручним для наступних оптимізаційних перетворень.

На другому етапі запит згідно зі своїм внутрішнім представленням піддається логічній оптимізації. При цьому можуть застосовуватися різні перетворення, які «покращують» початкове подання запиту. Серед них можуть бути еквівалентними перетворення, після проведення яких внутрішнє подання стає семантично еквівалентним початковому (наприклад, приведення запиту до деякої канонічної форми).

Перетворення можуть бути й семантичними. Коли одержуване представлення не є семантично еквівалентним початковому, проте гарантується, результат виконання перетвореного запиту збігається з результатом запиту в початковій формі з дотриманням обмежень цілісності, що існують у базі даних. У будь-якому випадку після виконання другої фази обробки запиту його внутрішнє представлення залишається не процедурним, хоча і є в певному розумінні ефективнішим за початкове.

Рис. 2.2. Етапи обробки запиту в реляційних СУБД

Третій етап обробки запиту полягає у виборі з урахуванням інформації, якою володіє оптимізатор, набору альтернативних процедурних планів виконання даного запиту відповідно до його внутрішнього представлення, отриманого на другій фазі. Крім того, для кожного плану оцінюється передбачувана вартість виконання запиту за ним. Під час оцінювання використовується статистична інформація про стан бази даних, доступний оптимізатору. З отриманих альтернативних планів вибирається найдешевший, і саме його внутрішнє представлення тепер відповідає оброблюваному запиту.

На четвертому етапі за внутрішнім представленням найбільш оптимального плану виконання запиту формується виконуване представлення плану. Воно може бути програмою в машинних кодах, якщо, як у випадку System R, система орієнтована на компіляцію запитів у машинних кодах, чи бути машинно-незалежним, але зручнішим для інтерпретації, якщо, як у випадку INGRES, система орієнтована на інтерпретацію запитів. У нашому випадку це не принципово, оскільки четверта фаза обробки запиту вже не пов'язана з оптимізацією.

На п'ятому етапі обробки запиту відбувається його реальне виконання відповідно до виконуваного плану запиту. Це або виконання відповідної підпрограми, або виклик інтерпретатора з передачею йому для інтерпретації виконуваного плану.

2.4.3. Логічна та семантична оптимізації запитів

При класичному підході до організації оптимізаторів запитів на етапі логічної оптимізації виробляються деякі еквівалентні перетворення внутрішнього представлення запиту, що «покращують» початкове внутрішнє представлення за конкретним фіксованим в оптимізаторі планом. Такі «покращання» мають досить умовний характер, тобто тісно пов'язані зі специфікою загальної організації оптимізатора, зокрема з тим, як побудований третій етап обробки запиту.

Отож доволі складно навести повну характеристику й класифікацію методів логічної оптимізації. Обмежимося лише декількома

прикладами, а в основній частині підрозділу розглянемо один частковий, але важливий клас логічних перетворень, які стосуються складних запитів, побудованих на мові SQL [97].

Загальновідомий клас логічних перетворень становлять перетворення, пов'язані з приведенням предикатів, що задають умову вибірки в певному запиті, до канонічного вигляду (представлення). Тут беруться до уваги предикати, що містять операції порівняння простих значень. Загалом такий предикат має вигляд «арифметичний вираз ор арифметичний вираз», де арифметичні вирази лівої й правої частин зазвичай містять імена полів і константи (такі предикати допускаються, наприклад, у мові SQL, причому серед констант можуть бути як літеральні константи, так і імена глобальних змінних, значення яких стають відомими тільки під час реального виконання запиту).

Семантична оптимізація запитів. Перетворення запитів, розглянуті в попередньому підрозділі, базувались на семантиці мови запитів. У них, однак, зовсім не використовувалася семантика бази даних, до якої адресується запит. Кожне зі згадуваних перетворень може бути зроблене незалежно від того, яка конкретна база даних задіюється. Насправді ж, при кожній реальній реляційній базі даних зберігається і певний набір знань, що визначають, наприклад, цілісність бази даних.

Маючи на увазі, скажімо, бази даних, підтримуваних System R, такі знання зберігаються в системних каталогах бази даних у вигляді попередньо сформульованих обмежень цілісності. Оскільки СУБД гарантує цілісність бази даних відповідно до таких обмежень, їх можна розглядати як аксіоми, навколо яких формулюються запити до бази даних.

Семантика представлень потребує лише їх точності. Це означає, що в момент виконання запиту над представленням одержувана інформація повинна точно відповідати поточному стану збереженої частини бази даних. Вважається, що виконання запиту над представленням потребує його матеріалізації, тобто обчислення відношення, визначеного запитом, який пов'язаний із представленням. Загалом можливі два підходи до матеріалізації представлень.

Перший підхід допускає постійне збереження матеріалізованого представлення з підтримкою його узгодженості з поточним станом бази даних. Такий підхід має свої недоліки й переваги. До недоліків належать непередбачувані накладні витрати на модифікацію матеріалізованих представлень при модифікації збережених відношень. Очевидна перевага — відсутність потреби матеріалізації при кожному використанні представлення. Зазначимо, що описаний підхід не є поширеним у централізованих базах даних, проте досить інтенсивно починає застосовуватися стосовно розподілених баз даних, де матеріалізація представлення може бути затратною.

Другий підхід, який ми і наводимо як приклад семантичних перетворень запитів, полягає в тому, що матеріалізація представлення з одержанням реального відношення, як правило, не здійснюється взагалі. Надалі пояснимо, що охоплює тезис «як правило». При застосуванні другого підходу представлення зберігаються в каталогах бази даних у внутрішній формі, яка одержується після виконання граматичного аналізу запиту, що визначає це представлення. Може бути доцільним також додаткове проведення логічної оптимізації цього запиту, зберігаючи вже перетворену внутрішню форму.

2.4.4. Оцінка альтернативних планів виконання запитів

Усі оптимізуючі перетворення запитів, розглянуті в попередніх підрозділах, залишали внутрішнє представлення запиту не процедурним. Навіть якщо говорити про процедурність на рівні виконання реляційних операторів у виразах реляційної алгебри, то вона є дуже умовною, оскільки, наприклад, порядок виконання операцій реляційного об’єднання зовсім не мусить збігатися з порядком їх послідовності у виразі. Окрім того, при реальному виконанні запиту ту саму операцію об’єднання можна виконати багатьма способами.

Процедурним представленням, чи планом виконання запиту, будемо називати таке представлення, в якому детально відображе-

ний порядок виконання операцій доступу до бази даних фізичного рівня. Як правило, у реляційних СУБД, орієнтованих на використання традиційної апаратури без використання спеціалізованих процесорів чи пристроїв зовнішньої пам'яті, виділяється підсистема керування даними на фізичному рівні.

Наприклад, у System R така підсистема називається RSS (Relational Storage System) і забезпечує простий інтерфейс, який дає можливість послідовно переглядати кортежі відносин, що задовольняють задані умови щодо значення полів з використанням індексів чи простим скануванням сторінок бази даних. RSS також дозволяє створювати відсортовані тимчасові файли і заносити, видаляти та модифіковувати індивідуальні кортежі. Докладніше інтерфейс RSS описується в [111]. Подібні підсистеми явно чи неявно виділяються у всіх аналогічних СУБД [98, 101, 103, 114].

Очевидно, в таких умовах до виконання запиту необхідно виробити його процедурне представлення. Крім того, оскільки таке представлення загалом вибирається неоднозначно, потрібно вибрати серед альтернативних планів запиту один, найбільш прийнятний згідно з деякими критеріями. Як правило, критерієм вибору плану виконання запиту є мінімізація затрат виконання.

Таким чином, під час обробки запиту на стадії, яка настає після логічної оптимізації, розв'язується дві задачі. Перша — на основі одержуваного після вироблення логічною оптимізацією внутрішнього представлення запиту й інформації, яка характеризує керівні структури бази даних (наприклад, індекси), вибрати набір потенційно можливих планів виконання цього запиту. Друга задача — оцінити вартість виконання запиту відповідно до кожного альтернативного плану і вибрати варіант з найменшою вартістю.

РОЗДІЛ 3
ІНФОРМАЦІЙНІ ТА СИСТЕМНІ ТЕХНОЛОГІЇ ПОБУДОВИ ДАНИХ У СТРУКТУРІ ОПЕРАТИВНОГО УПРАВЛІННЯ СОЦІООРІЄНТОВАНИМИ СИСТЕМАМИ

3.1. Використання провідних СУБД для побудови сховищ даних

Головним джерелом інформації, що надходить в оперативну БД, є діяльність корпорації. Для проведення аналізу даних потрібне застосування зовнішніх джерел інформації (наприклад, статистичних звітів). Сховище даних повинно містити як внутрішні корпоративні дані, так і зовнішні. Для оперативного опрацювання необхідні свіжі дані за декілька останніх місяців, для проведення достовірного аналізу й прогнозування в сховище даних потрібно мати інформацію про діяльність корпорації та стан ринку протягом декількох років. Обсяг аналітичних БД значно більший, ніж оперативних.

У корпораціях одночасно існує декілька оперативних ІС з власними БД. Оперативні БД можуть містити семантично еквівалентну інформацію, подану в різних форматах, із неоднаковою вказівкою часу її надходження, іноді навіть суперечливу. Сховище даних повинно містити однотипно подану й узгоджену інформацію, що максимально відповідає утриманню оперативних БД.

Оперативні ІС створюються для розв'язання конкретних задач. Інформація з БД вибирається часто й невеличкими порціями. Зазвичай набір запитів до оперативної БД відомий уже під час проектування. Набір запитів до аналітичної бази даних передбачити неможливо. Сховища даних існують, щоб відповідати на нерегламентовані запити аналітиків. Можна розраховувати тільки на те, що запити надходитимуть не дуже часто й обробляють великі обсяги інформації. Розміри аналітичної БД стимулюють використання запитів з агрегатами (сума, мінімальне, максимальне, середнє значення тощо).

Оперативні БД за своєю природою є надзвичайно мінливими, що враховується у типових СУБД (нормалізована структура БД, рядки зберігаються невпорядковано, B-дерева для індексації, транз-

акційність). При малій мінливості аналітичних БД (тільки при завантаженні даних) доцільними є впорядкованість масивів, швидші методи індексації при масовій вибірці, сховище заздалегідь агрегованих даних. Для оперативних ІС звичайно достатньо захистити інформацію на рівні таблиць. Інформація аналітичних БД настільки критична для корпорації, що потрібний великий ступінь захисту (індивідуальні права доступу до визначених полів таблиці) [58].

Інформаційна технологія. Сховище даних — предметно-орієнтований, інтегрований, незмінний, що підтримує хронологію, набір даних, організований для цілей підтримки керування [92]. Підхід до побудови СД для інтеграції неоднорідних джерел даних принципово відрізняється від підходу динамічної інтеграції різнорідних БД. Реально будується нове масштабне сховище, в якому керування даними відбувається за іншими правилами, ніж у вихідних оперативних БД [110].

В основі концепції сховища даних містяться дві головні ідеї.

1 Інтеграція роз'єднаних деталізованих даних (деталізованих у контексті, що вони описують деякі конкретні факти, властивості, події тощо) у єдиному сховищі. У процесі інтеграції повинно виконуватися узгодження деталізованих даних і, можливо, їх агрегація. Дані можуть надходити з історичних архівів корпорації, оперативних баз даних, зовнішніх джерел.

2 Поділ наборів даних і додатків, які використовуються для оперативного опрацювання й застосування для вирішення задач аналізу.

Предметна орієнтація: на відміну від БД, у традиційних СУБД-системах, де дані підібрані відповідно до конкретних додатків, інформація в *DW (Data Warehause)* орієнтована на задачі підтримки прийняття рішень. Для системи підтримки прийняття рішень необхідні «історичні» дані — факти продажу за визначений інтервал часу. Добре спроектовані структури даних *DW* відображають розвиток усіх напрямів бізнесу компанії в часі [59]. Оскільки в *DW*-технології об'єкти даних виходять на перший план, особливі вимоги приділяються до структур БД, які використовуються для створення інформаційних сховищ. Принципово відрізняється і структу-

ра баз даних для СУБД- і *DW*-систем. Інакше до них надходить тільки та інформація, що може бути корисною для роботи систем підтримки прийняття рішень.

Інтегрованість даних: дані в інформаційне сховище надходять із різноманітних джерел, де поля можуть мати різні імена, атрибути, одиниці виміру й способи кодування. Після завантаження в *DW* дані очищуються від індивідуальних ознак, тобто ніби приводяться до спільного знаменника. З цього моменту поля рекомендуються користувачеві у вигляді єдиного інформаційного простору. Якщо в чотирьох різних додатках стать клієнта кодувалася чотирма різноманітними способами, в інформаційному сховищі буде використана єдина для всіх даних схема кодування (наприклад, м, ж).

Інваріантність у часі: у СУБД-системах істинність даних гарантована тільки в момент читання, оскільки вже в наступну мить записи можуть змінитися в результаті чергової транзакції. Важлива відмінність *DW* від СУБД-систем — дані в них зберігають свою істинність у будь-який момент читання. У СУБД-системах інформація часто модифікується як результат виконання будь-яких транзакцій. Тимчасова інваріантність даних у *DW* досягається у результаті введення полів з атрибутом «час» (день, тиждень, місяць) у ключі таблиць. Як наслідок, записи в таблицях *DW* ніколи не змінюються, вони становлять знімки даних, зроблені у визначені періоди часу. У *DW* утримуються ніби моментальні знімки даних. Кожний елемент у своєму ключі явно або опосередковано зберігає тимчасовий параметр, наприклад день, місяць або рік.

Цілісність — стабільність інформації. У СУБД-системах записи можливо регулярно добавляти, видаляти і редагувати. У *DW*-системах, як випливає з вимоги тимчасової інваріантності, раз завантажені дані теоретично ніколи не змінюються. Стосовно них можливі тільки дві операції: початкове завантаження і читання (доступ). Це і визначає специфіку проектування структури бази даних для *DW*. Якщо при створенні СУБД-систем розроблювачі повинні враховувати такі моменти, як відкочення транзакцій після збою сервера, журналізація подій, боротьба зі взаємними блокуваннями процесів, зберігання цілісності даних, для *DW* окреслені

проблеми не настільки актуальні — перед розробниками постають інші задачі, пов'язані, наприклад, із забезпеченням високої швидкості доступу до даних.

Мінімізація надмірності інформації: оскільки інформація в *DW* завантажується з СУБД-систем, постає питання, чи не веде це до надмірної надлишковості. Її немає, стверджує Білл Інмон. Насправді надмірність мінімальна (приблизно 1%!), що пояснюється певними причинами: при завантаженні інформації з СУБД-систем у *DW* дані фільтруються та обробляються. Багато з них взагалі не потрапляє в *DW* — вони позбавлені змісту з погляду використання в системах підтримки прийняття рішень; інформація в СУБД-системах має, як правило, оперативний характер, і дані, які втратили актуальність, будуть видалені. У *DW*, навпаки, зберігається історична інформація, і щодо цього перекриття змісту *DW* даними СУБД-систем є дуже незначним; у *DW* зберігається деяка підсумкова інформація, що в базах даних СУБД-систем взагалі відсутня; під час завантаження в *DW* записи сортуються, очищаються від непотрібної інформації та приводяться до єдиного формату. Після такого опрацювання це вже зовсім інші дані.

3.1.1. Головні компоненти інформаційного сховища даних і знань

ПЗ проміжного прошарку забезпечує мережний доступ і доступ до баз та сховищ даних. Завантаження і попереднє опрацювання: цей рівень містить набір засобів для завантаження даних із СУБД-систем і зовнішніх джерел. Виконується, як правило, у поєднанні з додатковою обробкою: перевіркою даних на чистоту, консолідацію, форматуванням, фільтрацією.

Рівень інформаційного доступу забезпечує безпосереднє спілкування користувача з даним *DW* за допомогою стандартних систем маніпулювання, аналізу й надання даних типу *MS Excel*, *MS Access*, *Lotus 1-2-3* тощо.

Рівень керування (адміністрування) відстежує виконання процедур, необхідних для відновлення інформаційного сховища або

підтримки його стану. Тут використовуються процедури довантаження даних, перебудови індексів, реплікації даних, побудови звітів, формування повідомлень користувачам, контролю цілісності.

Проблеми інтеграції даних. Зупинимося лише на деяких проблемах реалізації сховища даних: неоднорідність програмного середовища; розподілений характер організації; підвищені вимоги до безпеки даних; необхідність наявності багаторівневих довідників метаданих; потреба в ефективному збереженні й обробці дуже великих обсягів інформації.

Неоднорідність програмного середовища: сховище даних практично ніколи не створюється на порожньому місці. Майже завжди кінцеве вирішення буде різнорідним, тобто в ньому використовуватимуться автономно розроблені програмні засоби. Насамперед це стосується формування інтегрованого узгодженого набору даних, що може надходити з різнорідних баз даних, електронних архівів, комерційних електронних каталогів, довідників. У процесі побудови сховища даних необхідно вирішувати задачу побудови єдиної, узгодженої інформаційної системи на основі неоднорідних програмних засобів і рішень. Під час вибору засобів реалізації сховища даних треба враховувати множину чинників, що охоплюють різні рівні сумісності різноманітних ужиткових компонентів.

Розподілений характер організації: у концепції сховища даних визначено, що операційна аналітична обробка може виконуватися в будь-якому вузлі мережі, незалежно від місця розташування головного сховища. Хоча при аналітичній обробці дані тільки зчитуються і потреба в синхронізації відсутня, для досягнення ефективності необхідно підтримувати реплікацію даних у різних вузлах мережі. Насправді все не так просто. Одна з вимог до сховищ даних — щоб свіжа інформація надходила сюди якнайшвидше. Тобто потенційно будь-яка модифікація оперативної БД може ініціювати додавання даних до сховища даних, тоді буде потрібно обновити і всі записи, для чого синхронізація все-таки потрібна.

Підвищення вимог до безпеки даних: зібрана разом узгоджена інформація про історію розвитку корпорації, її успіх і невдачі, про

взаємовідносини з постачальниками і замовниками, історію і стан ринку, дає можливість проаналізувати минулу й поточну діяльність корпорації та побудувати прогнози на майбутнє. Ця інформація настільки цінна для корпорації, що не можна допустити можливості її витоку (насправді, якщо сховище даних однієї корпорації потрапить у руки аналітиків іншої корпорації, всі аналітичні прогнози першої корпорації відразу видадуть найціннішу інформацію). У системах, заснованих на сховищах даних, недостатній захист даних у стилі мови *SQL*, що забезпечують звичайні комерційні СУБД (цей рівень захисту відповідає класу С2 згідно з класифікацією Міністерства оборони США). Для забезпечення належного рівня захисту доступ до даних контролюється не тільки на рівні таблиць і їх стовпчиків, а й на рівні окремих рядків (відповідає класу В1). Необхідно також вирішувати питання аутентифікації користувачів, захист даних при їх переміщенні в сховищі даних з оперативних баз даних і зовнішніх джерел, захист даних при їх передачі мережею.

Потреба ефективного збереження й опрацювання дуже великих обсягів інформації полягає в наступному. Вже нині відомі приклади сховищ даних, що містять терабайти інформації. За даними консалтингової компанії Meta Group, близько половини корпорацій, що використовують або планують використовувати сховища даних, передбачають виділити дисковий простір до сотень гігабайт. Проблемою таких великих сховищ є те, що накладні витрати на зовнішню пам'ять зростають нелінійно зі збільшенням обсягу сховища.

Дослідження, проведені на основі тестового набору TPC-D, показали, що для баз даних обсягом у 100 гігабайт буде потрібна зовнішня пам'ять обсягом у 4,87 разу більша, ніж потрібно, власне, для корисних даних. При подальшому зростанні баз даних такий коефіцієнт підвищується.

Реалізація сховища і вітрини даних. В основі віртуального сховища даних — репозиторій метаданих, що описують джерела інформації (БД транзакційних систем, зовнішні файли), *SQL*-запити для їх зчитування і процедури обробки та надання інформації. При цьому надмірність даних нульова. Кінцеві користувачі фактично

працюють із транзакційними системами дуже широкого спектра, плюсами (доступ до «живих» даних у реальному часі) і мінусом (інтенсивний мережевий трафік, зниження продуктивності СУБД-систем і реальної погрози їх відмови внаслідок невдалих дій користувачів-аналітиків).

Вітрина даних (Data Mart) за своїм вихідним визначенням — це набір тематично пов'язаних баз даних, що містять інформацію, яка стосується окремих аспектів діяльності корпорації. Фактично, вітрина даних — це полегшений варіант сховища даних, що містить тільки тематично об'єднані дані. Цільова база даних максимально наближена до кінцевого користувача і може містити тематично орієнтовані агрегатні дані. Вітрина даних, закономірно, істотно менша за обсягом від корпоративного сховища даних, і для її реалізації не потрібна особливо потужна обчислювальна техніка.

3.2. Вибір альтернативних планів виконання запитів у СУБД

Усі оптимізуючі перетворення запитів, що розглядалися в [32, 44], залишали внутрішнє представлення запиту непроцедурним. Навіть якщо говорити про процедурність на рівні виконання реляційних операторів у виразах реляційної алгебри, така процедурність є дуже умовною, оскільки, наприклад, порядок виконання операцій реляційного об’єднання зовсім не зобов’язаний збігатися з порядком їх послідовності у виразі. Тим паче, при реальному виконанні запиту ту ж операцію об’єднання можна виконати багатьма способами [40–41, 112].

Процедурним представленням, чи планом виконання запиту, називатимемо таке його представлення, в якому детально відображений порядок виконання операцій доступу до бази даних фізичного рівня. Як правило, у реляційних СУБД, орієнтованих на використання традиційної апаратури без застосування спеціалізованих процесорів чи пристроїв зовнішньої пам’яті, виділяється підсистема керування даними на фізичному рівні. Наприклад, у *System R*, така підсистема називається *RSS* (Relational Storage System) і забезпечує простий інтерфейс, який дозволяє послідовно переглядати

кортежі відносин, що задовольняють заданим умовам на значення полів, з використанням індексів чи простим скануванням сторінок бази даних. Крім того, *RSS* дозволяє створювати відсортовані тимчасові файли й заносити, видаляти і модифіковувати індивідуальні кортежі. Подібні підсистеми явно чи неявно виділяються у всіх аналогічних СУБД.

Очевидно, в таких умовах до виконання запиту необхідно виробити його процедурне представлення. Крім того, оскільки таке представлення взагалі-то вибирається неоднозначно, необхідно вибрати серед альтернативних планів запиту один, найбільш прийнятний згідно з деякими критеріями. Як правило, критерієм вибору плану виконання запиту є мінімізація затрат виконання. Таким чином, при обробці запиту на стадії, яка настає за логічною оптимізацією, розв'язується дві задачі. Перша — на основі одержуваного після вироблення логічною оптимізацією внутрішнього представлення запиту й інформації, яка характеризує керівні структури бази даних (наприклад, індекси), вибрати набір потенційно можливих планів виконання цього запиту. Друга задача — оцінити вартість виконання запиту відповідно до кожного альтернативного плану і вибрати варіант з найменшою вартістю.

Методи розв'язання прикладних задач оптимізації: при традиційному підході до організації оптимізаторів обидві задачі розв'язуються з використанням фіксованих вбудованих в оптимізатор алгоритмів (в останні роки з'явилася низка праць, в яких зазначені недоліки такого «жорсткого» способу організації оптимізаторів і пропонуються альтернативні підходи). Наприклад, оптимізатор може бути розрахованим на те, що обмеження будь-якого відношення відповідно до заданого предиката може виконуватись шляхом послідовного перегляду відношення з використанням кожного з існуючих для нього індексів і прямим послідовним переглядом.

Так, запит:

```
SELECT EMPNAME FROM EMP WHERE DEPT# = 6          (3.1)
                AND SALARY > 15000
```

може виконуватися:

— послідовним скануванням відношення *EMP* з вибором кортежів з

$$DEPT\# = 6 \text{ і } SALARY > 15000; \tag{3.2}$$

— скануванням відношення через індекс I1, визначений для поля *DEPT#*, з умовою доступу до індексу *DEPT#* = 6 і умовою вибірки кортежу *SALARY* > 15000;

— скануванням відношення через індекс I2, визначений для поля *SALARY*, з умовою доступу до індексу *SALARY* > 15000 та умовою вибірки кортежу *DEPT#* = 6.

Аналогічно, фіксованими є й стратегії виконання складніших операцій — реляційних об'єднань відношень, обчислення агрегатних функцій на групах кортежів відношення і т. д. Наприклад, у *System R* для (екві) об'єднання двох відношень використовуються дві основні стратегії: метод вкладених циклів і метод сортувань зі злиттями [77].

Якщо, наприклад, заданий запит:

```
SELECT EMP.EMPNAME, DEPT.DEPTNAME FROM EMP,
DEPT WHERE EMP.DEPT# = DEPT.DEPT#
```
(3.3)

(тут допускається, що в базі даних, яка описує структуру організації, відношення *DEPT* розширюється ще одним полем *DEPTNAME*, яке містить назву відділу), при застосуванні методу вкладених циклів передбачається наявність двох циклів сканування відношень *EMP* і *DEPT* відповідно. У зовнішньому циклі (наприклад, для відношення *EMP*) вибирається черговий кортеж *EMP*, і для кожного такого кортежу виконується повне сканування відношення *DEPT* з вибором таких кортежів, значення полів *DEPT#* яких задовольняє предикат об'єднання. До того ж розглядаються всі можливі способи сканування відношень.

При використанні методу сортувань зі злиттями спочатку здійснюється сортування обох відношень відповідно до значень поля *DEPT#* кожного відношення, у результаті чого утворюються два відсортованих тимчасових файли *EMP*1 і *DEPT*1. Далі проводиться

«паралельне» сканування обох файлів, під час якого виконується «злиття» відношень у сенсі операції об'єднання. Детальніше, вибирається перший кортеж *EMP*1. Нехай значення поля *DEPT*# цього кортежу дорівнює *C*1. Далі вибирається перший кортеж з *DEPT*1 так, щоб значення поля *DEPT*# цього кортежу *D*1 >= *C*1. Якщо знайдено кортеж *DEPT*1 з *D*1 = *C*1, це означає, що вдалося увійти в групу кортежів *DEPT*1, з'єднаних з поточним кортежем *EMP*1. Здійснюється об'єднання кожного кортежу цієї групи з поточним кортежем *EMP*1, після чого вибирається наступний кортеж *EMP*1, і якщо значення поля *DEPT*# у ньому не змінилося, об'єднання з поточною групою кортежів *DEPT*# повторюється. Процес триває доти, доки з *EMP*1 не буде вибрано кортеж зі значенням поля *DEPT*# *C*2 > *C*1. Після цього здійснюється наступне сканування *DEPT*1, але з позиції, наступної за поточною групою, — доки не буде вибрано кортеж зі значенням поля *DEPT*# *D*2 >= *C*2 і т. д. Якщо в *DEPT*1 знайдено кортеж зі значенням поля *DEPT*# *D*1 таким, що *D*1 > *C*1, у тимчасовому файлі *EMP*1 здійснюється пошук кортежу зі значенням поля *DEPT*# *C*2 >= *D*1, і надалі *EMP*1 та *DEPT*1 міняються місцями.

Подано загальну схему алгоритму, а насправді використовуються «витонченіші» варіанти, які враховують можливості буферизації частин відсортованих файлів в оперативній пам'яті.

Описані деякі конкретні алгоритми виконання реляційних операцій підтверджують твердження про існування в традиційних оптимізаторах запитів фіксованих стратегій, на основі яких виробляються плани виконання запитів. Відповідні компоненти оптимізаторів мають досить складну організацію; генерація плану виконання складного запиту — багатоетапний процес, під час якого враховуються властивості створюваних при виконанні запиту за цим планом тимчасових об'єктів бази даних. Наприклад, нехай запит заданий над трьома відношеннями, і в ньому задано два предикати об'єднання:

SELECT R1.C1, R2.C2, R3.C3 FROM R1, R2, R3 WHERE R1.C4 = R2.C5 AND R2.C5 = R3.C6. (3.4)

Тоді, якщо в плані запиту вибирається порядок виконання об'єднань спочатку $R1$ з $R2$, а потім отриманого тимчасового відношення — з $R3$, причому для виконання першого об'єднання вибирається метод сортувань зі злиттям, то тимчасове відношення буде відсортоване по $C5$, і одне сортування не буде потрібним, якщо й друге об'єднання виконуватиметься за тим самим методом.

Зі сказаного вище випливає, що компонент оптимізатора, який породжує множину альтернативних планів виконання запиту, базується на стратегіях декомпозиції запиту на елементарні складові та стратегіях виконання елементарних складових. Перша група стратегій забезпечує простір пошуку оптимального плану виконання запиту, друга — спрямована на те, щоб у цьому просторі справді існували ефективні плани виконання запиту. Очевидно, способом забезпечення ефективного виконання складного запиту є наявність ефективних стратегій виконання елементарних складових. Це винятково важливе питання, якому надається величезна увага в теорії реляційних баз даних. Особливо багато праць присвячено виробленню ефективних алгоритмів реалізації об'єднань відношень і подібних операцій. Такі питання детальніше розглядаються в наступному розділі, тут акцентуємо увагу лише на проблемі обґрунтованого вибору плану виконання запиту з множини альтернативних планів.

Справді, якщо оптимізатору вдалося згенерувати множину планів виконання запиту на основі «розумних» стратегій декомпозиції й ефективних стратегій виконання елементарних операцій, цього недостатньо — необхідно, окрім того, вибрати один план, відповідно до якого здійснюватиметься реальне виконання запиту. Якщо вибір буде неправильним, запит виконуватиметься неефективно. Насамперед треба визначити, як слід розуміти ефективність виконання запиту. Загалом це поняття неоднозначне й залежить від специфіки операційного середовища СУБД. В одних умовах можна вважати, що ефективність виконання запиту визначається часом його виконання, тобто реактивністю системи стосовно оброблюваних нею запитів. В інших умовах визначальною є загальна про-

пускна здатність системи стосовно суміші паралельно виконуваних запитів. Тоді мірою ефективності запиту можна вважати кількість системних ресурсів, необхідних для його виконання: чим менше ресурсів потрібно, тим запит ефективніший. Якщо користувачі СУБД працюють в умовах бюджетної системи, розумною мірою ефективності може бути бюджетна вартість виконання запиту і т. д. Зауважимо, що в будь-якому випадку оцінка ефективності виконання запиту базується на ресурсних характеристиках відповідного плану; при використанні різних мір вони по-різному беруть участь в оцінці.

Використовуючи прийняту термінологію, замість оцінки ефективності виконання запиту говоритимемо про вартість плану виконання запиту. Основними ресурсами, які витрачаються під час виконання запиту, є ресурси: процесора, пристроїв зовнішньої пам'яті та мережеві. Останній вид ресурсів, природно, задіюється тільки в розподілених системах, побудованих на основі апаратури обчислювальних мереж. Специфіка оптимізації виконання запитів у таких системах розглядається в окремому розділі, тут беремо до уваги лише централізовані СУБД, для виконання запитів у яких не потрібні мережеві ресурси.

