ІВАНО–ФРАНКІВСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
НАФТИ І ГАЗУ

На правах рукопису

ПІТУХ ІГОР РОМАНОВИЧ

УДК 681.215; 621.865

МЕТОДИ ОРГАНІЗАЦІЇ РУХУ ДАНИХ В РОЗПОДІЛЕНИХ КОМП'ЮТЕРНИХ СИСТЕМАХ НА ОСНОВІ МАТРИЧНИХ МОДЕЛЕЙ

Спеціальність 05.13.13 – обчислювальні машини, системи та мережі

Дисертація на здобуття наукового ступеня кандидата технічних наук

Науковий керівник
Николайчук Ярослав Миколайович
доктор технічних наук, професор

Івано–Франківськ– 2007

ЗМІСТ

РОЗДІЛ 3

РОЗДІЛ 4

РОЗДІЛ 5

РЕАЛІЗАЦІЯ В ПРОМИСЛОВОСТІ РОЗРОБЛЕНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ОРГАНІЗАЦІЇ РУХУ ДАНИХ В РКС.....123

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

КС – комп’ютерна система

МПЗ – модель подання знань

КМ – комп’ютерна мережа

МРД – модель руху даних

ММ – матрична модель

ММРД – матрична модель руху даних

СО – системний об’єкт

СПД – система передавання даних

Р – процесор

Д – дані

ОУ – об’єкт управління

О – оператор

МММ – модифікована матрична модель

ТММ – трьохвимірна матрична модель

ЕРД –епюра руху даних

ΔЕРД – диференціальна ЕРД

$\int$ ЕРД – інтегральна ЕРД

GЕРД – глобальна ЕРД

ДММРД – двовимірна модифікована модель руху даних

ОЗП– оперативно запам’ятовуючий пристрій

ПЗП – постійний запам’ятовуючий пнристрій

ТЧБ – теоретико– числовий базис

ЛСІМ – логіко–статистична інформаційна модель

ДІКМ – диференціально–імпульсна кодова модуляція

СЧГ – суміщений часовий граф

РКС – розподілена КС

ВСТУП

Актуальність теми. Сучасні досягнення в галузі комп'ютерної техніки, систем автоматизованого проектування, створення та застосування розподілених комп’ютерних систем (РКС) потребують відповідного вдосконалення методів організації руху даних, представлення та формалізованого опису моделей руху даних. Тому актуальною науковою задачею є розроблення ефективних методів формалізації руху даних, розвиток теоретичних основ побудови моделей та створення програмних засобів діагностики РКС реального часу, особливо це стосується комп’ютерних мереж з розпаралеленими інформаційними потоками.

Питаннями розвитку теорії проектування та оптимізації комп'ютерних мереж з глибоким розпаралеленням інформаційних потоків активно займалися відомі зарубіжні вчені: В.Столлінгс, Е. Таненбаум, Дж.Мартін та інші. При цьому американський системотехнік Дж.Мартін, який ввів поняття одиниці руху даних, під цим поняттям розуміє формалізацію та організацію процесу формування, передавання, цифрового оброблення, ідентифікації, зберігання та використання інформаційних даних в комп’ютерних системах (КС). Значний внесок у розвиток теорії оброблення інформації, в тому числі, в обчислювальних середовищах, а також в комп'ютерних мережах, внесли українські вчені: Грицик В.В., Палагін О.В., Николайчук Я.М., Стеклов В.К., Локазюк В.М., Широчин В.П.

Розвиток інформаційних технологій формалізації руху даних на основі моделювання системних функцій комп'ютерних мереж являє собою складну багатокритеріальну задачу, яка вирішується на базі теорії графів, мереж Петрі, дивергенції та розрізів потоків даних, алгоритмів Форда, Фалкерсона, тупикових потоків, найкоротших шляхів, локально-максимального збільшення, порозрядного скорочення неузгодженостей, моделей на основі матриць суміжностей та інциденцій, а також теорії інтервалів.

В той же час, названі результати розвитку технології проектування КС в недостатній мірі або зовсім не враховували економічні аспекти проектних та експлуатаційних рішень, а також ступінь використання системних ресурсів мережевих об'єктів.

Таким чином, розроблення та вдосконалення методів організації руху даних в РКС, а також інформаційних технологій проектування та діагностики їх ефективності є актуальною науковою задачею.

Зв’язок роботи з науковими програмами, планами, темами.

Представлені в дисертації дослідження виконані в рамках плану наукових досліджень, які проводяться на факультеті комп'ютерних інформаційних технологій кафедрою комп'ютерних технологій в системах управління та автоматики Івано-Франківського національного технічного університету нафти і газу (ІФНТУНГ) за держбюджетними темами: Д-3-07-П “Науково - методологічні основи діагностування і управління у нафтогазовій галузі для оптимізації витрат енергоресурсів” (номер державної реєстрації 0107U001560) та “Розробка методологічних основ (теорії, моделей, алгоритмів, процедур і технічних засобів) діагностування і автоматизованого управління об’єктами нафтового комплексу України” (номер державної реєстрації ТЗ ИГАУ-573/99).

Мета і задачі дослідження. Метою роботи є розвиток методології і розроблення теоретичних засад формалізації та організації руху даних в розподілених комп'ютерних системах, які забезпечують спрощення процедури проектування, аналізу та діагностики комп'ютерних систем шляхом побудови сукупності моделей руху даних на основі базової матричної моделі КС.

У відповідності з поставленою метою дисертаційна робота включає розв'язання таких задач:

– аналіз існуючих технологій проектування, діагностування та моделювання РКС;

– формалізація характеристик системних об'єктів РКС;

– вдосконалення та підвищення інформативності матриць інциденцій шляхом конкретизації атрибутики джерел інформації, пунктів обробки та зберігання даних;

– обґрунтування сукупності та розроблення інформаційних технологій побудови похідних моделей руху даних, які будуються на основі матричної моделі КС;

– дослідження та формалізація матричних моделей архітектур РКС;

– розроблення теоретичних основ та принципів побудови епюр собівартості руху даних в КС;

– розроблення методів побудови тривимірних та модифікованих двовимірних матричних моделей;

– розроблення програмних засобів проектування, аналізу та діагностики комп'ютерних систем на основі моделей руху даних.

Об'єктом дослідження є процеси формування, передавання, оброблення та використання даних у розподілених комп'ютерних системах на основі інформаційних технологій побудови моделей руху даних.

Предмет дослідження - методи організації та моделі руху даних в розподілених комп'ютерних системах.

Методи дослідження базуються на використанні теорії інформації, теорії складних систем, теорії моделювання, теорії графів, теорії кореляційного аналізу та теоретичних основ кодування джерел інформації. Розроблення та реалізація інформаційних технологій проектування та аналізу КС здійснювалась з використанням систем автоматизованого проектування.

Наукова новизна одержаних результатів визначається розробкою ефективних методів організації руху даних на етапах проектування, аналізу та діагностики розподілених комп'ютерних систем, які базуються на використанні методів формалізації архітектур КС, побудови двовимірних, тривимірних та модифікованих двовимірних моделей руху даних та на їх основі реалізація сукупності похідних моделей руху даних, в тому числі таких, які враховують коефіцієнт використання ресурсів і собівартість руху даних.

1. Вперше отримані аналітичні вирази та розроблена методологія обґрунтування критеріїв якості розподілених комп'ютерних систем на основі оцінки ентропії сукупності факторів, математичного сподівання та середньоквадратичного відхилення експертних оцінок якості.

2. Запропоновані функціонали характеристик системних об'єктів КС на основі глобальної моделі, яка включає процесори, дані, систему передавання даних, об'єкти управління (ОУ) та оператори, які об'єднані між собою через інтерфейсні зв'язки. Розроблені функціонали включають параметри часу функціонування, об'єм пам'яті, ресурси зчитування та запам'ятовування даних, швидкість обміну даними та ймовірності помилок в каналах зв'язку, а також характеристики моделей об'єктів управління, що дозволило суттєво спростити процеси проектування та оптимізації КС на основі моделей руху даних.

3. Отримала подальший розвиток теорія графів в частині розширення та конкретизації атрибутів направлених графів та матриць інциденцій шляхом розробки продукційних моделей подання знань на основі матричних моделей руху даних, що дозволило розробити сукупність похідних моделей руху даних і суттєво упростило інженерні розрахунки по проектуванню, діагностиці та оптимізації комп'ютерних систем.

4. Вперше розроблено метод побудови тривимірних та двовимірних модифікованих моделей руху даних на основі додатково введених атрибутів оцінки ступеня використання ресурсів в активних вузлах комп'ютерної системи, що дозволило підвищити якість моделювання комп'ютерних систем та практичну результативність діагностики їх ефективності та оптимізації характеристик.

Практичне значення одержаних результатів:

– розроблено програмне забезпечення для побудови розширеної сукупності моделей руху даних на основі архітектури КС та відповідної двовимірної модифікованої моделі руху даних, що дозволило автоматизувати процеси проектування КС та знизити їх вартість;

– основні результати дисертаційної роботи впроваджено в Інституті мікропроцесорних систем керування об'єктами електроенергетики (м. Львів), а

саме, інформаційну технологію та програмні засоби кореляційного аналізу в процесі експлуатації енергетичних об’єктів, що дозволило підвищити ефективність виявлення передаварійних та попередження аварійних станів об’єктів електроенергетики. Математичне та програмне забезпечення побудови інформаційних моделей технологічних об’єктів та модифікована матрична модель руху даних впроваджені на підприємстві "Бучач-Цукор", що дозволило покращити ефективність формування потоків та оброблення даних, а в результаті підвищити економічну ефективність використання інформаційно-керуючої КС.

– крім того, результати дисертаційної роботи використані та впроваджені в навчальному процесі на кафедрі комп’ютерних технологій в системах управління та автоматики Івано-Франківського національного технічного університету нафти і газу при проведенні лекційних, лабораторних занять та при виконанні курсових робіт з дисциплін: „Основи побудови АСУ", „Низові обчислювальні мережі".

Особистий внесок здобувача. Всі основні результати дисертаційного дослідження отримано автором особисто. У друкованих працях, опублікованих у співавторстві, автору належить:

– принципи формалізації атрибутів, інформаційна технологія та методи побудови тривимірних матричних моделей руху даних, які дозволили розширити функціональні можливості моделей КС за рахунок врахування ступеня використання ресурсів системи в активних вузлах КС [31];

– методологія реалізації стратегії проектування КС на основі характеристик: готовності підприємства до реалізації функцій; затрат на постановку та запуск функцій; економічного ефекту від впровадження; затрат часу на реалізацію функцій, що дозволило оптимізувати процедури впровадження комп'ютерної системи [37];

– методи та інформаційні технології організації руху даних в РКС на основі матричних моделей, які дозволили збільшити ступінь формалізації та адекватності моделей руху даних з врахуванням використання ресурсів та собівартості руху даних [80];

– обґрунтовані та викладені теоретичні засади побудови кореляційних та логіко-статистичних інформаційних моделей цифрової обробки даних на низових рівнях КС, що дозволило суттєво зменшити обсяг інформаційних потоків на виході активних вузлів цифрової обробки даних КС [79].

Апробація результатів дисертації. Основні результати дисертаційної роботи доповідались на:

– міжнародній конференції “Proceedings of International Conference on Modern Problems of Telecommunications, Computer Science and Engineers Training”. – Львів–Славське, 2000;

– міжнародній конференції “Вимірювальна та обчислювальна техніка в технологічних процесах” – Хмельницький, 2000;

– VII міжнародній конференції CADSM 2003 “Досвід розробки та застосування приладо–технологічних САПР в мікроелектроніці”, – Львів–Славське, 2003;

– міжнародній конференції „Сучасні проблеми радіоелектроніки, телекомунікацій та комп’ютерної інженерії” TSET 2004, – Львів–Славське, 2004;

– VIII міжнародній конференції CADSM 2005 „Досвід розробки та застосування приладо–технологічних САПР в мікроелектроніці”, – Львів–Свалява, 2004;

– міжнародній науково–практичній конференції „Автоматизація виробничих процесів” МНПК АВТ – 2005 – Хмельницький, 2005;

– міжнародній конференції “Proceedings of the third IEEE workshop on Intelligent data acquisition and advanced computing systems: Technology and applications” IDAACS 2005. – Sofia, Bulgaria, 2005;

– міжнародній конференції „Сучасні проблеми радіоелектроніки, телекомунікації та комп’ютерної інженерії” TSET 2006, Львів–Славське.

Публікації. Матеріали дисертаційної роботи в повному обсязі висвітлені в 13 друкованих працях автора, в тому числі – 5 статей одноосібних, 8 статей у фахових журналах, рекомендованих ВАК України, та 5 у матеріалах та тезах науково-технічних конференцій.

Структура дисертації. Дисертаційна робота складається зі вступу, п’яти розділів, висновків, списку використаних джерел та додатків. Загальний обсяг роботи становить 187 сторінок з них 132 основного тексту, 67 рисунків, 19 таблиць, 3 додатки, список використаних джерел 158 найменувань.

РОЗДІЛ 1
ЗАГАЛЬНА ХАРАКТЕРИСТИКА МЕТОДІВ ОРГАНІЗАЦІЇ РУХУ ДАНИХ ТА МОДЕЛЮВАННЯ СКЛАДНИХ РОЗПОДІЛЕНИХ СИСТЕМ

1.1. Аналіз досліджень в галузі організації руху даних, моделювання та діагностування комп'ютерних систем

Сучасні комп'ютерні системи, які належать до класу складних систем, знаходять все ширше застосування в усіх галузях промисловості. Широкомасштабне впровадження таких систем забезпечує можливості глобальної інформатизації всіх сфер діяльності людини [4, 5, 18, 22, 46, 60, 72].

Світова практика створення та аналізу складних систем базується на фундаментальних теоретичних дослідженнях відомих зарубіжних та українських вчених, які виконувались починаючи з 70–х років минулого століття. Вирішальний вклад в теорію систем внесли К.Шеннон [81], який розробив теорію інформації, Н.Вінер та Б.М.Маліновський [82, 83], які розробили теорію кібернетичних систем, Т.Харрісон та В.М.Глушков [84, 85], які розробили теоретичні основи побудови автоматизованих систем управління, Р.Фрітч та П.Орнацький [86, 87], які розробили основи теорії інформаційно–вимірювальних систем, Дж.Мартін та С.Т.Пуртов [17, 78], які розробили системотехніку проектування комп'ютерних систем, А.В.Палагін та Я.М.Николайчук [16, 46, 88], які розробили основи та принципи реалізації розподілених та низових комп'ютерних систем реального часу.

Успішна розробка складних комп'ютерних систем в значній мірі пов'язана з ефективним застосуванням новітніх інформаційних технологій, САПР та використанням прогресивних засобів моделювання. Процеси проектування комп'ютерних систем мають такі специфічні особливості [2, 20, 47, 55, 56, 58]:

– відсутність моделі об'єкта, який проектується;
– фрактальність характеристик об'єкта, які одночасно є елементом складної системи і складним модулем більш низьких рівнів системи.

У зв’язку з цим процеси проектування, як правило, носять ітераційний багатоваріантний характер.

Кожна проектна процедура складається з елементарних проектних операцій, які включають:

- розв’язання математичних задач;
- побудову графічних залежностей;
- побудову алгоритмів функціонування елементів системи.

Кожен конкретний процес проектування має свою стратегію і технологію.

Стратегія проектування включає визначення доцільної послідовності операцій з метою розв’язання поставленої задачі. Стратегії проектування можуть бути лінійними, циклічними, розгалуженими та адаптивними. Найбільш результативна адаптивна стратегія [3, 10, 14, 32, 72], яка дозволяє оптимізувати послідовність проектних операцій, залежно від результату попередніх етапів проектування.

Для успішного проектування складних багаторівневих розподілених систем необхідне застосування методології, що використовує фундаментальні положення теорії складних систем, теорії багаторівневих ієрархічних систем, математичної логіки і теорії алгоритмів, а також теорії інформації, рішень та моделювання.

В той же час, як показано в роботах [10, 11, 18, 66, 69, 129, 141, 144], моделювання та діагностування комп’ютерних систем реального часу є більш складним завданням, ніж моделювання систем, оскільки в цьому разі необхідно враховувати набагато більше число факторів і забезпечити виконання багатьох суперечливих вимог. При аналізі роботи системи, що складається з багатьох окремих компонентів, які функціонують в реальному часі, складно вибрати оптимальні методи операційного аналізу, що орієнтовані на обчислювальні системи, а також побудувати сукупність необхідних моделей, які б адекватно описували їх функції.