Отже, у нашому випадку вартість виконання запиту визначається на основі необхідних ресурсів процесора і пристроїв зовнішньої пам'яті. Як уже зазначалося, в традиційних реляційних СУБД, які не використовують спеціалізованої апаратури, як правило, явно виділяється підсистема керування доступом до даних на зовнішній пам'яті (*RSS* у *System R*). Дані у зовнішній пам'яті традиційно зберігаються в блоковому вигляді так, що база даних займає певний набір блоків одного чи декількох дискових томів. При цьому не суттєво, чи працює СУБД із блоками зовнішньої пам'яті безпосередньо через відповідні драйвери, чи користується послугами пристроїв тієї чи іншої файлової системи, яка забезпечує розподіл зовнішньої пам'яті і доступ до блоків файла за їх логічними номерами. У будь-якому випадку передбачається, що засоби підтримки доступу до блоків зовнішньої пам'яті породжують лише надмір малі накладні витрати.

Як правило, підсистеми керування доступом до даних на зовнішній пам'яті здійснюють буферизацію блоків бази даних в оперативній пам'яті. Кожен блок бази даних, прочитаний в оперативну пам'ять для виконання запиту, зберігається в одному з буферів СУБД доти, доки не буде витиснутий з нього іншим блоком бази даних. Така особливість СУБД, звичайно, дуже істотна для підвищення загальної ефективності системи, але практично вона не враховується при оцінках вартостей планів виконання запиту. У будь-якому випадку той компонент вартості виконання запиту, який пов'язаний з ресурсами пристроїв зовнішньої пам'яті, монотонно залежить від кількості блоків зовнішньої пам'яті, доступ до яких потрібний буде під час виконання запиту.

Окрім того, при будь-якому способі відображення відношень бази даних на блоки зовнішньої пам'яті — незалежно від того, чи розташовуються в одному блоці зовнішньої пам'яті кортежі тільки одного відношення, чи допускається розміщення в один блок кортежів декількох відношень, кількість блоків зовнішньої пам'яті, доступ до яких потрібен при виконанні запиту, монотонно залежить від кількості кортежів, задіяних запитом. Пояснимо це на прикладі.

Нехай задано запит:

SELECT EMP.EMPNAME, (3.5)

DEPT.DEPTNAME WHERE EMP.SALARY = 20000

AND EMP.DEPT# = DEPT.DEPT#.

Розглянемо набір планів запиту, у яких спочатку виконується обмеження відношення *EMP* відповідно до предикату *EMP.SALARY* = 20000 з формуванням тимчасового відношення *EMP*1, а потім здійснюється об'єднання *EMP*1 з *DEPT* відповідно до предикату об'єднання *EMP1.DEPT*# = *DEPT.DEPT*#. Це саме і є множина планів запиту, оскільки не деталізовувались ні спосіб виконання обмеження, ні спосіб виконання об'єднання.

Обмеження *EMP* потенційно можна виконати одним із двох способів. Перший спосіб — послідовне сканування відношення *EMP* з вибором кортежів, які задовольняють предикат обмеження.

При цьому буде здійснено доступ до всіх кортежів *EMP*, і отже, до всіх блоків бази даних, які містять хоча б один кортеж з *EMP*. Другий спосіб — сканування *EMP* з використанням індексу на поле *EMP.SALARY* з умовою *EMP.SALARY* = 20000 (тут передбачається, що такий індекс існує і, крім того, забезпечує послідовний доступ до кортежів у порядку зростання чи спадання значення ключа. Таку умову задовольняє, наприклад, загальноприйнята організація індексів на основі технології *B*-дерев). У такому випадку нам буде потрібно число звертань до блоків зовнішньої пам'яті, яке не перевищує кількість кортежів відношення *EMP* зі значенням поля *SALARY*, рівним 20000.

Зауважимо, що в будь-якому випадку тимчасове відношення *EMP*1 міститиме саме стільки кортежів, скільки кортежів відношення *EMP* задовольняє предикат обмеження, і отже, саме стільки блоків, скільки буде потрібно для розміщення цих кортежів. Таким чином, при оцінках вартості виконання об'єднання тим чи іншим способом буде отримано достатньо точну інформацію про розміри відношення — першого операнда. На такому простому прикладі показано важливість показника предикату обмеження, що характеризує частку кортежів відношення, які задовольняють цей предикат, названого ступенем селективності предикату.

Ступінь селективності простих предикатів виду:

$$\text{R.C op const}, \tag{3.6}$$

де *R.C* — ім'я поля відношення бази даних, *op* — операція порівняння (=, !=, >, <, >=, <=), а const константа є основою для оцінок вартості планів запиту.

Чим точніше вдається визначити ступінь селективності, тим точнішими будуть загальні оцінки та правильніше вибиратиметься оптимальний план запиту. Звичайно, насправді ступінь селективності предикату *R.C op* const залежить від виду операції порівняння, значення константи та розподілу значень поля *C* відношення *R* у базі даних, якій адресується запит. Якщо перші два параметри селективності відомі під час проведення оцінок, правильні розподіли значень полів відношень, як правило, невідомі. Проте існує ряд

підходів до наближених визначень розподілів на основі статистичної інформації. Далі розглянемо найцікавіші з них.

Якщо відомо ступінь селективності предикату обмеження відношення, відомою є й потужність результуючого відношення (потужності збережених відношень відомі при обробці запиту). В формулах враховується необхідна для виконання запиту кількість обмінів із зовнішньою пам'яттю. У наведеному вище прикладі, коли відношення *EMP* сканується з використанням індексу на поле *SALARY*, передбачувана кількість кортежів, які задовольняють умову *SALARY* = 20000, звичайно, є верхньою оцінкою необхідного числа блоків, але ця оцінка може бути дуже завищеною. У цьому випадку все залежить від розподілу кортежів по блоках зовнішньої пам'яті. Зазвичай можна одержати точнішу оцінку кількості блоків.

Отже, для складних запитів, що містять, наприклад, об'єднання більше двох відношень, потрібно оцінювати розміри виникаючих у процесі виконання запиту проміжних тимчасових відношень. Як відомо, об'єднання двох відношень *R*1 і *R*2 відповідно до предикату об'єднання:

$$\text{R1.C1 op R2.C2}, \tag{3.7}$$

де *op* — операція порівняння, концептуально інтерпретується як обмеження відношення *R*, отриманого в результаті декартового добутку відношення *R*1 і *R*2, предикатом R1.C1 op R2.C2. Щоб оцінити потужність результуючого відношення загалом, необхідно знати ступінь селективності предикату об'єднання стосовно прямого добутку відношень-операндів. Загалом ступінь селективності такого предикату неможливо визначити на основі простої статистичної інформації, тож зазвичай застосовуються досить «грубі» евристичні оцінки, хоч і пропонуються підходи, що забезпечують достатню точність.

3.2.1. Плани виконання запитів

Перш ніж розпочати розгляд сучасних підходів до розв'язання описаних проблем, наведемо короткий огляд підходу, застосовуваного для оцінок вартості планів виконання запиту в *System R*, від-

повідно до якого ставляться дві мети. По-перше, такий підхід є досить поширеним і, за твердженням, наприклад [122], виправданим на практиці. По-друге, у викладі постараємося виокремити основні його недоліки, подоланню яких присвячено чимало праць [73–74, 84].

Підхід *System R* базується на двох основних припущеннях про розподіли значень атрибутів відношень. По-перше, передбачається, що значення полів усіх відношень бази даних розподілені рівномірно. По-друге, вважається, що значення будь-яких двох полів розподілені незалежно. Перше припущення дає можливість без труднощів оцінювати селективність простих предикатів на основі дуже обмеженої статистичної інформації про базу даних. На другому припущенні ґрунтуються оцінки кількості блоків, у яких розташовується відома кількість кортежів. Зауважимо, що означені два припущення є гострим предметом критики *System R*, оскільки вони зроблені винятково для спрощення оптимізатора й не можуть бути теоретично обґрунтовані. Можна навести безліч прикладів баз даних, для яких такі припущення не є виправданими. При цьому, закономірно, оптимізатор *System R* буде свідомо неправильно оцінювати плани виконання запитів.

У каталогах бази даних для кожного відношення *R* підтримується така інформація: кількість кортежів у даному відношенні (*T*); кількість блоків зовнішньої пам'яті, у яких розташовуються кортежі відношення (*N*); для кожного поля *C* відношення зберігається кількість різних значень цього поля (*CD*), мінімально збережене значення цього поля (*CMin*) і максимальне значення (*CMax*). Для точнішої оцінки доступу до відношення через індекси для кожного індексу зберігається кількість його рівнів і кількість аркушевих сторінок. Оскільки у *B*-деревах допускається наявність недозаповнених сторінок, інформація про кількість рівнів індексу і кількість аркушевих сторінок не виводиться з інформації про загальну кількість кортежів у відношенні та число різних значень поля, на якому визначений індекс. На нашому рівні викладу статистична інформація про індекси несуттєва.

Зазначимо, що в ранніх версіях *System R* (наприклад, [77]) статистики про значення полів підтримувалися тільки для полів, на які визначені індекси, проте потім були запроваджені й для неіндексованих полів, для появи можливості оцінювати потужність тимчасових відношень, отриманих обмеженням базового відношення відповідно до простого предикату на неіндексованому полі [73].

За наявності такої інформації в умовах базового припущення про рівномірність розподілу значень кожного поля відношення ступінь селективності простих предикатів обчислюється дуже просто (і надзвичайно точно, якщо розподіл насправді є рівномірним). Будемо позначати ступінь селективності предикату P як $SEL(P)$. Тоді $SEL(R.C = \text{const}) = CD / (CMax - CMin)$ (очевидно, що при рівномірному розподілі ступінь селективності такого предикату не залежить від значення константи). $SEL(R.C > \text{const}) = (CMax - \text{const} / (CMax - CMin)$. Аналогічно, $SEL(R.C < \text{const}) = (\text{const} - CMin) / (CMax - CMin)$ (знову ж таки очевидно, що при рівномірному розподілі значень поля частка кортежів, які задовольняють умову потрапляння значення заданого поля в зазначений інтервал значень, лінійно зростає зі збільшенням розміру інтервалу). $SEL(R.C <= \text{const})$ — рівна $SEL(R.C < \text{const})$, а $SEL(R.C >= \text{const})$ — рівна $SEL(R.C > \text{const})$.

Під час оцінювання кількості блоків, у яких можуть розташовуватися кортежі, які задовольняють предикат $R.C$ *op* const, розрізняються два випадки: відношення кластеризовано по полю C або не кластеризовано по цьому полю. Зазначимо, що оцінювати цю кількість блоків має сенс тільки тоді, коли сканування відношення для його обмеження відповідно до предикату здійснюється з використанням індексу на поле C. Для варіанта сканування без використання індексу потрібно завжди звернутися до N блоків зовнішньої пам'яті.

Нагадаємо, що кластеризацією кортежів відношень відповідно до значень якогось поля називається таке фізичне розміщення кортежів цього відношення в блоках зовнішньої пам'яті, коли кортежі розташовуються в блоки з урахуванням порядку значень виділеного поля. Якщо відношення кластеризовано, в кожному блоці, що

містить кортежі цього відношення, є група кортежів, значення виділеного поля яких створять інтервал, і жоден кортеж зі значенням виділеного поля, який потрапляє всередину цього інтервалу, не міститься в іншому блоці.

У цьому випадку селективність предикату стосовно кортежів поширюється і на блоки, які займають ці кортежи. Наприклад, якщо в блоці зовнішньої пам'яті бази даних розміщується *K* кортежів відношення *R* (параметр *K* у *System R* визначається при створенні відношення і не змінюється протягом часу його існування), кількість блоків кластеризованого відношення, які містять кортежі, що задовольняють предикат *R.C op* const, визначається за формулою *T***SEL*(*R.C op* const) / *K* (перший член формули задає кількість кортежів, що задовольняють предикат).

При оцінках кількості блоків, у яких можуть розташовуватися кортежі відношення, що задовольняють предикат *R.C op* const, у випадку, коли відношення не кластеризовано по полю *C*, при розробці початкового варіанта оптимізатора виходили з того, що кортежі можуть бути довільним чином розкидані блоками бази даних, які містять кортежі даного відношення. Отже, оцінка здійснювалася за формулою *T* * *SEL*(*R.C op* const). Однак згодом було зауважено, що наведена формула дає завищений результат, якщо кількість кортежів відношень, які задовольняють предикат, достатньо велика (селективність предиката є низькою) [73].

Записи аркушевих блоків індексу відношення *R* на поле *C* у *System R* мають таку структуру: значення ключа, список унікальних ідентифікаторів кортежів відношення *R*, у яких поле *C* має те саме значення, що і ключ. Унікальний ідентифікатор кортежу (*tid*) забезпечує прямий доступ до кортежу і містить, зокрема, номер блока, в якому розміщується кортеж.

Якщо при організації *tid*-списку в записі індексу підтримувати в кожному списку впорядкованість *tid*-індетифікаторів відповідно до номерів блоків, що містять відповідні кортежі, при послідовному перегляді кортежів з фіксованим значенням ключа з'являється деякий елемент упорядкованості. В удосконаленому варіанті оптимізатора *System R* для оцінки числа блоків, звертання до яких по-

винні відбутися при такому перегляді, використовується формула $N = (1 - (1 - 1/N)CD)$. Ця формула уперше виведена та обґрунтована в [125]. Зрозуміло, що на її основі можна оцінити й кількість блоків, звертання до яких будуть потрібні при скануванні відношення через індекс для обмеження відношення відповідно до предикату з наперед відомою селективністю.

Зазначимо, що й при такому підході до оцінки кількості блоків зі строгим поділом випадків кластеризованості відношення *R* по полю *C* і відсутності такої кластеризованості виявляється деяка недосконаліть (обмеженість) *System R*. У реальних умовах кластеризованість відношення по одному полю не обов'язково означає повну відсутність кластеризованості того ж відношення по інших полях (припущення *System R* про незалежність розподілів значень різних полів відношення). Далі буде розглянуто один із запропонованих останнім часом підходів до більш адекватного обліку реально існуючих кореляцій атрибутів відношень.

Припущення про рівномірність розподілів значень атрибутів відношень дозволяють достатньо просто оцінювати й потужності відношень, які є результатами двомісних реляційних та теоретико-множинних операцій над відношеннями. Розглянемо, наприклад, як можна оцінити ступінь селективності двох предикатів об'єднання — $R1.C1 = R2.C2$ і $R1.C1 > R2.C2$.

Розпочнемо з предикату еквіоб'єднання $R1.C1 = R2.C2$. Нехай *Ti* і *CDi* — число кортежів у відношенні *Ri* і число різних значень поля *Ci* відповідно. Тоді у відношенні *Ri* існує *Ti*/*CDi* груп кортежів з однаковими значеннями поля *Ci*. Очевидно, що при виконанні операції об'єднання кортежі кожної групи кортежів відношення *R*1 можуть бути об'єднані з кортежами не більш ніж однієї групи відношення *R*2 (і навпаки). Якщо кортежі двох груп об'єднуються, в результуючому відношенні утвориться (*T*1/*CD*1)* (*T*2/*CD*2) кортежів (оцінка числа кортежів у кожній групі у вигляді *Ti*/*CDi* є правомірною відповідно до припущення про рівномірність розподілу). Очевидно, що об'єднатися можуть не більше, ніж *Min*(*CD*1, *CD*2) груп. Отож верхньою оцінкою потужності результуючого відношення може бути *Min*(*CD*1, *CD*2) * (*T*1/*CD*1) * (*T*2/*CD*2). Відповід-

но, ступінь селективності предикату еквіоб'єднання можна оцінити, як *Min*(*CD*1, *CD*2)/(*CD*1 * *CD*2). Зазначимо, що така оцінка є досить точною, якщо відношення «добре» об'єднуються, як, наприклад, відношення *EMP* і *DEPT* з нашого прикладу по полю *DEPT*#.

Формула для оцінки ступеня селективності предикату об'єднання *R*1.*C*1 > *R*2.*C*2 виводиться трохи складніше. Позначимо через *CiMax* і *CiMin*, відповідно, максимальне і мінімальне значення поля *Ci*. Нехай кортежі з *j*-ї групи кортежів відношення *R*1 мають значення поля *C*1, рівне *cj*. Тоді кількість кортежів відношення *R*2, що задовольняють предикату об'єднання щодо *j*-ї групи відношення R1, можна оцінити як:

$$((C2Max - cj) / (C2Max - C2Min)) * T2. \quad (3.8)$$

Кількість кортежів у результуючому відношенні, які виникають при об'єднанні *j*-ї групи кортежів відношення *R*1 з кортежами відношення *R*2, оцінюється як:

$$(T1/CD1) * ((C2Max - cj) / (C2Max - C2Min)) * T2. \quad (3.9)$$

Для одержання оцінки загальної кількості кортежів у результуючому відношенні необхідно просумувати отриману формулу по *j* для всіх груп відношення *R*1. При цьому можна скористатися тим, що з огляду на припущення про рівномірність розподілу значень поля *C1*, суму *cj* можна оцінити як:

$$CD1 * AVG(cj) = CD1 * (C1Max\ C1Min) / CD1 = C1Max - C1Min. \quad (3.10)$$

Результуюча формула для оцінки потужності результуючого відношення має вигляд:

$$T1 * T2 * (C2Max - C1Max + C2Min) / (C2Max - C2Min). \quad (3.11)$$

Відповідно, селективність предикату оцінюється за формулою:

$$(C2Max\ C1Max + C2Min) / (C2Max - C2Min). \quad (3.12)$$

Аналогічно створюються формули для оцінок потужностей результатів інших двомісних операцій.

Оцінки селективності предикатів використовуються також у формулах оптимізатора для оцінки витрат ресурсів процесора. Такі оцінки базуються на припущенні (підтверджується на практиці) про те, що основна частина процесорної обробки здійснюється в

RSS. Отож процесорні витрати вимірюються просто кількістю звертань до *RSS*, що, своєю чергою, визначається кількістю кортежів, одержуваних при скануванні збереженого чи тимчасового відношення, тобто селективністю логічного виразу, який складається з простих предикатів (нагадаємо, що під час сканування відношення при кожній операції взяття наступного кортежу *RSS* одержує параметр — логічний вираз, який складається з простих предикатів і є умовою, яку повинен задовольняти наступний кортеж. Насправді при використанні *RSS* у *System R* ця логічна умова завжди є тією самою в межах кожного сканування і становить предикат обмеження відповідного відношення). Керуючись припущеннями про рівномірність і незалежність значень різних полів відношення, селективність логічного виразу, складеного з простих предикатів, легко оцінити за відомих ступенів селективності простих предикатів.

Насправді в *System R* оцінюється не селективність предикатів у чистому вигляді. Замість цього оцінки оптимізатора ґрунтуються на фіксованому наборі елементарних формул, орієнтованих на статистичну інформацію про базу даних. Як було зазначено, *System R* підтримує досить малий набір статистик по кожному відношенні. Кожна формула відповідає деякій елементарній дії системи, причому виконання будь-якого запиту полягає в комбінації певних елементарних дій. Прикладами елементарних дій можуть бути виконання обмеження відношення шляхом його сканування через кластеризований індекс, сортування відношення, об'єднання двох раніше відсортованих відношень і т. д. Загальна оцінка плану виконання запиту здійснюється в ітераційному процесі, починаючи з оцінок елементарних операцій над збереженими відношеннями.

Наприклад, нехай для виконання запиту:

```
SELECT EMP.EMPNAME, DEPT.DEPTNAME FROM EMP,
    DEPT WHERE EMP.SALARY = 20000                    (3.13)
AND DEPT.DEPT# > 4 AND EMP.DEPT# = DEPT.DEPT#
```

оцінюється наступний план виконання: виконати обмеження відношення *EMP* з використанням некластеризованого індексу на

поле *EMP.SALARY* і породженням тимчасового відношення *EMP*1 (елементарна операція ОП1); виконати обмеження відношення *DEPT* з використанням кластеризованого індексу на поле *DEPT.DEPT#* і породженням тимчасового відношення *DEPT*1 (елементарна операція ОП2); відсортувати відношення *EMP*1 відповідно до значень поля *EMP.DEPT#* з породженням відсортованого файла *EMP*2 (елементарна операція ОП3); відсортувати відношення *DEPT*1 відповідно до значень поля *DEPT.DEPT#* з породженням відсортованого файла *DEPT*2 (елементарна операція ОП4); об'єднати файли *EMP*2 і *DEPT*2 за полем *DEPT#* (елементарна операція ОП5). Вартість елементарних операцій ОП1 і ОП2 оцінюється за формулами і статистичною інформацією про відношення *EMP* і *DEPT*. Вартість елементарних операцій ОП3, ОП4 і ОП5 оцінюється на основі формул і оцінок операцій ОП1 і ОП2 (потужності відношень *EMP*1 і *DEPT*1). Оцінка загальної вартості плану виконання запиту складається з оцінок для ОП1, ОП2, ОП3, ОП4 і ОП5.

Останнє, що варто зазначити з підходу *System R*, стосується підтримки статистичної інформації про відношення бази даних. Зрозуміло, що підтримувати в динаміці змін бази даних точну інформацію, наприклад, про кількість різних значень полів відношення, було б надто складно (особливо для полів, на які не визначені індекси). Отож така інформація не коректується при кожній зміні бази даних. У *System R* існує спеціальна утиліта збору статистики, яка, власне, і робить відповідні корекції або періодично, або за явною вказівкою оператора.

Перейдімо до розгляду пропонованих підходів, які дають змогу точніше оцінювати вартості виконання планів запиту. Підходи можна розподілити на два класи. Відповідно до підходів першого класу оптимізатор зберігає жорстку структуру, аналогічну структурі оптимізатора *System R*, але під час вироблення оцінок використовуються не припущення *System R* про рівномірність розподілів значень полів відношень і незалежності розподілів значень різних полів відношень, а найточніша статистична інформація, яка характеризує реальні розподіли значень. Підходи другого класу новітніші

й базуються на тому, що для вироблення планів виконання запитів і їх оцінок оптимізатор повинен забезпечуватися деякою характерною для конкретної області використання інформацією.

Природно, якщо відмовитися від припущення про рівномірність розподілу значень поля відношення, необхідно якимсь чином вміти встановити реальний розподіл значень. Як зазначалося, існує два базові підходи до оцінок розподілу значень поля відношення: параметричний і базований на методі сигнатур. Підхід *System R* є тривіальним частковим випадком методу параметричної оцінки розподілу — будь-який розподіл оцінюється як рівномірний.

Метод оцінки розподілу на основі сигнатур у загальному можна описати в такий спосіб. Область значень поля розбивається на декілька інтервалів. Для кожного інтервалу певним чином встановлюється число значень поля, що потрапляють у цей інтервал. Усередині інтервалу значення вважаються розподіленими за деяким фіксованим законом (як правило, приймається рівномірне наближення). Розглянемо тепер детальніше два альтернативних підходи, пов'язані з сигнатурним описом розподілів.

Традиційний підхід, який описується, полягає в тому, що область значень поля розподіляється на *N* інтервалів однакової довжини, і для кожного інтервалу підраховується число значень полів з кортежів цього відношення, які потрапляють в інтервал. Наприклад, припустимо, що наше відношення реєстрації співробітників підприємства *EMP* розширене ще одним полем *AGE* — вік співробітника. Нехай усього в організації працює 60 працівників віком від 10 до 60 років. Тоді гістограма, яка відображає розподіл значень поля *AGE,* може мати вигляд, показаний на рис. 3.1. Гістограма побудована на основі розбивки діапазону значень поля *AGE* на десять інтервалів.

Розглянемо, як можна оцінювати селективність простих предикатів, які задаються на поле *AGE*, з використанням такої гістограми. Нехай в інтервал значень *AGE Si* потрапляє *Ki* значень. Тоді *SEL* (*EMP.AGE* = const), якщо значення константи міститься в інтервалі значень *Si*, можна оцінити в такий спосіб: 0 <= *SEL* (*EMP.AGE*) <= *Ki*/*T* (*T* — загальне число кортежів у відношенні *EMP*). Звідси серед-

ня оцінка ступеня селективності предикату — *Ki* / (2**T*). Наприклад, *SEL* (*AGE* = 29) оцінюється в 40/200 = 0.2, а *SEL* (*AGE* = 16) оцінюється в 5/200 = 0.025. Це, звичайно, суттєво точніші оцінки порівняно з тими, які можна одержати, використовуючи припущення про рівномірність розподілів. Проте не так все просто з оцінками селективності простих предикатів з нерівностями.

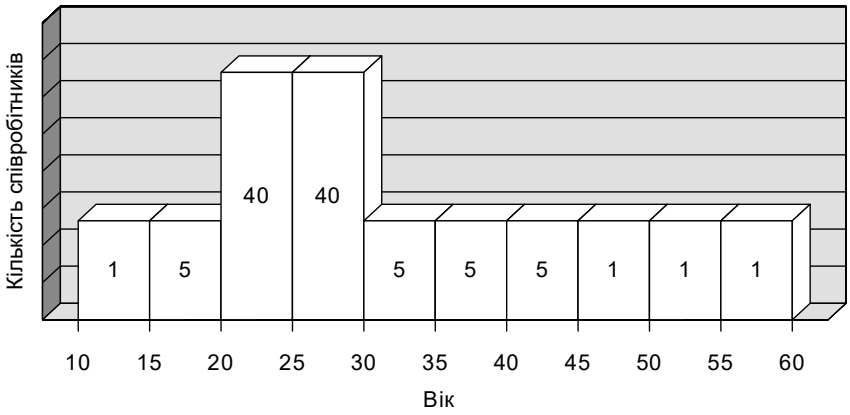


Рис. 3.1. Проста гістограма розподілу віку працівників

Наприклад, нехай потрібно оцінити ступінь селективності предикату *EMP.AGE* < const. Якщо значення константи потрапляє в інтервал *Si, a SUMi* — сумарна кількість значень *AGE*, що попадають в інтервали *S*1, *S*2, ..., *Si*, то *SUMi* - 1/*T* <= *SEL*(*AGE* < const) <= *SUMi*/*T*. Тоді середня оцінка ступеня селективності (*SUMi* - 1 + *SUMi*) / (2**T*), і помилка оцінки можуть досягати половини ваги від області, в яку потрапляє значення константи предикату. Найнеприємнішим є те, що помилка оцінки залежить від значення константи, і тим більша, чим більше значень поля міститься у відповідному інтервалі гістограми. Наприклад, *SEL* (*AGE* < 29) оцінюється як 46/100 <= *SEL* (*AGE* < 29) <= 86/100, звідки оцінка ступеня селективності (46 + 86) / 200 = 0.66; причому помилка оцінки може сягати 0.2. Водночас *SEL* (*AGE* < 49) оцінюється суттєво точніше.

Для усунення такого недоліку оцінок на основі гістограмного опису розподілу в [125] запропоновано інший підхід до опису

розподілів значень поля відношення. Основна ідея підходу полягає в тому, що, на відміну від чистого методу гістограм, множина значень поля розподіляється на інтервали, розмір яких вибирається таким чином, щоб у кожен інтервал (окрім останнього) потрапляла однакова кількість значень поля. Як і в чистому методі гістограм, кількість інтервалів вибирається з урахуванням обмежень по пам'яті, і чим воно більше, тим точнішими є одержувані оцінки. При виборі розбивки області значень на десять інтервалів одержувана псевдогістограмна картина розподілів значень поля *AGE* показана на рис. 3.2.

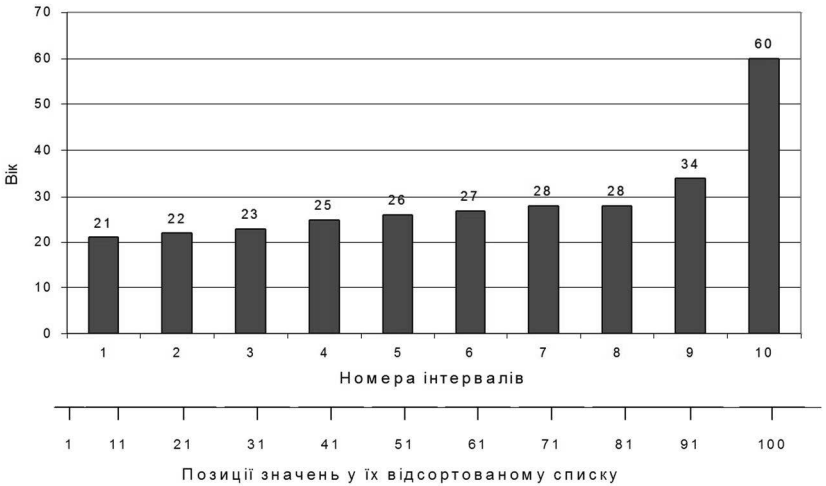


Рис. 3.2. Псевдогістограма розподілу віку працівників

Область значень поля *AGE* відношення *EMP* розподілена на десять інтервалів таким чином, що в кожен з них попадає рівно по десять значень поля *AGE*. При цьому, очевидно, інтервали мають різні величини. Граничні значення інтервалів показані над вертикальними лініями. Особливість псевдогістограми в тому, що в ній можливі, зокрема, інтервали, права і ліва границі яких збігаються. У нашому випадку таким інтервалом є (28, 28), що утворився через наявність у відношенні *EMP* великої (понад десять) кількості кортежів зі значенням *AGE* = 28.

3.3. Стратегії виконання процедури об'єднань блоків даних у розподілених базах

З розвитком апаратного та програмного забезпечення мереж ЕОМ (локальних і територіальних) проблема інтеграції роз'єднаних баз даних стає щораз актуальнішою. Існує багато підходів до вирішення цієї проблеми, починаючи від забезпечення віддаленого мережевого доступу до локальних баз даних і закінчуючи побудовою складних розподілених систем керування базами даних, які забезпечують для користувача прозорість місця розташування даних, що його цікавлять. В останньому випадку користувачі взаємодіють із глобальною розподіленою базою даних на основі її глобальної схеми.

Унаслідок обмеженості обсягу роботи неможливо навести повний аналіз проблем оптимізації запитів і структур даних та стратегій виконання реляційних операцій у всьому спектрі розподілених систем баз даних. Проте найважливіші особливості оптимізації в деякому широкому класі розподілених систем баз даних спробуємо розглянути. До цього класу віднесемо розподілені системи керування базами даних, що побудовані на однорідних локальних системах керування базами даних. Прикладом таких систем можуть бути СУБД *Oracle* або *System R*.

Додатковою вимогою цього класу є вимога локальної автономності вузлів мережі, на яких виконуються локальні версії СУБД (частково модифіковані версії). Це означає відсутність потреби в централізованому адмініструванні глобальною базою даних, відсутність централізованого каталогу бази даних і т. д. Допускається локальне створення й видалення нових відношень та індексів у локальній базі даних. При цьому знищення індексу не повинно призводити до неможливості повторного виконання вже відкомпільованих глобальних запитів. Вимога локальної автономності також істотно впливає і на допустимі способи оптимізації глобальних запитів (тут і далі глобальний запит треба розуміти як запит, для виконання якого потрібен мережевий доступ до віддалених локальних баз даних).

Зазначимо, що логічні рівні оптимізації запитів, включно зі семантичною оптимізацією, фактично не пов'язані з розподіленою чи локальною природою БД (не беручи до уваги певної специфіки обробки запитів через представлення). Розподілений характер баз даних торкається головно «фізичних» рівнів оптимізації, пов'язаних з вибором і оцінкою планів виконання запиту.

Інша проблема розподілених систем БД стосується алгоритмів виконання бінарних реляційних операцій над відношеннями, що зберігаються в різних вузлах розподіленої системи. Ця проблема існує і при традиційному збереженні кожного відношення повністю в одній локальній БД, але стає ще складнішою і такою, яка допускає більшу кількість розв'язань, у випадку так званих розділених (*partitioned*) і розкооперованих реплікованих (*replicated*) баз даних. Такі нетрадиційні організації розподілених БД дуже активно обговорюються в літературі, проте досі невідомою є яка-небудь завершена система, побудована на таких принципах.

3.3.1. Стратегії виконання об'єднань у розподілених базах даних

Як зазначалось раніше, у розподілених базах даних із традиційною (відношення повністю зберігаються в одному і тільки одному вузлі) або зі складнішою (розкооперованою чи розділеною) організацією потенційно можлива величезна кількість стратегій виконання об'єднань. Чимало публікацій, присвячених цій темі, розкривають відсутність єдиного рішення і незавершеність досліджень. Обмежимося розглядом лише окремих, на нашу думку найцікавіших, пропозицій розв'язання висвітленої проблеми.

Розглянемо об'єднання двох віддалених відношень S і T, розташованих у вузлах 1 і 2 відповідно, під керуванням предиката еквіоб'єднання:

$$S.C1 = T.C2. \tag{3.14}$$

Результат об'єднання повинен бути отриманий у вузлі 1. Передбачається також наступне: жодне з відношень не кластеризовано по полях об'єднання; відсутні індекси на полях, відмінні від

полів об’єднання; у запиті не потрібна проекція (тобто кортежі результату включають усі поля *S* і *T*); у запиті відсутні предикати обмеження відношень *S* і *T*; відношення *S* і *T* зберігаються в окремих блоках зовнішньої пам'яті (будь-який блок бази даних, що містить кортежі *S* чи *T*, не містить кортежів інших відношень).

Перший пропонований алгоритм полягає в тому, що відношення *T* повністю передається у вузол 1 і тут для нього створюється тимчасовий індекс на поле *C2*. Після цього у вузлі 1 вибирається найкращий локальний план виконання об’єднання відношення *S* і тимчасової копії відношення *T*.

Другий алгоритм [41] ґрунтується на використанні напівоб’єднань. Обидва відношення сортуються відповідно до значень полів *C1* і *C2* з утворенням тимчасових відсортованих файлів *S'* і *T'*. Здійснюється проекція *S'* на *C1* (знищуються дублікати) і результат передається у вузол 2. У цьому вузлі виконується напівоб’єднання *T'* з отриманим унарним відсортованим відношенням (тобто вибираються ті кортежі *T'*, значення поля *C2* яких збігається з якимнебудь значенням *C1* отриманого списку). Отримане проміжне, як і відсортоване, відношення *T''* передається у вузол 1, де виконується об’єднання *S'* і *T''* методом об’єднання (обидва відношення уже відсортовані) і виробляється остаточний результат.

За наявності індексів на полях відношень *S.C1* і *T.C2* може бути використаний або той самий алгоритм, або його модифікація, в якій відсутня операція сортування (використовуються сканування відношень через індекси).

Третій альтернативний алгоритм об'єднання двох відношень базується на використанні так званих фільтрів Блюма [78]. Відповідно до цього, якщо індекси на полях *S.C1* і *T.C2* не визначені, спершу у вузлі 1 для відношення *S* по полю *C1* будується фільтр Блюма. Фільтр є бітовим масивом, ініціалізований нулями. Для формування фільтра проводиться сканування відношення *S*, і до значення поля *C1* кожного кортежу застосовується хеш-функція, яка ставить у відповідність цьому значенню позицію біта в масиві. Біт встановлюється в одиницю.

Сформований фільтр передається у вузол 2. У ньому здійснюються сканування відношення *T* і до значень поля *C2* застосовується та сама хеш-функція, яка задає позицію відповідного біта у фільтрі. Якщо значення біта дорівнює 1, кортеж відношення *S* посилається у вузол 1 у потоці кортежів *T'*. У цьому вузлі виконується об'єднання *S* і *T'* та формується результат.

Наявність індексів *S.C1* і *T.C2* дає змогу модифікувати алгоритм. Зокрема, в такому випадку фільтр Блюма для відношення *S* можна побудувати, скануючи лише записи індексу без звертання до кортежів відношення.

Проблеми об'єднань у розподілених базах даних із нетрадиційною організацією: основна характеристика нетрадиційної організації розподіленої бази даних — одне відношення не обов'язково повністю зберігається в одному і тільки одному вузлі (в деякій локальній базі даних). Тобто в реплікованих (*replicated*) базах даних одне відношення може розміщуватись (повністю) у декількох вузлах мережі. Це може мати важливе значення в умовах територіальної мережі, коли мережеві витрати на взаємодію різних вузлів суттєво відрізняються. Запит може бути виконаний набагато ефективніше, якщо копія необхідного відношення зберігається у вузлі, доступ до якого не надто затратний. Крім можливого підвищення ефективності під час виконання запитів, репліковані бази даних загалом забезпечують доступ до розподіленої бази даних при втратах цілісності (зв'язності) мережі, якщо копії необхідних даних доступні в межах зв'язного розділу, що утворився в мережі. Звичайно, при цьому виникає цілий набір проблем, пов'язаних з підтримкою узгодженості копій відношення, особливо при порушеннях зв'язності. Питання не належать до тематики оптимізації, але зазначимо, що більшість рішень зводяться до виділення серед зв'язних розділів мережі, які утворилися, одного «головного» розділу, де, власне, і продовжується робота. Для вибору використовуються різноманітні варіанти алгоритмів «голосування».

Вертикально та горизонтально розділені бази даних: у горизонтально розділених базах даних у декількох вузлах мережі можуть розміщуватися підвідношення одного відношення, тобто набори

повних кортежів цього відношення, сформовані за певними правилами. Зазвичай вважається, що підвідношення відповідає обмеженню відношення під керуванням деякого предикату обмеження. Таким чином, допускається наявність у декількох підвідношень загальних кортежів. При такій організації збільшення ефективності системи є можливе завдяки відсутності потреби в ряді випадків виконання обмеження відношення на стадії виконання запиту (якщо існує відповідне підвідношення).

У більш загальному випадку наявність фрагментів відношення в декількох вузлах мережі дає можливість розпаралелити виконання об'єднання. Наприклад, якщо, як у [90], фрагменти відношення отримані в результаті застосування мультиатрибутного хешування, вдається досягти дуже високого ступеня розпаралелювання. Зауважимо, що розподілені системи, орієнтовані на паралельне виконання реляційних операцій, зазвичай будуються на основі апаратури високошвидкісних локальних мереж, і до них можна звертатись як до машин баз даних. Питання оптимізації в таких системах є своєрідними і загалом відрізняються від тих, що властиві розподіленим системам розглянутого класу.

У вертикально розділених базах даних у різних вузлах мережі також зберігаються підвідношення одного відношення, але кожне підвідношення є деякою проекцією вихідного відношення, тобто складається не з повних кортежів. Подібно, як у випадку горизонтального поділу декілька підвідношень одного відношення могли містити загальні кортежі, при вертикальному поділі підвідношення можуть включати загальні поля вихідного відношення. Якщо врахувати, що, як правило, об'єднання виконуються над попередньо спроектованими відношеннями, а операція проекції у своєму строгому визначенні потребує усунення дублікатів (тобто є затратною), застосування вертикального поділу може значно підвищити ефективність об'єднань.

Відзначимо для коректності, що питання щодо дублікатів є простим лише в чистій реляційній алгебрі. Відразу, коли йдеться про можливість використання вбудованих функцій, таких як COUNT і AVG, ситуація істотно ускладнюється. Допустимі різні

комбіновані рішення, до яких можна віднести можливість підтримки в розподіленій базі даних матеріалізованих представлень бази даних. Наявність такої можливості, звичайно, також сприяла б підвищенню ефективності системи при виконанні запитів на вибірку.

3.4. Системний аналіз основних функціональних компонентів автоматизованого облікового комплексу для соціоорієнтованих структур

3.4.1. Склад і характеристика автоматизованих інформаційних систем

Розв'язання всього комплексу задач, на який повинен бути орієнтований інформаційний комплекс, забезпечується чотирма функціональними контурами: адміністративного управління; оперативного управління; управління виробництвом; бухгалтерського обліку.

Модульний принцип побудови комплексу допускає як ізольоване використання окремих програмних модулів, так і їх довільні комбінації, залежно від виробничо-економічної необхідності. На рис. 3.3 зображена структура автоматизованого комплексу. Модуль *«Управління документообігом»* винесений за межі контуру адміністративного управління, адже він забезпечує взаємодію всіх користувачів комплексу.

В основі моделі побудови програмного інформаційного комплексу повинні бути такі концептуальні положення:

1. Метою функціонування будь-якого підприємства (організації) є одержання прибутку від результатів своєї діяльності.

2. Усі взаємодії між юридичними суб'єктами (підприємствами, організаціями) зводяться до висновку та реалізації угоди. При цьому одна зі сторін є продавцем, друга — покупцем. Предметом угоди може бути товарно-матеріальна цінність (ТМЦ), робота, послуга чи їх комбінація.

3. При здійсненні будь-якої господарської операції формується документ, що підтверджує її реалізацію (операційний документ). Сукупність операційних документів творить документообіг підприємства.

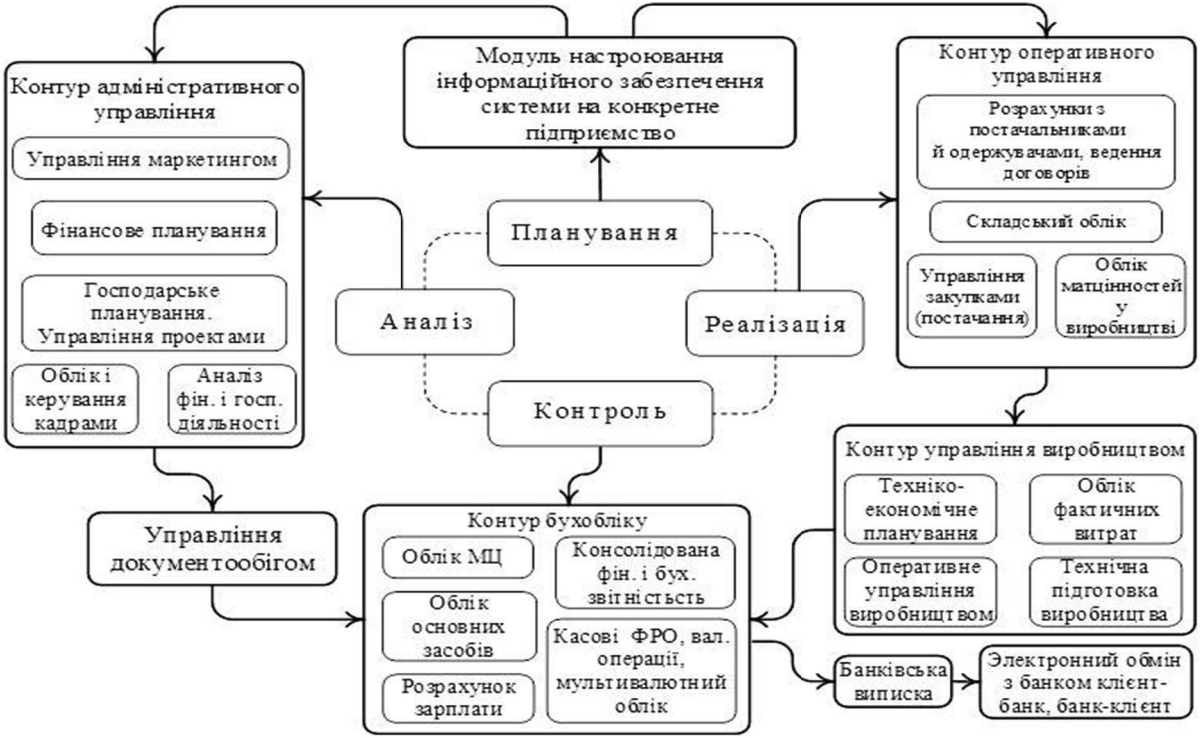


Рис. 3.3. Структура автоматизованого комплексу

4. Операційні документи належать до одного з двох класів. Перший клас документів — документи-підстави, тобто документи, що регламентують операції між юридичними особами. До цього класу належать прості й багатоступінчасті договори, рахунки, рахунки-фактури, контракти, вимоги, гарантійні листи тощо.

Другий клас документів — супровідні, тобто операційні документи, що віддзеркалюють суть фактично виконуваних операцій.

Усі супровідні документи можна розділити на дві групи:

— документи, що підтверджують переміщення ТМЦ, або операції виконання робіт, послуг; до них належать накладні різноманітних видів, складські ордери, акти на виконання робіт (послуг);
— фінансові супровідні документи, що підтверджують операції переміщення наявних і безготівкових фінансових засобів; до них належать банківські й касові документи. Супровідні документи, як правило, пов'язані з документами-основами.

5. Робота користувачів контуру оперативного управління полягає в реєстрації вхідних або формуванні вихідних документів-підстав і супровідних документів, що підтверджують виконання господарської операції.

При чітко налагодженій організаційній схемі функціональної експлуатації програмного комплексу кожний виконавець виконує визначені для нього інструкцією дії, одержуючи інформацію в обсязі, необхідному і достатньому для здійснення конкретних посадових обов'язків. У результаті роботи всіх користувачів комплексу відбувається наповнення БД підприємства (організації) оперативною інформацією про хід виконання конкретних господарських операцій, що належать до різноманітних напрямів діяльності. Опрацювання оперативної інформації дає змогу, з одного боку, проаналізувати взаємовідносини з контрагентом на основі зведень про прямування матцінностей, послуг, робіт і фінансових засобів, а з другого — оцінити ефективність роботи підприємства у різноманітних напрямках господарської діяльності. При цьому забезпечується:

— принцип однократного запровадження до БД інформації і, як наслідок, відсутність дублювання функцій користувачів, упорядкування документообігу;

— легкість контролю на коректність і цілісність даних, персоніфікація дій користувача;
— контроль за регламентом виконання господарських операцій;
— швидка перебудова комплексу, зміна експлуатаційної схеми комплексу при зміненні бізнесу-процесу (технології управління).

Адміністрація підприємства (організації), використовуючи для управління виробничими процесами інформаційний комплекс, одержує можливість: оперативного отримання достовірної інформації про поточну діяльність підприємства; оперативного управління фінансами; контролю виконання договірних відношень; контролю взаємних зобов'язань; контролю й управління матеріальними, трудовими і технічними ресурсами; формування та контролю бізнес-плану; планування й обліку виконання внутрішнього бюджету.

3.4.2. Аналіз фінансових ресурсів

Реалізація програмного модуля «*Аналіз фінансової і господарської діяльності»* у контурі адміністративного управління комплексу має на меті забезпечити керівника набором наочних графічних і текстових звітів для швидкого огляду господарсько-фінансового стану підприємства (групи консолідованих підприємств) та прийняття управлінських рішень [10, 20, 24].

Дані для аналізу — результати роботи оперативних підрозділів, що пройшли бухгалтерське опрацювання (синтетична й аналітична інформація з рахунків бухгалтерського обліку). Зібрана інформація розглядається в динаміці з можливістю урахування індексу цін. Аналіз господарсько-фінансової діяльності підприємства провадиться на основі типових форм («Баланс підприємства», «Звіт про фінансові результати») та інших показників ефективності господарсько-фінансової діяльності підприємства і внутрішніх звітів підприємства. Програмний модуль має бути орієнтований насамперед на фінансового директора, головного бухгалтера, фінансового менеджера (аналітика), співробітників відділу планування підприємства. За бажанням можна створювати звіти для податкових органів, потенційних партнерів та інвесторів.

Основні можливості модуля:

— вертикальний аналіз (структура) типових форм звітності за будь-який інтервал часу;
— горизонтальний аналіз (динаміка) типових форм звітності за будь-який інтервал часу;
— вертикальний і горизонтальний аналіз типових форм звітності щодо обраного базового моменту часу;
— аналіз динаміки і структури показників господарсько-фінансової діяльності підприємства (оцінка майнового становища, ліквідності, фінансової усталеності, ділової активності, рентабельності, становища підприємства на ринку цінних паперів);
— аналіз динаміки перевищення чи заниження показниками господарсько-фінансової діяльності, що рекомендуються, значень дозволяє зробити висновок про можливе погіршення або поліпшення господарсько-фінансового стану підприємства, показує небезпечне або нераціональне співвідношення господарсько-фінансових засобів підприємства;
— розрахунок провадиться за обраним консолідованим звітом, тобто по підгрупі підприємств, що входить до повної групи консолідованих підприємств;
— набір розрахункових форм відповідає обраному плану рахунків і, таким чином, може задовольнити найрізноманітніших споживачів аналітичної інформації, зокрема іноземних партнерів;
— усі розраховані значення рекомендуються і в текстових, і в графічних звітах;
— аналіз провадиться з тим рівнем деталізації, який підтримують інші контури системи;
— розрахунки провадяться в усіх валютах, інформацію про курси яких містить система, причому суми всіх операцій перераховуються за курсом відповідної валюти на день операції;
— створення й перегляд звітів, похідних від типових форм, дає змогу групувати і співвідносити розраховані значення практично за будь-якою методикою фінансового менеджменту;

— розробка будь-яких внутрішніх звітів підприємства в термінах мови проектування бухгалтерських і економічних розрахунків системи має дозволяти скористатися всіма перерахованими вище можливостями модуля для одержання управлінської інформації.

3.4.3. Облік і управління кадрами та обігом документів

Програмний модуль *«Облік і управління кадрами»* призначений для:

— автоматизації процесу ведення особистих справ працівників;
— планування й управління штатним розкладом і резервом на заміщення посад;
— планування й обліку робочого часу співробітників;
— одержання звітів з кадрової інформації про співробітників підприємства.

На рис. 3.4 подано схему функціонування модуля, в якій відображено взаємозв'язок задач, розв'язуваних системою, з даними інформації, що зберігається в базі даних підприємства. Показано взаємодію з іншими модулями системного комплексу. *Особиста справа* співробітника складається з дев'яти розділів, що містять у сумі приблизно 150 показників. Розділ «*Анкетні дані»* дає змогу сформувати анкету з необхідним набором питань. Крім того, до особистої справи можна «підшити» довільну кількість додатків, що містять текстову (автобіографія, характеристики тощо), графічну та іншу обов'язкову інформацію.

Управління штатами забезпечує упорядкування штатного розкладу підприємства з вказівкою основних характеристик робочих місць. Якщо є потреба, можна ввести додаткові характеристики, склад яких визначається користувачем (вимоги до кваліфікації, віку, посадові інструкції, оснащення робочого місця тощо). Усі призначення і переміщення виконуються відповідно до штатного розкладу. Один співробітник може займати декілька ставок, зокрема неповних. Щодо кожного робочого місця можна скласти список співробітників, котрі входять у резерв на заміщення певної посади.

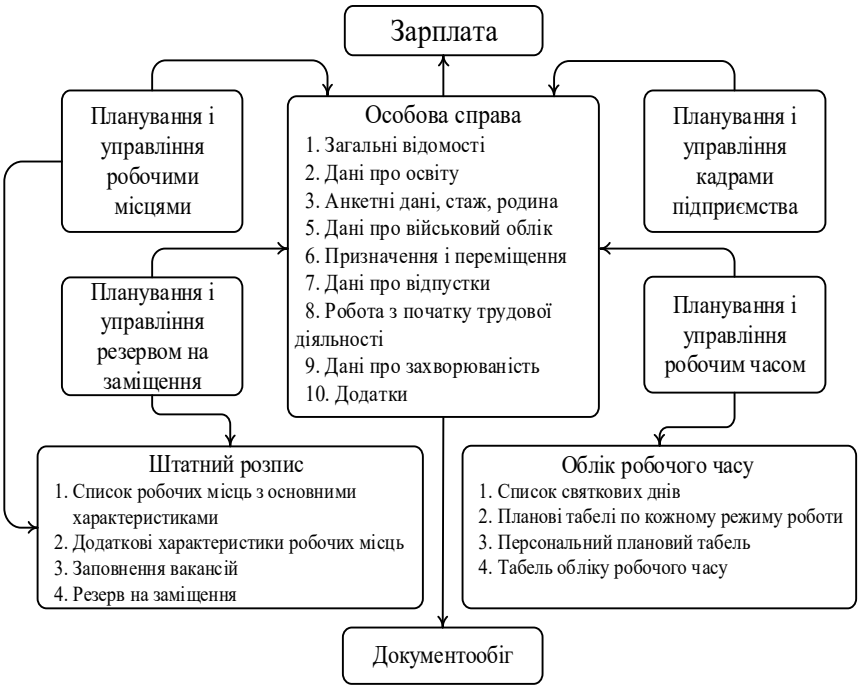


Рис. 3.4. Схема функціонування модуля обліку й управління кадрами

Облік робочого часу забезпечує ведення планового і фактичного табелів урахування робочого часу інтегровано з відпустками, лікарняними, призначеннями і переміщеннями. На одного співробітника можна вести декілька табелів одночасно — по одному на кожну зайняту ставку.

Засоби підготування звітів пропонують можливість гнучкого налаштування вихідних документів на потреби конкретного підприємства. Створення звіту передбачає визначення правил добору співробітників у звіт, порядок їх сортування і вибір форми вихідного документа. Користувач може створити свій власний звіт як додаток до вже наявного і внести його до списку стандартних звітів для подальшого використання.

Модуль обліку й управління кадрами пропонує гнучкі засоби для адаптації до найрізноманітніших вимог:

— склад і структура каталогів цілком визначаються потребами підприємства, користувач може створювати власні каталоги для заповнення анкет і додаткових характеристик робочих місць;
— анкета дає можливість вводити довільні, не передбачені програмою дані про співробітника і використовувати їх під час добору та сортування виведених у звіт записів;
— додатки дають змогу зберігати довільну додаткову інформацію про співробітника — як текстову, так і графічну;
— склад наявних звітів легко розширюється завданням правил добору співробітників у звіт і порядку їх сортування. Форми вихідних документів проектуються за допомогою текстового редактора.

Модуль обліку й управління кадрами може бути встановлений як у локальному варіанті, коли вся робота виконується на одному комп'ютері, так і в мережевому — робота над тими самими даними може виконуватися на декількох комп'ютерах одночасно, що дозволяє гнучко розподіляти обов'язки між працівниками відділу кадрів та іншими споживачами кадрової інформації. Модуль обліку й управління кадрами працює разом із модулями «Зарплата» та «Управління документообігом». Програмний модуль *«Управління документообігом»* призначений для урахування, збереження й опрацювання документів і облікових карток паперових документів (договорів, листів, наказів, протоколів нарад та ін.) в електронній формі, а також для організації спілкування користувачів у процесі розв'язання виробничих задач. Документи, що входять у документообіг, можуть бути отримані скануванням паперових документів, які надійшли електронною поштою, а можуть бути підготовлені з допомогою різноманітних текстових редакторів.

Модуль «*Управління документообігом»* надає користувачу можливості: створення і ведення номенклатури справ фірми; створення повнотекстових документів; створення класифікації документів і використання її в процесі роботи; ведення стадій опрацювання документів і контроль за їх виконанням; пошук документів за формальними полями, що привласнені конкретному документу; просування документів за маршрутом опрацювання; масове розсилання до-

кументів у підрозділи; реєстрація звітів комплексу як документів; виведення на екран списку всіх перелічених документів, пов'язаних із документами-основами комплексу (наприклад, рахунками за продані товари або послуги) і напрямами робіт господарського плану; перегляд будь-якого із зазначених документів. Зв'язок із документами-основами встановлює користувач. Схема документообігу на підприємстві (організації) зображена на рис. 3.5.

3.4.4. Контур оперативного управління

До контура оперативного управління належать задачі, безпосередньо пов'язані з реалізацією виробничих планів підприємства (рис. 3.6). Серед них можна виділити задачі, актуальні для всіх типів організацій (постачання, складський облік), а також задачі, характерні тільки для державного управління.

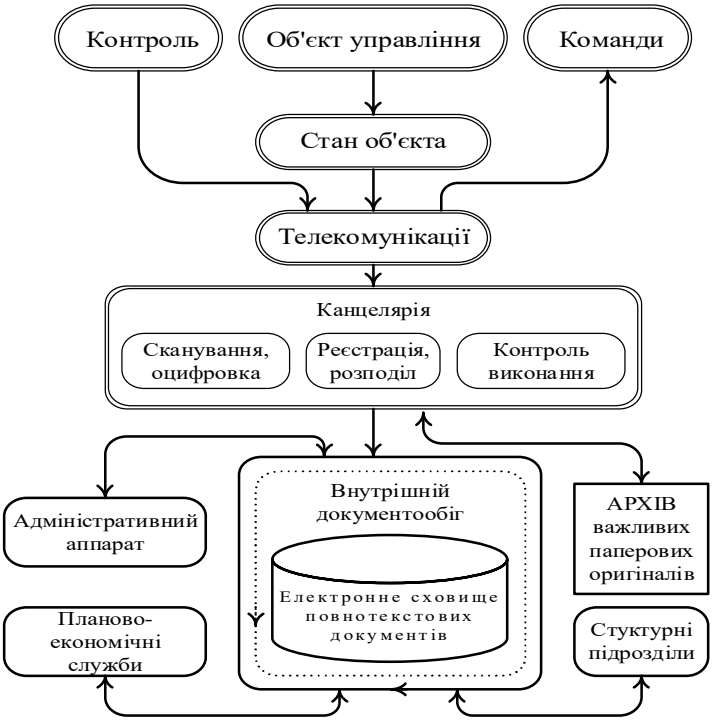


Рис. 3.5. Схема документообігу

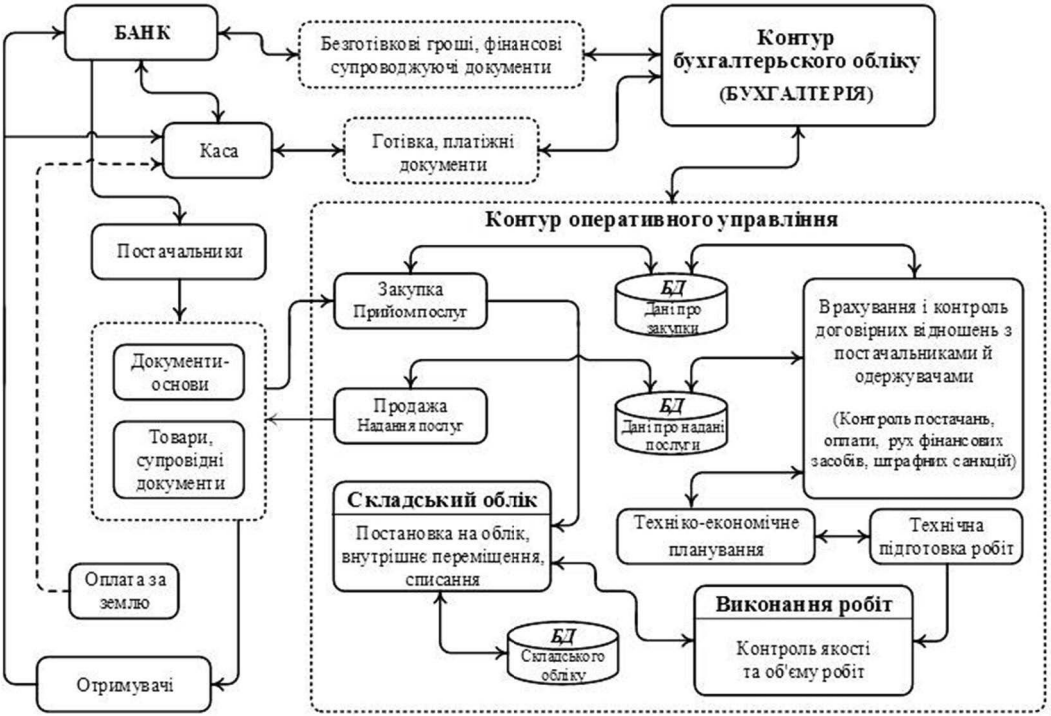


Рис. 3.6. Схема інформаційних потоків контура оперативного управління

Управління закупівлями (матеріально-технічне постачання). Стандартні функції підрозділу, відповідального за закупівлі необхідних підприємству товарів і матеріалів, звичайно передбачають ведення картотеки пропозицій потенційних постачальників, відстеження вимог, що надходять від інших підрозділів, заявок на придбання, упорядкування плану закупівель відповідно до укладених договорів і довгострокових контрактів, вибір конкретного постачальника і формування замовлення на постачання, реєстрацію документів, на підставі яких провадиться закупівля (рахунки, договори, контракти, гарантійні листи), оформлення доручень на одержання по складах, контроль стану договорів і платіжних документів на придбання, одержання різноманітних звітів у розрізі номенклатури, що простежується, партій, груп і використовуваних систем класифікації.

Одержання й відстеження заявок на придбання здійснюється модулем *«Управління документообігом»*. Заявки на постачання продукції — «*Управління закупівлями».* Для контролю взаєморозрахунків із постачальниками розроблений модуль *«Постачальники, одержувачі».* Безпосередньо в модулі *«Управління закупівлями»* зосереджені операції щодо роботи з конкретними документами на придбання.

Унаслідок глибокої інтеграції модулів і наявності єдиної бази даних усі зведення, необхідні для оформлення документів-підстав, накладних і доручень, можуть бути введені шляхом вибору з наявних таблиць бази даних. Підключення потрібної таблиці (класифікатора, довідника) у момент заповнення тієї чи іншої графи документа забезпечується автоматично. Інтеграція модуля «*Управління закупівлями»* зі складським обліком і плануванням виробництва дає можливість у модулі *«Складський облік»* відслідковувати встановлені рівні нормативних запасів (матеріалів, що комплектуються, товарів), виявляти дефіцит і продукцію, яка не користується попитом (неліквіди).

Відмінні ознаки реалізації задачі управління закупівлями: використання в документах на закупівлю як товарних, так і нематеріальних позицій (послуг); облік партій товарів, що закуповуються, відстеження термінів збереження, термінів дії ліцензій та сертифікатів; підтримка різноманітних валют і міжнародних закупівель; облік мита, транспортних та інших витрат під час обчислення облікової вартості конкретних виробів, що закуповуються; облік повернень по рекламації; автоматизований розподіл товарів по складах; автоматичне формування прибуткових складських ордерів по групі накладних; формування платіжних документів на оплату по документах-основах; формування доручень на одержання матеріальних цінностей; одержання зведень про документи-підстави по обраних контрагентах за допомогою «картки постачальника»; повна інтеграція з модулем «*Постачальники, одержувачі»*; відбиток у бухгалтерському контурі всіх операцій закупівлі матеріальних цінностей і послуг за допомогою механізму типових господарських операцій.

Технологія розв'язання задачі управління закупівлями відображена схемою на рис. 3.7.

Стандартні звіти: звіти цін закупівель товарів і послуг у розтині номенклатури, постачальників, груп, партій, зовнішньої класифікації; звіти по зареєстрованих прибуткових і сформованих поворотних накладних; звіт про стан виконуваних замовлень на закупівлю; реєстри документів-підстав і накладних; звіт про невідповідності в документах на закупівлю.