Створення апаратних, програмних та діагностичних засобів КС є однією з актуальних задач вдосконалення та оптимізації характеристик КС. Моделювання, контроль та діагностування апаратних і програмних засобів РКС реалізується

найбільш складними інформаційними технологіями. Вирішення цих проблем на високому методологічному та теоретичному рівні достатньо повно відображено в роботах В.М.Локазюка [18, 73], М.П.Дивака [10] та інших вчених в цій галузі. Ефективний рівень реалізації процесів діагностування на структурному рівні базується на теорії графів, орграфів, матриць інциденцій та відповідних інформаційних моделях. Важливим інструментом у ефективному вирішенні задач діагностики РКС є побудова моделей, які базуються на основі теорії логічних граф–схем та блок–схем алгоритмів руху даних [6, 32, 142, 143]. При цьому важливим є ефективне вирішення задачі розробки єдиної методології побудови складних розгалужених блок–схем, які б забезпечували мінімізацію числа операторів, структурну регулярність та мінімізацію числа перевірок умовних операторів при діагностуванні РКС реального часу.

1.2. Інформаційні технології та рівні моделювання складних розподілених систем

Моделювання як технологія розв’язування задач в середовищі комп’ютерних систем широко застосовується під час аналізу і проектування інформаційних систем з метою дослідження їх ефективності, ступеня використання ресурсів, оцінки пропускної спроможності та економічної ефективності руху даних в системах.

З 1952 року існує всесвітнє товариство міжнародного комп’ютерного моделювання SCS, основними завданнями якого є вивчення використання і вдосконалення методів комп’ютерного моделювання. Регіональні ради SCS організовані у США, Канаді, країнах Європи та Китаї [11, 78].

Європейська федерація товариств моделювання EUROSIM, товариство моделювання та технології імітації EUROSIS, а також інститути науки моделювання McLeod активно виконують дослідницькі роботи в галузі розвитку теорії та інформаційних технологій моделювання. У Великій Британії існує ряд груп дослідників в університетах, що працюють в цій галузі, наприклад в

Лондонській школі економіки, Імперіал–коледжі, університетах Варвік, Саумптомському та Ланкастера [48].

Під моделюванням розуміють процес адекватного відображення найбільш істотних характеристик досліджуваного об'єкта з точністю, що необхідна для інженерних потреб. Моделювання також узагальнює формалізований підхід до дослідження складних систем.

Теоретичною базою моделювання є теорія подібності, яка дає можливість математично описати формальні параметри тотожних об'єктів. При цьому можуть бути побудовані функції переходу від параметрів формалізованого об'єкту до параметрів модельованого об'єкту. Таким чином, моделювання – це процес подання об'єкта дослідження адекватною йому моделлю. Проведення експериментів з моделлю є джерелом інформації про об'єкт дослідження. Інакше, модель – це фізична або абстрактна система, що адекватно є об'єктом дослідження.

Розрізняють фізичні та абстрактні моделі [11,17,18] (табл.1.1).

Кожна з цих моделей відбиває лише деякі сторони, тобто проекції, оригіналу об'єкту дослідження, тому на практиці користуються сукупністю моделей.

Класи моделей *Таблиця 1.1*

№ п/п	Назва моделі	Характеристика моделі
1	Фізична	Реалізується сукупністю матеріальних об'єктів у вигляді макету або дослідних взірців
2	Абстрактна	Об'єкт дослідження описується системою понять, а не фізичних елементів (словесний опис, креслення, схеми, алгоритми, програми, аналітичні вирази).
2.1.	Гносеологічна	Базується на теорії пізнання та установленні об'єктивних законів.
2.2.	Інформаційна	Адекватно описує формальні характеристики об'єкту оригіналу в статиці та динаміці.
2.3.	Сенсуальна	Базується на рецепторних характеристиках біологічних видів та людини.
2.4.	Концептуальна	Визначає причинно–наслідкові зв'язки залежно від мети дослідження.
2.5.	Математична	Подається на мові математичних співвідношень, має форму функціонально–кількісних залежностей між параметрами, що враховуються концептуальною моделлю.

Для побудови сукупності моделей таких складних об’єктів, як розподілені комп’ютерні системи, а також в більш загальних випадках проектування складних інформаційних систем, використовують системний підхід [61–64]. Застосування принципів системного підходу дає можливість вирішити задачу побудови складної системи із врахуванням усіх факторів пропорційно їх значимості, на всіх етапах дослідження системи та побудови її моделей. При цьому побудова моделі системи також є системною задачею, при розв’язанні якої використовують велику кількість вихідних даних. Завдяки системному підходу можна не тільки обгрунтувати характеристики реального об’єкта, але й оцінити показники його функціонування для знаходження найбільш ефективного варіанту побудови і оптимізації режимів його функціонування в реальних умовах [66, 67, 70].

У процесі проектування складних систем та моделювання їх елементів і функціональних модулів виконується кілька етапів або рівнів проектування. Так наприклад, як показано В.К. Стекловим [11], для інформаційно - обчислювальних систем етапи проектування включають:

– оцінку продуктивності, економічності та швидкодії процесорів, які є елементом моделювання і виконують окремі операції;
– одержання орієнтованих характеристик на рівні пристроїв;
– уточнення функціональної взаємодії компонентів обчислювальних пристроїв на рівні регістрів та логічних елементів;
– одержання точних часових характеристик та якісних показників (тривалостей і форм перехідних процесів, фронтів сигналів, часових затримок, значень логічних рівнів, завадостійкості) на рівні логічних елементів та вентилів.

Аналогічна етапність процедур проектування повинна бути реалізована при проектуванні складних розподілених комп’ютерних систем, наприклад:

– проектування топографічної структури КС [14, 20];
– визначення числа об’єктів, де відбувається формування, цифрова обробка, архівація та затвердження даних [48];

– проектування трас комунікаційних ліній зв’язку між об’єктами системи [5, 55, 56];

– побудова матриці інциденцій та суміжності взаємодії об’єктів та даних [11, 18];

– визначення номенклатури та типів даних, які формуються у вузлах КС [78];

– уточнення вимог до типів та характеристик ліній зв’язку [5, 57];

– побудова двовимірної моделі руху даних, яка описує символіку об’єктів та структуру руху даних від джерел інформації до пунктів їх затвердження та архівізації [31, 33];

– конкретизація часових характеристик руху даних, які відображаються на матричній моделі руху даних (ММРД) стосовно параметрів початку, тривалості та типів виконуваних операцій в її вузлах [32];

– формальна побудова на основі ММРД сукупності похідних моделей наступних типів [89, 80]:

– модель граф– розгалужене дерево;

– параметрична часова модель;

– структурно–часова модель;

– мережевий графік;

– суміщений часовий граф;

– блок–схема алгоритму руху даних;

– задання проектних значень витрат на реалізацію задач у вузлах ММРД, які включають: затрати на будівництво комп’ютерних центрів, придбання комп’ютерної техніки, периферійних засобів, телекомунікаційного обладнання, навчання операторів, витрати на енергоносії та експлуатацію приміщень, витрати на розхідні матеріали (папір, оптичні диски, флеш пам’ять);

– прогнозування проектних прибутків від експлуатації комп’ютерної системи з конкретизацією для кожного вузла ММРД та врахуванням собівартості руху даних;

– проектування вартісних характеристик циклів руху даних, число яких відповідає всім можливим топологічним схемам руху даних згідно ММРД від джерел інформації до пунктів їх архівації і затвердження;

– побудова сукупності вартісних епюр руху даних, в тому числі, для кожного циклу руху даних: прибутково–затратну епюру руху даних, диференціальну собівартісну, інтегральну собівартісну та для рівня всієї архітектури комп'ютерної системи сумарну глобальну епюру собівартості руху даних. На базі останньої розраховується глобальна оцінка собівартості руху даних проектованої комп'ютерної системи на основі незваженого або зваженого математичного сподівання.

Аналогічна технологія проектування та моделювання характеристик складної комп'ютерної системи виконується на більш високих або низьких рівнях КС [16]. Причому, на найнижчому технологічному рівні системи повинен бути описаний комплекс характеристик об'єкта дослідження або керування, який охоплює наступне [79, 89, 90, 91, 43, 41]:

– клас об'єкта управління по ознаці стаціонарності, квазістаціонарності або нестаціонарності;

– статистичні характеристики станів об'єкта (математичне сподівання, дисперсія, середньоквадратичне відхилення, в тому числі, ковзне та зважене).

– автокореляційні та взаємокореляційні характеристики об'єкта (знакова, релейна, коваріаційна, кореляційна, нормована кореляційна, структурна, модульна та еквівалентності);

– спектральні моделі в різних теоретико–числових базисах (унітарний, Хаара, Крейга, Радемахера, Уолша, Галуа);

– ентропійні моделі згідно інформаційних мір Хартлі, Шеннона, „$3\sigma_x$" та на основі автокореляційних функцій [42, 28];

– логіко–статистичні інформаційні моделі, які реагують на відхилення станів об'єкта від норми по амплітуді, динаміці, фазі, спектру та глобальній дисперсії [42, 89];

– часові та кластерні моделі квазістаціонарних об'єктів ;

– моделі–фрейми, які подають пакети даних, що циркулюють в КМ на рівні об’єкта дослідження, СПД, оператора та бази даних архіву.

Таким чином, методика моделювання безпосередньо залежить від рівня моделювання та ступеня деталізації описування об’єкту, а кожному рівню моделювання відповідає визначений структурний рівень моделювання, який враховує зовнішні впливи на систему.

Класифікують пять основних рівнів моделювання складних систем [11, 18, 30]:

1. Рівень структурного або імітаційного моделювання на основі застосування алгоритмічних моделей, моделюючих алгоритмів, спеціальних мов моделювання, теорії множин, формальних граматик, графів, масового обслуговування та статистичного моделювання.
2. Рівень логічного моделювання баз даних або функціональних схем, елементів та вузлів системи, які будуються з застосуванням апарату двозначної або багатозначної алгебри логіки.
3. Рівень дискретного моделювання для цифрових процесорів на основі прикладної теорії цифрових автоматів та програмних алгоритмів синтезу процесорів на програмованих логічних матрицях.
4. Рівень кількісного моделювання принципових рішень елементів складних систем, моделі яких подаються як системи лінійних та нелінійних алгебраїчних, діофантових або інтегро–диференціальних рівнянь і досліджуються із застосуванням методів функціонального аналізу, теорії полів Галуа, диференціальних рівнянь, статистичних та кореляційних характеристик.
5. Рівень інформаційного моделювання на основі теорії моделей руху даних.

Отримана в результаті проектування ієрархічна система розкриває взаємозв’язок різних сторін описування об’єкта на основі сукупності моделей об’єкта на структурному, логічному, кількісному та інформаційному рівнях.

На структурному етапі моделюються зв’язки належності об’єктів ієрархічної підпорядкованості та інцидентності, суміжності і порядку.

На логічному рівні моделювання множині булевої матриці бінарних відношень або структурного графа відповідають набори логічних відношень між вхідними та вихідними характеристиками, які враховують причинно–наслідкові зв'язки.

Як показано в [11], на структурному рівні моделюється склад елементів об'єкта у вигляді деякої множини елементів $V = (v_1, v_2, ..., v_n)$, на рівні описування об'єкта $B(V) = (b_1, b_2, ..., b_m)$, а також структурними відношеннями між елементами та описуваннями, куди входять бінарні відношення ієрархічного підпорядкування, порядку суміжності, спряжності та функціонального зв'язку.

На структурному рівні, на основі топологічної моделі об'єкта, будується орієнтований граф (орграф) $G(V, B)$, упорядкування якого змістовно описується множиною вершин V, способом дії об'єкта у вигляді множини ребер B. При цьому вершинами орграфа є функціонально закінчені модулі системи, а ребрами b_i інформаційні зв'язки між ними.

Структурні відношення між елементами множини описуються матрицею суміжності [18]

$$\left\lfloor C_{ij} \right\rfloor_V = [n \times n],$$

рядки і стовпчики якої відповідають вершинам орграфа структурної моделі, а її C_{ij}–й елемент – кількості ребер, спрямованих від вершини V_i до V_j. Відношення між елементами множини V і B, що відповідає опису відношень між вершинами і ребрами орграфа, описується у вигляді булевої матриці інциденцій

$$[A_{ij}]_{V,B} = [n \times m],$$

рядки якої відповідають вершинам, а стовпчики– ребрам орграфа, при цьому a_{ij} – елементу присвоюється „+1", якщо V_i – початкова вершина b_j і „–1", якщо V_i – кінцева вершина ребра b_j.

На рис 1.1 показаний орграф з вершинами $V_1 - V_7$ та ребрами $b_1 - b_{12}$

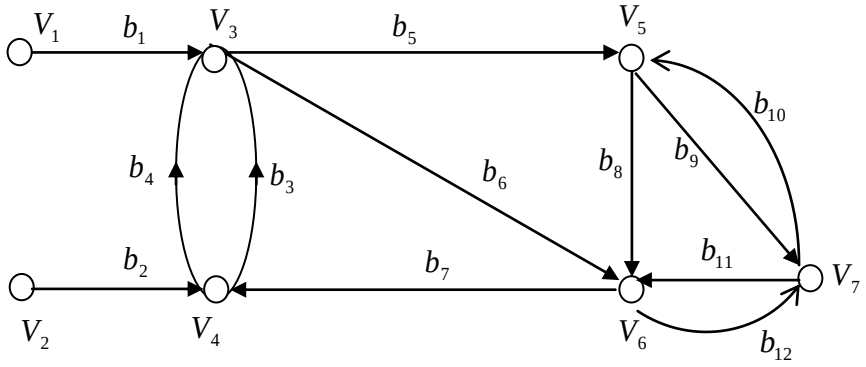


Рис. 1.1. Оргграф булевої матриці інциденцій

Матриця суміжності C_{ij} та інциденцій A_{ij} орграфа мають вигляд :

	V_1	V_2	V_3	V_4	V_5	V_6	V_7
V_1			1				
V_2				1			
V_3				1	1	1	
V_4			1			1	
V_5						1	1
V_6							1
V_7					1		

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}
V_1	–1											
V_2		–1										
V_3	+1		–1	+1	–1	–1						
V_4		+1	+1	–1			–1					
V_5					+1			–1	–1	+1		
V_6						+1	+1	+1			+1	–1
V_7									+1	–1	–1	+1

Отримана матриця інцидентності відображає топологічну архітектуру комп’ютерної системи і направленість інформаційних потоків між її вузлами. Дана матриця є базою для побудови матриці двовимірної моделі руху даних, в якій конкретизуються атрибути комп’ютерної системи у вигляді символів джерел інформації, пунктів цифрового оброблення даних, зберігання та архівації даних [32]. При цьому актуальною задачею є розробка комплексу інформаційних технологій, які б враховували проблемну орієнтацію проектованої КС та необхідну деталізацію сукупності моделей, доступних для інструментарію проектування інженерної практики.

У загальному випадку, розробка теоретичних основ побудови моделей та їх формалізація для конкретних об’єктів промисловості пов’язана зі значними труднощами. Запропонована в роботах [31, 32, 33, 90, 91] інформаційна технологія проектування КС на основі матричних моделей руху даних, в значній мірі спрощує рішення такої задачі шляхом відображення архітектури багаторівневих КС різної топології адекватною багаторівневою моделлю руху даних.

1.3. Характеристика методів моделювання складних систем

В основу моделювання складних систем покладені інформаційні процеси, які базуються на описі характеристик об’єктів дослідження, інформаційних технологій формування, оброблення та зберігання даних з врахуванням процесів передавання даних [11, 48, 68].

Складні системи при проектуванні потребують врахування наступних властивостей [11]:

1. Проблемна орієнтованість, яка визначається ступенем цілеспрямованості та метою функціонування системи.
2. Цілісність і складність, яка потребує опису системи однією моделлю.
3. Невизначеність системи, яка описується оцінкою ентропії, що відображає кількість керуючої інформації, необхідної для досягнення заданого стану системи.
4. Адаптивність, яка враховує можливості пристосування системи до різноманітних зовнішніх факторів у певному діапазоні зміни впливів зовнішнього середовища, що описується імовірнісними моделями живучості та надійності.
5. Універсальність, яка представлена математичними моделями, що мають однакову структуру незалежно від класу об’єктів дослідження.