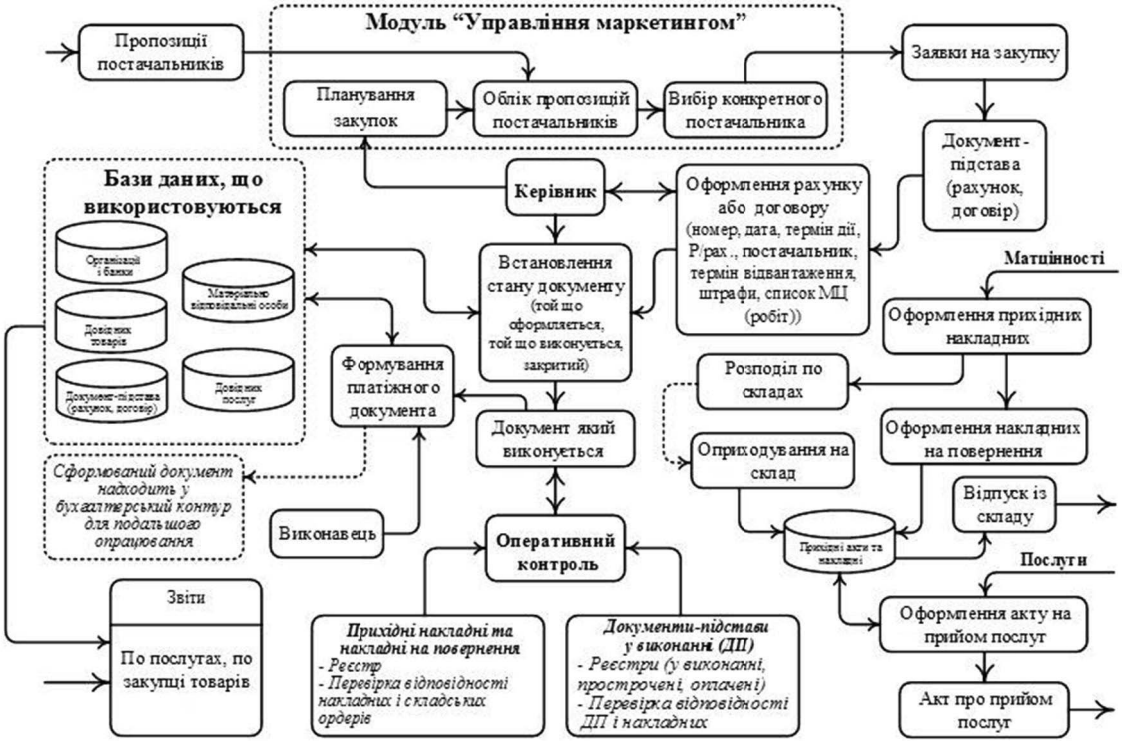


Рис. 3.7. Схема реалізації модуля «Управління закупівлями»

РОЗДІЛ 4
ПОБУДОВА ІНФОРМАЦІЙНИХ РОЗПОДІЛЕНЬ СИСТЕМ, ОРІЄНТОВАНИХ НА АНАЛІЗ РІЗНОВИДНИХ ДАНИХ

4.1. Банки даних у структурі управління

Сьогодні практично в будь-якій організації склалася усім добре знайома ситуація: інформація, начебто, десь і є, її навіть надто багато, але вона не нормалізована, розрізнена, не завжди достовірна, її практично неможливо вчасно отримати і застосувати (використати). Саме на усунення наявної суперечності — відсутність інформації за наявності і навіть її надлишку, — націлена концепція банків даних (*Data Warehouse*).

В основі концепції банків даних містяться дві основні ідеї:

1) інтеграція раніше роз'єднаних деталізованих даних: історичні архіви;

2) дані з традиційних баз даних (БД); дані із зовнішніх джерел у єдиній БД, їх узгодження.

Розподілено набори даних, які використовуються для оперативного опрацювання, і набори даних, використовувані для розв'язання задач аналізу.

Автором концепції банків даних (Data Warehouse) є Б. Інмон, що визначив банки даних як: «…предметно-орієнтовані, інтегровані, незмінні, що підтримують хронологію набору даних, організовані для цілей підтримки керування», покликані виступати в ролі «одного й єдиного джерела істини» та забезпечувати менеджерів і аналітиків достовірною інформацією, необхідною для оперативного аналізу і прийняття рішень. Річард Хакаторн — інший основоположник цієї концепції, вважає, що ціль банків даних — забезпечити для організації «єдиний образ існуючої реальності» [92].

Найпоширенішою сьогодні помилковою думкою є спроба знайти в концепції банків даних якийсь завершений рецепт реалізації інформаційної аналітичної системи. Тим паче, єдиний готовий програмний продукт або якесь готове універсальне рішення. Ціль концепції банків даних — з'ясувати відмінності в характеристиках

даних в операційних і аналітичних системах, визначити вимоги до даних, розміщенихв цільовій БД банку даних, встановити загальні принципи й етапи її побудови, головні джерела даних, дати рекомендації для вирішення потенційних проблем, що виникають у разі їх перевантаження, очищення, узгодження, транспортування й завантаження в цільову БД.

Предметом концепції банку даних є самі дані. Після того як традиційна БД реалізована і починає функціонувати, вона стає таким самим самостійним об'єктом реального світу, як і будь-який, виробничий процес. Дані, які є одним із кінцевих продуктів такого виробництва, набувають таких самих властивостей й характеристик, як і будь-який промисловий продукт: термін придатності, місце складування (збереження), сумісність з даними інших виробництв БД, ринкова вартість, переміщення, комплектність тощо. І саме з цього погляду розглядаються дані в БД, тобто його предметом є не способи опису й відображення об'єктів предметної галузі, а власне, дані як самостійний об'єкт предметної галузі, утвореної в результаті функціонування раніше створених інформаційних систем.

Для правильного розуміння цієї концепції необхідне усвідомлення декількох принципових моментів:

1) концепція банків даних — це не концепція аналізу даних, скоріше це концепція підготовки даних для аналізу;

2) концепція банків даних не зумовлює архітектури цільової аналітичної системи, вона вказує на те, які процеси повинні виконуватися в системі, а не де конкретно і як саме процеси мають виконуватися;

3) концепція банку даних допускає не просто єдиний логічний зміст даних організації, вона передбачає реалізацію єдиного інтегрованого джерела даних.

Останній пункт є принциповим, тому дослідимо його детальніше. Сьогодні досить популярними є рішення, що допускають інтеграцію різноманітних БД на основі єдиного довідника метаданих (підтримуючого єдиний логічний зміст даних організації), але не єдиного інтегрованого джерела даних (віртуального банку да-

них). При цьому передбачається динамічне розвантаження (для кожного нового запиту) даних із різноманітних операційних джерел (БД), їхнє динамічне узгодження, транспортування до користувача. Очевидно, що для певних класів додатків це рішення цілком коректне. Проте варто заздалегідь розуміти всі обмеження, що накладаються. Крім єдиного довідника метаданих, засобів розвантаження, агрегації й узгодження даних, концепція банків даних передбачає: інтегрованість, незмінність, підтримку хронології й узгодженість даних. І якщо дві перших властивості (інтегрованість і незмінність) впливають на режими аналізу даних (як буде показано нижче, без інтегрованої бази даних, у якій використовуються спеціалізовані методи збереження й доступу, принаймні сьогодні, важко говорити про реалізацію інтерактивного динамічного аналізу), останні дві — підтримка хронології й узгодженість — істотно звужують список розв'язуваних аналітичних задач.

Без підтримки хронології (наявності історичних даних) не можна говорити про розв’язання задач прогнозування й аналізу тенденцій. Але найбільш критичними і болючими є питання, пов’язані з узгодженням даних. Головна вимога аналітика — це навіть не так оперативність, як достовірність відповіді. А достовірність, у кінцевому підсумку, й визначається узгодженістю. Поки не проведена робота з взаємного узгодження значень даних із різноманітних джерел, важко говорити про їх достовірність. Класична схема надходження та обробки даних зображена на рис. 4.1.

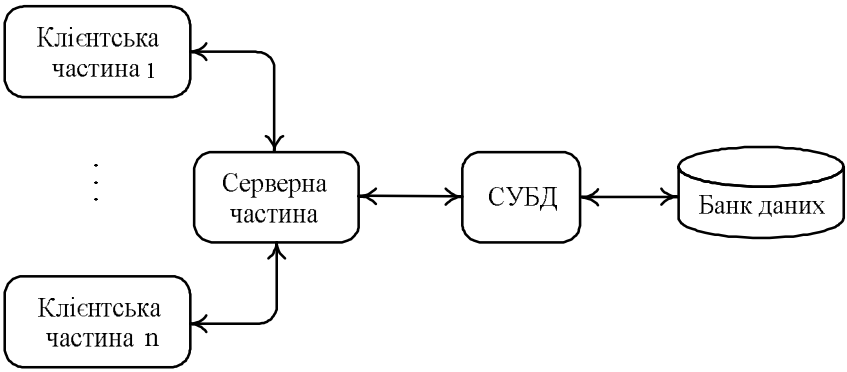


Рис. 4.1. Класична схема надходження та опрацювання даних

Практично в будь-якій організації питання узгодженості даних з різноманітних інформаційних систем постають надзвичайно гостро. І часто менеджер стикається з ситуацією, коли на ті самі запитання різноманітні системи можуть дати і, звичайно, дають різні відповіді. Це може бути пов'язано як із несинхронністю моментів модифікації даних, відмінностями в трактуванні тих самих подій, понять і даних, зміною семантики даних у процесі розвитку предметної галузі, елементарними помилками під час встановлення системи й опрацювання даних, частковою втратою окремих фрагментів архівів і т. д. Очевидно, що врахувати і заздалегідь визначити алгоритми всіх можливих колізій малоймовірно. Тим паче, це нереально зробити в оперативному режимі, динамічно, безпосередньо в процесі формування відповіді на запит.

Традиційно на банки даних спираються інформаційно-аналітичні системи. Аналітичні системи завжди пред'являли істотно вищі, ніж традиційні пошукові системи, вимоги до апаратного й програмного забезпечення. Розпочинаючи побудову аналітичної системи, варто розуміти, що її реалізація практично неможлива без вирішення таких питань, як: неоднорідність програмного середовища; захист даних від несанкціонованого доступу; побудова і ведення багаторівневих довідників метаданих; ефективне збереження й обробка дуже великих об'ємів даних.

Основна відмінність банку даних від традиційних БД полягає в тому, що вони практично ніколи не створюються на порожньому місці. І тому майже завжди кінцеві рішення будуть різнорідними (з погляду виробників програмних засобів, принципів побудови, операційних систем). Основою банку даних є не внутрішні, як у більшості традиційних БД, а зовнішні джерела даних: різноманітні інформаційні системи, електронні архіви, загальнодоступні й комерційні електронні каталоги, довідники, статистичні збірники. Як правило, сьогодні в будь-якій організації реально функціонує множина незв'язаних або слабопов'язаних БД. Здебільшого вони створювалися в різний час, різними колективами розробників і реалізовані на основі різноманітних програмних і апаратних засобів. Таким чином, сама основа, на якій будуватиметься банк даних, зазвичай є вкрай неоднорідною. Додайте сюди засоби розвантаження,

транспортування, реалізації цільової БД банку даних. Очевидно, що в таких умовах навіть говорити про однорідність програмних засобів надзвичайно складно.

Практично завжди задача побудови банку даних — це задача побудови єдиної узгоджено функціонуючої інформаційної системи на основі неоднорідних програмних засобів і рішень. Уже безпосередньо вибір засобів реалізації банку даних стає надзвичайно складною задачею. Тут повинна враховуватися множина чинників, зокрема взаємна сумісність різноманітних програмних компонент, легкість їх освоєння і використання, ефективність функціонування, стабільність і навіть форма, рівень і потенційна перспективність взаємовідносин різноманітних фірм-виробників.

Банки даних уже за своєю природою є розподіленим рішенням. В основі концепції банку даних міститься фізичне розділення вузлів, у яких виконується оперативна обробка, від вузлів, у яких виконується аналіз даних (рис. 4.2). І хоча у процесі реалізації такої системи нема необхідності суворої синхронізації даних у різноманітних вузлах (наприклад, на основі засобів двофазної фіксації транзакцій), засоби асинхронної асиметричної реплікації даних є невід'ємною частиною практично будь-якого рішення.

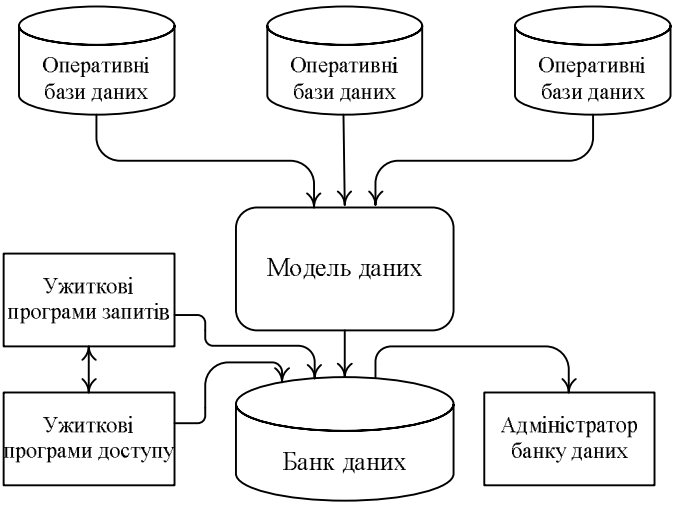


Рис. 4.2. Банки даних у структурі управління

Наявність метаданих і засобів їх представлення кінцевим користувачам є одним із основних чинників успішної реалізації банку даних. Окрім того, без наявності актуальних, максимально повних і легко зрозумілих користувачам описів даних банк даних перетворюється в звичайний, але дуже дорогий електронний архів. Головною ж задачею, з якою стикаєшся під час проектування й реалізації інформаційної системи на основі концепції побудови банків даних, є необхідність одночасної роботи з найрізноріднішими зовнішніми джерелами даних, неузгодженістю їх структур і форматів, масштабами і кількістю архівів, що повинні бути перероблені й завантажені. Отож при побудові такої системи розробнику складно обійтися без високотехнологічних засобів опису інформаційної моделі системи, причому така модель має містити описи не тільки цільових структур даних у БД банку даних, а й структур даних у джерелах їх одержання (різноманітних інформаційних системах, архівах, електронних довідниках тощо), правила, процедури і періодичність їх вибірки й розвантаження, процедури й місця узгодження та агрегації.

Надзвичайно важливою проблемою при проектуванні інформаційних систем на основі ідеології побудови банків даних є розробки спеціалізованих програмних модулів, спрямованих на забезпечення захисту й обмеження прав доступу до банків даних. Зібравши в одному місці всю інформацію історію розвитку організації, її успіхи й невдачі, про взаємовідносини з постачальниками і замовниками, історію розвитку і стан ринку, менеджери отримують унікальну можливість для аналізу минулої діяльності, сьогодення та побудови обґрунтованих прогнозів на майбутнє. Однак не варто забувати і про те, що у разі незабезпечення належних засобів захисту й обмеження прав доступу, можливо постачати здобутою інформацією і конкурентів. Отож одним з найважливіших питань під час обговорення проекту банків даних є захист даних.

Суто психологічно багатьох лякають не так затрати на реалізацію системи банків даних, як те, що доступ до критично значимої інформації може одержати будь-хто, не маючи на це права. Для побудови таких інформаційних систем часто надають недостатньо

уваги захисту та збереженню інформації, забезпечених у стандартних конфігураціях комерційних СУБД (зазвичай рівень захисту по класу «*C2 Orange Book*»). Регіональний менеджер повинен бачити лише ті дані, що стосуються його регіону, а менеджер підрозділу не має ознайомлюватися з даними щодо усієї фірми. Проте для підвищення ефективності доступу до даних у цільовий БД банків даних усі потрібні дані, як правило, зберігаються у вигляді єдиної фактологічної таблиці. Внаслідок цього засоби реалізації повинні підтримувати обмеження доступу не тільки на рівні окремих таблиць і їх колонок, а й окремих рядків у таблиці (клас «*B1 Orange Book*»).

Не менше гостро постають і питання авторизації та ідентифікації користувачів, захисту даних у місцях їх перетворення й узгодження, в процесі їх переміщення мережею (шифрування паролів, тексти запитів, даних).

4.1.1. Напрями і типи задач створення СППР

1. Не варто шукати в концепції банків даних щось цілком принципово нове, про що не говорилося та не писалося раніше, і чому не можна знайти аналогії у минулому. Реальне значення полягає в тому, що концепція банків даних являє собою: «частину відповіді з боку інформаційних технологій на питання: «Що ми робимо далі?». І подібно до багатьох новацій у технологіях, цей термін використовується для того, щоб описати концепцію, яка обеззброює своєю простотою, і що має потенціал розвинутися згодом у щось складніше й значуще. Водночас, як було показано вище, сьогодні можна твердити, що поява цієї концепції стала серйозним стимулом для розвитку внутрішньої архітектури сучасних СУБД, їх програмного оточення, інструментальних засобів кінцевого користувача, різноманітних міжкорпоративних стандартів.

2. Незважаючи на те, що вартість аналітичних систем і досі залишається досить високою, а методології та технології реалізації таких систем перебувають на стадії становлення, вже сьогодні економічний ефект, забезпечуваний ними, істотно перевищує ефект

від традиційних оперативних систем. Ефект від правильної організації, стратегічного й оперативного планування розвитку бізнесу важко заздалегідь оцінити в цифрах, але незаперечно, що він у десятки і навіть сотні разів може перевершити витрати на реалізацію таких систем. Однак варто пам'ятати, що ефект забезпечує не сама система, а люди, які з нею працюють. Отож не зовсім коректними є декларації типу: система банків даних буде допомагати менеджеру приймати правильні рішення.

Підтримка прийняття управлінських рішень на основі нагромадженої інформації може здійснюватися в трьох основних областях [15]: деталізованих даних; агрегованих показників; закономірностей.

Основні елементи архітектури інформаційних систем, орієнтованих на аналіз даних, зображені на рис. 4.3.

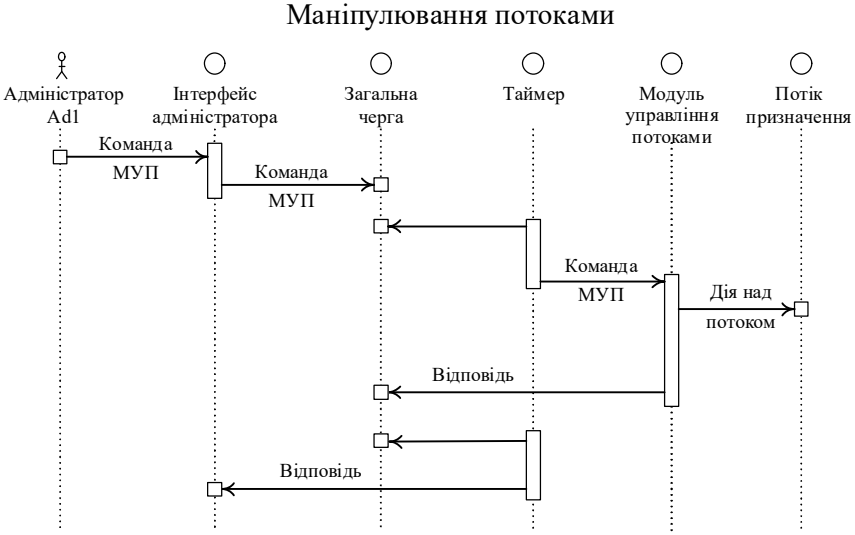


Рис. 4.3. Концепція архітектурного проектування розподілених систем

Сучасні аналітичні системи не є системами штучного інтелекту і вони не можуть ні допомогти, ні перешкодити в ухваленні рішення. Їх ціль — своєчасно забезпечити менеджера всією інфор-

мацією, необхідною для ухвалення рішення. Яка ж інформація буде затребувана і яке рішення буде прийнято на її основі, залежить тільки від конкретної людини.

4.2. Концепція архітектурного проектування розподілених систем

Великі системи завжди можна розподілити на підсистеми. Архітектурним проектуванням називають перший етап процесу проектування, на якому визначаються підсистеми, а також структура керування і взаємодії підсистем. Метою архітектурного проектування є опис архітектури програмного забезпечення.

Перший етап у моделі процесу проектування — архітектурне проектування, що служить з'єднувальною ланкою між процесом проектування і процесом розробки вимог до системи, яка створюється. В ідеальному випадку специфікація вимог не повинна містити інформації про структуру системи. Насправді таке справедливо лише для невеликих систем. Архітектурна декомпозиції системи потрібна для структуризації й організації системної специфікації. Модель системної архітектури часто служить відправною точкою для створення специфікації різних частин системи. У процесі архітектурного проектування розробляється базова структура системи, тобто визначаються основні компоненти системи і взаємодія між ними.

Існують різні підходи до процесу архітектурного проектування, вони залежать від професійного досвіду, а також майстерності та інтуїції розробників. Можна виділити кілька етапів, спільних для всіх процесів архітектурного проектування:

1. Структурування системи. Програмна система структурується у вигляді сукупності відносно незалежних підсистем; також визначаються взаємодії між підсистемами.

2. Моделювання управління. Розробляється базова модель управління взаємовідносинами між частинами системи.

3. Модульна декомпозиція. Кожна визначена на першому етапі підсистема розподіляється на окремі модулі; визначаються типи модулів і типи їх взаємозв'язків.

Як правило, означені етапи межують і накладаються один на одного. Етапи повторюються для детальнішої розробки архітектури, доки архітектурний проект не задовольнятиме системні вимоги. Чітких відмінностей між підсистемами і модулями немає, але деякі визначення будуть корисними.

1. *Підсистема* — це система, операції (методи) якої не залежать від сервісів, наданих іншими підсистемами. Підсистеми складаються з модулів і мають визначені інтерфейси, за допомогою яких взаємодіють з іншими підсистемами.

2. *Модуль* — зазвичай компонент системи, що надає один або кілька сервісів для інших модулів. Модуль може використовувати сервіси, що підтримуються іншими модулями. Як правило, модуль ніколи не розглядається як незалежна система, вони звичайно складаються з ряду інших, простіших компонентів.

Результатом процесу архітектурного проектування є документ, що відображає архітектуру системи. Він складається з набору графічних схем представлення моделей системи з відповідним описом. В описі зазначено, з яких підсистем складається система і які модулі утворюють кожну з підсистем. Графічні схеми моделей системи дають можливість поглянути на архітектуру з різних сторін. Як правило, розробляються чотири архітектурні моделі:

1. Статична структурна модель, у якій представлені підсистеми або компоненти, що надалі розробляються незалежно.

2. Динамічна модель процесів, у якій представлена організація процесів під час функціонування системи.

3. Інтерфейсна модель, що визначає сервіси, які надаються кожною підсистемою через загальний інтерфейс.

4. Моделі відносин, у яких показані взаємини між частинами системи, наприклад, потік даних між підсистемами.

При описі архітектури систем дослідники пропонують використовувати спеціальні мови опису архітектур, де основними архітектурними елементами є компоненти і конектори (об'єднувальні ланки); ці мови також пропонують принципи і правила побудови архітектур. Однак, аналогічно іншим спеціалізованим мовам, вони мають один недолік — вони зрозумілі лише для фахівців, які їх засво-

їли, а на практиці майже не використовуються. Фактично використання мов опису архітектур лише ускладнює аналіз систем. Отож для опису архітектур доцільно використовувати неформальні моделі й системи нотації, наприклад, уніфіковану мову моделювання UML.

Архітектура системи будується відповідно до визначеної архітектурної моделі. Важливо знати такі моделі, їх недоліки, переваги і можливості застосування [92]. Водночас архітектуру великих систем неможливо описати за допомогою однієї моделі. При розробці окремих частин великих систем можна використовувати різні архітектурні моделі, але при цьому архітектура системи може виявитися надто складною, оскільки буде побудована на комбінації різних архітектурних моделей. Розробник повинен підібрати найкращу модель, потім модифікувати її з урахуванням вимог ПЗ, що розробляється.

Архітектура системи впливає на продуктивність, надійність, зручність супроводу та інші її характеристики, тому моделі архітектури, вибрані для конкретної системи, можуть залежати від нефункціональних системних вимог:

1. *Продуктивність.* Якщо критичною вимогою є продуктивність системи, варто розробити таку архітектуру, щоб за всі критичні операції відповідало якомога менше підсистем з максимально малою взаємодією між ними. Для зниження взаємодії між компонентами краще використовувати великомодульні компоненти, а не дрібні структурні елементи.

2. *Захищеність.* У такому випадку архітектура повинна мати багаторівневу структуру, в якій найбільш критичні системні елементи захищені на внутрішніх рівнях, а перевірка безпеки цих рівнів здійснюється на вищому рівні.

3. *Безпека* — архітектуру варто проектувати так, щоб за всі операції, що впливають на безпеку системи, відповідало якнайменше підсистем. Такий підхід дає змогу знизити вартість розробки і вирішує проблему перевірки надійності.

4. *Надійність.* У цьому випадку варто розробити архітектуру з включенням надлишкових компонентів, аби можна було їх замінювати й оновлювати, не перериваючи роботи системи (рис. 4.4).

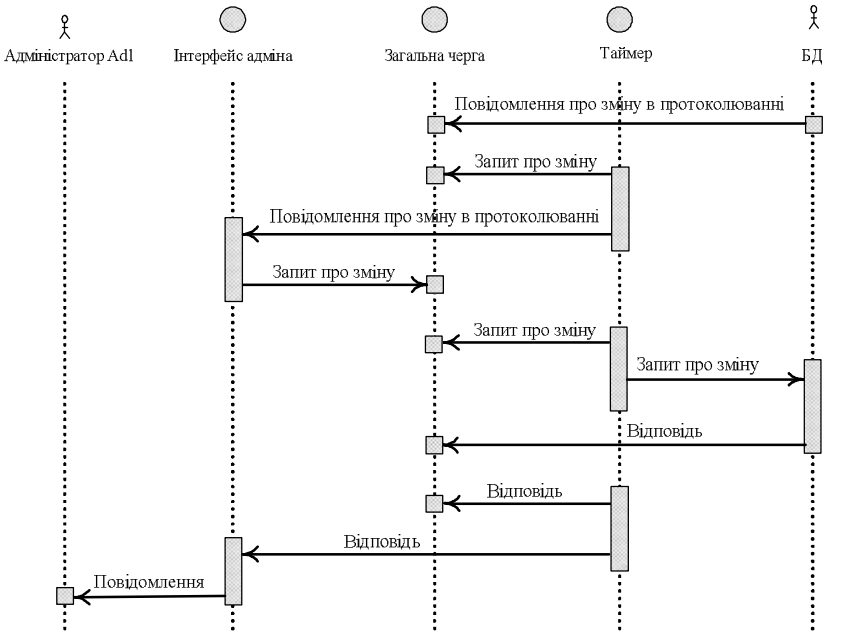


Рис. 4.4. Архітектура з включенням надлишкових компонентів

5. *Зручність супроводу* — архітектуру системи варто проектувати на рівні дрібних структурних компонентів, які можна легко змінювати. Програми, що створюють дані, відділені від програм, які керують даними.

4.3. Методи та алгоритми оптимізаційної роботи пошукової машини в системі підтримки прийняття рішень

У загальному випадку механізм пошуку, який закладається в пошукових машинах, складається з трьох головних складових. Першою й основною складовою є пошуковий павук (spider, crawler чи bot) — спеціальна програма, яка переглядає яку-небудь сторінку чи файл у мережі, ознайомлюється з її вмістом і потім згідно з механізмом навігації переходить на інші сторінки чи вузли в мережі. Павук повертається до вузла приблизно один раз у місяць чи

частіше, щоб віднайти зміни. Конкретний механізм пошуку є «ноу-хау» розробників.

Усі дані, які знаходить пошуковий павук, запущений пошуковою системою, входять у другу складову механізму пошуку, яка називається індексом. Індекс, який іноді називають каталагом, є подібним до величезної картотеки, що містить копію кожної сторінки (чи файла) мережі, які знайшла програма — пошуковий павук. Якщо надалі сторінка змінюється, під час подальшого перегляду пошуковий павук фіксує такі зміни і на основі знайденої інформації модифікує картотеку (індекс).

Програмне забезпечення механізму пошуку — третя складова пошукової системи, яка суттєво відрізняється в різних системах. Загалом це програма, яка переглядає величезну кількість сторінок, записаних в індексі, щоб знайти пари (шаблони) до пошуку та впорядкувати їх у визначеному порядку.

Завдання павука може полягати в створенні міні-покажчика вихідної сторінки для порівняння контекстів зв'язків із фрагментом запиту. Нечітка логіка допомагає павуку впорядковувати сторінки за їх відповідністю запиту (наприклад, вона може відкидати зв'язки, для яких імовірність доцільності їх відстежування становить 66%). Далі павук запитує сторінку чи документ, які відповідають найбільш імовірному зв'язку. Після того, як сторінка знайдена та занесена у базу даних, вона позначається як прочитана (оброблена); її вміст індексується, а вихідні зв'язки заносяться до списку пошукової системи. Виконавши своє завдання, павук тимчасово припиняє роботу. Надалі цикл повторюється заново.

Повторне оновлення інформації павуком супроводжується значними затратами, що виливається в часі його роботи та завантаженні трафіка. Пропонований у роботі алгоритм дає змогу покращити механізм повторного оновлення інформації та скоротити затрати роботи пошукової системи.

Алгоритм оптимізації роботи пошукової машини. Нехай на деякому вузлі пошукова система, окрім пошуку нової, постійно оновлює деяку інформацію. Визначимо інформаційний потік у вигляді набору:

$$A = A\,(A_1 \ldots, A_n), \tag{4.1}$$

де A_j — окремий вид інформації; j — індекс виду інформації у загальному наборі.

Для інформаційного набору А побудуємо табл. 4.1.

Таблиця 4.1

A_1	A_1
...	...
A_m	a_m

де m — кількість видів інформації в інформаційному наборі A; a_j = (0,1) — логічна змінна, яка відображає результат оновлення інформації j-го виду. Тут 0 означає, що інформація A_j не оновлювалась, 1 — інформація A_j оновлювалася. Індекс $j = \overline{1,m}$ вказує на причетність до виду інформації з інформаційного набору A.

Таблиця будується в деякий момент часу. Тобто зробивши в часі $T = T(t_1, ..., t_n)$ певну кількість вибірок, отримуємо часові набори:

Таблиця 4.2

t_1			t_n	
A_1	$a_1(t_1)$	...	A_1	$a_1(t_n)$
...	...		...	...
A_m	$a_m(t_1)$		A_m	$a_m(t_n)$

де t_i — момент часу, коли зроблена вибірка, $i = \overline{1,n}$; $a_j(t_i)$ = (0,1) — результат вибірки A_j -ої інформації в момент часу t_i.

Згідно з табл. 4.2 для кожного виду інформації A_j будується парна кореляція:

$$(t_1, a_j(t_1)), \ldots, (t_n, a_j(t_n)), \tag{4.2}$$

яку запишемо у вигляді:

$$(t_1, a_j^1), \ldots, (t_n\, a_j^n)\,, \tag{4.3}$$

де $a_j^i = a_j(t_i)$. (4.4)

Оцінивши згідно з [19] для кожного A_j перші моменти:

$$\overline{a}_j = \frac{1}{n}\sum_{i=1}^{n} a^i_j, \qquad \overline{t} = \frac{1}{n}\sum_{i=1}^{n} t_i, \qquad j = \overline{1,m}, \tag{4.5}$$

та другі моменти:

$$\hat{\mu}_{20_j} = \hat{\sigma}^2_T = \frac{1}{n}\sum_{i=1}^{n}\left(t_i - \overline{t}\right)^2, \qquad \hat{\mu}_{02_j} = \hat{\sigma}^2_{A_j} = \frac{1}{n}\sum_{i=1}^{n}\left(a^i_j - \overline{a}_j\right)^2, \qquad j = \overline{1,m},$$
$$\hat{\mu}_{11_j} = \hat{\mathrm{cov}}(T, A_j) = \frac{1}{n}\sum_{i=1}^{n}\left(t_i - \overline{t}\right)\left(a^i_j - \overline{a}_j\right), \qquad j = \overline{1,m}, \tag{4.6}$$

можна обчислити коефіцієнт парної кореляції:

$$\hat{\rho}_j = \frac{\hat{\mathrm{cov}}(T, A_j)}{\hat{\sigma}_T \hat{\sigma}_{A_j}} = \frac{\sum_{i=1}^{n}\left(t_i - \overline{t}\right)\left(a^i_j - \overline{a}_j\right)}{\sqrt{\sum_{i=1}^{n}\left(t_i - \overline{t}\right)^2 \sum_{i=1}^{n}\left(a^i_j - \overline{a}_j\right)^2}}, \; j = \overline{1,m}, \tag{4.7}$$

та коефіцієнти наближення лінійних регресій

$$\hat{\beta}_{1_j} = \hat{\rho}_j \frac{\hat{\sigma}_{A_j}}{\hat{\sigma}_T}, \qquad \hat{\beta}_{2_j} = \hat{\rho}_j \frac{\hat{\sigma}_T}{\hat{\sigma}_{A_j}},$$
$$\hat{\alpha}_{1_j} = \overline{t} - \hat{\beta}_{1_j}\overline{a}_j, \quad \hat{\alpha}_{2_j} = \overline{t} - \hat{\beta}_{2_j}\overline{a}_j, \quad j = \overline{1,m}\ . \tag{4.8}$$

Визначивши коефіцієнти (4.8), побудуємо кореляційну залежність між a_j^i та t_i за формулою

$$a_j = \hat{\alpha}_{1_j} + \hat{\beta}_{1_j} t, \qquad t = \hat{\alpha}_{2_j} + \hat{\beta}_{2_j} a_j. \tag{4.9}$$

Формули (4.5) – (4.9) становлять основу алгоритму оптимізації роботи пошукової системи стосовно вирішення завдань оновлення та індексації інформації в інформаційних системах.

4.3.1. Побудова таблиць кореляційних залежностей даних

Припустимо, що стосовно оновлення інформації (A_1, A_2, A_3,) на вузлі у моменти часу t = (0,3,4,10) існує певний набір даних (табл. 4.3–4.4):

Таблиця 4.3

			t			
			0	3	4	10
A	A_1	a_1^i	1	1	1	1
	A_2	a_2^i	0	0	1	1
	A_3	a_3^i	1	1	0	0

Таблиця 4.4

Зведені дані

Ni	t	$t^i - \bar{t}$	$(t^i - \bar{t})^2$	a_1^i	$a_1^i - \bar{a}_1$	$(a_1^i - \bar{a}_1)^2$	a_2^i	$a_2^i - \bar{a}_2$	$(a_2^i - \bar{a}_2)^2$	a_3^i	$a_3^i - \bar{a}_3$	$(a_3^i - \bar{a}_3)^2$
1	0	-4,75	22,5625	1	0	0	0	-0,5	0,25	1	0,5	0,25
2	3	-1,75	3,0625	1	0	0	0	-0,5	0,25	1	0,5	0,25
3	6	10,25	1,5625	1	0	0	1	00,5	0,25	0	-0,5	0,25
4	10	50,25	27,5625	1	0	0	1	00,5	0,25	0	-0,5	0,25
∑	19	54	54,75	4		0	2		1	2		1

Тоді результати розрахунку кореляційних характеристик, здійснені за формулами (4.5)–(4.8), будуть такими:

$$\hat{\sigma}_T^2 = 13,69; \quad \hat{\sigma}_{A_1}^2 = 0; \quad \hat{\sigma}_{A_2}^2 = 0.25; \quad \hat{\sigma}_{A_3}^2 = 0.25;$$

$$\hat{\mu}_{11_1} = 0; \quad \hat{\mu}_{11_2} = 1.63; \quad \hat{\mu}_{11_3} = -1.63; \tag{4.10}$$

$$\hat{\rho}_1 = 0; \quad \hat{\rho}_2 = 0.88; \quad \hat{\rho}_3 = -0.88;$$

$$\hat{\beta}_{1_1} = 0; \quad \hat{\beta}_{1_2} = 6.5; \quad \hat{\beta}_{1_3} = -6.5;$$

$$\hat{\alpha}_{1_1} = 1; \quad \hat{\alpha}_{1_2} = -30.4; \quad \hat{\alpha}_{1_3} = 31.4;$$

Кореляційні залежності частоти оновлення кожного з видів інформації від часу наведено на рис. 4.5.

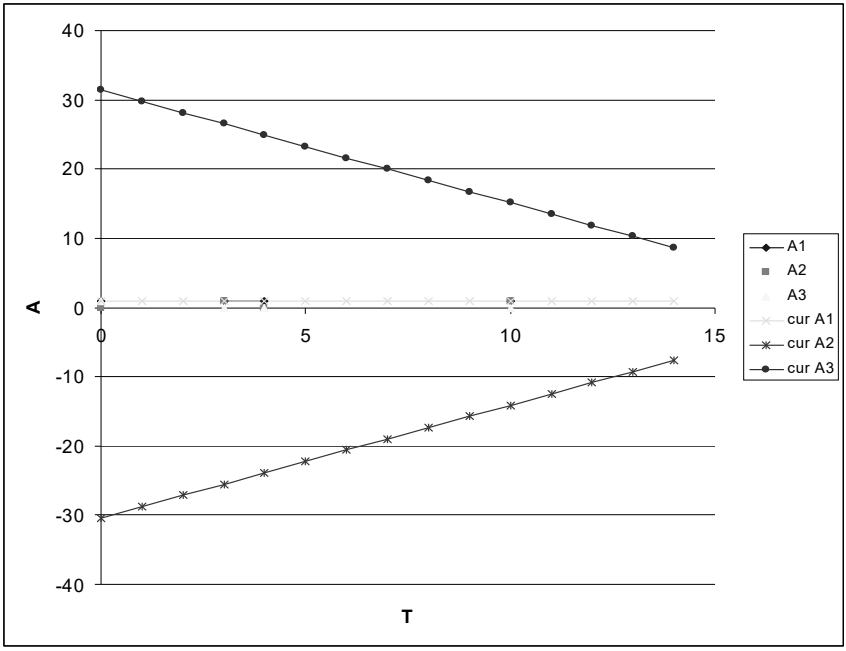


Рис. 4.5. Кореляційна залежність частоти обновлення

Як випливає, потреба в оновленні інформації A_2 зменшується в часі, а інформації A_3, навпаки, зростає. Потреба оновлення інформації A_1 є постійною.

4.3.2. Метод прискореного пошуку даних з символьним представленням

Вимоги до рівня професійної підготовки оперативного й адміністративного персоналу зумовили появу нових та розвиток наявних різноманітних форм навчання, наприклад, корпоративної, дистанційної, електронної тощо. Окрім того, стрімкий розвиток новітніх інформаційних технологій (ІТ) спричинив широке їх використання практично в усіх (традиційних та нетрадиційних) формах навчання. Тим паче, стали з'являтися та розвиватися наукові концепції використання ІТ в навчальному процесі. З урахуванням цього з'явилася й активно впроваджується в практичне використання

коцепція систем управління навчанням. У загальному випадку її основним завданням є спрощення навчального процесу як з боку викладача, так і з боку студента, а також спрощення управління безпосередньо навчальним процесом [52, 54]. Залежно від форми навчання основна скерованість системи може зміщуватися в бік вирішення одного чи другого завдання.

Розвиток концепції системи управління навчанням сукупно з розвитком комп'ютерної техніки сприяв появі теорії архітектур таких систем. Так, зокрема, системи, які володіють достатньо широкими можливостями та використовують технології комп'ютерних мереж, будуються за клієнт-серверною технологією. Типова схема зображена на рис. 4.6 [54].

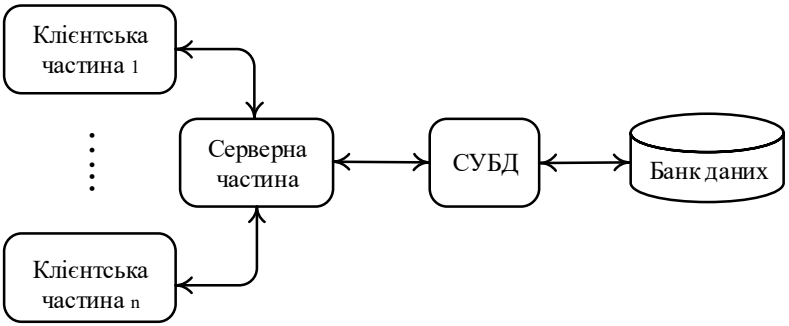


Рис. 4.6. Типова схема системи управління навчанням

Як бачимо, основою таких систем є система управління БД. У більшості випадків БД системи містить навчальну інформацію, інформацію про учасників навчального процесу, нормативні документи тощо [52, 54].

У міру використання системи розміри БД значно зростають і виникає проблема пришвидшення вибірки інформації зі СД. Ця проблема особливо актуальна у випадку використання розподілених та віддалених СУБД. Оскільки якість транспортних засобів передавання досить великих обсягів інформації є низькою (головно у випадку використання Інтернету), а її покращення потребує значних коштів, вирішення описаної проблеми полягає в оптимізації роботи з БД [51].

4.4. Процеси вибірки і пошуку даних

Сьогодні існує досить багато засобів оптимізації процесу вибірки чи пошуку даних із СУБД. Усіх їх умовно можна класифікувати на певні групи:

1. Засоби безпосередньо СУБД (тобто алгоритми вибірки чи пошуку даних, які вбудовані в саму СУБД).

2. Оптимізаційні засоби [112]. Йдеться про попередню логічну, оптимізаційну обробку складних запитів до СУБД.

3. Апаратні засоби передбачають фізичне оновлення (Upgrade) пристроїв, на яких працює СУБД.

4. Логічні засоби — це засоби, які умовно також можна розподілити на такі дві категорії: засоби першої категорії передбачають внесення в СУБД додаткової інформації, яка забезпечує пришвидшення пошуку даних; засоби другої категорії ґрунтуються на методах передбачення даних.

5. Інші засоби. Сюди належать засоби, які не входять у жодну з описаних груп. Як приклад можна навести логічну оптимізацію диску, на якому зберігається БД.

Оскільки наявні системи управління навчанням використовують стандартні комерційні реляційні СУБД, практично не існує жодної можливості розглядати оптимізаційні алгоритми самої СУБД. Стосовно групи оптимізаційних засобів варто зауважити, що вони є досить повно досліджені та описані в науковій літературі [112]. Показано, що не існує єдиного ефективного механізму покращення SQL-запиту, який забезпечував би найкращу (з точки зору швидкості) вибірку даних у середовищі будь-якої СУБД. Для вирішення цього завдання потрібно проводити цілий набір оптимізаційних дій (логічна оптимізація, семантична тощо). Проведені в сукупності дії дають прийнятний результат лише у випадку баз даних великого об’єму. При цьому, однак, поля в таблицях БД повинні бути або числовими, або символьними короткої довжини. Коли ж поля таблиці бази є довгими рядками або бінарними наборами даних, ефективність логічних засобів суттєво знижується. Це зумовлено значними витратами ресурсів самої СУБД при порівнянні великих масивів одного екземпляра даних.

Засоби фізичного оновлення апаратних компонентів комп'ютера, на якому працює СУБД, ми не розглядаємо. Додамо лише, що недотримання вимог, зумовлених виробником конкретної СУБД, може призвести до суттєвого сповільнення роботи сервера БД. Засоби, які належать до групи інших, тут також не розглядаємо, оскільки зазвичай вони залежать від персоналу, який адмініструє дану СУБД.

Логічні засоби, на відміну від оптимізаційних, виявляють свою ефективність у випадку, коли поля таблиць є великими за розміром. Так, зокрема, прогностичні засоби логічної групи з достатньою імовірністю дають змогу спрогнозувати дані, які будуть потрібними в конкретний момент часу. Проте, зауважимо, ці засоби меншою мірою використовуються для пришвидшення пошуку. Значно більшою мірою вони використовуються під час вирішення завдань розпізнавання логічного змісту, закладеного в дані.

Оскільки таблиці бази даних систем управління навчанням можуть містити змішані набори даних, найефективнішим є використання цілого комплексу оптимізаційних засобів — від оптимізаційних до інших. Здійснимо розробку ефективного логічного засобу першої категорії, який у великих за обсягом БД давав би можливість пришвидшити пошук та вибірку інформації при існуванні полів великої довжини.

Характеристика методу пошуку на основі нечітких мір. Пошук інформації є найпростішою задачею теорії алгоритмів і може бути представлений як пошук інформації, яка збережена з конкретним ідентифікатором. Загалом вважається, що існує набір записів, і завдання полягає в знаходженні кожного з них. Допускається, що кожен запис має спеціальне поле, яке називається ключем, атрибутом чи характеристикою запису. Це поле повинно однозначно (або з деякою похибкою у випадку розпізнавання) ідентифікувати запис, але при цьому мати значно менший розмір. Полів, які задіються в однозначному визначенні запису, може бути декілька (випадок поділу ключів на первинні та вторинні і т. д.). Але тоді однозначна ідентифікація визначається комбінацією цих полів. Тобто у загальному випадку вважається, що в кожному записі міститься декілька

атрибутів, і необхідно віднайти усі записи з деякими значеннями цих атрибутів.

На основі сформульованого завдання пошук запису зводиться до пошуку за ключем чи характеристикою, які за довжиною є значно меншими від самого запису.

Метод прискореного пошуку. Будь-яке слово чи фраза, яку можна розглядати як набір слів, надалі S, складається з символів (s_i), кожен з яких має свій код (d_i) у таблиці UNICODE. Це означає, що для кожного S, яке подається у вигляді суми символів і може бути подане у вигляді

$$S \to \sum_{i=1}^{m} s_i \,, \tag{4.11}$$

тут m — кількість символів у слові чи фразі, можна побудувати дискретний сигнал:

$$S \to \sum_{i=1}^{m} (d_i, s_i) \,. \tag{4.12}$$

Якщо додатково розглядати ще й імовірність появи символа, то S можна подати у вигляді:

$$S \to \sum_{i=1}^{m} (p_i, n_i) \,, \tag{4.13}$$

де n_i — порядковий номер символа у слові; p_i — імовірність появи символа, яка є наперед визначеною статистичною характеристикою кожного символа.

Тобто слово стає функцією від коду символа та імовірності появи даного символа:

$$S \to f(p_i, n_i) \,. \tag{4.14}$$

Функціональну залежність (4.14) можна посилити, якщо n_i-му поставити у відповідність код символа, тобто n_i розглядати як лінійну функцію від коду:

$$n_i = n_i(d_i) \,. \tag{4.15}$$

На основі (4.14) введемо в розгляд посимвольну інформативність у вигляді:

$$P' = \int_l pdl\,, \tag{4.16}$$

де $l = l(n_i)$ — траєкторія слова чи фрази. Беручи до уваги (4.16), введемо першу ознаку — питому посимвольну інформативність слова чи фрази, яка визначатиметься за формулою:

$$P = \frac{P'}{L} = \frac{1}{L}\int_l pdl, \tag{4.17}$$

де $L = L(l)$ – довжина S.

Оскільки окрім p_i існує ще одна статична характеристика, а саме g_i — імовірність появи окремих складів (у цьому випадку двосимвольних), подібно до (4.16) та (4.17) можна отримати другу характеристику для S:

$$G' = \int_{l_g} gdl_g\,; \tag{4.18}$$

$$G = \frac{G'}{L} = \frac{1}{L}\int_{l_g} gdl_g\,, \tag{4.19}$$

де $l_g = l_g\ (n_i)$ — траєкторія слова відносно g_i. Характеристику G назвемо питомою сполучною характеристикою S.

Таким чином, кожному слову чи фразі зіставляється характеристична пара:

$$S \to (P, G)\,, \tag{4.20}$$

де P, G, які є цілими додатними числами, повинні зберігатися в базі поряд з кожною фразою. Власне, на основі них і прийматиметься рішення про збігання шуканої фрази з поточною в базі. Рішення про збіжність пропонується приймати на основі введеної міри подібності:

$$\mu\left(S_1; S_2\right) = \mu\left(\left(P_1, G_1\right); \left(P_2, G_2\right)\right) < \varepsilon\,, \tag{4.21}$$

де ε — точність збігання. Міру (4.21) можна розглядати як набір мір, тобто:

$$\mu\left(\left(P_1,G_1\right);\left(P_2,G_2\right)\right)=\begin{cases}\mu\left(P_1;P_2\right)<\varepsilon_P;\\ \mu\left(G_1;G_2\right)<\varepsilon_G,\end{cases} \tag{4.22}$$

де ε_P, ε_G є мірами збіжності за окремими характеристиками P і G. У загальному випадку вони можуть бути рівними. Кожна з мір в (4.22) — звичайний модуль, тобто:

$$\mu\left(P_1;P_2\right)=\left|P_1-P_2\right|<\varepsilon_P;\qquad \mu\left(G_1;G_2\right)=\left|G_1-G_2\right|<\varepsilon_G. \tag{4.23}$$

Реалізації алгоритму посимвольного пошуку. Для апробації запропонованого алгоритму було попередньо проведено статистичний аналіз частоти зустрічання українських символів та пар українських символів. Розглядався текст завдовжки 289387 символів. На його основі отримано частоти зустрічання українських літер рис. 4.7.

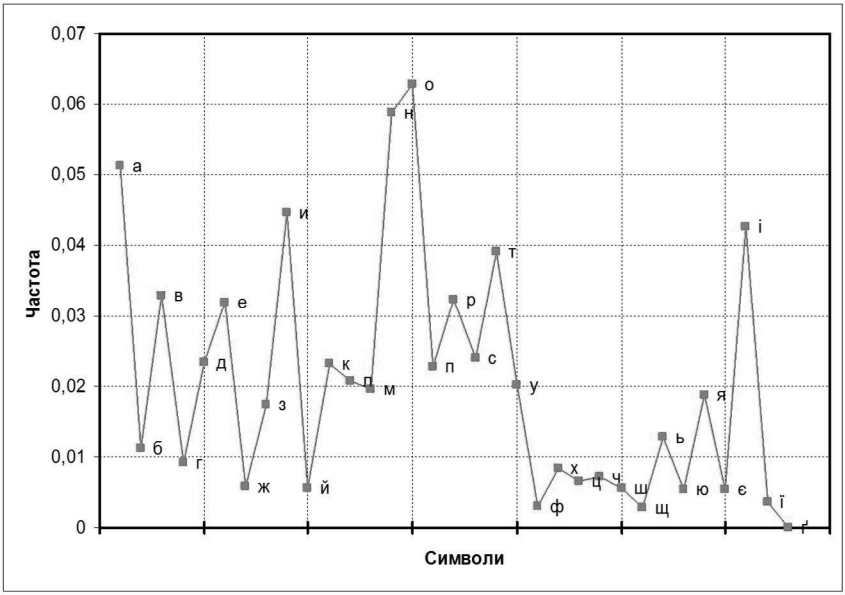


Рис. 4.7. Частота зустрічання українських символів

Для пар українських символів була зроблена вибірка, до якої ввійшли пари, частота зустрічання яких перевищувала значення 0.003. Графік частоти пар українських символів, які трапляються найчастіше, наведено на рис. 4.8.

Для практичної реалізації запропонованого алгоритму на основі отриманих результатів була побудована база даних у СУБД MySql. Побудована база даних містила одну таблицю (розмір її становив 12316 записів), яка складалася з чотирьох полів, а саме:

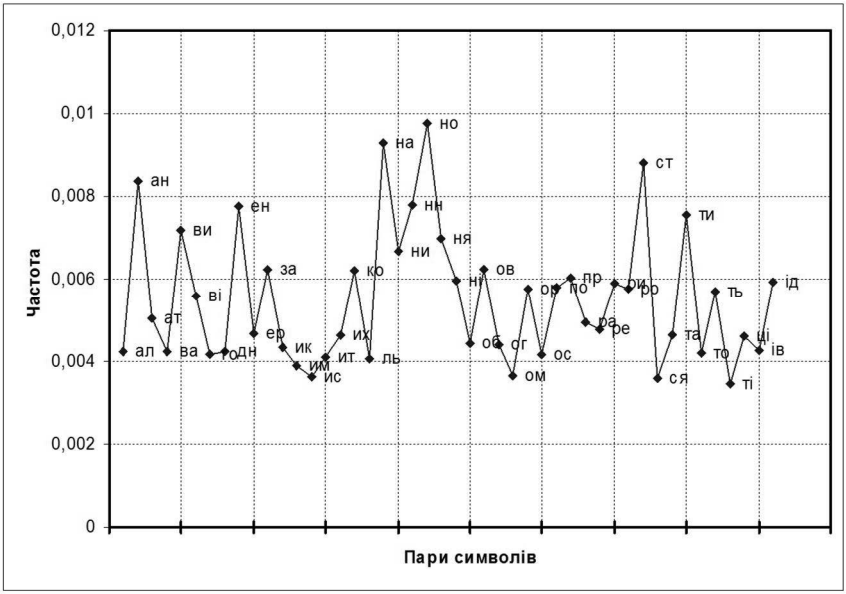


Рис. 4.8. Частота найуживаніших пар українських символів

— Code — цілого (довжиною 11 символів);
— Produce — символьного (довжиною 75 символів);
— Realized — символьного (довжиною 50 символів);
— FreqSymb — цілого (довжиною 4 символи);
— FreqPair — цілого (довжиною 4 символи).

Для спрощення практичної реалізації вибірка проводилася полем Produce, інші поля (за винятком двох останніх) не були ключовими. Власне, два останніх поля таблиці містили попередньо обчи-

слені на основі запропонованої методики характеристики P і G. Дані характеристики, які є додатними дійсними числами, були приведені до цілого числа з проміжку [0; 9999] шляхом їх множення на додатковий ваговий коефіцієнт, що дорівнює 10000. Тобто в базу даних як характеристики P і G фактично заносились:

$$P = \text{intr}(k \cdot P); \qquad G = \text{intr}(k \cdot G), \tag{4.24}$$

де k = 10000 — додатково введений ваговий коефіцієнт.

Це зроблено з метою пришвидшення пошуку, який є найоптимальнішим для цілих чисел. Кількість розрядів у ваговому коефіцієнті вибрано не випадково. Для даної бази вже при такому ваговому коефіцієнті було досягнуто 100% правильності вибірки, тобто забезпечувалась абсолютна точність вибірки. При меншому ваговому коефіцієнті необхідно задіювати повторну вибірку, щоправда, з меншою кількістю записів. Останній випадок наразі ми не досліджували.

У випадку вибірки без урахування коефіцієнтів P і G поля FreqSymb і FreqPair не вибирались.

Результи вибірки наведено на рис. 4.9–4.10 і в табл. 4.5. Час вибірки замірявся від початку посилання запиту на вибірку (зв'язок з базою був попередньо встановлений) і до моменту отримання результатів без виведення їх на екран.

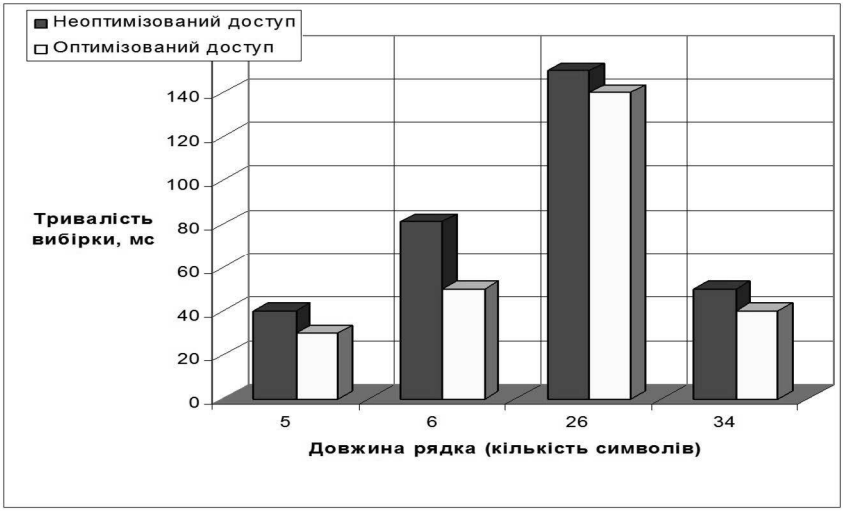


Рис. 4.9. Тривалість вибірки рядків з різною довжиною

Таблиця 4.5

Результати вибірки та ефективність у відсотковому значенні

Слово	Кість слів	Довжина слова в символах	Час доступу в мс		Характерис-ки		Відсоток ефективності %
			Неоптимізований доступ	Оптимізований доступ	Посимвольна	Сполучна	
Факси	2	5	40	30	2923	0	25
Горіхи	1	6	81	50	3334	165	38
Роботи будівельно-ремонтні	125	26	150	140	3524	179	7
Послуги еміграційно-консультаційні	5	34	50	40	3050	161	20

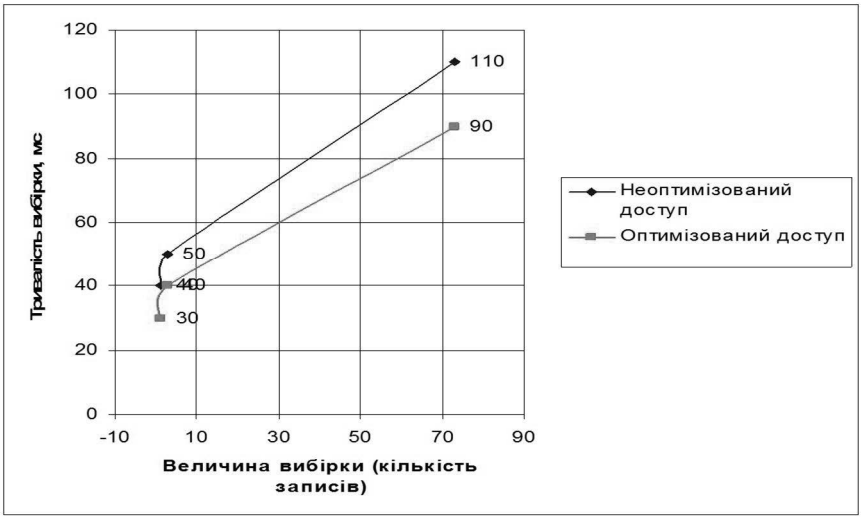


Рис. 4.10. Результати вибірки слів завдовжки 25 символів

Як зображено, запропонований алгоритм дає менший час вибірки, проте його ефективність залежить як від величини вибірки, так і від довжини зустрічного поля (в нашому випадку Produce), на основі якого здійснювалась вибірка.

РОЗДІЛ 5
ІНФОРМАЦІЙНІ ТА ВЕБ-ТЕХНОЛОГІЇ СТВОРЕННЯ НАВЧАЛЬНИХ СИСТЕМ ДЛЯ ПІДВИЩЕННЯ КВАЛІФІКАЦІЇ АДМІНІСТРАТИВНОГО ПЕРСОНАЛУ

5.1. Програмні засоби для керованого навчання

Навчальні системи для підвищення кваліфікації адміністративного персоналу являють собою програмні засоби перенесення в ієрархічну систему курсу наявних електронних документів (текстових та графічних). Більшість із таких засобів для ефективного використання потребує базових знань *HTML* та порівняно невеликого досвіду програмування.

Програмні засоби для керованого навчання типово мають можливості реєстрації студента в стандартному *Java*-браузері, централізовану базу даних плану навчання зі зв'язками (лінками) на внутрішні або зовнішні веб-ресурси, тестові середовища з динамічною генерацією питань, групи дискусій та інтегровану електронну пошту.

Courseinfo (http://product.blackboard.net/courseinfo/) — цей продукт компанії *Blackboard* протягом останніх двох років замінив *Topclass* та *WebCT* у багатьох кампусах вищих закладів освіти. Його перевагами є агресивна цінова стратегія та порівняна легкість використання. *Courseinfo* є класичною, базованою на шаблонах системою управління курсами (*Course Management System*). Може використовуватись на платформах *Unix* та *Windows.*

Lotus LearningSpace (http://www.lotus.com/products/learningspace.nsf) є програмою-додатком, написаною на *Lotus Notes*. Орієнтована передусім на корпоративних користувачів.

WBT Topclass (*http://www.wbtsystems.com/*) — найстарший представник у цій категорії. *Topclass* був найзавершенішим продуктом, проте втратив свій вплив у Північній Америці з появою продукту *Courseinfo*.

Total Knowledge Management (*TKM*, раніше *Generation 21,* — http://www.gen21.com) поділяє створення курсу на невеликі «частини» інформації — динамічні навчальні об'єкти (*Dynamic Learning*

Objects). Ці частини навчання можуть бути зв'язані в необмеженій кількості конфігурацій для курсів різного рівня знань, уміння, слухачів та форматів. Використовуючи тестування для визначення рівня знань слухача, вибирають лише необхідні для заповнення певних прогалин в освіті частини знань. Крім того, користувачі мають можливість здійснювати пошук в усій базі даних інформації, що необхідна для розв'язання задачі чи отримання відповіді на запитання.

Avilar WebMentor (http://www.avilar.com/) — середовище підготовки з повним набором засобів для розробки, адміністрування та донесення інформації з використанням *Web* через Інтернет, Інтранет та Екстранет. Цей продукт є одним з декількох, що дає змогу створювати модульну інформацію з можливістю повторного використання.

Wineducation's The Learning Manager (www.wineducation.com/-TLM.htm) — продукт на платформі *Windows NT* з програмними характеристиками, порівнюваними з характеристиками систем управління навчанням, але без об'єктно-орієнтованої бази даних.

COSE (*http://www.staffs.ac.uk/COSE/) – Creation of Study Environments* (*COSE*) — програмна система, що надає систему та засоби для створення навчальних середовищ. Дає змогу створювати середовища, що використовують матеріал з широкого діапазону носіїв та забезпечують механізми для підтримки, забезпечення зворотного зв'язку, спільної роботи та самотестування. Для створення цих середовищ не потрібно знати мови розмітки та програмування.

5.1.1. Програмні засоби для синхронного навчання

Ключовими характеристиками систем цього класу є використання браузера, передавання аудіоінформації через протокол *IP*, спільне використання програмних засобів, дошка оголошень, елементи тестування в режимі онлайн. Деякі з продуктів цієї категорії для кращого подання та засвоєння навчального матеріалу використовують відео типу «*talking head»* (потокове відео з зображенням інструктора).