Як показано в роботі В.М.Локазюка [18], при створенні діагностичних моделей та реалізації інформаційних моделей діагностики РКС додатково необхідно враховувати теорію технічної діагностики, а також одночасно врахувати апаратну програмну складові функціонування КС.

1.4. Постановка задач оптимізації проектування комп’ютерних систем та мереж

Однією з найважливіших задач оптимізації КС є визначення основних показників її функціонування на основі критеріїв якості, які є вихідними параметрами системи $D = \{D_1, ..., D_l\}$. До таких вихідних параметрів належить:

адекватність та достовірність повідомлень станам об’єктів системи, імовірність помилок при передаванні по каналах зв’язку, цифровій обробці та зберіганні даних, середній час безвідмовної роботи, ступінь використання ресурсів у вузлах руху даних, собівартість руху даних.

За умовами задачі проектування, значення параметрів якості системи можуть бути фіксованими або варіювати в певних межах. При цьому показником якості системи може бути така числова характеристика, яка пов’язана з її функцією якості строго монотонною залежністю. Це дозволяє в процесі проектування уникнути прийняття безумовно гірших рішень, а також полегшити та прискорити знаходження кращого рішення.

З врахуванням вищевикладеного, сукупність $D = \{D_1,..., D_l\}$ поділяють на такі підгрупи сукупностей [11]:

– умови, які обмежують функції системи $Y = \{Y_1,..., Y_p\}$;

– обмеження на структуру і параметри проектованої системи $O_s = \{O_{s1},..., O_{sq}\}$;

– показники якості у вигляді векторів, які мають враховуватися в процесі синтезу системи $K =< k_1,..., k_m >$;

– обмеження, які накладаються на показники якості $O_k = \{O_{k1},..., O_{kr}\}$;

– обмеження O_k, що накладається на величини показників якості $k_1,..., k_m$ можуть бути типу рівності, нерівності та функціонального зв’язку.

Слід зауважити, що в залежності від постановки задачі проектування, числові характеристики можна розглядати як показники якості, умови або обмеження. При цьому варіант побудови системи C, що задовільняє сукупності обмежень і умов $\{Y, O_s\}$, називається допустимим. Допустима система, що задовільняє сукупність обмежень O_k називається строго допустимою, тобто $D = \{Y, O_s, K, O_k\}$ [11].

У процесі проектування умовам строго допустимої системи може задовільняти кілька варіантів проектних рішень. При цьому оптимальною вважається система $S_{ОПТ}$, яка забезпечує найкращі значення вектора K показників якості та умову, яка задовільняє наперед вибраний критерій переваги. Така

стратегія проектування КС успішно розвивається в останні роки у світовому масштабі і реалізується на практиці різними проектними фірмами.

Світовий досвід проектування КС різними фірмами (Simens, Motorola, Philips) показує, що оптимізацію варіантів проектних рішень названі фірми виконують тільки на рівні власного обладнання [90, 91, 92].

Проектування та пошук оптимальної системи називається синтезом. Як показано в [11], задача синтезу формулюється наступним чином: синтезувати таку систему S, що задовільняє сукупності обмежень та умов $\{Y, O_s, K, O_k\}$ вихідних даних і має при цьому значення вектора $K =< k_1, ..., k_m >$ показників якості, які найкраще відповідають обраному критерію переваг. Розрізняють наступні технології синтезу складних систем:

– інженерний, заснований на об'єднанні математичних та еврістичних методів;

– векторний, який виконується з врахуванням декількох показників якості на основі вектора $K =< k_1, ..., k_m >$ (даний тип синтезу називають векторною оптимізацією, оптимізацією за векторним критерієм або багатокритерійною);

– скалярний, що проводиться за єдиним показником якості;

– частковий, в якому при синтезі враховуються не всі суттєві показники якості;

– глобальний, що виконується з врахуванням усіх суттєвих показників якості включаючи економічні та конструктивні.

При цьому, якщо не враховано хоча б один із суттєвих показників якості, необхідний для практичного застосування системи, вона не може вважатися оптимальною.

Інженерний синтез комп'ютерної системи та її компонентів містить рішення таких основних задач:

– синтез оптимальної структури мережі;

– вибір оптимальних значень системних характеристик та оптимізація ресурсних параметрів КС;

– вибір оптимального варіанта побудови системи із скінченної кількості проектних рішень.

Таким чином, синтез КС складається з синтезу структури, оптимізації параметрів та дискретного вибору оптимального проектного варіанту.

1.5. Аналіз архітектур та векторний синтез комп'ютерної системи

При рішенні задачі векторного синтезу показники якості $K_1,...,K_i,...,K_m$, доцільно привести до стандартного вигляду, що задовільняє умову [11]:

$$K_i \geq 0 \ (i = 1,2,...,m). \qquad (1.1)$$

При цьому чим менша величина

$$\sum_{i=1}^{m} K_i = \min, \qquad (1.2)$$

тим проектована система краща і може вважатися оптимальною. Оптимальна система вважається ідеальною, якщо $\sum_{i=1}^{m} K_i = 0$.

В окремих випадках умова оптимальності (1.1) сукупності стандартних показників якості на основі експертного обгрунтування може подаватися у вигляді адитивної або мультиплікативної функціональної залежності, а також у вигляді відношення суми позитивних показників якості до суми негативних, тобто таких, що погіршують оптимальні характеристики системи.

У загальному випадку критерій оптимальності системи може бути поданий в умовних одиницях з визначеним однаковим для всіх показників якості діапазоном градації або представлений через функціональні залежності. Наприклад, логарифмічні, квадратичні, експоненціальні та інші функції, які називають цільовими, дозволяють привести глобальний критерій оптимальності до фіксованого діапазону значень. При цьому повинна досягатися максимально можлива різниця між оцінками критеріїв оптимальності різних проектних варіантів системи та реалізація достовірно–значимого розрізнення всієї сукупності варіантів. Більш точною оцінкою критерія оптимальності може

вважатися глобальний критерій якості, в якому враховуються не тільки числові рішення показників якості, але й їх інформаційна, соціальна, стратегічна та інші семантичні значимості, що враховують проблемну орієнтацію проектованої системи. В даному випадку критерій глобальної оптимальності (1.2) проектованої системи отримає вигляд: $\sum_{i=1}^{m} \alpha_i \cdot K_i = \min$, де $\alpha_i \quad (i = 1,2,\ldots,m)$ - відповідні коефіцієнти значимості числових векторних оцінок якості системи.

При аналізі архітектур комп'ютерних систем, який включає 15 варіантів, обгрунтовано застосування вектора із 11 наступних показників якості, кожен з яких заданий у діапазоні $0 \div 9$ експертних одиниць (табл.1.2) [40].

Аналіз архітектур КС *Таблиця 1.2.*

Тип архітектури	**Критерії**											
	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	$K_{ефект}$
1. Зіркова	1	0	5	5	1	0	4	3	0	10	5	2,4
2. Ієрархічна	4	0	4	8	10	0	4	1	0	10	5	2,83
3. Магістральна	4	10	1	7	1	10	4	1	10	10	5	2,5
4. Кільцева	8	0	2	8	1	10	4	1	1	10	5	4
5. Систолічна	10	0	10	9	1	10	4	10	10	10	10	3,4
6. Зірково–магістральна	10	0	8	1	10	10	4	10	10	10	10	8,2
Безпровідні (радіо)												
7. Безретрансляційні	10	0	10	9	1	10	7	10	10	10	10	3,57
8. З пасивним ретранслятором	1	10	2	7	1	0	7	3	0	10	5	1,47
9. З активним ретранслятором	3	0	3	5	1	5	7	4	0	10	7	4,75
10. З активним ретранслятором та кільцевою структурою	2	0	2	5	1	5	7	6	4	10	7	6,14
11. Сотова	9	0	5	7	5	10	7	6	9	10	8	5,41
З відкритим оптичним каналом зв'язку												
12. Високошвидкісні дуплексні	3	0	1	5	0	10	9	1	10	10	5	6,8
13. Середньошвидкісні кільцеві	5	0	2	5	1	10	9	3	4	10	7	6,1
14. Низькошвидкісні кільцеві	8	0	2	5	7	10	9	3	4	10	5	6,2
15. Розгалужені	10	0	2	7	0	8	9	4	5	10	6	5,7

Визначено наступні показники: S_1–надійність, S_2–наявність колізій, S_3–кількість каналів зв'язку та їх вартість, S_4–складність міжрівневих зв'язків, S_5–багаторівневість, S_6 –прямий зв'язок з об'єктом управління, S_7–тип зв'язку, S_8–емерджентність, S_9–наявність прямих зв'язків між процесорами, S_{10}–можливість використання загальносистемних ресурсів, S_{11}–реалізація принципу паралелізму.

В результаті отримаємо формулу [40] :

$$K_{ефект} = \frac{S_1 + S_5 + S_6 + S_7 + S_8 + S_9 + S_{10} + S_{11}}{S_2 + S_3 + S_4}, \quad (1.3)$$

Згідно табл. 1.2. та рівняння (1.3) побудована діаграма векторного синтезу оптимальної системи (рис.1.2).

З рис.1.2 видно, що кращим можливостям вибору оптимальної системи відповідає відповідно краща роздільна здатність побудованої діаграми у діапазоні $0..9$, коли $\left|K_{ei} - K_{ej}\right| = \max$, для $i \in \overline{1,m}$, $j \in \overline{1,n}$, причому $K_{ei} > K_{ej}$.

Правильність вибору формули критерію, при заданій сукупності векторних показників якості, а також характеристик їх сумісної глобальної оцінки у вигляді адитивної, мультиплікативної, відносної або функціональної залежності, може бути оцінена шляхом статистичної обробки діаграми на основі оцінки математичного сподівання M_K та середньоквадратичного відхилення σ_X згідно виразів [40]:

$$M_K = \frac{1}{N}\sum_{j=1}^{N} K_{ej}, \ \sigma_x = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}(K_{ej} - M_K)^2}, \quad (1.4)$$

де N – число варіантів системи, $0 \le K_{ej} \le A$, A–діапазон зміни K_{ej}.

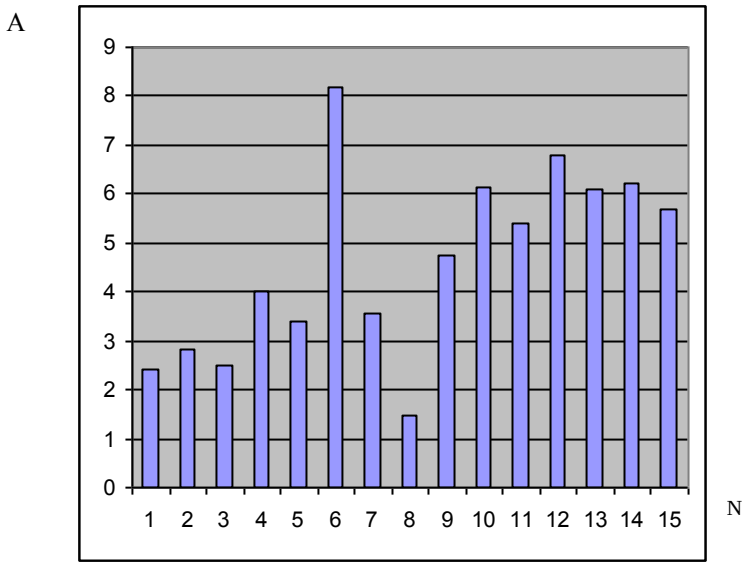


Рис.1.2. Діаграма векторного синтезу оптимальної архітектури системи.

При цьому умовою оптимальності вибору аналітичного виразу коефіцієнта ефективності та числових оцінок вектора показників якості є задоволення наступній системі рівнянь:

$$\left.\begin{array}{c} |M_K - 0{,}5A| = \min \\ \sigma_x = \max \end{array}\right\}. \qquad (1.5)$$

Недопустимо, щоби в результаті зміни компонентів або характеру аналітичного виразу (1.3) результати побудови діаграми (рис.1.2) були суперечливими і не дозволяли однозначно вибрати оптимальний варіант системи, а також змінювати порядок нумерації варіантів оптимальних систем при їх впорядкуванні по зростанню (рис.1.3) [40]. При такій технології обгрунтування критерію ефективності вибору оптимальної системи досягається максимальна ентропія діаграми, показана на (рис.1.2) та (рис.1.3), тобто оптимальні характеристики критерію вибору оптимальної системи можна оцінити на основі інформаційної міри ентропії Хартлі:

$$I_{опт} = \hat{E}[\log_2 A], \qquad (1.6)$$

де $\hat{E}[\cdot]$– цілочисельна функція з округленням до більшого цілого.

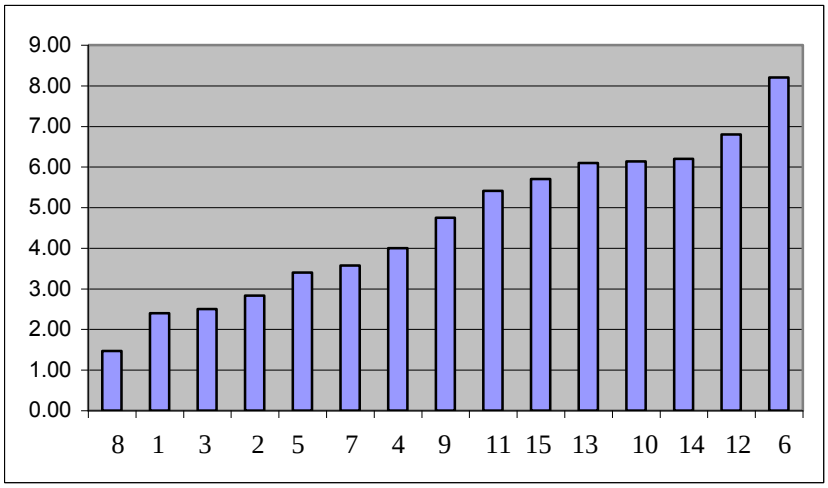


Рис.1.3. Діаграма векторного синтезу оптимальної системи, впорядкована по зростанню.

У випадку, коли для оцінки оптимального проекту системи використовується декілька різних векторів показників якості та різних аналітичних виразів оцінки коефіцієнта ефективності, глобальний критерій вибору оптимальної системи може бути розрахований на основі виразу:

$$\sum_{K=1}^{P} \hat{E}[\log_2 A_K]\max = \max \ , A_K = 3\sigma X_K \ . \qquad (1.7)$$

Базовою теорією для досягнення максимуму (1.3) або (1.7), безсумнівно, може служити теорія інтервального моделювання [10], яка дозволяє шляхом варіації числових значень показників якості обгрунтувати умови досягнення максимуму в рівняннях (1.3) та (1.7).

1.6. Методи організації руху даних в РКС та постановка задачі досліджень

При дискретному виборі системи з сукупності допустимих оптимальних варіантів, отриманої на основі векторного синтезу, оптимальне рішення отримують у вигляді дискретної множини точок у $m-$ мірному просторі R^m показників якості системи.

Під час оптимізації параметрів системи варіюється сукупність характеристичних параметрів у вигляді вектора $X =< x_1,..., x_n >$, що забезпечує найкраще значення вектора показників якості $K =< k_1,...,k_m >$. В загальному випадку кожен із показників якості $k_1,...,k_m$ може залежати від багатьох параметрів проектованої системи. Ці залежності називають цільовими функціями. При цьому обмеження типу рівностей або нерівностей накладаються не тільки на значення кожного з параметрів окремо, а також і на зв’язки між ними. Таким чином, задача оптимізації параметрів при дискретному виборі складається з двох основних етапів:

1) визначення аналітичних виглядів цільових функцій та функцій зв’язку між компонентами вектора K;

2) оптимізація параметрів системи, що відповідає обмеженням і забезпечує найкраще значення вектора показників якості.

У математичному плані дана задача зводиться до дослідження скінченної кількості цільових функцій вектора параметрів системи X і забезпечення оптимального значення вектора K.

Формування задачі оптимізації параметрів системи у m–вимірному просторі R^m показників якості виконується шляхом оптимізації у трьох просторах показників якості, варійованих параметрів та обмежень. При цьому у просторі показників якості кожній системі відповідатиме одна точка, і навпаки, кожному значенню вектора відповідатиме тільки одна з систем. Врахування простору обмежень дозволяє виділити строго допустиму область, якій відповідає скінченна множина точок строго допустимих варіантів системи.