Classpoint (www.wpine.com/Products/ClassPoint/index.html). Продукт забезпечує передавання голосу через Інтернет. Поєднує версію клієнта компанії *WhitePine CU-See Me* (з компонентами аудіо та відео) із ПЗ для серверів *Meetingpoint Control Server*.

Placeware Conference Center (http://www.placeware.com/). За допомогою середовища *PlaceWare Auditorium* користувач показує слайд-шоу та відповідає на запитання аудиторії. Можливий також зворотний зв'язок, голосування та оцінка виступу.

Astound (http://www.astound.com/main.htm) — продукт забезпечує передавання голосу через Інтернет. Забезпечує сервіс голосової комунікації, спільного використання програмних засобів та слайд-шоу.

Educata Classroom (http://educata.com/) — продукт забезпечує передавання голосу через Інтернет, дає можливість інструкторам створювати, інтегрувати, налагоджувати та передавати навчальний матеріал, використовуючи єдине гнучке середовище.

5.1.2. Програмні засоби групового навчання

Програмні засоби групового навчання складаються із систем персональних середовищ спільної роботи та систем колективної роботи. *Персональні середовища спільної роботи* (*Personal Collaborative Environments*) є новою категорією програмних засобів, які дають змогу особам спілкуватися між собою або в межах невеликих груп.

Microsoft Messenger (*http://messenger.msn.com/*) — внесок корпорації *Microsoft* у галузь програм обміну повідомленнями.

ICQ (*http://www.icq.com/*) — одна з найпопулярніших програм обміну повідомленнями.

AOL's Instant Messenger (*http://aim.aol.com/*) використовується клієнтами *America On-Line* в Північній Америці.

Tribal Voice's PowWow (*http://www.powwow.com/*) — доступна у двох формах: версія для приватних користувачів забезпечує повідомлення про наявність та обмін текстовими повідомленнями; версія для приватних мереж надає додаткові можливості як спільне

використання веб-сайта, систему дискусій та передачу файлів. Адміністратори отримують можливість реєстрації сесій, управління безпекою та масштабування.

Навчальні портали є видом персональних середовищ спільної роботи (*Personal Collaborative Environments*), який поєднує студентські сервіси та механізм створення спільноти (*community*) за допомогою веб-систем, подібно до пошукових порталів типу *Yahoo* та *Lycos*. Навчальні портали широко використовуються в корпоративних навчальних центрах.

Jenzabar (*http://www.jenzabar.com/*) — система дає змогу створювати персоналізовані календарі, інформаційні каталоги студентів, інформацію про курси з використанням веб-інтерфейсу.

Mascot Network (*http://www.mascotnetwork.com/*) орієнтований на інтернет-мережу кампуса-сервіс, комбінацію комунікаційних засобів, інформацію про кампус, персоналізовану інформацію для студентів, викладачів та працівників.

Засоби колективної роботи *Teamware* є новим поколінням засобів, призначених для підтримки роботи віртуальних груп. У типову конфігурацію входить: система дискусій, текстовий чат та система обміну файлами. Розширена конфігурація містить розклад роботи групи та системи управління проектами. Засоби *Teamware* потребують певного досвіду адміністрування проектними роботами після інсталяції для забезпечення простоти створення, редагування та закінчення проектів.

Instinctive Technology's eRoom (*http://www.instinctive.com/*) — найдовершеніший продукт у категорії.

Lotus Quickplace (*http://www.lotus.com/home.nsf/tabs/quickplace*) — продукт для роботи з текстовими документами та для співпраці в межах групи.

Involv Intranet (*http://www.involv.com/*) створено на платформі *Lotus Domino,* виконує функції інформаційного бюро.

Програмні засоби, що застосовуються у навчанні з використанням Web, класифікуються за такими функціональними категоріями:

1. Система донесення навчального матеріалу (*Educational Delivery System*) — програмний продукт чи сім'я, що забезпечує

донесення навчального матеріалу та взаємодію слухачів з використанням *Web*, але не обов'язково виконання адміністративних задач.

2. Система управління курсами (*Course Management System* (*CMS*)) — програмний продукт, що об'єднує систему донесення навчального матеріалу та інтегровані засоби для оцінки результатів навчання окремих слухачів або їх груп. Прикладами систем *CMS* є *BlackBoard's CourseInfo* та *WebCT*.

3. Система управління навчанням (*Learning Management System* (*LMS*)) — програмний продукт, який забезпечує слухача інтегрованою інформацією з курсу та про виконану роботу згідно з планом навчання. *LMS*-системи використовуються в департаментах великих корпорацій та корпоративних університетах.

5.1.3. Засоби розроблення навчальних систем з використанням Web

DazzlerMax (*http://www.dazzler.net/dazzler/*) — засіб швидкого розроблення матеріалу, який дає змогу швидко створити «інтелектуальний» навчальний матеріал. Підтримка об'єктно-орієнтованої парадигми та парадигми «*drag and drop*» надає всю потужність засобів розробки для програмістів, не вимагаючи знання мов програмування. Матеріал подається у вигляді *Java*-аплета, що забезпечує крос-платформенність та не потребує додаткових засобів для його перегляду. Використовується для навчальних середовищ із компонентами тестування, спілкування та інтерактивності.

Link Systems TechBoard (*http://www.link-systems.com/*) — потокова система повідомлень для технічних дискусійних груп. Зручні засоби управління мають інтуїтивний інтерфейс.

Link Systems WWWhiteboard (*http://www.link-systems.com/*) — це інтерактивна графічна програма-чат, яка надає можливість спілкування в реальному часі. Забезпечує підтримку наукових та математичних символів, геометричних примітивів та графічних зображень.

Macromedia (*http://www.macromedia.com/*) — виробник програмних засобів створення інтерактивного навчального матеріалу *Authorware™* та *Director™*.

Netsage (*http://www.netsage.com/*) — розробник інтелектуального анімованого ПЗ для персоналізації комп'ютерних середовищ *Sages*. Ці програмні агенти використовуються у навчанні з використанням *Web*, системах електронної комерції та системах підтримки роботи з персоналом.

WebFuse (*http://webfuse.cqu.edu.au/*) — засіб для створення веб-аудиторій.

Для зберігання та донесення навчального матеріалу доцільно використовувати системи управління базами даних. Платформа для реалізації навчальних систем на базі *Web* повинна забезпечувати доступ будь-якого клієнта до даних різних типів: текст; відео; аудіо; XML; файли; структуровані дані.

Особливо важливою характеристикою є можливість зберігання мультимедійних даних, використання яких в освіті та інших сферах діяльності стрімко розширюється.

Промислові стандарти для навчання з використанням Web. Для забезпечення взаємодії та вільного обміну матеріалом, що існує в різних програмних середовищах, створюються різні групи для розробки стандартів. Розробники та комітети стандартизації визнають потребу розроблення онлайн матеріалу за тими самими принципами, за якими розробляється програмне забезпечення комп'ютерів, а саме — використовуючи технологію об'єктно-орієнтованого програмування. Для описання «навчальних об'єктів» об'єднання *Anvanced Distributed Learning Network* (*ADL*) розробляє стандарти «моделі опису об'єкта курсу спільного використання» — *Sharable Course Object Reference Model* (*SCORM*).

Aviation Industry CBT (*Computer-Based Training*) *Committee* (*AICC*) (*http://www.aicc.org/*). Авіаційна промисловість є давнім користувачем розподілених навчальних інформаційних систем. Починаючи з 1980-х років, в авіаційній промисловості була розроблена множина стандартів програмного забезпечення онлайн-курсів, разом з рекомендаціями *AICC Guidelines & Recommendations* (*AGR's*). *AICC* розробляє директиви щодо розробки й тестування систем *CBT* та інших навчальних технологій в авіаційній промисловості, проте ці стандарти визнані в усій галузі розроблення систем *CBT*.

Advanced Distributed Learning Network (*ADL*) (*http://adlnet.org/*) — це підрозділ *IMS*, який фінансується урядом США. Департамент оборони (*Department of Defense, DoD*) та офіс Білого дому з науки та техніки (*White House Office of Science and Technology Policy OSTP*) ініціювали створення *Advanced Distributed Learning Network* (*ADL*). Призначенням цієї мережі є:

— розробка директив для широкомасштабного створення та впровадження ефективного розподіленого навчання;
— визначення та підтримка бізнес-моделей і економічних стимулів для покупців та розробників розподіленого навчання;
— організація мережі спільноти покупців навчальних систем;
— заохочення спільної роботи організацій, що розробляють засоби для навчання;
— виявлення технічних проблем, що гальмують розвиток навчальних систем та ініціювання спільних досліджень і розроблення програм розв'язання цих проблем;
— поширення досвіду та прискорення розробки нових об'єктно-орієнтованих відкритих середовищ для *Advanced Distributed Learning.*

IEEE Standard for Computer-Based Learning (*http://www.educause.edul*) — *IEEE* є однією з найбільших організацій з розробки стандартів (*Standards Development Organizations, SDO*), акредитованою в Американському національному інституті стандартів (*American National Standards Institute, ANSI*). Розробляє стандарти для «освітньої технології» (*Learning Technology*) *IEEE Learning Technology Task Force*. Передбачається об'єднання специфікацій з *AICC*, *IMS* та *ADL* в єдиний стандарт *IEEE*.

National Center for Supercomputing Applications — Educational Division. (*http://www.ncsa.uiuc.edu/edu/html/lete.html*) *NCSA* є давнім лідером у розробці програмного забезпечення для Інтернету, і тепер розробляє ряд проектів, що можуть вплинути на промислові стандарти в галузі навчальних систем.

5.2. Структурний аналіз систем управління навчальним процесом

5.2.1. Інформаційні та телекомунікаційні технології забезпечення навчального процесу

Основними напрямами використання ІСТ (інформаційних та телекомунікаційних технологій) в навчальному процесі є [50]:

— поширення інформації, зокрема публікацій;
— комунікація між викладачами і студентами та студентами між собою;
— організація роботи в групах — спільні дискусії, проекти тощо;
— організація пошуку інформації (ресурсів) — пошукові машини, мультимедійні бази даних тощо;
— специфічні потреби навчального процесу — інтерактивне консультування, завдання для самостійної роботи, тести, відеоконференції в рамках лекцій;
— інтегрована система організації та ведення навчального процесу — системи управління, створеної з використанням веб-технологій.

За оцінками західних дослідників, однією з основних причин застосування ІСТ у навчальному процесі є зміни в демографічних характеристиках контингенту студентів. На сьогодні, окрім традиційного контингенту віком від 18 до 23 років, які вступають в університети відразу після закінчення середньої освіти і навчаються на стаціонарі, до вищих навчальних закладів приходять нові категорії студентів, що прагнуть здобути вищу освіту в зрілому віці та на основі різноманітних форм попередньої освіти, часто на базі виробничого досвіду. Крім того, такі студенти зазвичай хочуть мати можливість поєднувати навчання із сімейним та професійним життям. У зв'язку з цим гостро постає проблема забезпечення якнайбільшої гнучкості пропонованих навчальних програм.

Бажана гнучкість може бути реалізована на основі низки підходів:

— гнучкість за місцем здобуття освіти — можливість частково навчатися поза межами основного кампусу університету;

— гнучкість у навчальних програмах — можливість вибору конкретних дисциплін у зв'язку з попереднім навчальним та професійним досвідом;
— гнучкість у видах взаємодії протягом навчання — не всі студенти повинні працювати в межах груп, не всі члени групи в процесі виконання завдання мають знаходитися поряд;
— гнучкість у формах комунікації в межах навчальної дисципліни — студенти та викладачі, окрім традиційного способу спілкування «обличчям до обличчя», повинні мати інші можливості комунікування;
— гнучкість у виборі навчально-методичних матеріалів — розширити доступ до засобів інформації.

5.2.2. Веб-орієнтовані системи управління навчальним процесом

Сьогодні існує велика кількість електронних систем управління навчанням (надалі СУН). Переважна їх кількість орієнтована на Web і технології, які реалізовуються засобами Web. Усі такі системи класифікуються за певними функціональними категоріями.

1. Система донесення навчального матеріалу (Educational Delivery System) — програмний продукт чи сімейство, яке забезпечує донесення навчального матеріалу та взаємодію слухачів з використанням Web, але не обов'язково виконання адміністративних задач. Прикладами таких систем є Placeware Auditorium та Centra's Conference.

2. Система управління курсами (Course Management System (CMS)) — програмний продукт, що об'єднує систему донесення навчального матеріалу та інтегровані засоби для оцінки результатів навчання окремих слухачів або їх груп. Прикладами систем CMS є BlackBoard's CourseInfo та WebCT.

3. Система управління навчанням (Learning Management System (LMS) — програмний продукт, який забезпечує слухача інтегрованою інформацією з курсу та про виконану роботу згідно з планом навчання. LMS-системи використовуються в департамен-

тах підготовки великих корпорацій та корпоративних університетах. Прикладами таких систем є Docent Enterprise та Knowledgesoft Enterprise.

Спільною для систем усіх категорій є концептуальна схема, подана на рис. 5.1.

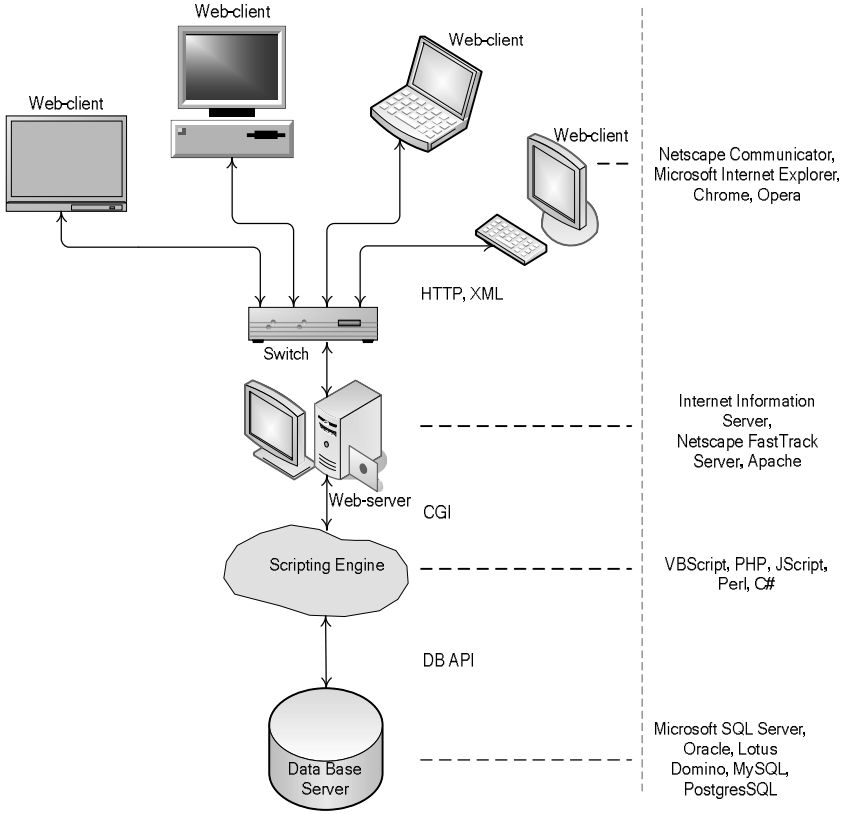


Рис. 5.1. Концептуальна схема управління навчальним процесом

Складовими частинами цієї схеми є:

— ядро, яке базується на веб-сервері, підтримує відкриті Web і прикладні інтерфейси та забезпечує обробку логіки системи через механізм скриптів;

— інформаційне сховище даних, побудоване на базі однієї із сучасних реляційних СУБД з підтримкою розподілених технологій;

— користувацьке середовище, яке через механізм взаємодії з ядром дає змогу користувачеві інтерактивно працювати в системі.

Наявність цих складових зумовлює потребу як проектування системи загалом, так і детального проектування окремих частин, що передбачає моделювання підсистем управління інформації, даних тощо.

5.2.3. Аналіз структури задач управління інформаційними потоками в системах управління навчанням

Розглянемо три основні класи задач, пов'язаних з керуванням інформацією в середовищах СУН.

1. *Моделювання і запити в СУН*. Припустимо, що СУН розглядається як орієнтований граф, вузли якого є веб-сторінками, а хорди — зв'язками між сторінками. Першою задачею є формулювання запитів для пошуку визначених сторінок. При цьому запити можуть будуватись на основі як змісту потрібних сторінок, так і з урахуванням структури зв'язків між цими сторінками. Найпростішим прикладом такої задачі є пошук сторінки на основі слів, що містяться в ній (завдання для пошукових машин). Просте узагальнення такого запиту полягає в використанні складніших предикатів відносно змісту сторінки (наприклад, знайти сторінки, які містять слово «Мама» після зв'язку, який вказує на зображення). Найповнішими у цьому випадку є запити, які задіюють усю структуру системи, починаючи з кореня.

2. *Вибірка й інтеграція інформації*. Деякі СУН можуть розглядатися на тоншому рівні гранулярності, аніж сторінки, як контейнери структурованих даних (набори кортежів, набори об'єктів і т. д.). У зв'язку з цим набувають актуальності дві задачі. Перша задача полягає в тому, щоб фактично здійснювати вибірку даних, представлених у структурованому вигляді (наприклад, набір кортежів) з *HTML* — сторінок, які їх містять. Така задача розв'язується за допомогою набору програм-оболонок (*wrapper*), створення і підтримка яких породжує ряд проблем. Якщо розглядати СУН такого класу, як автономні неоднорідні бази даних, виникає друга задача — формулювання запитів, які потребують інтеграції даних. Друга

задача розв'язується за допомогою систем медіаторів (чи систем інтеграції даних).

Особливо важливого значення задачі інтеграції інформації набувають при впровадженні СУН за умови наявності інших інформаційних систем університету (бухгалтерії, відділу кадрів, студентського відділу тощо). Для успішної інтеграції СУН необхідна підтримка протоколів доступу до Directory-систем — LDAP (Lightweight Directory Access Protocol, ADSI – Active Directory Service Interfaces). Directory-системи (системи каталогів) зберігають інформацію про ресурси мережевого середовища організації (програмні ужитки, файли, принтери, користувачі) та єдиний шлях до опису, пошуку, доступу, керування та захисту інформації про ці ресурси. Саме Directory-системи (Novell eDirectory, Microsoft Active Directory) є стрижневою компонентою при побудові високоефективних інформаційних систем корпорацій, підприємств, університетів.

3. *Розробка і реструктуризація СУН*. Інший аспект використання концепцій і технологій баз даних — розробка і реструктуризація СУН, а також керування ними. На відміну від попередніх двох класів задач, пов'язаних із уже наявними СУН, тут розглядається процес розробки нових систем. Їх проектування може розпочинатися або з деяких вихідних даних (збережених у БД чи у структурованих файлах), або шляхом реструктуризації наявних систем. Виконання цієї задачі потребує використання певних методів моделювання структури веб-сайта (за умови, коли СУН будується на основі Web) і мов реструктуризації даних таким чином, щоб вони задовольняли необхідну структуру. Не розглядатимемо питань, пов'язаних із застосуванням концепцій баз даних стосовно WWW, зокрема кешування і тиражування даних, обробка транзакцій і безпека у веб-орієнтованих СУН.

5.3. Представлення даних для побудови веб-орієнтованих систем управління навчанням

Створення систем для розв'язання кожної з окреслених вище задач потребує вибору якого-небудь методу для моделювання її предметної галузі. Зокрема, необхідно в таких задачах моделювати

саму СУН, структуру системи, внутрішню структуру сторінок, і зрештою, зміст СУН з детальнішим ступенем гранулярності. Розглянемо головні фактори, що характеризують моделі даних, які використовуються у веб-орієнтованих СУН [31].

1. *Графові моделі даних:* як уже зазначалося, для деяких з розглядуваних тут програм необхідно моделювати множини сторінок Web, а також зв'язок між ними, які можуть розміщуватись на декількох або на одному сайті. У такій моделі вузли становлять сторінки Web (або внутрішні компоненти сторінок), а хорди — зв'язки між сторінками. Мітки на хордах можуть розглядатися як імена атрибутів.

2. *Моделі слабкоструктурованих даних*. Другий аспект моделювання даних СУН полягає в тому, що в багатьох випадках структури цих даних не є постійними. Тобто при моделюванні структури веб-сайта веб-орієнтованої СУН не існує заданої наперед якоїнебудь фіксованої схеми. Окрім того, у випадку моделювання даних, які надходять з множини джерел, описи деяких атрибутів можуть різнитись для різних джерел.

Загалом слабкоструктурованими називаються дані, яким властива хоча б одна з таких характеристик.

Схема не задана заздалегідь і може неявно міститися в даних:

— схема порівняно велика (у сенсі обсягу даних) і може часто змінюватися;

— схема є описовою, а не примусовою, тобто вона описує поточний стан даних; порушення цієї схеми все-таки допускаються;

— дані не є строго типованими, тобто для різних об'єктів значення того самого атрибуту можуть мати різні типи.

Моделі слабкоструктурованих даних були побудовані на визначених орієнтованих графах [71, 75]. У моделі слабкоструктурованих даних не накладається жодного обмеження на множину хорд, що виходять з даного вузла в графі, чи на типи значень атрибутів. У зв'язку зі згаданими вище характеристиками слабкоструктурованих даних стає важливою додаткова можливість — запитувати схему (тобто мітки хорд у графі). Така можливість забезпечується в мовах запитів для слабкоструктурованих даних за-

вдяки змінним — хордам, що пов’язані з мітками хорд, але не з вузлами графа.

3. *Моделі даних з веб-конструкціями* є окремим випадком моделей слабкоструктурованих даних, але, на відміну від них, вони містять специфічні для веб-конструкції у представленні даних. Наприклад, у деяких моделях розрізняють відношення унарне, яке ідентифікує сторінки, і бінарне — для зв’язків між сторінками. Крім того, допускається розрізняти зв'язки всередині системи і зовнішні. Важлива причина розрізняти відношення зв'язків полягає в тому, що вони загалом можуть обходитися тільки в прямому напрямку.

До властивостей другого порядку, за якими розрізняються обговорювані тут моделі даних, можна віднести: здатність моделювання деякого порядку на множині елементів у базі даних; моделювання вкладених структур даних; підтримка типів колекцій (множин, мультимножин, масивів).

Важливий аспект мов запитів даних у СУН — необхідність генерувати складні структури в результаті обробки запиту. Наприклад, результатом деякого запиту в системі може бути граф, який моделює цю систему. Отже, фундаментальна характеристика багатьох з мов, які можуть використовуватися в СУН, полягає в тому, що їх вирази запитів поряд із традиційними компонентами фільтрації даних повинні містити компоненти структурування.

5.3.1. Структурна організація веб-орієнтованої системи навчання

Інформаційно-аналітична система для навчання та управління навчальним процесом (ІАУН) виникла як результат спільної роботи в межах проекту Tempus Tasis в 2000–2001 роках і на сьогодні є завершеною розподіленою системою (робоча назва DigLE — Digital Learning Environment), розгорнутою і впровадженою в навчальний процес на кафедрі АСУ НУ «Львівська політехніка». Ця система забезпечує практично всі наявні на цьому етапі форми електронного навчання, зокрема й дистанційного.

Цільовим призначенням системи є забезпечення навчального процесу як з боку навчанння (отримання освіти), так і стосовно управління навчальним процесом на основі сучасних інформаційних технологій. Головні складові цієї системи зображені на рис. 5.2.

Основними завданнями її є:

— спрощення (шляхом комп'ютерної автоматизації) управління як навчальним процесом, так і підрозділом;
— розширення можливостей навчального процесу через введення в нього мультимедійних і симуляційних можливостей подання інформації;
— спрощення роботи викладача як щодо управління навчальним процесом через автоматичне підбиття результатів роботи за місяць (семестр, навчальний рік), так і стосовно проведення занять, тобто забезпечення усіх слухачів навчальним матеріалом, проведення навчального процесу на основі наперед складеного та заздалегідь відомого графіку тощо;
— забезпечення реклами та маркетингу підрозділу.

Логічна структура системи зображена на рис. 5.3.

На кафедрі автоматизованих систем управління в рамках проекту Tempus/Tacis UM_CP-20527-1999 «Національний центр інноваційних технологій у навчанні» було створено систему управління начальним процесом. Система створена для доступу до всіх матеріалів та нормативних документів, які регламентують навчальний процес на кафедрі. Використовуючи систему, користувачі отримують доступ до: навчальних планів підготовки спеціалістів, кваліфікаційної характеристики спеціалістів, інформації про кафедру та викладачів, оперативних новин та форумів, а також до навчальних матеріалів з дисциплін.

Система є інтегрованою з загальноуніверситетською системою, що забезпечує єдиний шлях доступу до ресурсів як самої системи, так і до університетських ресурсів.

Розроблений прототип системи містить традиційні для систем такого класу засоби організації взаємодії — дискусії та електронну дошку оголошень; оболонка дисципліни має стандартизований вигляд і містить такі сервіси:

Рис. 5.2. Схема управління навчальним процесом

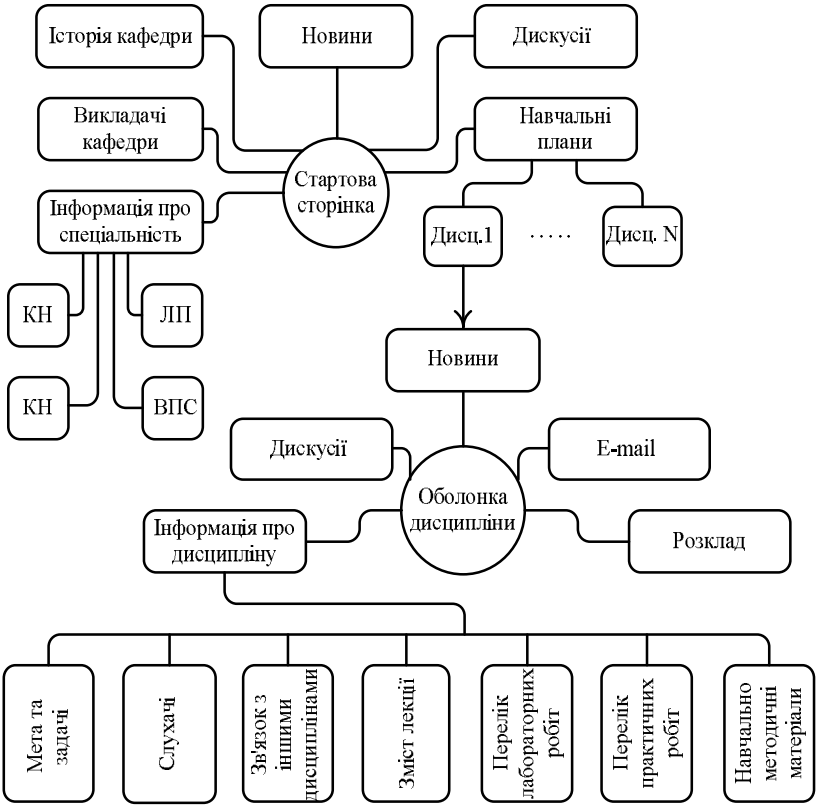


Рис.5.3. Логічна структура веб-системи

1. Сервіс новини дисципліни.

2. Інформація про дисципліну: мета та завдання дисципліни; слухачі дисципліни — список студентів; зв'язок з іншими дисциплінами — перелік дисциплін, знання яких необхідні студентові для вивчення конкретної дисципліни: лекційні заняття; лабораторні заняття; практичні заняття; самостійна робота; модульні контролі; навчально-методичні матеріали; питання для контролю залишкових знань.

Розклад є консолідованим джерелом інформації про навчальний процес дисципліни і містить упорядковану потижнево інфор-

мацію про всі види робіт. Розклад передбачає механізм виставлення оцінок за певні види робіт.

3. Середовище викладача.

4. Сервіс відправлення електронної пошти.

5. Сервіс дискусій.

Інформація про дисципліну за методами подання в системі поділяється на певні категорії: текстова інформація (мета та завдання дисципліни; зв'язок з іншими дисциплінами — перелік дисциплін, знання яких необхідні студентові для вивчення цієї дисципліни; навчально-методичні матеріали; питання для контролю залишкових знань). Для редагування цієї інформації в системі передбачено вікно редагування; гіпертекстова інформація (лекційні заняття; лабораторні заняття; практичні заняття; самостійна робота; модульні контролі); містить дані про тиждень, тему, кількість годин та балів (за необхідності), а також лінк на супровідний матеріал. Методичний матеріал додається в систему одразу (при виборі «Записати файл» або пізніше, використовуючи поле «Файл» у таблиці).

Вигляд оболонки курсу залежить від статусу користувача: для викладача, який читає дисципліну, доступні піктограми редагування інформації та можливість викладення на сервер документів; для студента або інших викладачів (які не відповідають за дисципліну) такі можливості недоступні.

Фрагмент роботи системи подано на рис. 5.4.

До проблем впровадження системи належать:

— стереотип стосовно проведення і лекційних, і практичних та лабораторних занять;
— складність переходу до використання системи, а саме — необхідність проведення певних тренінгових занять, особливо для викладачів, мало ознайомлених з комп'ютером, і затрати на переведення курсів в електронну форму;
— складність проведення усіх видів контролю, за винятком контролю, базованого на поточному тесті;
— технічне забезпечення студентів умовами для самостійної роботи.

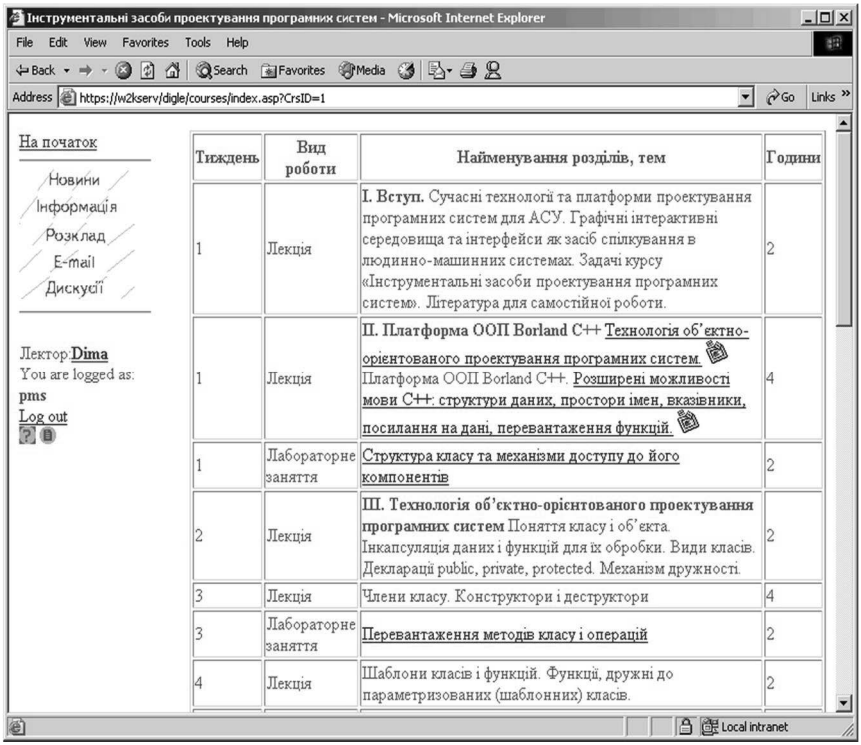


Рис. 5.4. Фрагмент роботи системи

Основними перевагами системи є:

— можливість донесення будь-яких обсягів матеріалу до користувача;

— вироблення та розвиток навичок самоостійної роботи студента;

— перетворення процесу навчання зі статичного у динамічний, тобто можливість швидкого донесення нового матеріалу відразу до усіх слухачів та використання нових підходів до подання інформації;

— забезпечення принципу навчання будь-де і будь-коли.