У загальному випадку задача синтезу може бути абсолютно інваріантною для заданого набору показників якості, обмежень та функцій системи. Тоді задача вибору оптимізованої системи може бути успішно вирішена на основі скорочення або розширення системи, стратегії її довговічності, морального старіння обладнання та інше, що в свою чергу приведе до однозначної різниці між варіантами системи.

При дослідженні та оптимізації процесів оптимізації руху даних в РКС актуальним є вирішення наступних задач:

– аналіз існуючих технологій проектування, діагностування та моделювання РКС;

– формалізація характеристик системних об’єктів РКС;

– формалізація матриць інциденцій шляхом конкретизації атрибутики джерел інформації, пунктів обробки та пунктів зберігання даних;

– обґрунтування сукупності похідних моделей руху даних, які будуються на основі матричної моделі КС;

– розробка інформаційних технологій побудови сукупності моделей руху даних КС;

– дослідження та формалізація моделей архітектур РКС;

– розробка теоретичних основ та принципів побудови епюр собівартості руху даних;

– розробка методів формалізації та інформаційних технологій побудови трьохвимірних та модифікованих матричних моделей;

– розробка програмних засобів проектування, аналізу та діагностики комп'ютерних систем на основі формалізованої сукупності та інформаційних технологій побудови моделей руху даних.

Результатом успішного вирішення сформульованих інформаційних задач є реалізація наступних методів організації руху даних в РКС, які описуються алгоритмом послідовного виконання системних функцій формування, передавання, оброблення, зберігання і використання даних при їх русі від джерел інформації до пунктів зберігання та затвердження.

У загальному випадку відомі методи організації руху даних в РКС описуються алгоритмом послідовного виконання системних функцій у вигляді:

$$F_{PKC} = F_1[G(V,B)] \Rightarrow F_2\lfloor C_{ij} \rfloor \Rightarrow F_3\lfloor A_{ij} \rfloor_{V,B} \Rightarrow F_4[D,K],$$

де F_1, F_2, F_3, F_4 - відповідно формалізований опис процесів організації руху даних у вигляді подання топології РКС, процесів оброблення даних та їх використання для оптимізації параметрів РКС;

$[G(V,B)]$ - орієнтований граф топологічної моделі;

$V = (v_1, v_2, ..., v_n)$ - множина вершин елементів архітектур;

B - множина ребер, яка представляє рівні описування об'єкта $B(V) = (b_1, b_2, ..., b_m)$;

$\lfloor C_{ij} \rfloor$ - матриця суміжності, яка описує структурні відношення між елементами системи;

C_{ij} - елемент кількості ребер, спрямованих від вершин V_i до V_j;

$\lfloor A_{ij} \rfloor_{V,B}$ - булева матриця інциденцій;

$D = \{D_1, ..., D_l\}$ - сукупність показників критеріїв якості, які поділяються на підгрупи умов, що обмежують функції системи $Y = \{Y_1, ..., Y_p\}$, умов, що обмежують структуру та параметри проектованої системи $O_s = \{O_{s1}, ..., O_{sq}\}$, вектори показників якості, які враховуються в процесі синтезу системи $K =< k_1, ..., k_m >$.

Результатом функціоналу F_4 є визначення параметрів допустимої системи $D = \{Y, O_s, K, O_k\}$, що задовільняє сукупності обмежень O_s та умов Y, причому $\sum_{i=1}^{m} K_i = \min$ відповідає умові оптимальної системи.

В роботі запропоновано методи організації руху даних, які характеризуються розширенням функціональних можливостей проектування та діагностування ефективності РКС та інформаційною технологією проектування РКС на основі вдосконалення атрибутів двовимірних матричних моделей, побудови сукупності похідних моделей, а також реалізацією тривимірних та модифікованих двовимірних моделей руху даних, на основі яких виконується глобальна оцінка собівартості руху даних шляхом побудови сукупності епюр руху даних.

Суть запропонованих методів організації руху даних полягає у послідовному виконанні функцій наступних алгоритмів:

$$F_1[\odot, \bigcirc, \otimes] \Rightarrow \left[\left\{\begin{array}{c} E_p = F(T,V,M,S) \\ E_d = F(T,M,V_R,V_W,S) \\ E_{OУ} = F(T,M,I,S) \\ E_{СПД} = F(V_R,V_W,P_i,S) \\ E_O = F(T,V_R,V_W,S,M) \\ X_{OУ} = F(X(t),M_x,D_x,\sigma_x,R_{xx},R_{xy},M_{ij},S(w),K_{ij},ЛСІМ,I_x) \end{array}\right.\right] \Rightarrow F_3[K_{ed}, P-V] \Rightarrow$$

$$\Rightarrow F_4[M_1, ТМРД, ДММРД, ..., M_n] \Rightarrow F_5\left[\left\{\begin{array}{c} ЕРД \\ \Delta ЕРД(t_j) \\ \int \Delta ЕРД(T)dT \\ \sum_{i=1}^{n} \int \Delta ЕРД(T)dT \end{array}\right.\right] \Rightarrow F_6[G_{КС}],$$

де $\odot$, $\bigcirc$, $\otimes$- відповідно атрибути джерела, пункту оброблення та пункту затвердження даних; $E_p, E_d, E_{OУ}, E_{СПД}, E_O$- відповідно характеристики системних об'єктів РКС (процесорів, даних, об'єктів управління, систем передавання даних, операторів); T- часовий параметр, V-швидкість перетворення даних, M - об'єм використовуваної пам'яті, S- системні ресурси; $X(t)$-поточне значення параметру, M_x-математичне сподівання, D_x-дисперсія, σ_x- середньоквадратичне

відхилення, R_{xx} - автокореляційна функція, R_{xy} -взаємокореляційна функція, M_{ij} - матриця нормованих коефіцієнтів взаємокореляції, $S(w)$ - спектральні моделі, K_{ij} - матриця імовірностей переходу в різні стани, $ЛСІМ$ - логіко-статистична інформаційна модель, I_x - ентропійна модель; K_{ed} - коефіцієнт ефективності руху даних, $P-V$ - собівартість виконання операції в активному вузлі матричної моделі; M_1 - матрична модель руху даних, $ТМРД(S_0, S_i, C_o, C_i, G)$ - тривимірна матрична модель руху даних (S_0 – максимальне число записів, S_i – реальне число записів, C_0 – швидкість створення та передавання даних, C_i – проектна швидкість створення та передавання даних, G – завантаженість), $ДММРД(\alpha_0, \alpha_1, \alpha_2, d_0, d_i)$ -двовимірна матрична модель руху даних (α_0 - ресурси зчитування даних; α_1 -ступінь використання ресурсів зчитування даних; α_2 -ступінь використання ресурсів записів; d_0, d_i - відповідно ресурси та ступінь використання швидкості передавання даних в каналі зв'язку); M_n - похідні моделі; $ЕРД$ -сигнальна ЕРД, $\Delta ЕРД(t_j)$ - диференціальна ЕРД, $\int \Delta ЕРД(T)dT$ -інтегральна ЕРД, $\sum_{i=1}^{n} \int \Delta ЕРД(T)dT$ -сумарна інтегральна ЕРД; $G_{КС}$ -глобальна характеристика ефективності комп’ютерної системи.

Теоретичне обгрунтування реалізації функціоналів, їх моделювання та аналіз, а також ефективності застосування запропонованих методів організації руху даних в РКС є предметом дослідження даної дисертації.

ВИСНОВКИ ПО ПЕРШОМУ РОЗДІЛУ

1. Викладена загальна характеристика теоретичних основ та інформаційних технологій проектування, діагностування та моделювання складних розподілених систем.

2. Досліджені рівні моделювання та інформаційні технології складних розподілених систем.

3. Систематизовані характеристики та властивості методів моделювання складних розподілених систем.

4. Виконана постановка задачі оптимізації проектування комп’ютерних систем на основі синтезу структури, оптимізації параметрів та дискретного вибору оптимального проекту.

5. Досліджена задача векторного синтезу складної системи на основі запропонованої класифікації архітектур, критеріїв оптимальності та обґрунтувано умови оптимальності вибору аналітичного виразу коефіцієнта ефективності та числових оцінок вектора показників якості.

6. Викладені алгоритмічн-функціональні основи відомих і запропонованих методів організації руху даних та поставлена задача дисертаційного дослідження.

РОЗДІЛ 2
ФОРМАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ ХАРАКТЕРИСТИК СИСТЕМНИХ ОБ'ЄКТІВ КС

2.2 Глобальна модель, системні об'єкти, системні функції, функціональні об'єкти комп'ютерних систем

Глобальна модель комп'ютерної системи вперше була запропонована Николайчуком Я.М. у вигляді взаємодії пятьох типів системних об'єктів (рис.2.1) [16].

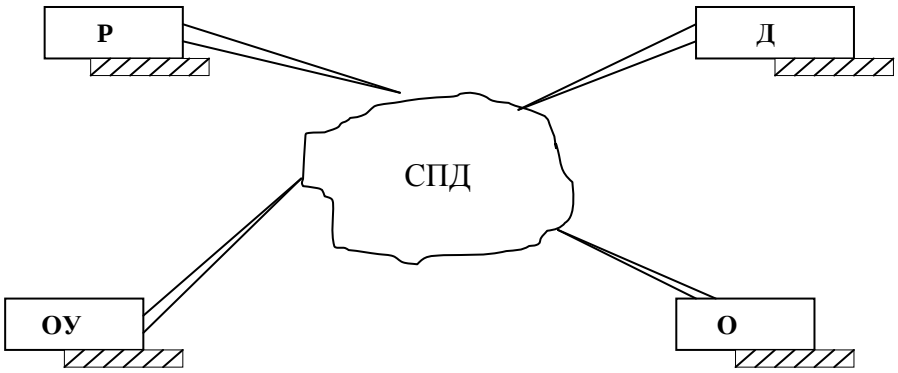


Рис.2.1. Глобальна модель комп'ютерної системи:
Р – процесор, Д – дані, СПД – система передавання даних, ОУ – об'єкт управління, О – оператор.

Кожен з наведених системних об'єктів може виконувати в середовищі КС чотири системні функції:

– формування даних ;

– передавання даних;

– цифрова обробка даних;

– приймання та зберігання даних.

Тобто, кожен з системних об'єктів (СО) може бути одним з функціональних об'єктів наступного типу:

– джерело інформації (ДІ);

– середовище передавання інформації (СПІ);

– середовище цифрової обробки інформації (СОІ);

– приймач інформації (ПІ).

Отже, системні об’єкти КС характеризуються дуальними (поліфункціональними) властивостями, що в значній мірі ускладнює методологію проектування та теоретичні основи оптимізації параметрів КС.

Виходячи з класифікації п’яти системних об’єктів КС, можна побудувати таблицю пар їхньої взаємодії через інтерфейсні засоби комунікацій (табл.2.1).

Таблиця 2.1

	Р	Д	СПД	ОУ	О
Р	Р→Р	Р→Д	Р→СПД	Р→ОУ	Р→О
Д	Д→Р	Д→Д	Д→СПД	Д→ОУ	Д→О
СПД	СПД→Р	СПД→Д	СПД→СПД	СПД→ОУ	СПД→О
ОУ	ОУ→Р	ОУ→Д	ОУ→СПД	ОУ→ОУ	ОУ→О
О	О→Р	О→Д	О→СПД	О→ОУ	О→О

Очевидно, що для вивчення названих інтерфейсних взаємодій СО та використання їх при проектуванні КС необхідно описати взаємодію 25–ти їхніх пар. В той же час, враховуючи, що теорія, методологія та техніка реалізації КС на основі стандартних технічних засобів, міжнародних протоколів та інтерфейсів достатньо повно подана у відповідних виданнях та інструкціях, при проектуванні низових рівнів проблемно–орієнтованих та спеціалізованих КС особливу увагу слід надавати вивченню системних характеристик об’єктів управління та їх взаємодії з іншими об’єктами КС . Дані взаємодії ОУ та СПД з іншими СО відображені в табл.2.1 відповідним фоном. Широка різноманітність реальних ОУ (наприклад, космічні апарати, літаки, підводні та наземні човни, атомні станції, енергетичні та нафтопромислові системи, транспортні засоби, інформаційні системи, телекомунікації та інше) вимагає відповідної проблемної орієнтації та адаптації КС до характеристик ОУ при проектуванні та іналізі діючих КС. Ця адаптація потребує по - новому осмислити формалізацію опису характеристик системних об’єктів [124, 126, 133, 134].

2.2. Формалізація опису характеристик системних об’єктів КС

У загальному випадку ресурсні характеристики СО проектованої КС можуть бути достатньо повно описані функціоналом, який заданий коефіцієнтом Eco та четвіркою параметрів

$$Eco = F(T, V, M, S), \tag{2.1}$$

де T –час використання ресурсу, V –швидкість виконання системних операцій (формування, передавання, цифрова обробка та зберігання даних), M –об'єм використовуваного ресурсу пам'яті, S –системні функції.

При цьому границі зміни параметрів задаються системою нерівностей

$$\begin{aligned} 0 \le T \le T_0;\\ 0 \le V \le V_0;\\ 0 \le M \le M_0;\\ 0 \le S \le S_0, \end{aligned} \tag{2.2}$$

де T_0 – час формування, передавання, цифрової обробки та зберігання даних, використання технічного засобу та інше, V_0 –пропускна здатність каналу зв'язку, максимальна швидкість читання/запису, максимальна частота обміну даними та інше, M_0 – максимальний об'єм пам'яті, що використовується (ОЗП, ПЗП, магнітних, оптичних, та твердих копій носіїв), S_0 – максимальний ресурс системних функцій (операційні системи, пакети прикладних програм тощо).

Якщо задати нормовані границі зміни кожного параметра (2.1) в діапазоні від 0 до 1, то реалізація коефіцієнта Eco отримає вигляд:

$$Eco = F(0.3, 0.4, 0.7, 0.2).$$

Даний коефіцієнт доцільно привести до безрозмірної форми на основі функції адитивності $E_{co} = \dfrac{T + V + M + S}{4}$, що забезпечує діапазон зміни Eco в межах $0 \le E_{co} \le 1$ та відповідає гіпотезі про статистичну незалежність ресурсних параметрів СО (2.1). При цьому економічна собівартість руху даних може бути обчислена на основі рівняння [40]:

$$P_{co} = E_{co} \cdot P_0 - V_0, \tag{2.3}$$

або

$$P_{co} = \frac{T + V + M + S}{4} \cdot P_0 - V_0 \ , \tag{2.4}$$

де P_0, V_0 – відповідно прибутки та затрати на реалізацію функцій СО.

На рисунку 2.2 показані характеристики собівартості руху даних на рівні формалізованого опису системного об'єкту.

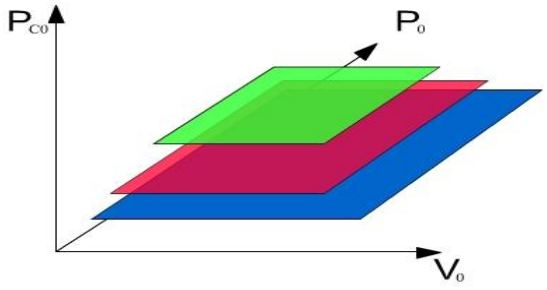


Рис.2.2. Характеристики собівартості руху даних системного об'єкту

Аналогічні формули можуть бути застосовані для інших системних об'єктів, в яких формальні параметри S будуть відрізнятися наступними системними функціями:

P – процесор (апаратне, системне та прикладне програмне забезпечення);

$Д$ – дані (зберігання даних в архівах, способи кодування даних, захист від помилок, захист від несанкціонованого доступу, семантичні властивості даних і.т.д.) ;

$СПД$ – мережеве програмне забезпечення, оптимізація маршрутів передавання даних, використання спецканалів, інформаційна технологія моделювання руху даних ;

$ОУ$ – характеристики стаціонарності, нестаціонарності, квазістаціонарності, інформаційні технології кодування станів та контролю їхнього відхилення від норми, статистичні, кореляційні та ентропійні моделі, функції керування та побудови моделей руху даних;

$О$ – система знань та професійних навиків і.т.д.

Функціонал характеристик системних об'єктів в багатьох випадках доцільно розширити до п'яти параметрів шляхом диференціації параметру швидкості виконання системних операцій, тобто V_R –швидкість запису (вхідний інформаційний потік), V_W –швидкість зчитування (вихідний інформаційний потік), звідки характеристики системного об'єкта будуть описуватися параметрами:

$$Eco = F(T, V_R, V_W, M, S). \qquad (2.5)$$

Характеристика (2.5) дозволяє врахувати асиметричність характеристик швидкодії вхідних та вихідних пристроїв системних об'єктів.

Розроблені аналітичні моделі характеристик системних об'єктів комп'ютерної мережі є базою для теоретичної формалізації руху даних. На основі вказаних моделей проводиться аналіз ефективності та розробка проектів комп'ютерних мереж з проблемною орієнтацією для конкретних технологічних процесів та підприємств [10, 18, 90, 91].

2.3. Системні характеристики процесора

Процесори (Р) містять в своєму складі інтелектуальні оснащені однокристальними спецпроцесорами сенсори, сигнальні процесори, мікро– та міні–контролери, контролери низових мереж, комутаційні процесори, сервери та комп'ютерні кластери [19]. Системні характеристики Р достатньо повно описуються функціоналом (2.1)

$$Ep = F(T, V, M, S),$$

де T – час використання ресурсу, V – швидкість виконання системних операцій V_R / V_W, M – об'єм використовуваного ресурсу пам'яті, S – системні функції операційної системи та прикладного програмного забезпечення.

Параметр часу використання ресурсу T оцінюється у відсоткових значеннях його безперервної роботи протягом доби. Використання ресурсу даної характеристики залежить від типу та призначення процесорів. Наприклад, мікропроцесори, технологічні мікроконтролери та промислові процесори призначені для цілодобової роботи, тобто $T = 1.0$. Для процесорів ЕОМ режим роботи може складати 6, 8 або 16 годин в добу, що відповідає використанню ресурсу відповідно $T_i = 0.25, 0.3, 0.6$.

Характеристика використання об'єму ресурсів пам'яті процесора M також залежить від конкретної інформаційної технології її застосування. Наприклад, мікроконтролери масового застосування на основі 8 та 16 розрядних

мікропроцесорів, які оперують з обмеженою пам'яттю, використовують її ресурси практично на 100 відсотків, тобто $M_i = 1.0$. В процесорах універсального призначення, якими оснащено ПЕОМ, використання ресурсів пам'яті залежить від характеру задач, які вирішує оператор та динамічності їх реалізації в часі, тобто дану оцінку треба розраховувати на основі математичного сподівання :

$$\overline{M}_i = \frac{1}{k}\sum_{i=1}^{k} M_i \,, \qquad (2.6)$$

де M_i – поточні або миттєві коефіцієнти використання ресурсів пам'яті процесора. Аналогічно може бути обчислена характеристика використання ресурсів функціональних характеристик процесора (S_i).

Приклад вхідних та вихідних інформаційних сигналів процесорів поданий в табл.2.2.

Таблиця 2.2

Формувачі сигналів процесора

Формувачі вхідних та вихідних інформаційних сигналів процесора	Характеристики інформаційних потоків
Паралельний порт	1 Гбіт/с
Послідовний порт	100 Мбіт/с
Маніпулятор мишка	8 біт/с
Клавіатура	8–12 біт/с
Модем	10–100 Мбіт/с
Мікрофон	2000 * 8=біт/с 2,4–3,6 кГц
Технологічний сканер	100 Мбіт/с

Аналіз табл.2.2 показує, що швидкість створення повідомлень процесорів може змінюватися в широких межах. Діапазон швидкості обміну даними процесорів може змінюватися в межах $1–10^9$ біт/с.

2.4. Системні характеристики даних

Сучасна систематизація даних охоплює три їх основні класи [45,135, 136]:

– фізичні дані;

– логічні дані;

– віртуальні дані.

Ресурсні характеристики СО „Дані” – представлені часом зберігання, швидкістю запису, швидкістю зчитування, об’ємом та захистом даних від помилок

$$E_d = F(T, M, V_R, V_W, S),$$

де – T – час зберігання, M – об’єм, V_R – швидкість запису, V_W – швидкість зчитування, S – захист від помилок та несанкціонованого доступу.

Фізичні дані – це дані, які представлені в адресному просторі фізичних носіїв (жорстких магнітних дисків, гнучких магнітних дисків, оптичних та лазерних дисків), а також на твердих носіях (документи, таблиці, графіки).

Поняття логічних даних використовується в теорії та практиці опису логічних моделей баз даних, а також в програмних продуктах як формальні параметри.

Віртуальні дані– це такий тип даних, які відсутні в фізичному вигляді в КС і можуть формуватися в процесі рішення задач, розрахунків, а також бути в динамічному стані при передаванні в КМ.

Способи кодування даних визначаються теоретико–числовими базисами (ТЧБ), які застосовуються для їх представлення [106-109]. Найбільш широко вживаними ТЧБ в сучасних КС є наступні базиси: унітарний, Хаара, Крейга, Грея, Радемахера, Крестенсона, Уолша та Галуа, кодові матриці яких подані на рис.2.3 [110].

$$M_{Uni} = \begin{vmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 1 & 0 & \dots & 0 & 0 \\ 1 & 1 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & 1 & \dots & 1 & 1 \end{vmatrix} \quad M_{har} = \begin{vmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{vmatrix} \quad M_{Gr} = \begin{vmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 1 & 1 & 1 \\ 1 & 1 & 1 & \dots & 1 & 1 & 0 \\ 1 & 1 & 1 & \dots & 0 & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \end{vmatrix} \quad M_{Rad} = \begin{vmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 1 \\ 1 & 0 & 0 & \dots & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & 1 & \dots & 1 & 1 \end{vmatrix}$$

а) б) в) г)

$$M_{LibCr} = \begin{vmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 1 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & 1 & \dots & 1 & 1 \\ 0 & 1 & 1 & \dots & 1 & 1 \\ 0 & 0 & 1 & \dots & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{vmatrix} \quad M_{Cres} = \begin{vmatrix} P_1 & P_2 & \dots & P_n \\ 0 & 0 & \dots & 0 \\ 1 & 1 & \dots & 1 \\ 2 & 2 & \dots & 2 \\ 0 & 3 & \dots & 3 \\ 1 & 4 & \dots & 4 \\ 2 & 0 & \dots & 5 \\ 0 & 1 & \dots & 6 \\ \dots & \dots & \dots & \dots \\ a_1 & a_2 & \dots & a_n \end{vmatrix} \quad M_{Gal} = \begin{vmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{vmatrix}$$

д) е) є)

Рис. 2.3. Кодові матриці дискретних базисів: а) – унітарного; б) –Хаара; в) – Грея; г) – Радемахера; д) –Крейга; е) –Крестенсона; є)–Галуа

Кожен з названих базисів характеризується визначеним об'ємом кодової матриці для представлення даних. При цьому найбільш надлишковим базисом є унітарний, в якого кодова матриця $V = N^2$, а число активних кодових елементів $n = N^2/2$, де $N-$ діапазон кодування даних. Аналогічну надлишковість забезпечує базис Хаара, в два рази меншу надлишковість забезпечує базис Крейга, тобто $V = N^2/4$, а $n = N^2/8$. Максимально широке застосування для кодування даних в сучасних КС отримали базиси Радемахера та Крестенсона, в яких $V = N\log_2 N$. Дані базиси відповідно породжують двійкову систему числення та систему числення залишкових класів [9].

Базис Уолша максимально широко використовується в сучасних телекомунікаційних КС [5]. Даний базис породжує систему ортогональних шумоподібних сигналів, які використовуються в сотових системах мобільного зв'язку [28].

Найменшу надлишковість кодування даних забезпечує базис Галуа, кодова матриця якого $V = N$, а $n = N/2$ [44, 152].

Згідно викладеного, характеристики ТЧБ кодування даних, як системного об'єкта, подані в табл.2.3.

Характеристики потоків даних

Таблиця 2.3

Формувачі вхідних та вихідних інформаційних сигналів даних	Характеристики інформаційних потоків даних
Унітарний базис	$V = N^2;\ n = N^2/2$
Базис Хаара	$V = N^2,\ n = N$
Базис Крейга	$V = N^2/4,\ n = N^2/8$
Базис Радемахера	$V = N\cdot\log_2 N,\ n = \frac{N}{2}\log_2 N$
Базис Крестенсона	$V = N\cdot\log_2 N$
Базис Уолша	$V = N^2, n = N^2/2$
Базис Галуа	$V = N,\ n = N/2$

2.5. Дослідження характеристик моделей об'єктів управління

Об'єкт управління адекватно може бути описаний характеристичними параметрами – часом, ентропією, моделлю об'єкта та системними функціями:

$$E_{OY} = F(T, M, I, S),$$

де T – час, M – модель об'єкта, I – ентропія, S – системні функції.

Найважливішими системними характеристиками ОУ є модель об'єкта, ентропія та системні функції.

Найважливіші моделі ОУ представлені в табл.2.4. [41].

Типи моделей *Таблиця 2.4.*

№	Типи моделей ОУ	Аналітичний вираз
1	2	3
1	Сигнальні аналогові	$M = X(t)$
2	Сигнальні дискретизовані і квантовані	$M = X_i$, $i \in \overline{1,n}$, $0 \le x_i \le A$, де X_i –дискретизоване квантоване значення ОУ, n –об'єм вибірки, A –діапазон квантування
3	Дискретні диференціальні	$M = \Delta X_i = X_{i+1} - X_i$, де ΔX_i –перші прирости станів ОУ.

Продовж. табл.2.4.

4	Дискретні інтегральні	$M = \sum_{i=1}^{k} X_i$, де k –число сумувань дискретних станів ОУ.
5	Статистичні:	
5.1	вибіркове математичне сподівання	$M_x = \frac{1}{n}\sum_{i=1}^{n} x_i$
5.2	ковзне математичне сподівання	$M_j = \frac{1}{m}\sum_{i=1+j}^{m+j} X_{i+j}$, $j = 0,1,2,\ldots$ де $j = 0,1,2,\ldots$–дискретний зсув;
5.3	вагове математичне сподівання	$M_v = \sum_{i=1+j}^{m+j} V_{i-j} \cdot X_{i+j}$,

		де V_i –вагова функція;
5.4	дисперсія	$D_x = \frac{1}{n}\sum_{i=1}^{n}(X_i - M_x)^2$
5.5	середньоквадратичне відхилення	$\sigma_x = \sqrt{D_x}$
6	Автокореляційні моделі	
6.1	знакова	$B_{xx}(j) = \frac{1}{n}\sum_{i=1}^{n} sign\, \overset{o}{x}_i \cdot sign\, \overset{o}{x}_{i+j}$ $sign\, \overset{o}{x}_i = \begin{cases} +1, \overset{o}{x}_i \geq 0 \\ -1, \overset{o}{x}_i < 0 \end{cases};$
6.2	релейна	$H_{xx}(j) = \frac{1}{n}\sum_{i=1}^{n} \overset{o}{x}_i \cdot sign\, \overset{o}{x}_{i+j}$
6.3	коваріаційна	$K_{xx}(j) = \frac{1}{n}\sum_{i=1}^{n} x_i \cdot x_{i+j}$
6.4	кореляційна	$R_{xx}(j) = \frac{1}{n}\sum_{i=1}^{n} \overset{o}{x}_i \cdot \overset{o}{x}_{i+j}$
6.5	нормована кореляційна	$\rho_{xx}(j) = \frac{R_{xx}(j)}{D_{xx}}$
6.6	структурна	$C_{xx}(j) = \frac{1}{n} \cdot \sum_{i=1}^{n}(x_i - x_{i+j})^2$
6.7	модульна	$G_{xx}(j) = \frac{1}{n} \cdot \sum_{i=1}^{n}\lvert x_i - x_{i+j}\rvert$

Продовж. табл.2.4.

6.8	нормована модульна	$g_{xx}(j) = \frac{C_{xx}(j)}{M_x} - M_x$
6.9	еквівалентна	$F_{xx}(j) = \frac{1}{n} \cdot \sum_{i=1}^{n} \overset{\vee}{Z}_{ij};$ $\overset{\vee}{Z}_{ij} = \begin{cases} x_i, & x_i < x_{i+j} \\ x_j, & x_i \geq x_{i+j} \end{cases}.$
7	Взаємокореляційні моделі між двома параметрами ОУ	

7.1	взаємознакова	$B_{xy}(j)=\frac{1}{n}\sum_{i=1}^{n} sign\, \overset{o}{x}_i \cdot sign\, \overset{o}{y}_{i+j}$
7.2	взаєморелейна	$H_{xy}(j)=\frac{1}{n}\sum_{i=1}^{n} \overset{o}{x}_i \cdot sign\, \overset{o}{y}_{i+j}$
7.3	взаємоковаріаційна	$K_{xy}(j)=\frac{1}{n}\sum_{i=1}^{n} \overset{o}{x}_i \cdot \overset{o}{y}_{i+j}$
7.4	взаємокореляційна	$R_{xy}(j)=\frac{1}{n}\sum_{i=1}^{n} \overset{o}{x}_i \cdot \overset{o}{y}_{i+j}$
7.5	нормована взаємокореляційна	$P_{xy}(j)=\frac{R_{xy}(j)}{\sqrt{D_x+D_y}}$, якщо $j=0$, то $Pxy(0)$ – нормований коефіцієнт взаємокореляції $P_{xy}(0)=\frac{R_{xy}(0)}{\sqrt{D_x+D_y}}$.
8	Взаємокореляційна матрична модель ОУ	$\begin{vmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1j} & \cdots & \rho_{1m} \\ \rho_{21} & \rho_{22} & \cdots & \rho_{2j} & \cdots & \rho_{2m} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \rho_{i1} & \rho_{i2} & \cdots & \rho_{ij} & \cdots & \rho_{im} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \rho_{ni} & \rho_{n2} & \cdots & \rho_{nj} & \cdots & \rho_{nm} \end{vmatrix}$, де $\rho_{ij}=\frac{R_{ij}(0)}{\sqrt{D_i+D_j}}$ –нормований коефіцієнт взаємокореляції між параметрами ОУ, R_{ij} – взаємокореляційна модель між i та j параметром.

Крім цих моделей розглянуті наступні:

2.5.1. Спектральні моделі ОУ. Ці моделі ОУ будуються на основі ортогональних функцій різних базисів [31, 108, 111]:

– Фур’є, синусоїдальні функції;

– Хаара, фазоімпульсні функції;

– Крейга, широтно–імпульсні функції;

– Крестенсона, пилоподібні трикутні функції;

– Радемахера, меандрові прямокутні функції;

– Уолша, в якому використовуються шумоподібні дискретні функції;

– Галуа, в якому використовуються зсуви однієї з шумоподібних функцій Уолша.

Спектральні характеристики $S_{(W)}$ можуть бути представлені в класі ортогональних функцій різних теоретико–числових базисів Фурє, Радемахера, Хаара, Крейга, Крестенсона, Уолша та Галуа у вигляді мультиплікативної інтегральної оцінки на основі коефіцієнта взаємокореляції

$$S_{(W)} = \frac{1}{m}\sum_{j=1}^{m} Z_{xx}(j)\cdot W_j \ ,$$

де $Z_{xx}(j)$ – одна з автокореляційних моделей , W_j – гармоніка ортогональної функції відповідного теоретико - числового базису. Слід зауважити, що на практиці для реальних ОУ кореляційні моделі характеризуються нестаціонарною дисперсією, тобто $D_j \to 0, (j = 0,1,2...n)$. Тому для ліквідації ефекту появи "від'ємних частот ", коли $S_{(W)} < 0$, що протирічить фізичним станам ОУ, в рівняння вводять вагову функцію

$$W_j = e^{-\alpha j},$$

де α –коефіцієнт затухання ковзної дисперсії автокореляційної моделі. В результаті отримаємо загальну формулу для розрахунку спектральних характеристик ОУ у будь–якому теоретико–числовому базисі:

$$S_{(W)} = \frac{1}{m}\sum_{j=1}^{m} Z_{xx}(j)\cdot W_j \cdot e^{-\alpha j} \ .$$

Для забезпечення необхідної точності обчислення спектральної характеристики станів ОУ $S_{(W)}$ величина m вибирається з умови забезпечення інтервалу кореляції згідно обмеження

$$m = j, \rho_{xx}(j+1) \le 0.01.$$

2.5.2. Кластерні моделі квазістаціонарних ОУ. Інформаційна технологія побудови кластерних моделей квазістаціонарних ОУ базується на теорії побудови продукційних моделей подання знань[11, 18].