Можливості розширення системи: розроблена на цей час система орієнтована на підрозділ – кафедру, а тому перший шлях її розширення – це піднесення на рівень інституту, а в перспективі й університету. Очевидно, що без розвинутої інститутської інфра-

структури про жоден розвиток не йдеться. Отож першим кроком є розвиток саме внутрішньої інститутської інфраструктури. Другий шлях розвитку — відкриття зовнішнього доступу до системи, що дасть: по-перше, можливість самостійної роботи студентів удома та принаймні off-line спосіб спілкування з ними і, по-друге, проведення ширшої рекламної компанії як інституту, так і департаменту.

5.3.2. Перетворення вмісту бази даних у статичні документи

Протягом останніх десяти років відбувається бурхливий розвиток використання інформаційних та телекомунікаційних технологій (англійське скорочення — ICT) у навчальному процесі. Це стосується не тільки спеціалізованих так званих відкритих університетів (open university) чи університетів для дистанційного навчання (distance university), а й традиційних технічних чи гуманітарних університетів. У сучасному університеті вже не виникає питання про те, чи використовувати сучасні ICT для наукової та викладацької роботи. В одному з оглядів UNESCO стосовно ролі університетів в майбутньому інформаційному суспільстві безпосередньо констатується: «Нові інформаційні технології, особливо Інтернет, драматично змінюють доступ до інформації, навчальний процес та наукову роботу, спосіб дослідження, відкриття нового, викладання та навчання; майбутнє університетів прямо залежить від їх здатності адаптуватися в новому інформаційному суспільстві і відповідати вимогам ще більш чутливого до нових технологічних змін ринку праці». Сьогоднішні тенденції використання ICT полягають у створенні систем, здатних не лише виконувати інформаційно-аналітичні функції, а й давати можливість оперативно керувати навчальним процесом, бути ефективним середовищем організації та менеджменту освітянського процесу. Це дає змогу універсалізувати процес підготовки навчально-методичних матеріалів для різних форм навчання та різноманітних структур навчальних програм.

Організація розподілених інформаційно-навчальних систем. Інформаційні служби навчальних систем надають (чи повинні

надавати) набір сервісів, доступних із робочих місць як викладача, так і студента. Загалом поняття сервісу аж ніяк не обмежується навчальною системою, яка надає учасникам освітнього процесу доступ до корпоративних даних. Сервісом може бути доступ до тих чи інших файлів навчальної системи, які зберігаються в локальній мережі, робота з електронною поштою, доступ до Інтернету, використання мережевого принтера чи модему, проведення якихнебудь розрахунків, здавання і приймання звітів тощо [39]. Доступність того чи іншого сервісу в мережі нерідко визначається тим, які стандарти він підтримує (йдеться про стандартні програмні інтерфейси і стандартні протоколи обміну даними).

Якщо розглядати багатокористувацьку роботу з корпоративними даними в мережі, організовану на використанні будь-якої СУБД (у цьому випадку несуттєво, мережевої чи серверної), можна зауважити, що вона визначається деякими загальними дефініціями і складається переважно зі стандартного набору програмних компонентів і сервісів.

Наступним важливим компонентом такої системи є набір користувацьких програм, які використовуються для редагування й перегляду даних на робочих станціях студентів та викладачів. У такому випадку вважається, що програми містять презентаційну логіку навчальної системи. Як правило, користувацькі програми використовуються і для проведення інших операцій з даними (перевірка доступу, статистична обробка, генерація звітів тощо). При цьому сприймається, що така програма містить алгоритми прикладної обробки даних.

Ще один компонент, без якого робота мережевої навчальної системи неможлива, — це засоби забезпечення доступу до даних СУБД у клієнтській програмі. Набір таких засобів істотно залежить від того, чи є СУБД серверною. У всіх випадках він повинен містити засоби мережевого доступу, які базуються на мережевих можливостях операційних систем, використовуваних для експлуатації СУБД і клієнтських програм. Мережеві можливості операційних систем передбачають, як мінімум, підтримку мережевих протоколів, які забезпечують потрібний доступ.

У випадку серверних СУБД до цього набору додаються засоби взаємодії клієнтської програми і сервера баз даних, які використовують однакову підтримку мережевих протоколів операційними системами. Такі засоби зазвичай включають клієнтську частину серверної СУБД, яка містить, як правило, низькорівневе *API* (*Application Program Interface*) взаємодії із сервером баз даних.

Окрім того, засоби забезпечення доступу до даних нерідко мають бібліотеки, які, відповідно, містять високорівневі функції доступу до даних. Ці функції спрощують використання клієнтської частини, якщо СУБД є серверною, або реалізують стандартні операції з даними, якщо СУБД не серверна.

Один з найпоширеніших засобів забезпечення доступу до СУБД у відкритій навчальній системі — *WWW*-технології.

Компоненти веб-технології. Використання технологій WWW для забезпечення доступу до ресурсів навчальної системи передбачає наявність таких компонентів (рис. 5.5):

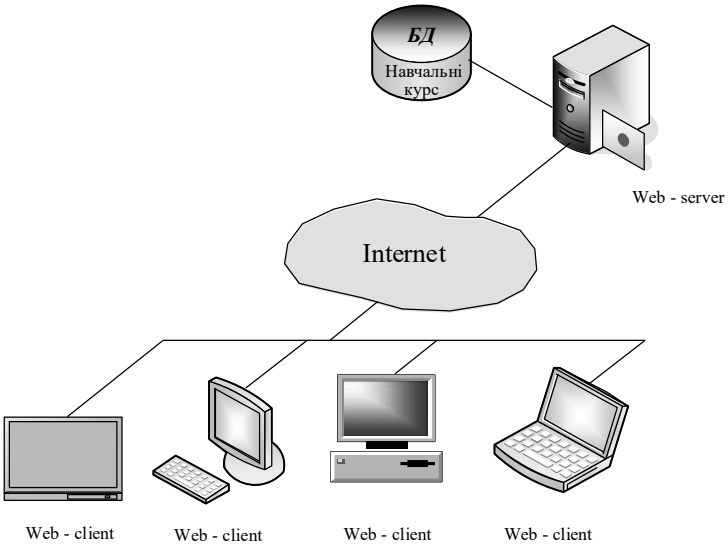


Рис. 5.5. Компоненти та сценарії WWW–доступу

1) *IP* — мережі з підтримкою базового набору послуг для передачі даних з єдиною політикою нумерації та маршрутизації, яка працює сервісом імен *DNS*;

2) виділеного інформаційного сервера — *WWW*-сервера, який забезпечує подання гіпертекстових документів через *IP*-мережу у відповідь на запити *WWW-клієнтів*, якими і будуть викладачі та студенти.

Передані гіпертекстові документи формуються в стандарті *HTML* — мові опису гіпертекстових документів. Ці документи можуть або зберігатися в статичному вигляді (сукупність файлів на диску), або динамічно компонуватися залежно від параметрів запиту спеціальним програмним забезпеченням. Для динамічного компонування *HTML*-документів *WWW*-сервер використовує різні технологічні рішення: тут і спеціальним чином оформлені програми — *CGI*-програми, написані на різних мовах, і програми, які є окремим програмним кодом як поза межами *HTML*-сторінки (аплети *Java*), так і всередині неї (*JavaScript, VBasicScript* скрипти).

До складу специфікації конкретної БД, на основі якої будується навчальна система, входять як технологічні основи — такі як тип СУБД, вид інтерфейсів, зв'язок між таблицями, обмеження цілісності, так і організаційні рішення, пов'язані з підтримкою актуальності бази даних і забезпеченням доступу до неї.

WWW-доступ до будь-яких баз даних навчальних систем може здійснюватися за одним із трьох основних сценаріїв. Нижче подано їх короткий опис і основні характеристики.

Перетворення вмісту БД у статичні документи: вміст БД переглядає спеціальна програма, яка створює набір файлів — зв'язних *HTML*-документів (рис. 5.6). Отримані файли можуть бути перенесені на один чи декілька WWW-серверів. Доступ до них здійснюватиметься аналогічно, як до статичних гіпертекстових документів сервера.

Означений варіант характеризується мінімальними початковими витратами. Він ефективний при невеликих масивах даних простої структури і з нечастим поновленням, а також за низьких вимог до актуальності даних, наданих через *WWW*. Крім того, очевидна повна відсутність механізму пошуку, хоча можливим є розвинутий механізм індексування.

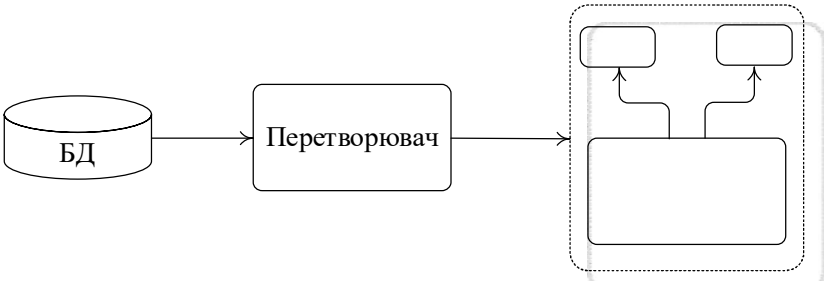


Рис. 5.6. Схема однократного перетворення вмісту БД навчальної системи у статичні документи

Перетворювачем може бути програмний комплекс, який автоматично чи напівавтоматично генерує статичні документи. Програма-перетворювач може бути самостійно розробленою програмою або бути інтегрованим засобом класу генераторів звітів.

Динамічне створення гіпертекстових документів на основі вмісту БД. У цьому варіанті доступ до БД навчальної системи здійснюється спеціальною програмою-скриптом, що запускається *WWW*-сервером у відповідь на запит *WWW*-клієнта. Програма, яка може виконуватись як на стороні клієнта (*Java, JavaScript*), так і на стороні сервера (*Java*,*CGI, php*), обробляючи запит, переглядає вміст БД і генерує вихідний *HTML*-документ, який повертається клієнту.

Таке рішення є дуже ефективним для великих баз даних зі складною структурою і в разі потреби підтримки операцій пошуку, а також при частих поновленнях і неможливості синхронізації перетворення БД у статичні документи з відновленням вмісту. У цьому варіанті можливо здійснювати зміну БД із *WWW*-інтерфейсів.

Недоліками методу є тривалий час обробки запитів, необхідність постійного доступу до основної бази даних, додаткове завантаження засобів підтримки БД навчальної системи, породжене обробкою запитів від *WWW*-сервера. Для реалізації такої технології треба використовувати взаємодію *WWW*-сервера з програмами, які запускаються. Вибір програмних засобів досить широкий — мови програмування, інтегровані засоби типу генераторів звітів та ін.

5.3.3. Створення інформаційного сховища даних

Для створення інформаційного сховища даних пропонується використання технології, яка здобула назву «інформаційного сховища» (ІС). Для обробки різноманітних запитів, зокрема й від *WWW*-сервера, використовується проміжна БД високої продуктивності. Інформаційне наповнення проміжної БД здійснюється спеціалізованим програмним забезпеченням на основі вмісту основних баз даних (рис. 5.7) [35].

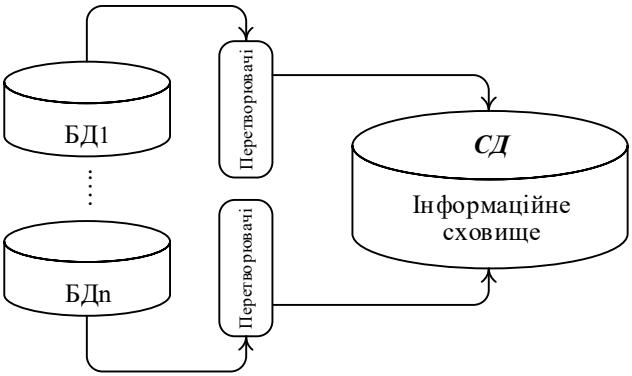


Рис. 5.7. Схема організації інформаційного сховища

Умовно процес роботи такої технології можна розподілити на два етапи:

— етап 1 — перевантаження даних;

— етап 2 — обробка запитів.

Цей варіант не має усіх недоліків попередньої схеми. Окрім того, після встановлення синхронізації даних інформаційного сховища навчальної системи з основними БД можливе перенесення користувацьких інтерфейсів на інформаційне сховище, що істотно підвищить надійність і продуктивність, дасть змогу організувати розподілені робочі місця.

Незважаючи на вдавану громіздкість такої схеми, для задач забезпечення *WWW*-доступу до вмісту декількох баз даних накладні витрати істотно зменшуються. Основою підвищення про-

дуктивності обробки *WWW*-запитів і різкого збільшення швидкості розробки *WWW*-інтерфейсів є використання внутрішніх мов СУБД інформаційного сховища для створення гіпертекстових документів.

Для завантаження вмісту основних БД в інформаційне сховище можуть використовуватися всі перелічені рішення (мови програмування, інтегровані засоби), а також спеціалізовані засоби перевантаження, які постачаються з *SQL*-сервером, і продукти підтримки інформаційних сховищ.

Реалізація розподіленої навчальної системи. Система управління навчальним процесом (Learning Management System — LMS) є програмним продуктом, який забезпечує *слухача* інтегрованою інформацією з курсу та про виконану роботу згідно з планом навчання; *викладача* (інструктора) — засобами системи донесення навчального матеріалу (Educational Delivery System) та оцінки результатів навчання слухачів [34]; *керівника підрозділу* — засобами управління курсами (Course Management System) та простеження поточного стану підготовки слухачів чи рівень підготовленості курсу [38].

Для розроблених систем управління навчальним процесом (Docent Enterprise, Knowledgesoft Enterprise) характерна проста система організації навчання студентів: студент для отримання конкретного рівня підготовки (бакалавр, магістр) має пройти певний набір курсів; він самостійно обирає дисципліну, яку вивчатиме в цей час, і оплачує курс навчання; менеджер дисципліни за фактом оплати формує групи, додає дисципліну до розкладу (ростер-roaster, transcript) студента, надаючи йому доступ до навчального матеріалу та завдань; менеджер курсу призначає інструктора (викладача), відповідального за викладання дисципліни.

Ростер є надзвичайно важливим у контексті управління навчальним процесом. Він має не лише звичний для нас перелік дисциплін, викладачів та дат викладання дисципліни, а й забезпечує систематизоване поєднання усіх видів навчальної роботи в межах дисципліни та засобів їх реалізації у веб-базованій системі.

Рішення створювати власну систему управління навчальним процесом було прийняте внаслідок впливу таких факторів:

— висока вартість систем управління навчальним процесом закордонних виробників;
— проблеми адаптації цих систем до навчального процесу в українських університетах.

Використовувати розроблені у світі системи в українських університетах неможливо передусім через значні відмінності в організації навчального процесу та наявність різних категорій студентів, котрі навчаються:

а) студенти за державним замовленням, які отримують доступ до дисципліни у складі навчальної групи (груповий запис);

б) комерційні студенти, які, крім виконання навчального плану, зобов'язані оплатити навчання (персоналізований підхід);

в) студенти-екстерни, які навчаються за індивідуальним навчальним планом;

— відсутність на ринку України програмних засобів систем управління навчальним процесом.

Доступ до інформації в системі є персоналізованим для викладачів, студентів та керівників.

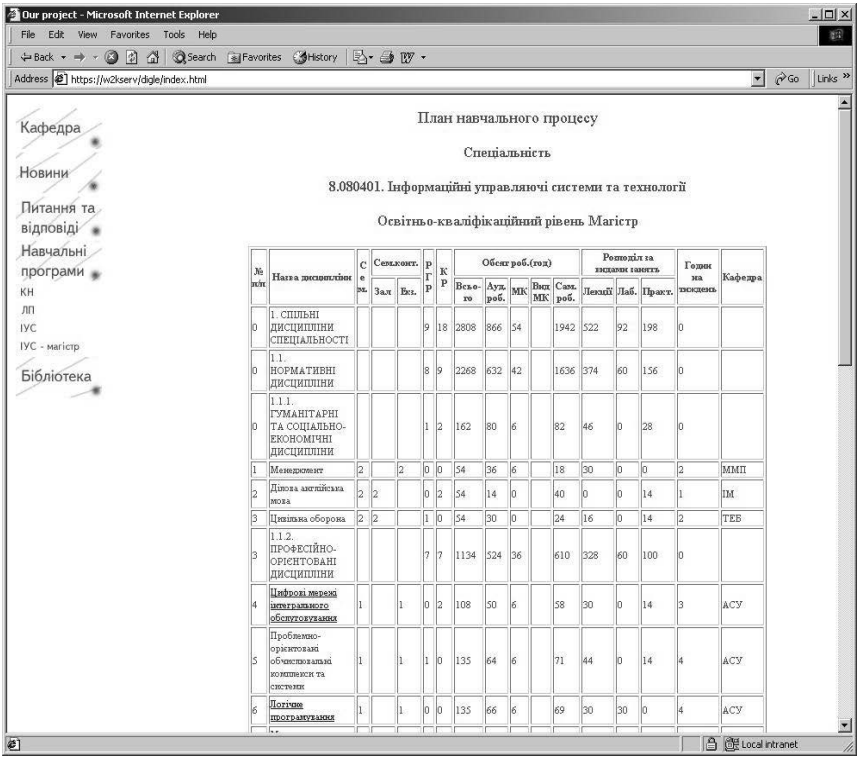
Для реєстрації в системі використовується облікова інформація з бази даних користувачів (Directory) університетської мережі на базі Novell NetWare. Використання єдиного джерела облікової інформації про користувачів спрощує адміністрування та підтримку системи. У разі потреби можливе використання власної бази облікової інформації про користувачів. Використання протоколів шифрованої передачі даних HTTPS та SSL дає можливість передавати пароль користувача в шифрованому вигляді, що підвищує захищеність системи.

Середовище системи складається із розділу інформації про підрозділ (кафедру) та множини оболонок дисциплін (рис. 5.8). Інформація та сервіси оболонки підрозділу розділені на категорії:

1. Кафедра: історія кафедри, центру, ВНЗ; загальна інформація про кафедру; штат кафедри (список викладачів кафедри з детальною інформацією та списком дисциплін, які читає викладач); характеристика спеціальностей.

2. Новини: розділ, в якому відповідальна особа розміщує оголошення для працівників кафедри. Цей розділ є недоступним для студентів.

3. Запитання та відповіді: система дискусій, в якій користувачі можуть обмінюватись електронними повідомленнями.



№ п/п	Назва дисципліни	Се ме.	Сем.конт.		РГР	КР	Обсяг роб.(год)					Розподіл та видачі занять			Годин за тиждень	Кафедра
			Зал	Екз.			Всього	Ауд. роб.	МК	Вид. МК	Сам. роб.	Лекції	Лаб.	Практ.		
0	1. СПІЛЬНІ ДИСЦИПЛІНИ СПЕЦІАЛЬНОСТІ				9	18	2808	866	54		1942	522	92	198	0	
0	1.1. НОРМАТИВНІ ДИСЦИПЛІНИ				8	9	2268	632	42		1636	374	60	156	0	
0	1.1.1. ГУМАНІТАРНІ ТА СОЦІАЛЬНО-ЕКОНОМІЧНІ ДИСЦИПЛІНИ				1	2	162	80	6		82	46	0	28	0	
1	Менеджмент	2		2	0	0	54	36	6		18	30	0	0	2	ММП
2	Ділова англійська мова	2	2		0	2	54	14	0		40	0	0	14	1	ІМ
3	Цивільна оборона	2	2		1	0	54	30	0		24	16	0	14	2	ТЕБ
3	1.1.2. ПРОФЕСІЙНО-ОРІЄНТОВАНІ ДИСЦИПЛІНИ				7	7	1134	524	36		610	328	60	100	0	
4	Цифрові мережі інтегрального обслуговання	1		1	0	2	108	50	6		58	30	0	14	3	АСУ
5	Проблемно-орієнтовані обчислювальні комплекси та системи	1		1	1	0	135	64	6		71	44	0	14	4	АСУ
6	Логічне програмування	1		1	0	0	135	66	6		69	30	30	0	4	АСУ

Рис. 5.8. Зріз системи управління навчальним процесом: навчальні плани; права доступу – анонімний користувач

4. Навчальні програми: список планів навчального процесу для спеціальностей, підготовка з яких проводиться на кафедрі; бакалавр за напрямом «Комп'ютерні науки»; магістр за спеціальністю «Інформаційні управляючі системи»; магістр за спеціальністю «Системи підтримки прийняття рішень».

План навчального процесу має стандартну форму, містить також переходи на оболонки внесених у систему курсів. Дані плану навчального процесу є основою для організації інформації про дисципліни.

5. Бібліотека — перехід на добірку електронних видань. Крім того, в розділі бібліотеки можуть бути розміщені курси, що мають веб-інтерфейс, посилання на які зберігаються в оболонці дисципліни.

Оболонка дисципліни має стандартизований вигляд і містить такі сервіси: сервіс новин дисципліни; інформація про дисципліну; мета та завдання дисципліни; слухачі дисципліни — список студентів.

Зв'язок з іншими дисциплінами — перелік дисциплін, знання яких необхідні студентові для вивчення конкретної дисципліни рис. 5.9: лекційні заняття; лабораторні заняття; практичні заняття; самостійна робота; модульний контроль; навчально-методичні матеріали; питання для контролю залишкових знань.

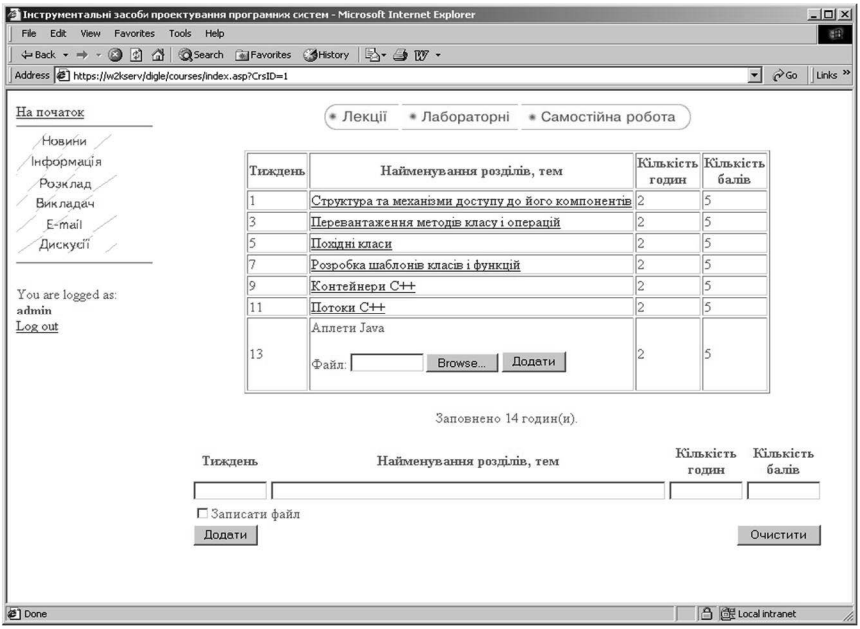


Рис. 5.9. Зріз системи управління навчальним процесом: лабораторні заняття; права доступу — викладач

1. Розклад.
2. Середовище викладача.
3. Сервіс відправлення електронної пошти.
4. Сервіс дискусій.

Вигляд оболонки курсу залежить від статусу користувача: для викладача, який читає дисципліну, доступні піктограми редагування інформації та пункт меню «Викладач»; для студента або інших викладачів ці можливості недоступні.

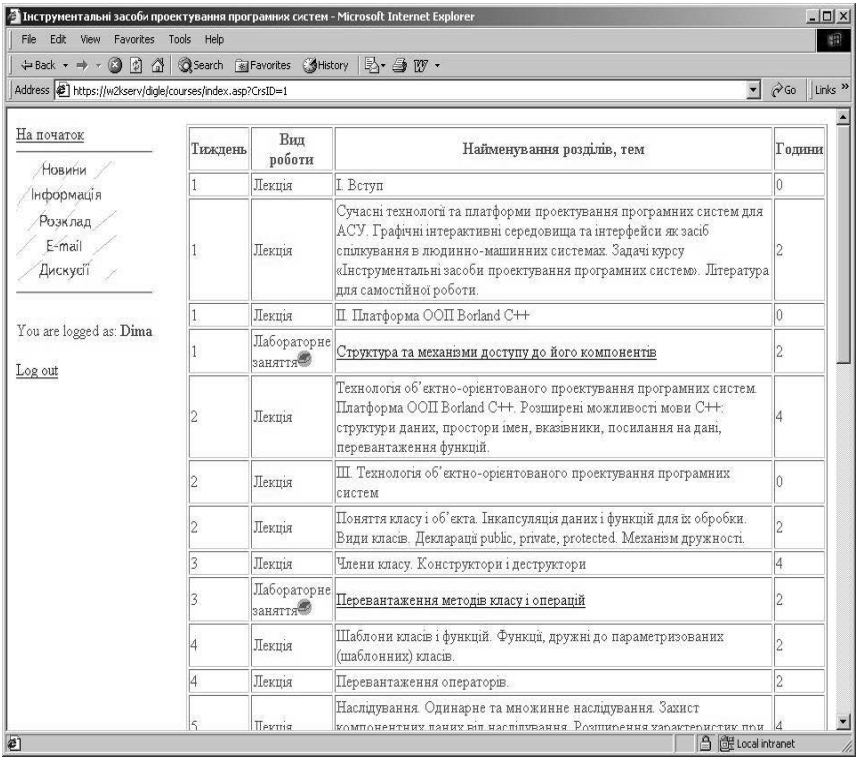
Редагування інформації про дисципліну: інформація за методами подання в системі поділяється на категорії:

— текстова інформація (мета та завдання дисципліни; зв'язок з іншими дисциплінами — перелік дисциплін, знання яких необхідні студентові для вивчення цієї дисципліни; навчально-методичні матеріали; питання для контролю залишкових знань). Для редагування цієї інформації в системі передбачено вікно редагування;

— гіпертекстова інформація (лекційні заняття; лабораторні заняття; практичні заняття; самостійна робота; модульні контролі) містить дані про тиждень, тему, кількість годин та балів. Методичний матеріал у систему додається одразу або пізніше, використовуючи поле «Файл» в таблиці.

Розклад є консолідованим джерелом інформації про навчальний процес дисципліни і містить впорядковану потижнево інформацію про всі види робіт (рис. 5.10). Тут також передбачено механізм виставляння оцінок за певні види робіт.

До засобів адміністрування системи належать: створення оболонки нового курсу; додавання навантаження викладача.



Тиждень	Вид роботи	Найменування розділів, тем	Години
1	Лекція	I. Вступ	0
1	Лекція	Сучасні технології та платформи проектування програмних систем для АСУ. Графічні інтерактивні середовища та інтерфейси як засіб спілкування в людинно-машинних системах. Задачі курсу «Інструментальні засоби проектування програмних систем». Література для самостійної роботи.	2
1	Лекція	II. Платформа ООП Borland C++	0
1	Лабораторне заняття	Структура та механізми доступу до його компонентів	2
2	Лекція	Технологія об'єктно-орієнтованого проектування програмних систем. Платформа ООП Borland C++. Розширені можливості мови C++: структури даних, простори імен, вказівники, посилання на дані, перевантаження функцій.	4
2	Лекція	III. Технологія об'єктно-орієнтованого проектування програмних систем	0
2	Лекція	Поняття класу і об'єкта. Інкапсуляція даних і функцій для їх обробки. Види класів. Декларації public, private, protected. Механізм дружності.	2
3	Лекція	Члени класу. Конструктори і деструктори	4
3	Лабораторне заняття	Перевантаження методів класу і операцій	2
4	Лекція	Шаблони класів і функцій. Функції, дружні до параметризованих (шаблонних) класів.	2
4	Лекція	Перевантаження операторів.	2
5	Лекція	Наслідування. Одинарне та множинне наслідування. Захист компонентних даних від наслідування. Розширення характеристик при	4

Рис. 5.10. Зріз системи управління навчальним процесом

ВИСНОВКИ

У монографії розроблено методи підвищення ефективності та якості прийняття управлінських рішень з використанням інформаційних технологій опрацювання запитів у базах даних і системах документообігу в соціальних структурах. При цьому отримано такі основні результати.

1. Уперше розроблено метод пришвидшення процедур запитів і проведено їх оптимізацію (це дає змогу підвищити опрацювання різновидних і слабоформалізованих даних прискорення вибірки при реалізації алгоритму від 7–38% залежно від кількості повторень слів та їх довжини в об'єктах соціальних структур), що підвищує ефективність прийняття управлінських рішень.

2. Одержали подальший розвиток методи інтеграції об'єднаних баз різнорідних даних у системі підтримки прийняття рішень, що дало змогу ефективніше сприймати та усвідомлювати знання комплексів і блоків даних для оцінки ситуації й адекватного прийняття рішень, забезпечивши адекватність прийняття рішень відповідно до ситуацій у соціоорієнтованих системах.

3. Удосконалено методи побудови автоматизованих інтегрованих мультимедійних комплексів для відображення динамічних ситуацій в об'єктах соціокомунальної структури, що дало можливість виявляти й ідентифіковувати просторові, структурні та соціальні зміни в системах нижнього управлінського рівня та, відповідно, оперативно коректувати документи і їх обіг.

4. Удосконалено методи підвищення професійного рівня оперативного персоналу на основі використання сучасних інформаційних технологій: мультимедіа та веб-технологій автоматизацій навчального процесу, що стало підставою формування управлінських команд з високим рівнем підготовки.

5. Отримані результати дослідження є основою вдосконалення та оптимізації SQL-запитів у неструктурованих базах даних для проектування систем з прийняття рішень у соціоорієнтованих структурах. Розроблений метод пришвидшення процедур запитів дає змогу прискорити опрацювання слабоформалізованих і нестру-

ктурованих даних для об'єктів соціоорієнтованих структур та забезпечує можливість ефективно оцінити поточну ситуацію, обравши правильне рішення.

6. Запропонована методика зменшення часу вибірки є ефективною у великих і надвеликих базах даних, а також у базах із достатньо складними логічними зв'язками. Інша область її використання — внесення до неї елементів аналізу посимвольних рядків для пошуку близьких за змістом чи споріднених рядків, що може суттєво розширити сферу використання цієї методики, особливо під час роботи в надвеликих сховищах даних.

CONCLUSIONS

The book presents the developed methods to improve the efficiency and quality of management decisions using the information technology of inquiries processing in databases and document management systems in social structures. The following main results have been achieved.

1. For the first time the method of speeding up inquiries procedures has been developed and their optimization has been made (it allows you to increase processing of heterogeneous and weakly-formalized data of selection acceleration in the implementation of the algorithm from 7-38% depending on the number of repetitions of words and their length in the objects of social structures), which increases the effectiveness of management decisions.