При цьому на коефіцієнти матриці $\left|P_{ij}\right|$

$$\begin{vmatrix} P_{11} & P_{12} & \dots & P_{1j} & \dots & P_{1n} \\ P_{21} & P_{22} & \dots & P_{2j} & \dots & P_{2n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ P_{i1} & P_{i2} & \dots & P_{ij} & \dots & P_{in} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ P_{m1} & P_{m2} & \dots & P_{mj} & \dots & P_{mn} \end{vmatrix},$$

де P_{ij} - ймовірність переходу об'єкту з *i*-го стану в *j*-й,

накладається додаткова логіко–статистична умова апертури α

$$K_{ij} = P_{ij} \cdot k_{ij},$$

$$k_{ij} = \begin{cases} 1, & P_{ij} \geq \alpha \\ 0, & P_{ij} < \alpha \end{cases} \quad 0 \leq \alpha \leq 1,$$

де α – коефіцієнт інформаційної значимості елемента P_{ij}. Коефіцієнт кластеризації K_{ij} приводить до кластеризації матриці P_{ij}, в якій значення елементів $P_{ij} \geq \alpha$ залишаються без змін, а елементи $P_{ij} < \alpha$ дорівнюють нулю.

Наприклад, маємо наступну матрицю P_{ij} для $n = 4$; $\alpha = 0{,}7$:

$$\begin{vmatrix} 0.8' & 0.4 & 0.2 & 0.75' \\ 0.1 & 0.9' & 0.6 & 0.4 \\ 1.0' & 0.3 & 0.7' & 0.1 \\ 0.8' & 0.2 & 0.9' & 0.5 \end{vmatrix},$$

де позначені інформативні елементи, для яких виконується умова $P_{ij} \geq 0.7$, що відповідає кластеризованій бінарній матриці, аналогічній матриці інциденції

$$\begin{array}{c} \\ S_1 \\ S_2 \\ S_3 \\ S_4 \end{array} \begin{array}{c} \begin{array}{cccc} S_1 & S_2 & S_3 & S_4 \end{array} \\ \begin{vmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{vmatrix} \end{array}.$$

Кластеризована матриця P_{ij} є основою для побудови трьох типів кластерних моделей:

1) таблична кластерна модель;
2) ієрархічна графова однорівнева кластерна модель;
3) повнозв'язна графова кластерна модель.

Таблична кластерна модель (рис. 2.4) відповідає регламентним переходам об’єкта ОУ з одного стану в інший. При відхиленні статистики переходів ОУ від нормативних в кластерній матриці з’являється фіксація ненормативних P_{ij} переходів, що демонструється індикацією зникнення регламентних переходів або появою нерегламентних. Недоліком даної кластерної моделі є її неієрархічність, яка відображена в графовій однорівневій кластерній моделі (рис.2.5).

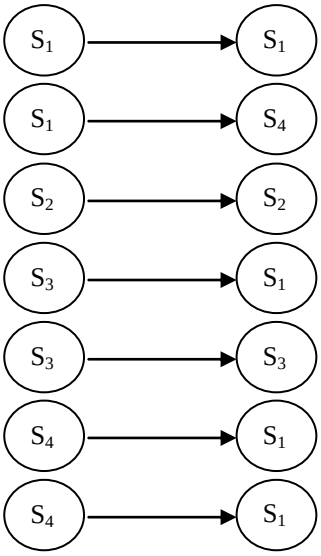


Рис.2.4. Таблична кластерна модель.

Приведені модифікації ієрархічних графових однорівневих кластерних моделей також характеризуються обмеженими демонстраційними та функціональними можливостями, оскільки не охоплюють повну взаємодію елементів кластеризованої матриці інциденцій (рис.2.5).

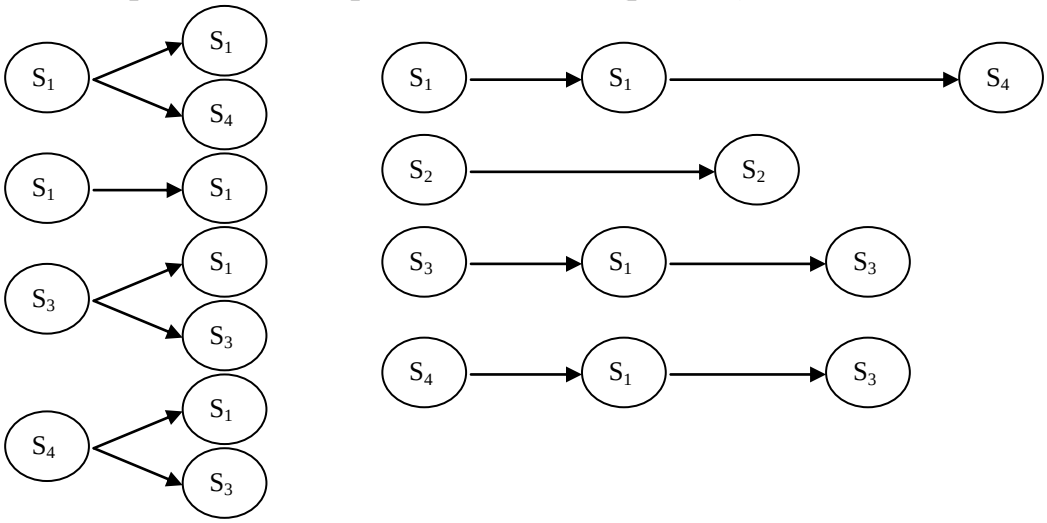


Рис. 2.5. Ієрархічна графова однорівнева кластерна модель.

Повнозв’язна графова кластерна модель дозволяє найбільш компактно та досконало подати кластеризовані інформаційно значимі ймовірнісні переходи ОУ з одних станів в інші (рис.2.6).

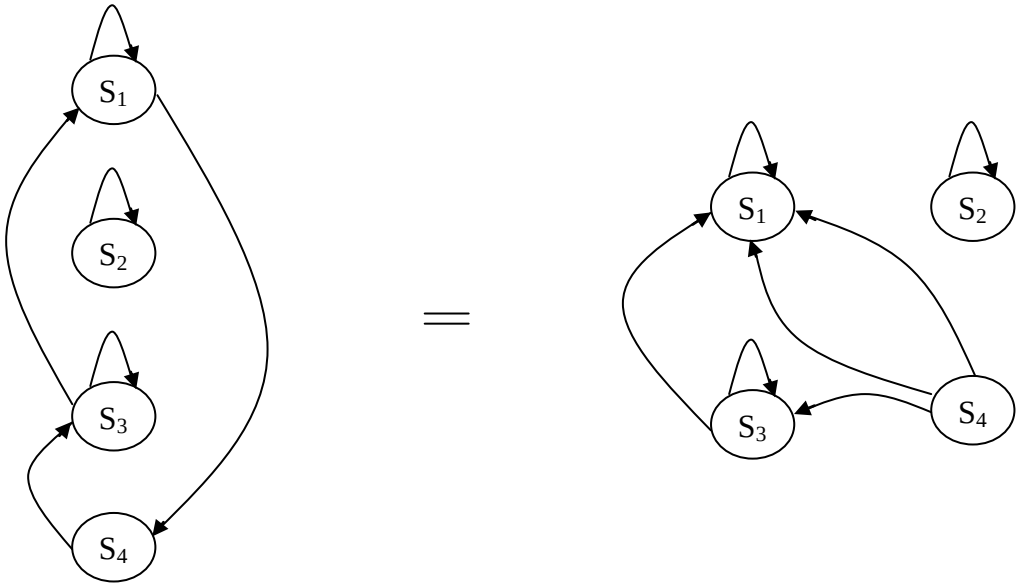


Рис.2.6. Повнозв’язна кластерна модель ОУ.

Викладена методологія дозволяє провести детальний автоматизований аналіз технологічних, інформаційних та перехідних станів ОУ, що в свою чергу спрощує діагностику аварійних та нормативних станів об’єктів. Позитивною характеристикою таких моделей є можливість зміни глибини кластеризації матриці імовірності P_{ij} шляхом зміни величини коефіцієнта α, а також введення більш складних видів апертур кластеризації матриць, наприклад $\alpha_1 \le P_{ij} \le \alpha_2$.

2.5.3. Логіко–статистичні інформаційні моделі (ЛСІМ).ЛСІМ є важливим інструментом контролю відхилень від норми станів об’єкта управління КС. Теоретичні основи побудови ЛСІМ охоплюють процедури контролю відхилень станів ОУ по амплітуді, динаміці, авто– та взаємокореляційних характеристик технологічних процесів, а також фазових та спектральних відхилень [42].

Одна з найпростіших ЛСІМ–1 традиційно використовується в системах контролю та автоматики для регулювання та стабілізації технологічних процесів. Приклад побудови такої моделі, яка реагує на відхилення стану об’єкту керування від норм по амплітуді, приведена на рис.2.7.

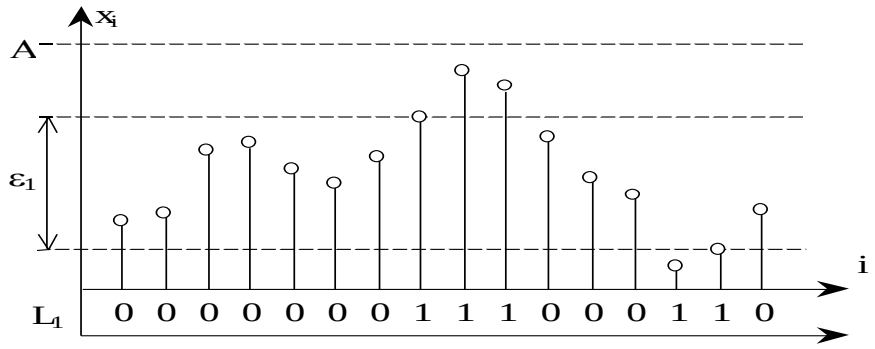


Рис.2.7. Вихідний сигнал ЛСІМ–1

Значення ЛСІМ–1 задається булевою змінною L_1, яка описується рівнянням:

$$L_1 = \begin{cases} 0, & X_i \in \varepsilon_1 \\ 1, & X_i \notin \varepsilon_1 \end{cases}$$

де ε_1 – апертура станів ОУ (x_i), яка має відповідний зміст: $X_i \in \varepsilon_1$, відповідає знаходженню X_i в границях апертури ε_1, а $X_i \notin \varepsilon_1$ – міститься на границях або поза границями апертури.

Проте, ЛСІМ–1 на основі контролю відхилення по амплітуді не реагують на відхилення по динаміці станів ОУ.

Чутливими до зміни динаміки контрольованого процесу є ЛСІМ–2 [43]. На рис.2.8 показано реакцію ЛСІМ–1 та ЛСІМ–2 на одну і ту ж послідовність станів ОУ.

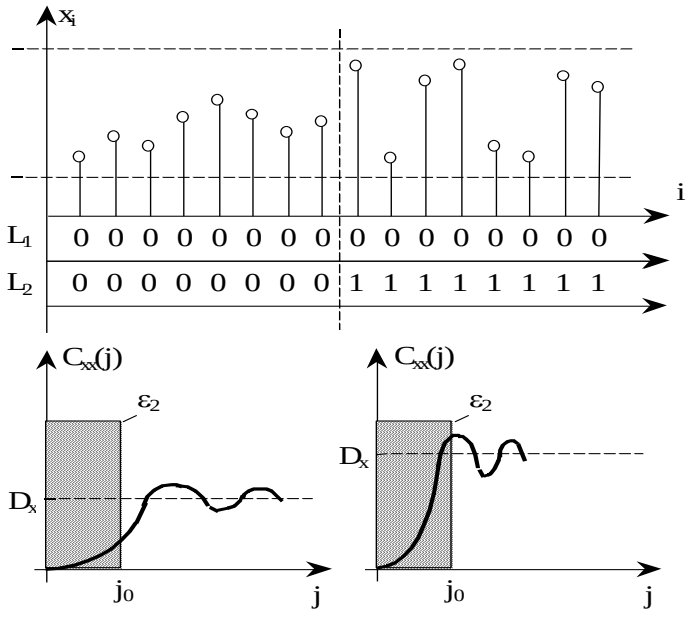


Рис. 2.8. Порівняльна характеристика ЛСІМ–2 на основі контролю структурної автокореляційної моделі станів ОУ з результатами реакції ЛСІМ–1.

На рис 2.8 показані умови контролю динаміки станів ОУ на основі ковзної структурної кореляційної моделі (L_2), яка описується виразом

$$L_2 = \begin{cases} 0, & C_{xx}(j) < \varepsilon_2, \\ 1, & C_{xx}(j) \geq \varepsilon_2. \end{cases} \quad \text{де} \quad C_{xx}(j+k) = \frac{1}{n} \cdot \sum_{i+k}^{n+k} (x_{i+k} - x_{i+k+j})^2, \quad k = 0, 1, 2 \ldots n; \quad i = 1, 2 \ldots n.$$

При цьому, умовою знаходження процесу X_i в границях апертури ε_2 є досягнення значення $C_{xx}(j)$ асимптотичного рівня дисперсії D_x на інтервалі j_0. Для побудови даної моделі ЛСІМ аналогічно можуть бути використані інші автокореляційні моделі, в тому числі подані в табл.2.5.

Вищенаведені моделі ЛСІМ–1 та ЛСІМ–2 також мають недолік. Вони є нечутливими до фазових змін параметрів ОУ. В таких випадках використовують ЛСІМ–3 [44], що дає змогу зафіксувати фазові зміни станів ОУ в границях апертури ε_1. На рис.2.9 показано, у порівнянні, реакцію ЛСІМ–1, ЛСІМ–2 та ЛСІМ–3 на задану послідовність станів ОУ.

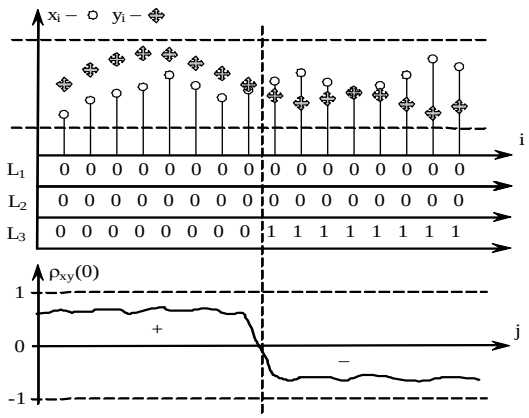


Рис.2.9. Порівняльна характеристика ЛСІМ–3 на основі нормованого коефіцієнта взаємокореляції процесів x_i та y_i станів ОУ з результатами реакції ЛСІМ–1 та ЛСІМ–2.

При цьому ЛСІМ–3 описується рівнянням:

$$L_3 = \begin{cases} 0, & \rho_{xy} > 0 \\ 1, & \rho_{xy} \leq 0, \end{cases}$$

де $\rho_{xy}(0)$ – нормований коефіцієнт взаємокореляції між процесами x_i та y_i, що описують стани ОУ в реальному часі, які обчислюються згідно наступного виразу:

$$\rho_{xy}(0) = \frac{\sum_{i+k}^{n+k} \overset{o}{x}_{i+k} \cdot \overset{o}{y}_{i+k}}{\sqrt{\sum_{i+k}^{n+k} (\overset{o}{x}_{i+k} - M_x)^2 \cdot \sum_{i+k}^{n+k} (\overset{o}{y}_{i+k} - M_y)^2}},$$

де M_x, M_y – відповідно вибіркові або ковзні математичні сподівання процесів x_i та y_i.

Модель ЛСІМ-4 успішно застосовується при складних та зашумлених процесах, коли на основі описаних моделей неможливо розрізнити відхилення контрольованої величини від норми. В таких випадках необхідно проводити спектральний аналіз сигналу. При його розбитті на частотні гармоніки можна досить чітко визначити появу небажаних складових (наприклад, шкідливих високочастотних гармонік). В таких випадках використовують ЛСІМ–4, яка основана на спектральному аналізі характеристик ОУ.

Аналогічно, в ЛСІМ–4 можуть бути реалізовані перетворенням сигналів в базисах Крейга, Крестенсона та Галуа.

При аналізі станів ОУ важливе значення має чітке визначення всіх можливих станів об’єкта, які можуть бути зображені у вигляді графа переходів (рис. 2.10). Вершинами цього графа будуть можливі стани об’єкта. Наведемо для прикладу граф переходів станів об’єкта управління, у якого є п’ять ймовірних станів.