2. The further development of integration methods of combined heterogeneous databases in the decision support system has been received, which allowed to perceive and understand better the knowledge of complexes and data blocks to assess the situation and adequate decision-making, ensuring adequate decision-making in the situations of socio-oriented systems .

3. The methods of design of automated integrated multimedia complexes for displaying dynamic situations in the objects of socio communal structure have been improved, making it possible to detect and identify the spatial, structural and social changes in the systems of the lower management level and, therefore, correct documents and their circulation promptly.

4. The methods of enhancing the professional development of operational staff through the use of modern information technology have been improved: multimedia and web technologies of automation of the educational process, which led to the formation of management teams with a high level of training.

5. The received results of the research are the basis for improving and optimizing of SQL-inquiries in non-structured databases for the design of decision-making systems in socio-oriented structures. The developed method to speed up the inquiries procedure results in faster processing of weakly-formalized and non-structured data for the objects

of socio-oriented structures and provides the ability to effectively assess the current situation, choosing the right solution.

6. The suggested technique of reducing the selection time is effective in large and enormous databases, as well as in the bases with quite complex logical connections. Another area of its use is to make it the element of analysis of character lines in search of similar in content or related lines, which could significantly expand the scope of application of this technique, especially when working in enormous data storage.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Батюк А. Є. Комп'ютеризований фінансовий аналіз підприємств при розробці інноваційних проектів / А. Є. Батюк, М. С. Пасєка, С. М. Смовженко // Регіональна науково-технічна політика: інноваційний розвиток та інформаційний простір: міжнар. наук.-практ. конф. — Львів, 2000. — 29–30 червня.
2. Батюк А. Є. Концепція побудови і реалізації інформаційних систем, орієнтованих на аналіз даних / А. Є. Батюк, М. С. Пасєка // Технічні вісті. — Львів, 2000. — Вип. 1(10)–2(11). — С. 76–79.
3. Блюменау Д. И. Информация и информационный сервис / Д. И. Блюменау. — Л. : Наука, 1989. — 180 с.
4. Буч Г. Объектно-ориентированное проектирование с примерами применения / Г. Буч ; пер. с англ. Конкорд. — М., 1992. — 519 с.
5. Бушуєв С. Д. Автоматизированные системы управления строительством / Бушуєв С. Д., Михайлов В. О., Лянко С. Д. — К. : Будівельник,1989. — 256 с.
6. Герчикова И. Н. Маркетинг: организация, технология / И. Н. Герчикова. — М.: Высшая школа, 1994. — 81 с.
7. Гирнык Н. Л. Стохастическая динамика управляемых экотехнических систем / Н. Л. Гирнык, Л. С. Сикора. — М.: НПО «Информация», 1991. — 330 с.
8. Дейт К. Введение в системы баз данных / К. Дейт. — М. : Наука, 1980. — 463 с.
9. Долан Э. Д. Макроэкономическая модель / Э. Д. Долан, Д. Е. Линдсей. — СПб. : Рынок, 1992. — 496 с.

10. Долан Э. Дж. Микроэкономика / Э.Дж. Долан, Д. Е. Линдсей ; пер. с англ. В.В. Лукашевич и др. — СПб.: АО «Санкт-Петербург оркестр», 1994. — 446 с.

11. Драган Я. П. Інформаційно-ресурсна концепція синтезу структур керування / Я. П. Драган, Л. С. Сікора, О. І. Мартиненко // УОиМ. — К., 1991. — № 5, № 9. — С. 79–86.

12. Драган Я. П. Системология, стохастические модели и теория целеустремленных систем как концептуальная основа и функционирования / Я. П. Драган, Л. С. Сикора // Технология программирования – 90 : междунар. конф. — К., 1991. — С. 21–22.

13. Дурняк Б. В. Інформаційна технологія та системологія управління регіональними структурами в умовах ризику / Б. В. Дурняк, М. С. Пасєка // Збірник наукових праць; Національна академія наук України. Інститут моделювання в енергетиці. — К. — 2012. — № 65. — С. 188–195.

14. Дурняк Б. В. Інформаційні та системні концепції в розробці автоматизованого робочого місця в структурі системи підтримки прийняття рішень / Б. В. Дурняк, М. С. Пасєка // Збірник наукових праць; Національна академія наук України. Інститут моделювання в енергетиці. — К., 2012. — № 64. — С.187–196.

15. Дурняк Б. В. Інформаційні технології ситуативного прийняття рішень в адміністративних та господарських структурах / Б. В. Дурняк, М. С. Пасєка, О. М. Назаренко // Збірник наукових праць; Національна академія наук України. Моделювання та інформаційні технології. — К. — 2012. — № 65. — С.192–195.

16. Дурняк Б. В. Інформаційні технології створення базису моделей для СППР управління адміністративно-господарськими структурами / Б. В. Дурняк, М. С. Пасєка // Збірник наукових

праць; Національна академія наук України. Інститут моделювання в енергетиці. — К., 2012. — № 66. — С. 207–214.

17. Карачодова Е.А. Автоматизированные рабочие места / Е. А. Карачодова, В. А.Антонов, В. Ф. Маслов. — К. : Техніка, 1989. — 128 с.

18. Кігель В. Методи і моделі підтримки прийняття рішень в ринковій економіці / В. Кігель. — К. : ЦУЛ, 2003. — 202 с.

19. Колемаев В. А. Теория вероятностей и математическая статистика / В. А. Колемаев, О. В. Староверов, В. Б. Турундаевский. — М. : Высш. шк., 1991. — 400 с.

20. Котлер Ф. Основы маркетинга : пер. с анг. / Ф. Котлер; общ. ред. и вступ. ст. Е. М. Пеньковой. — М.: Прогресс, 1990. — 736 с.

21. Кучеров О. Ю. Автоматизированнные системы управления предприятиями промышленности строительных материалов / О. Ю. Кучеров, А. Е. Рохварчер. — Л. : Стройиздат, 1989. — 374 с.

22. Лавинский Г. В. Теоретические основы автоматизации управления в экономических системах / Г. В. Лавинський, А. Д. Шарапов. — К.: Вища шк., 1988. — 178 с.

23. Лесечко М. Д. Технологія прийняття управлінських рішень в органах державного управління та органах місцевого самоврядування / Лесечко М. Д., Чемерис А. О., Рудніцька Р.М. — Львів : ЛРІДУ УАДУ, 2003. — 168 с.

24. Маркова В. Д. Маркетинг услуг / В. Д. Маркова. — М.: Фи, 1996. — 126 с.

25. Мейер Д. Теория реляционных баз данных / Д. Мейер. — М. : Мир, 1987. — 608 с.

26. Мескон М. Х. Основы менеджмента / М. Х. Мескон. — М. : Дело, 1990. — 702 с.

27. Міхеєв Є. К. Система підтримки прийняття рішень в рослинництві / Є. К. Міхеєв, В. А. Платонов // Аграрна наука: Вісник аграрної науки. — К., 1995. — № 10. — С. 41–48.

28. Паламарчук А. М. Общественно-территориальные системы (логико-математическое моделирование) / А. М. Паламарчук. — К. : Наук. думка, 1992. — 270 с.

29. Пасєка М. С. Ефективні алгоритми виконання запитів на основі оптимізації операцій об'єднання / М. С. Пасєка, Д. Д. Пелешко // Технічні вісті. — Львів, 2003. — № /1(16)–2(17). — С. 39–42.

30. Пасєка М. С. Вибір і оцінка альтернативних планів виконання запитів / М. С. Пасєка, Д. Д. Пелешко, Н. Д. Лотошинська, М. З. Лоза // Технічні вісті. — 2006. — № 1(22)–2(23). — С. 56–64.

31. Пасєка М. С. Використання Web-технологій для створення навчальних систем / М. С. Пасєка, А. Б. Стецюк // Комп'ютерна інженерія та інформаційні технології : Вісник Державного університету «Львівська політехніка». — Львів, 2000. — № 413. — С. 120–127.

32. Пасєка М. С. Глобальна оптимізація в реляційних системах керування базами даних / М. С. Пасєка, Д. Д. Пелешко // Праці 9-th international modelling school of AMSE – UAPL : [12-17 September]. — Alushta – Ukraine, 2004. — P.183–186.

33. Пасєка М. С. Логічна оптимізація запитів / М. С. Пасєка, Д. Д. Пелешко, Н. Д. Лотошинська // Технічні вісті. — Львів, 2007. — № 1(25)–2(26). — С. 37–43.

34. Пасєка М. С. Математична модель адаптивного тестування знань / М. С. Пасєка // Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту (ISDMCI'2009) : міжнар. наук. конф. — Євпаторія, 2009. — С. 100–102.

35. Пасєка М. С. Огляд можливостей використання провідних СУБД для побудови сховищ даних / М. С. Пасєка // Комп'ютерна інженерія та інформаційні технології : Вісник Державного університету «Львівська політехніка». — Львів, 1999. — № 386. — С. 191–198.

36. Пасєка М. С. Оптимізатори реляційних СУБД із гнучкою структурою / М. С. Пасєка, Д. Д. Пелешко, М. З. Лоза // Технічні вісті. — Львів, 2006. — № 3(24). — С. 98–101.

37. Пасєка М. С. Стратегії виконання об'єднань у розподілених базах даних / М. С. Пасєка, Д. Д. Пелешко // Технічні вісті. — Львів, 2004. — № 1(18)–2(19). — С. 72–74.

38. Пасєка М. С. Інструментальні засоби для створення систем навчання з використанням Web / М. С. Пасєка, А. Б. Стецюк // Праці семінару Національного центру інноваційних технологій у навчанні. — К., 2000. — С. 34–41.

39. Пелешко Д. Д. Web-технології як основа сучасних інформаційних технологій в навчанні / Д. Д. Пелешко, А. Б. Стецюк // Вісник НУ «Львівська політехніка». — Львів, 2000. — № 413. — С. 74–78.

40. Пелешко Д. Д. Деякі окремі алгоритми виконання реляційних операцій / Д. Д. Пелешко, М. С. Пасєка // Технічні вісті. — Львів, 2003. — № 1(16)–2(17). — С. 46–48.

41. Пелешко Д. Д. Ефективні алгоритми виконання запитів на основі оптимізації операцій об'єднання / Д. Д. Пелешко, М. С. Пасєка // Технічні вісті. — Львів, 2003. — № 1(16)–2(17). — С. 39–42.

42. Пелешко Д. Д. Метод оптимізації роботи пошукової машини / Д. Д. Пелешко, М. С. Пасєка // Комп'ютерна інженерія та інформаційні технології : Вісник Національного університету «Львівська політехніка». — Львів, 2002. — № 450. — С. 133–138.

43. Пелешко Д. Д. Семантична оптимізація запитів / Д. Д. Пелешко, М. С. Пасєка // Технічні вісті. — Львів, 2005. — № 1(20)–2(21). — С. 34–39.

44. Пелешко Д. Д. Стратегії виконання об'єднань у розподілених базах даних / Д. Д. Пелешко, М. С. Пасєка // Технічні вісті. — Львів, 2004. — № 1(18) –2(19). — С. 72–74.

45. Переудов Р. И. Информационные системы для руководителей / [ред. Р. И. Переудов]. — М. : Фин. и стат., 1989 — 176 с.

46. Подольчак Н. Ю. Стратегічний менеджмент / Н. Ю. Подольчак. — Львів : «Львів. політех.», 2012. — 400 с.

47. Пономаренко В. С. Інформаційні системи і технології в економіці / ред. В. С. Пономаренко. — К. : В.Ц. «Академія», 2002. — 542 с.

48. Райфа Г. Прикладная теория статистических решеней / Г. Райфа, Р. Шлейфер. — М. : Статистика, 1977. — 360 с.

49. Рашкевич Ю. М. Оптимізація процесу пошуку інформації в базах даних систем управління навчанням / Ю. М. Рашкевич, Д. Д. Пелешко, М. С. Пасєка // Комп'ютерні науки та інформаційні технології : Вісник Національного університету «Львівська політехніка». — Львів, 2002. — № 468. — С. 162–170.

50. Рашкевич Ю. М. Інформаційні технології у навчальному процесі / Ю. М. Рашкевич // Праці семінару національного центру інноваційних технологій у навчанні. — К., 2000. — С. 119–125.

51. Рашкевич Ю. М. Методи прискорення пошуку в системах зберігання даних / Ю. М. Рашкевич, Д. Д. Пелешко, М. С. Пасєка // Міжнародна конференція з індуктивного моделювання МКІМ-2002 : [20–25 травня]. — Львів, 2002. — С. 125–130.

52. Рашкевич Ю. М. Проектирование WEB-ориентированных распределенных учебных систем / Ю. М. Рашкевич, Д. Д. Пелешко, М. С. Пасєка, А. Б. Стецюк // Управляющие системы и машины. — К., 2002. — № 3/4. — С. 72–79.

53. Рашкевич Ю. М. Проектування WEB-орієнтованих розподілених навчальних систем / Ю. М. Рашкевич, Д. Д. Пелешко, М. С. Пасєка, А. Б. Стецюк // Telematics&life-longlearning : international Workshop : [october 15–17]. — Kyiv–Ukraine, 2001. — С. 143–152.

54. Рашкевич Ю. М. Структурний аналіз систем управління навчанням / Ю. М. Рашкевич, Д. Д. Пелешко, М. С. Пасєка, А. Б. Стецюк // Вестник Херсонского государственного технического университета. — Херсон, 2002. — № 1(14). — С. 464–470.

55. Рожнов В. С. Информационное обеспечение хозяйственной деятельности предприятия / В. С. Рожнов. — М. : Фин. и стат., 1987. — 144 с.

56. Рубинов А. М. Оптимальное управление в агрегированных моделях экономики / А. М. Рубинов, К. Ю. Борисов, В. Н. Десницкая, В. Д. Матвеенко. — М. : Наука, 1991. — 269 с.

57. Русинович М. Внутренее устройство Microsoft Windows: Windows Server 2003, Windows XP, Windows 2000 / М. Русинович, Д. Соломон. — М. : Питер, 2005. — 992 с.

58. Сахаров А. А. Концепции построения и реализации информационных систем, ориентированих на анализ данних / А. А. Сахаров // СУБД. — 1996. — № 4. — С. 55–70.

59. Синки Дж.-мл. Управление финансами в коммерческих банках / Дж.-мл. Синки. — М., 1994. — 17 с.

60. Сікора Л. С. Системологія управління рішень на управління в складних технологічних системах / Л. С. Сікора. — Львів: Каменяр, 1998. — 452 с.

61. Степанкова Т. М. Фінанси в умовах перехідної економіки / Т. М. Степанкова // Фінанси України. — 1998. — № 2. — С. 5–19.

62. Стечко Д. Н. Комплексное территориальное планирование народного хозяйства / Д. Н. Стечко. — К. : Вища школа, 1988. — 200 с.

63. Таненбаум Э. Современные операционные системы / Э. Таненбаум. — М.: Питер, 2002. — 1037 с.

64. Ульман Д. Основы систем баз данных / Д. Ульман. — М.: Финансы и статистика, 1983. — 335 с.

65. Умнова Э. А. Бухгалтерский учет на персональной ЭВМ: учебный практикум по ведению бухгалтерского учета в «1С: Бухгалтерии» : [5-ая версия] / Э.А. Умнова, Д. В. Чистов. — Изд. 2-ое, испр. и доп. — М. : Фирма «1С», 1994. — 331 с.

66. Фатхудинов Р.А. Стратегический менеджмент / Р. А. Фатхудинов. — М.: ЗАО «Бизнес-школа», 1997. — 304 с.

67. Ходаков В. Е. Системы информационного обслуживания руководителей предприятий / В. Е. Ходаков. — К. : Техника, 1992. — 200 с.

68. Цигичко В. Н. Руководителю о принятия решений / В. Н. Цигичко. — М.: Финн. и стат., 1991. — 210 с.

69. Яремко И. И. Бухгалтерский учет в условиях различных форм собственности и хозяйствования : учеб. пособ. / И. И. Яремко. — Львов: Леопрес, 1993. — 440 с.

70. Abiteboul S. Querying and updating the file [Електронний ресурс] / Abiteboul S., Cluet S., Milo T. // In Proc. of the Int. Conf. on Very Large Data Bases (VLDB). — Dublin, Ireland, 1993. — Режим доступу: link.springer.com/.../bbm%3A978-3-540-4923.

71. Abiteboul S. Querying semi-structured data : In Proc. of the Int. Conf. on Database Theory / S. Abiteboul // (ICDT). — Delphi, Greece: — 1997. — P. 167–172.

72. Aho A.V. Equivalence of Relational Expressions /Aho A. V., Sagiv Y., Ullman J.D. // SIAM J. Comput. — 1979. — № 2. — P. 218–246.

73. Astrahan M. M. Performance of the System R Acess Selection Mechanism / Astrahan M. M., Schkolnick M., Kim W. // Inf. Process., 80. Proc. IFIP World Comput. Congr. : [Sept., 1980]. — Melbourne: Amsterdam e.a., 1980. — P. 487–492.

74. Astrahan M. M. System R: A Relational Approach to Data Base Management / Astrahan M. M., Blasgen M. W., Chamberlin D. D., Eswaran K. P., Griffits P. P., King W., Lorie R. A., McJones P., Mehl J. W., Putzolu G. R., Traiger I. L., Wade B. W., Watson V. // ACM Trans. Database Syst. — 1976. — № 2. — P. 97–137.

75. Atzeni P. Design and maintenance of dataintensive web sites. In Proc. of the Conf. on Extending Database Technology [Електронний ресурс] / Atzeni P., Mecca G., Merialdo P. // (EDBT). — Valencia, Spain: — 1998. — Режим доступу : http://link.springer.com/article/10.1023%2FA%3A1012456311864#

76. Bell D. A. Issues in Relational Database Performance / D.A. Bell // Data and Knowledge Eng. — 1988. — № 1. — P. 46–61.

77. Blasgen M. W. Storage and Access in Relational Data Bases / Blasgen M. W., Eswaran K. P. // IBM Syst. J. — 1977. — 16, № 4. — P. 363–377.

78. Bloom B. H. Space/Time Trade-offs in Hash Coding with Allowable Errors / B. H. Bloom // Commun. ACM. — 1970. — 13, № 7. — P. 422–426.

79. Bodorik P. Distributed Query Processing Optimization Objectives / P. Bodorik, J.S. Riordon // 4th Int. Conf. Data Eng., West Berlin : [Sept., 13-15]. — 1988. — P. 320–329.

80. Ceri S. Optimizing Joins Between Two Partitioned Relations in Distributed Databases / S. Ceri, G. Gottlob // J. Parall. and Distrib. Comput. — 1986. — 3. № 2. — P. 183–205.

81. Ceri S. Translating SQL into Relational Algebra: Optimization, Semantics, and Equivalence of SQL Queries / S. Ceri, G. Gottlob // IEEE Trans. on Software Eng. — 1985. — 11, № 4. — P. 324–345.

82. Chacravarthy U. S. Multiple Query Processing in Deductive Databases Using Query Graphs / U.S. Chacravarthy, J. Minker // Proc. 12th Int. : Conf. Very Large Data Bases, Kyoto, Japan, Aug. 1986. Los Altos, Calif. — 1986. — P. 384–394.

83. Chakravarthy U. S. Semantic Query Optimization in Expert Systems and Database Systems / Chakravarthy U. S., Fishman D. H., Minker J. // Expert Database Syst. : Proc. 1st Int. Workshop, Menlo Park, Calif., Feb. 1986. New York. — 1986. — P. 326–341.

84. Chamberlin D. D. A History and Evaluation of System R / Chamberlin D. D., Astrahan M. M., Blasgen M. W., Gray J. N., King W. F., Lindsay B. G., Lorie L. A., Mehl J. W., Price T. G., Putzolu G. R., Selinger P. G., Schkolnick M. D., Slutz D. R., Traiger I. L., Wade B. W., Yost R. A. // Commun. ACM. — 1982. — 24, № 10. — P. 632–646.

85. Chandra A.K. Optimal Implementation of Conjuctive Queries in Relational Databases / A. K. Chandra, P. M. Merlin // Proc. 9th Ann. ACM Symp. Theory of Comput. ; [New York, May 1976]. — New York, 1976. — P. 77–90.

86. Chen J. S. J. Optimizing Joins in Fragmented Database Systems on a Broadcast Computer Networks / J.S.J. Chen, V.O.K. Li // 7th Int. Conf. Distrib. Comput. Syst. : [Berlin, Sept. 21–25, 1987]. — Merlin Washington, 1987. — P. 338–345.

87. Cornell D.W. A Vertical Partitioning Algorithm for Relational Databases / D. W. Cornell, P. S. Yu // 3rd Int. Conf. Data Eng. :

[Los Angeles, Ca, Febr. 3–5, 1987]. — Proc. Washington, 1987. — P. 30–35.

88. Date C. J. An Introduction to Database Systems / C.J. Date // V.1. 4th ed.- Reading. — Mass. : Addison-Wesley, 1984. — 639 p.

89. Dayal U. Of Nests and Trees: A Unified Approach to Processing Queries That Contain Nested Subqueries, Aggregates, and Quantifiers / U. Dayal // Proc. 13th Int. Conf. Very Large Data Bases. — Brington, England, Sept. 1987. — Los Altos, Calif., 1987. — P. 197–208.

90. DeWitt D. Multiprocessor Hash-Based Join Algorithms / D. DeWitt, R. Gerber // Proc. 11th Int. Conf. Very Large Data Bases. — Stockholm, Sweden, Aug. 1985. Palo Alto, Calif., 1985. — P. 151–164.

91. Howard P. Энциклопедия малого бизнеса, или Как начать свое дело : пер. с англ. / Dr. G. Howard Poteet, T.A Слука // Книга Одиссей. — М., 1994. — 286 с.

92. Hackathorn D. Reinventing Enterprise Systems Via Data Warehousing [Електронний носій] / D. Hackathorn. — Washington, DC: The Warehousing Institute Annual Conference. — 1995. — P. 29–53. — Режим доступу : citforum.ru/database/articles/art_11.shtml

93. Jarke M. Common Subexpression Isolation in Multiple Query Optimization / M.Jarke // Query Processing in Database Systems. — New York : Springer, 1985. — P. 191–205.

94. Jarke M., Koch J. Query Optimization in Database Systems /Jarke M., Koch J.// ACM Comput. Surv. — 1984. — 16, № 2. — P. 111–152.

95. Jhingram A. A Performance Study of Query Optimization Algorithms on a Database System Supporting Procedural Objects / A. Jhingram // Proc. 14th Int. Conf. Very Large Data Bases : [Los An-

geles, Calif., Aug.-Sept. 1988]. — Los Altos, Calif., 1988. — P. 88–99.

96. Kim W. Global Optimization of Relational Queries: A First Step / W. Kim // Query Processing in Database Systems. — New York : Springer, 1985. — P. 206–216.

97. Kim W. On Optimizing an SQL-Like Nested Query / W. Kim // ACM Trans. Database Syst. — 1982. — 7, № 3. — P. 443–469.

98. King J.J. QUIST: A System for Semantic Query Optimization in Relational Databases / J.J. King // Proc. 7th Int. Conf. Very Large Data Bases : [Cannes, France, Sept. 3-11, 1981]. — New York, 1981. — P. 510–517.

99. Kooi R. Query Optimization in INGRES / R. Kooi, D. Frankforth // IEEE Database Eng. Bull. — 1982. — 5, № 3. — P. 2–5.

100. Lee M. Implementing an Interpreter for Functional Rules in a Query Optimizers /Lee M., Freytag J., Lohman G. // Proc. 14th Int. Conf. Very Large Data Bases : [Los Angeles, Calif., Aug.-Sept. 1988]. — Los Altos, Calif., 1988. — P. 218–229.

101. Lohman G. M. Optimization of Nested Queries in a Distributed Relational Database / Lohman G.M., Daniels D., Haas L.M., Kistler R., Selinger P.G. // Proc. 10th Int. Conf. Very Large Data Bases : [Singapore, Aug. 27–31, 1984]. — New York, 1984. — P. 403–415.

102. Lohman G. M. Query Processing in R* / Lohman G.M., Mohan C., Haas L.M., Lindsay B.G., Selinger P.G., Wilms P.F., Daniels D. // Query Processing in Database Systems. — New York, 1985. — P. 31–47.

103. Lorie R. A. An Access Specification Language for a Relatiomal Data Base System / R.A. Lorie, J.F. Nilsson // IBM J. Res. and Dev., 1979. — 23, № 3. — P. 1–13.

104. Lu H. Some Experimental Results on Distributed Join Algorithms in a Local Network / H. Lu, M. Carey // Proc. 11th Int. Conf. Very

Large Databases : [Stockholm, Sweden, Aug. 1985]. — Los Altos, Calif., 1985. — P. 425–432.

105. Mackert L. Optimizer Validation and Performance Evaluation for Distrib-uted Queries / L. Mackert, G. R. Lohman // Proc. 12th Int. Conf. Very Large Data Bases : [Kyoto, Japan, Aug.]. — Los Altos, Calif., 1986. — P. 149–159.

106. Mackert L. R* Optimizer Validation and Performance Evaluation for Local Queries / L. Mackert, G. Lohman // Proc. ACM SIGMOD Int. Conf. Manag. : [Washington, D.C., May 28-30, 1986]. — New York, 1986. — P. 173–180.

107. Miyao J. Optimization of Multiple Queries in Relational Database Systems / Miyao J., Tominaga K., Kikuno T., Yoshida N. // Syst. and Comput. in Japan, 1988. — 19, № 4. — P. 56–65.

108. Motzkin D. Database Performance Optimization / D. Motzkin // AFIPS Conf. Proc.: Nat. Comput. Conf. ; [Chicago, Ill., July 15–18, 1985]. — Reston, Virg., 1985. — P. 555, 557–566.

109. Otoo E. J. Improving Semi-Join Evaluation in Distributed Query Processing / Otoo E.J., Santoro N., Rotem D. // 7th Int. Conf. Distrib. Comput. Syst. : [Berlin, Sept. 21–25, 1987]. — Washington, 1987. — P. 554–561.

110. Parsaye K. New Realms of Analisis: Surveying Decision Support / K. Parsaye // Database Programming & Design. — 1996. — № 4. — P. 26–33.

111. Pasyeka M. Mathematical Model of Adaptive Knowledge Testing / Pasyeka M., Sviridova T., Kozak I. // Perspektive technologies and metods in mems design, MEMSTECH 2009 : [22–24 квітня]. — Львів-Поляна, 2009. — P. 96–97.

112. Peleshko D. D. SQL-queries optimization. / D.D. Peleshko, M.S. Pasyeka // MS'2001 International Conference on Modeling &Simulation Proccedings. — Lviv, Ukraine, 2001. — P. 184–186.

113. Rashkevych Y.M. Optimization search process in database of learning system / Rashkevych Y.M., Peleshko D.D., Pasyeka M.S. // IEEE International Workshop on Intelligent data acquisition and advanced computing systems: technology and application ; [Lviv, Ukraine, 8–10 september]. — Lviv, 2003. — P. 358–356.

114. Rothnie J.R. Evaluating Inter-Entry Retrieval Expressions in a Relational Data Base Management System / J.R. Rothnie // AFIPS Conf. Proc. : Nat. Comput. Conf., Anaheim, Calif., May 1975. — Montvale, 1975. — P. 152–159.

115. Satoh K. Local and Global Query Optimization Mechanisms for Relational Databases / Satoh K., Tsuchida M., Nakamura F., Oomachi K. // Proc. 11th Int. Conf. Very Large Databases, Stockholm, Sweden, Aug. 1985. — Los Altos, Calif., 1985. — P. 405–417.

116. Selinger P.G. Acess Path Selection in a Relational Database Management System / Selinger P.G., Astrahan M.M., Chamberlin D.D., Lorie R.A., Price T.G. // Proc. ACM SIGMOD Int. Conf. Manag. Data, Boston, Mass., May 30–June 1, 1979. New York, 1979. — P. 23–34.

117. Sellis T. Global Query Optimization / T. Sellis // Proc. ACM SIGMOD Int. Conf. Manag. : Washington, D.C., May 28–30, 1986]. — New York, 1986. — P. 191–205.

118. Sellis T. Multiple-Query Optimization / T. Sellis // ACM Trans. Database Syst. — 1988. — 13, № 1. — P. 23–52.

119. Shenoy S.T. A System for Senantic Query Optimization / S.T. Shenoy, Z.M. Ozsoyoglu // Proc. ACM SIGMOD Int. Conf. Manag. : [San Francisco, Calif., May 1987]. — New York, 1987. — P. 181–195.

120. Smith J.M. Optimizing the Performance of a Relational Algebra Database Interface / J.M. Smith, P.Y. Chang // Commun. — 1975. — 18, № 10. — P. 568–579.

121. Stonebraker M. Implementation of Integrity Constraints and Views by Query Modification / M. Stonebraker // Proc. ACM SIGMOD Int. Conf. Manag. : [San Jose, Calif., May 23-26, 1975]. — New York, 1975. — P. 65–78.

122. Whang K.Y. Constructing Cost Formulas for Relational Database Query Optimizers: A Tutorial / K.Y. Whang // TENCON'87 : IEEE Reg. 10th Conf. : [Seoul, Aug. 25–28, 1987. Proc. Vol. 1]. — New York, 1987. — P. 132–141.

123. Whang K.-Y. Index Selection in Relational Databases / K.-Y. Whang // Found. Data Organ. Proc. Int. Conf. : [Kyoto, Japan, May 22–24, 1985]. — New York; London, 1987. — P. 487–500.

124. Wong E. Decomposition - A Strategy for Query Processing / E. Wong, K.Youssefi // ACM Trans. Database Syst. — 1976. — 1, № 3. — P. 223–241.

125. Yao S. B. Approximating Block Acess in Database Organizations / S.B. Yao // Commun. ACM. — 1977. — 20, № 4. — P. 260–261.

126. Yao S.B. Optimization of Query Evaluation Algorithms / S.B. Yao // ACM Trans. Database Syst. — 1979. — 4, № 2. — P. 133–155.

127. Yu C.T. Algorithms to Process Distributed Queries in Fast Local Networks /Yu C.T., Gun K.-C., Zhang W., Templeton M., Brill D., Chen A.L.P.// IEEE Trans. Comput. — 1987. — 36, № 10. — P. 1153–1164.

Наукове видання

Дурняк Богдан Васильович
Пасєка Микола Степанович
Майба Тарас Миронович

УПРАВЛІННЯ ЗАПИТАМИ В СИСТЕМАХ ДОКУМЕНТООБІГУ

Монографія

Обкладинка *В. І. Сабат*
Верстання *А. І. Шустакевич*
Редактор *У. Р. Чубай*
Коректор *О. В. Музичка*

Українська академія друкарства
79020, м. Львів, вул. Підголоско, 19
Свідоцтво про внесення до Державного реєстру
ДК № 3050 від 11.12.2007 р.

Підписано до друку 31.03.2016 р. Формат 60x84/16.
Умов. друк. арк. 11,16. Обл.-вид. арк. 12,64.
Друк офсетний. Тираж 300 примірників.
Зам. № _____.

Віддруковано в НВЛПТ УАД
79008, м. Львів, пл. Митна, 1