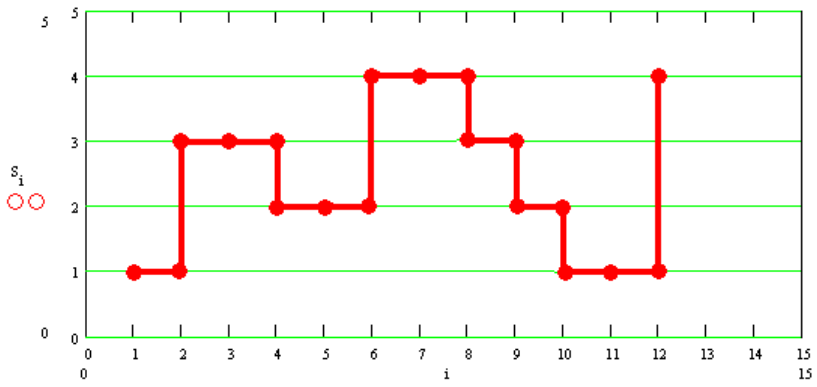


Рис.2.10. Граф переходів станів ОУ.

В даному випадку, S_i – це стани ОУ; $j=\overline{1,m}$ – кількість станів об'єкта керування, де $m=5$; $i=\overline{1,n}$ – кількість вибірок, де $n=12$.

Слід зазначити, що перехід ОУ з одного стану в інший можна оцінити ймовірнісними характеристиками P_{ij}, які утворюють матрицю ймовірності:

$$\rho=\begin{vmatrix} \rho_{11} & \rho_{12} & \dots & \rho_{1i} & \dots & \rho_{1m} \\ \rho_{21} & \rho_{22} & \dots & \rho_{2i} & \dots & \rho_{2m} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \rho_{j1} & \rho_{j2} & \dots & \rho_{ji} & \dots & \rho_{jm} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \rho_{n1} & \rho_{n2} & \dots & \rho_{ni} & \dots & \rho_{nm} \end{vmatrix}.$$

Для цієї кореляційної матриці характерні наступні властивості:

$$\rho_{ii}=1,\quad \rho_{ij}=\rho_{ji},\quad -1\le\rho_{ij}\le+1.$$

Отже, у відповідності до цих умов інформативну частину матриці розділяє діагональ .

Запишемо в ряд інформативні значення цієї матриці:

$$\begin{array}{cccccccccc} \rho_{12}, & \rho_{13}, \dots, & \rho_{1m}, & \rho_{23}, & \rho_{24}, \dots, & \rho_{2m}, \dots\dots, & \rho_{nm} \\ 1 & 2 & \dots & m+1 & \dots\dots\dots\dots\dots\dots\dots & & k \end{array}$$

Кількість значень утвореного центрованого та нормованого вектора (рис.2.11) визначається як $k=(m-1)+(m-2)+\dots+1$.

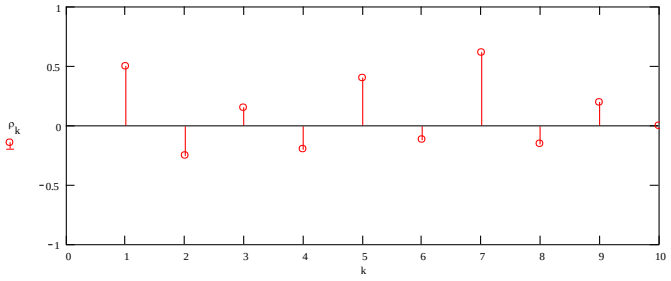


Рис.2.11. Інформаційний вектор ймовірнісних станів ОУ

З приведеного нормованого взаємокореляційного вектора ρ_k^* отримуємо глобальну дисперсію станів ОУ $D_{\rho_k^*}$ (рис.2.12):

$$D_{\rho_k^*}=\frac{1}{k}\cdot\sum_{i=1}^{k}(\rho_k^*)^2\,.$$

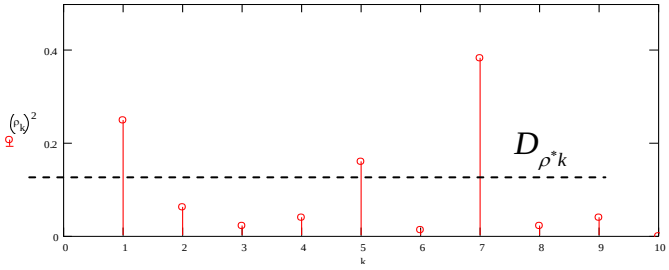


Рис.2.12. Оцінка глобальної дисперсії станів ОУ

Звідси, можна в загальному представити інформаційну технологію та математичні основи побудови ЛСІМ–5, принцип роботи якої базується на аналізі зміни глобальної дисперсії станів ОУ (рис.2.13).

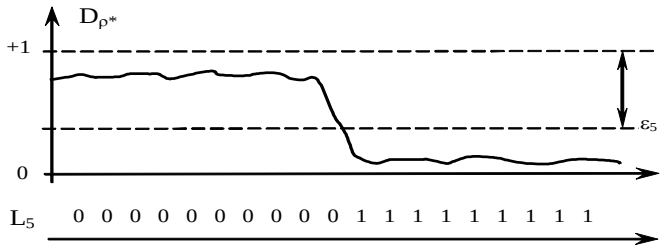


Рис.2.13. ЛСІМ–5 на основі глобальної дисперсії станів ОУ.

Аналітично ЛСІМ–5 описується виразом:

$$L_5 = \begin{cases} 0, & D_{\rho_k^*} > \varepsilon_5, \\ 1, & D_{\rho_k^*} \le \varepsilon_5. \end{cases}$$ де ε_5 – апертура для глобальної дисперсії.

2.5.4. Ентропійні моделі. Існують чотири найвживаніші оцінки ентропії, джерел інформації та ОУ, які можуть бути описані наступними інформаційними мірами [41]:

1. Хартлі

$$I_x = \hat{E}[\log_2 A];$$

2. Шеннона

$$I_x = P_i \cdot \log_2 \frac{1}{P_i};$$

3. “$3\sigma x$”

$$I_x = \hat{E}[\log_2 3\sigma x];$$

4. Кореляційною мірою шляхом використання різних автокореляційних моделей [29]:

– нормованої автоковаріаційної

$$I_x = \log_2 2\pi e + \frac{1}{2}\log_2\left([D_x - K_{xx}(j)]\cdot[D_x + K_{xx}(j)]\right);$$

– нормованої моделі автокореляції

$$I_x = \log_2 2\pi e\sqrt{D_x{}^2\left[1-\rho_{xx}{}^2(j)\right]};$$

– нормованої модульної моделі

$$I_x = \log_2 \frac{\pi e\sqrt{\pi}}{2} + \sum_{j=1}^{m}\log_2\left[M_x\cdot q_{xx}(j)\cdot\sqrt{8-\pi q_{xx}{}^2(j)}\right];$$

– моделі еквівалентності

$$I_x = \frac{1}{m}\sum_{j=1}^{m}\log_2\left[\pi^2 e\frac{M_x - F_{xx}(j)}{\sigma_x}\cdot\sqrt{\frac{8\sigma_x}{\pi} - M_x - F_{xx}(j)^2}\right].$$

Проведений аналіз та систематизація ОУ РКС показує, що в загальному випадку кожен ОУ, який може бути джерелом інформації, середовищем передавання, оброблення та зберігання даних, а також приймачем інформації може бути в повній мірі описаний характеристичним функціоналом:

$$D_{OY} = F(X(t), M_x, D_x, \sigma_x, R_{xx}R_{xy}, M_{ij}, S(w), K_{ij}, ЛСІМ, I_x),$$

де $X(t)$-поточне значення параметра, M_x-математичне сподівання, D_x-дисперсія, σ_x- середньоквадратичне відхилення, R_{xx}- автокореляційна функція, R_{xy}- взаємокореляційна функція, M_{ij}- матриця нормованих коефіцієнтів взаємокореляції, $S(w)$- спектральні моделі, K_{ij}-матриця імовірностей переходу в різні стани, $ЛСІМ$- логіко-статистична інформаційна модель, I_x- ентропійна модель.

На основі даного фрейму формуються моделі-фрейми, що реєструються на рівні периферійних контролерів КС, передаються у вигляді пакетів даних по каналах зв'язку, відображаються на рівні пристроїв відображення для операторів та зберігаються в архівах баз даних та баз знань [112].

В загальному випадку структура фрейму має наступний вигляд:

	Start	A_1	A_2	*Y*	D_{OY}	*KK*	*Stop*	

де *Start, Stop* – відповідні флаги конкретних протоколів та інтерфейсів, A_1, A_2 – адреси передавача та приймача, Y – тип фрейму, D_{OY} – дані про об'єкт, які описуються функціоналом, *KK* – контрольний код захисту даних від помилок.

2.6. Системні характеристики СПД

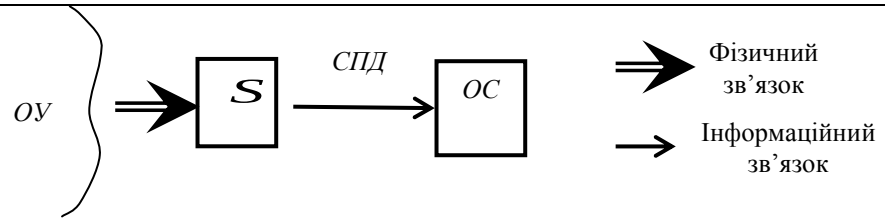
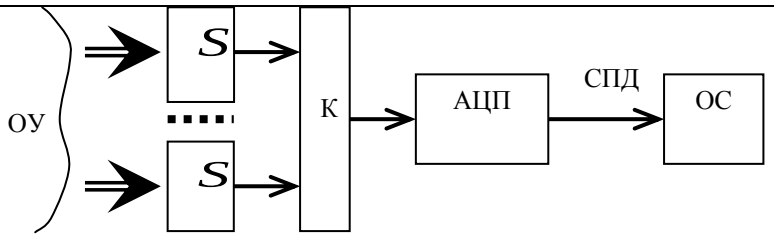
СПД – представлені швидкістю приймання, швидкістю передавання даних, імовірністю помилок, часом затримки в каналі

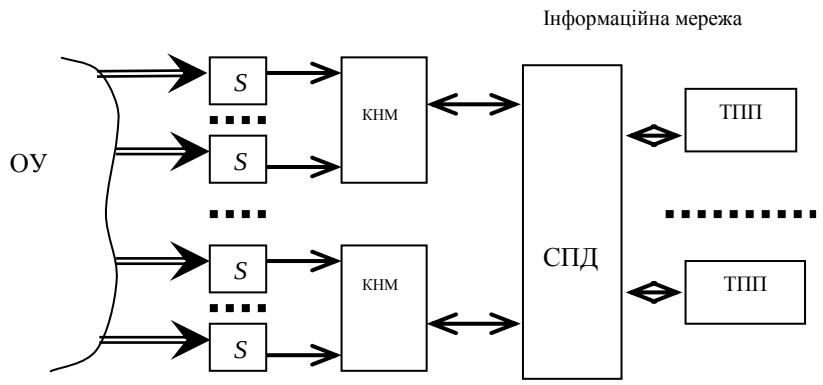
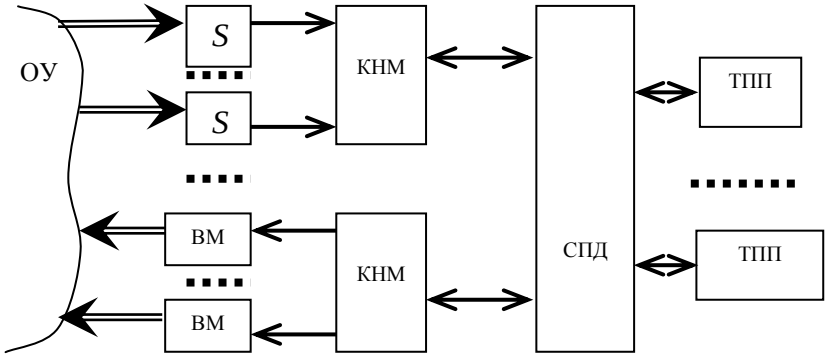
$$E_{СПД} = F(V_R, V_W, P_i, T),$$

де V_R – швидкість приймання даних, V_W – швидкість передавання даних, P_i – імовірність помилок, T – час затримки в каналі.

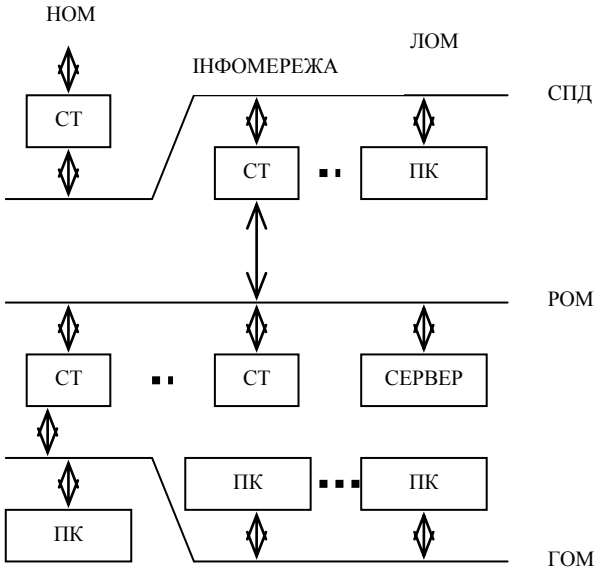
Системи передавання даних використовуються в наступних випадках (табл.2.5):

Структура СПД *Таблиця 2.5*

Сфера застосування	Структура	Компоненти
При контролі об'єктів управління		*ОУ* – об'єкт управління; *S* – сенсор; *ОС* - обчислювальна система
При зборі вимірювальної інформації		*К* – комутатор; *АЦП* – аналогово–цифровий перетворювач

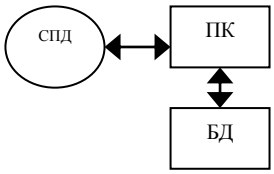
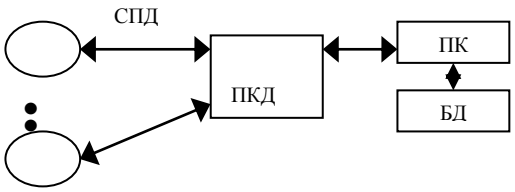
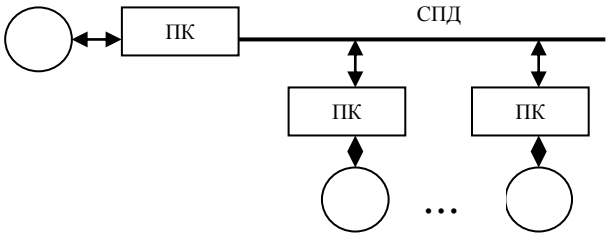
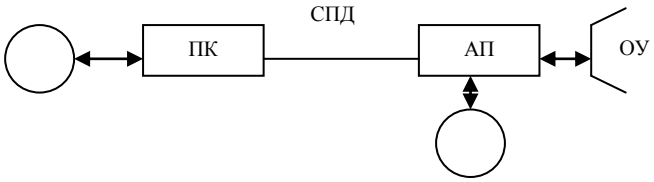
При організації низових обчислювальних мереж		*КНМ*– контролер низової мережі; *НОМ*– низова обчислювальна мережа.
При управлінні об'єктами в реальному масштабі часу		*ВМ*– виконавчий механізм; *ТПП*– технологічні промислові процесори

Продовж.табл.2.5.

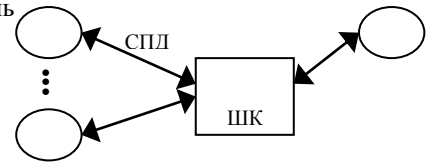
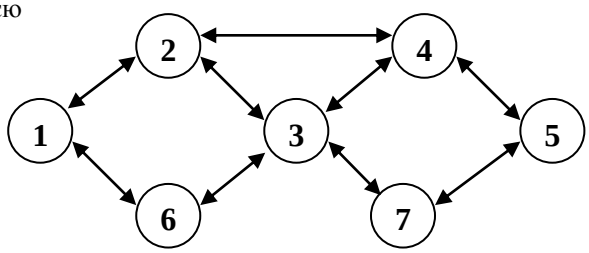
При організації локальних та глобальних мереж		*ЛОМ*– локальна обчислювальна мережа; *НОМ*– низова обч. мережа; *РОМ*– регіональна обч. мережа; *ГОМ*– глобальна обч. мережа

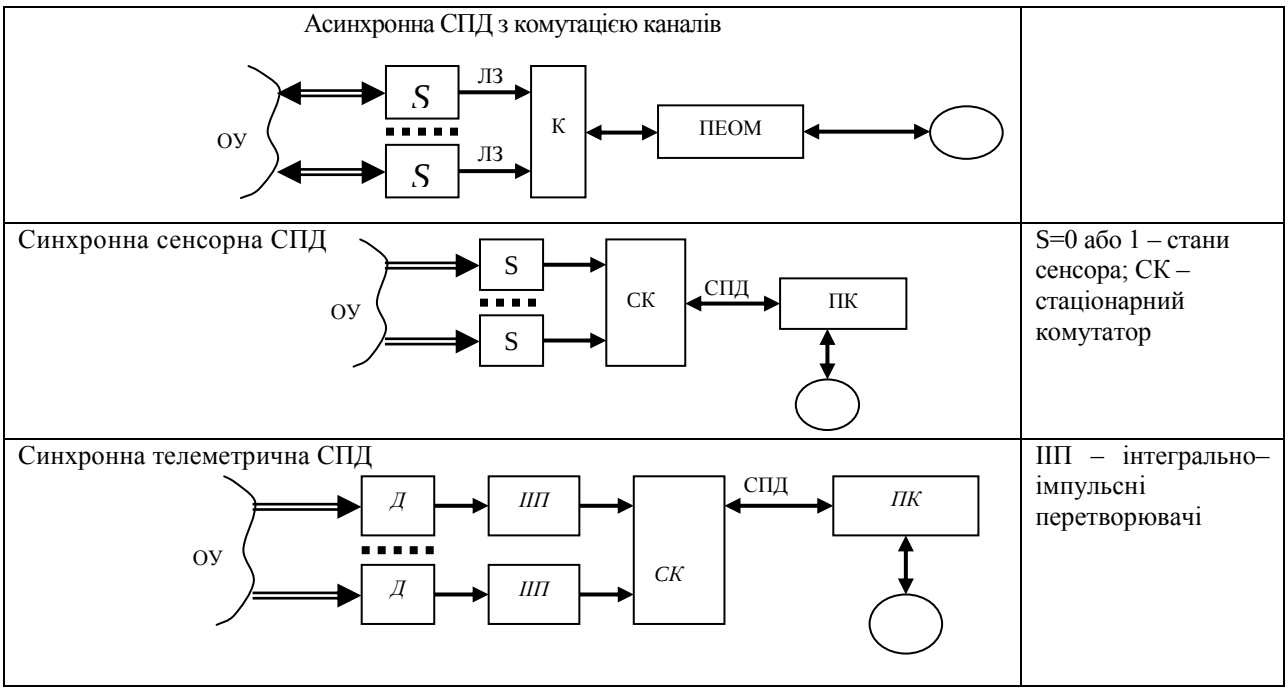
Структура та компоненти основних класів СПД представлені в табл. 2.6. [48, 56].

Класи СПД *Таблиця 2.6.*

Структура	Компоненти
Інтерактивна СПД – взаємодії людини з машиною 	БД- база даних
Довідкова СПД 	ПКД – пам’ять колективного доступу
СПД вводу і розділення даних 	ПК- персональний комп’ютер
Віддалений ввід завдань 	АП – абонентський пункт

Продовження таблиці 2.6.

СПД з комутацією повідомлень 	ШК – швидкісний комутатор
СПД з пакетною комутацією 	
Електронна пошта 	ТП – трансляційний пункт
Промислові СПД	К – комутатор



Класифікація каналів зв’язку СПД наведена в таблиці 2.7 [113].

Класифікація каналів зв'язку СПД *Таблиця 2.7*

Тип ЛЗ	Назва ЛЗ	ΔF , Гц	дб/км
ПНЛ	Пневматична	1.0	20
ГЛ	Гідравлічна	10	10
ХЛ	Хімічна (нейронний зв’язок)	10	0
ПЛ..	Повітряна	150000	0.1

Продовж.табл.2.7

СКЛ	Симетрична кабельна	10^6	5
ККЛ	Коаксіально–кабельна	$3\cdot10^9$	40
ХВМ	Хвилевід металічний	$50\cdot10^9$	16
ХВМ	Хвилевід діалектричний	$100\cdot10^9$	20
ВОЛЗ–П	Волоконно–оптична пластикова	10^{15}	20
ВОЛЗ–СТ	Волоконно–оптична ступінчаста	10^{15}	0.5
ВОЛЗ–ГР	Волоконно–оптична градієнтна	10^{15}	0.2
ВОЛЗ–ОМ	Одномодові волокна	10^{11}	0.1–0.05
ІКР	Інфрачервона	10^{14}	100
РРЛ	Радіорелейна	$300\cdot10^9$	20
СЛЗ	Супутникова	$30\cdot10^9$	20

2.7. Системні характеристики операторів

Оператор представлений часом праці, швидкістю запису, швидкістю зчитування, потенційною пам'яттю та знаннями

$$E_O = F\ (T, V_R, V_W, S, M),$$

де T – час праці, V_R – швидкість запису, V_W – швидкість зчитування, S – знання, M – пам'ять.

На рис 2.14. показано основні можливі інтерфейсні зв'язки оператора з іншими системними об'єктами КС.

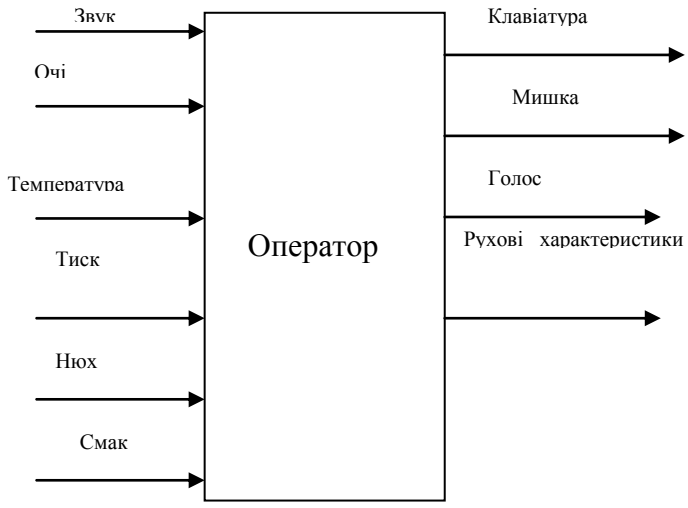


Рис. 2.14. Інтерфейсні зв'язки оператора КС

Системні характеристики оператора можуть змінюватися у широких границях в залежності від рівня знань, професійної підготовки, віку, стану здоров'я та оточуючого середовища, ступеня оснащення різними програмно–апаратними засобами процесорної, дисплейної та телекомунікаційної техніки, відповідальності за вчинення дії та інше [51].

В той же час вже відомі деякі базові характеристики оператора, як системного об'єкта РКС, в тому числі [114, 146, 147]:

– часовий режим роботи оператора протягом доби складає не більше 6–ти годин, тобто $T = 0{,}25$;

– допустимі часові ворота реакції на зміну даних та динаміку відображення інформації на дисплеї – $\Delta t = 1{,}8 \div 2{,}4$;

– швидкість сприймання аудіо– даних на основі стандартної ДІКМ цифрової телефонії $C = 64 Кбіт/с$;

– потенційні можливості сприймання кольорових відео–даних $V = 25 \cdot 10^{12}$ *пікселів*$/с$;

– рівень спеціальних знань оператора S можна оцінити сумарною системою знань та навиків їх використання, наприклад, володіння алгоритмічними мовами, операційними системами, САПР [85, 87, 88], $S = \sum_{i=1}^{n} S_i$;

– об'єм даних оперативної аудіо–відео пам'яті

$$W = (64 \cdot 10^3 + 25 \cdot 10^{12} \cdot T_0(біт)),$$

де $T_0 = 3600 \cdot 24 \cdot 365 \cdot 40$ – час життя оператора у секундах протягом 40 років, поки нарощується оперативна нейронна пам'ять людини;

– швидкість запису даних через клавіатуру ПК, $V_R = 12 біт/с$.

Очевидно, що системні характеристики оператора є набагато потужніші, якщо врахувати інші інтерфейси вводу/виводу даних, наприклад, тактильні, нюх, смак та інше. В той же час оператор є ненадійним елементом КС, може часто помилятися, виконувати протисистемні дії та інше.

ВИСНОВКИ ПО ДРУГОМУ РОЗДІЛУ

1. Вперше виконана ідентифікація характеристик КС на основі глобальної моделі, яка включає процесори, дані, систему передавання даних, об'єкти управління та оператори, які об'єднані між собою через інтерфейсні зв'язки та СПД. Запропонована систематизація є основою для диференціації досліджень та доцільного проектування комп'ютерної системи на основі формалізованої моделі руху даних.

2. Запропонована математична модель ресурсних характеристик системного об'єкта комп'ютерних систем, заданого четвіркою параметрів: T – час

використання ресурсу, V – швидкість виконання системних операцій, M – об'єм використовуваної пам'яті, S – ступінь використання системних функцій, а також його нормалізована форма, на основі якої проводиться оцінка собівартості руху даних з врахуванням прибутків та затрат на реалізацію функцій системного об'єкта.

3. Запропонований функціонал оцінки характеристик системних об'єктів, який диференціює параметри швидкодії відповідно на швидкість запису – V_w та швидкість зчитування – V_R інформаційних потоків, що дозволяє врахувати асиметричні характеристики швидкодії вхідних та вихідних інтерфейсних пристроїв системних об'єктів.

4. Визначені системні характеристики процесорів КС на основі оцінки математичного сподівання, яка враховує поточні або миттєві коефіцієнти використання ресурсів пам'яті процесора.

5. Класифіковані та досліджені системні характеристики даних на основі методів їх кодування в різних теоретико–числових базисах: унітарному, Хаара, Крейга, Радемахера, Крестенсона, Уолша та Галуа.

6. Систематизовані та аналітично описані системні характеристики та моделі об'єктів управління, які включають сигнальні, статистичні, авто– та взаємокореляційні, матричні взаємокореляційні, спектральні кластерні, логіко–статистичні інформаційні та ентропійні моделі ОУ.

7. Систематизовані структури СПД та характеристики каналів зв'язку, які використовуються при формалізації архітектур КС на основі матричних моделей руху даних.

8. Вперше визначені інтерфейсні зв'язки та системні характеристики операторів, як інформаційних елементів КС, необхідні для розрахунку ступеня використання ресурсів в активних вузлах моделей руху даних.

РОЗДІЛ 3
МЕТОДИ ОРГАНІЗАЦІЇ РУХУ ДАНИХ В КОМП'ЮТЕРНИХ СИСТЕМАХ НА ОСНОВІ МАТРИЧНИХ МОДЕЛЕЙ

3.1. Вдосконалення атрибутів матричної моделі

Матриці інциденцій, які описані в першому розділі, є теоретичною та методологічною основою для побудови матричних моделей руху даних. При цьому ММРД об'єднують характеристики матриці інциденцій та графових дерев і більш конкретизовані, оскільки доповнюються символами атрибутів, які відображають поняття джерела інформації, проміжного пункту цифрової обробки даних та приймача інформації [6, 31, 33].

Матрична модель руху даних (рис.3.1) визначається умовами та графом взаємодії об'єктів $O_1,...,O_4$ та документами $D_1,...,D_4$ [37].

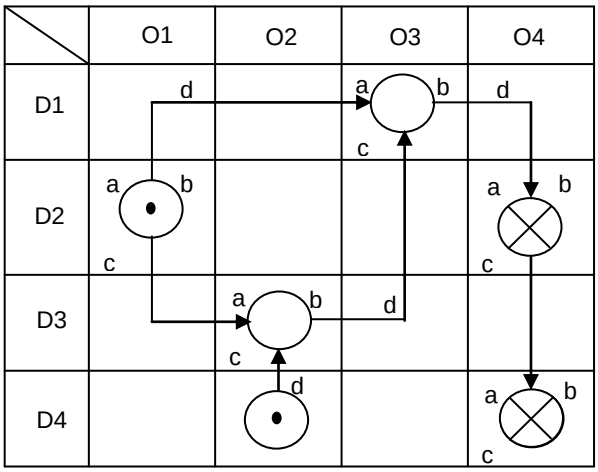


Рис. 3.1. Матрична модель руху даних

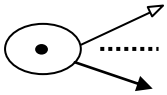
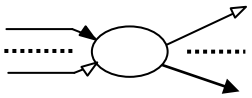
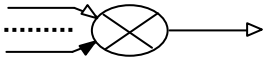
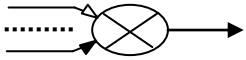
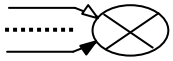
При цьому для кожного SO з координатами $D_i \cdot O_j$ повинні виконуватися умови несуперечливості:

- джерело інформації має не менше одного виходу і жодного входу ;
- пункт обробки даних має не менше одного входу і одного виходу ;
- залежний пункт затвердження даних має не менше одного входу і тільки один вихід ;
- незалежний пункт затвердження даних має не менше одного входу і жодного виходу (табл. 3.1).

Для формальної побудови сімейства моделей руху даних на основі відомої двовимірної ММ кожний елемент двовимірної ММ описується четвіркою параметрів: a – початок виконання операції; b – тривалість виконання операції; c – тип операції; d –час передавання даних у каналі зв'язку між об'єктами.

Таблиця 3.1

Символіка атрибутів ММРД

Символи атрибутів ММРД	Зміст атрибутів ММРД
	інформаційний потік
	матеріальний потік
	джерело
	пункт обробки ІМП (інформаційно–матеріальний потік)
	залежний пункт затвердження
	
	незалежний пункт затвердження і архівізації даних

Значний вклад в розвиток теорії проектування комп'ютерних мереж та автоматизованих систем вніс відомий американський вчений Дж. Мартін [14], який визначив поняття і ввів оцінку одиниці руху даних у вигляді :

$$K_d = \frac{R}{W}, \qquad (3.1)$$

де R – число зчитувань або запитів,

W –число записів або оновлень даних.

Дана оцінка дозволила розвинути Дж. Мартіном основи теорії проектування корпоративних комп'ютерних мереж і методологію побудови різноманітних проекцій їх моделей.

В той же час дана оцінка одиниці руху даних не дозволяє врахувати ефективність використання ресурсів в пунктах формування, обробки та реєстрації даних, що не дозволяє реалізувати оптимізаційне проектування комп'ютерних

мереж та розрахунок характеристик їх надійності, живучості, ймовірності перевантажень та відмов.

Дана оцінка руху даних практично не може бути використана для проектування та розрахунку системних характеристик мереж з глибоким розпаралеленням інформаційних потоків.

На основі коефіцієнта руху даних (3.1) можна визначити коефіцієнт ефективності руху даних, який враховує ресурси руху даних в конкретному вузлі матричної моделі

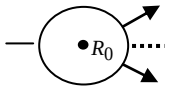
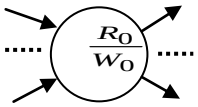
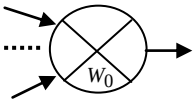
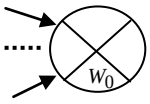
$$K_{ed} = \frac{R_i \cdot W_0}{R_0 \cdot W_i}, \qquad (3.2)$$

де R_i, R_0, W_i, W_0 – відповідно фактичне число запитів, максимально можливе число запитів, фактичне число записів або оновлень, максимально можливе число записів або оновлень у вузлі матричної моделі.

У зв'язку з введенням описаної оцінки в характеристики двовимірної матричної моделі необхідно ввести наступні позначення [6] (табл. 3.2).

Таблиця 3.2

Позначення характеристик двовимірної ММ

Тип вузла матричної моделі	Символ	Умова несуперечливості
Джерело даних		$R_0 > \sum_{i=1}^{n} R_i$
Пункт обробки даних		$R_0 > \sum^{n} R_i$ $W_0 > \sum^{m} R_j$
Залежний приймач даних		$W_0 > \sum^{m} R_j$
Незалежний приймач даних		$W_0 > \sum^{m} R_j$

Запропоноване розширення атрибутів ММРД дозволяє розробити інженерну методику та інформаційну технологію проектування РКС на основі реальних топологій промислових підприємств.

3.2. Аналіз топології промислового об’єкта управління та формалізація параметрів руху даних

На рис. 3.2 представлений приклад топологічної схеми збору та цифрової обробки даних в низовій комп’ютерній мережі обліку витрат енергоносіїв [123, 125, 130].

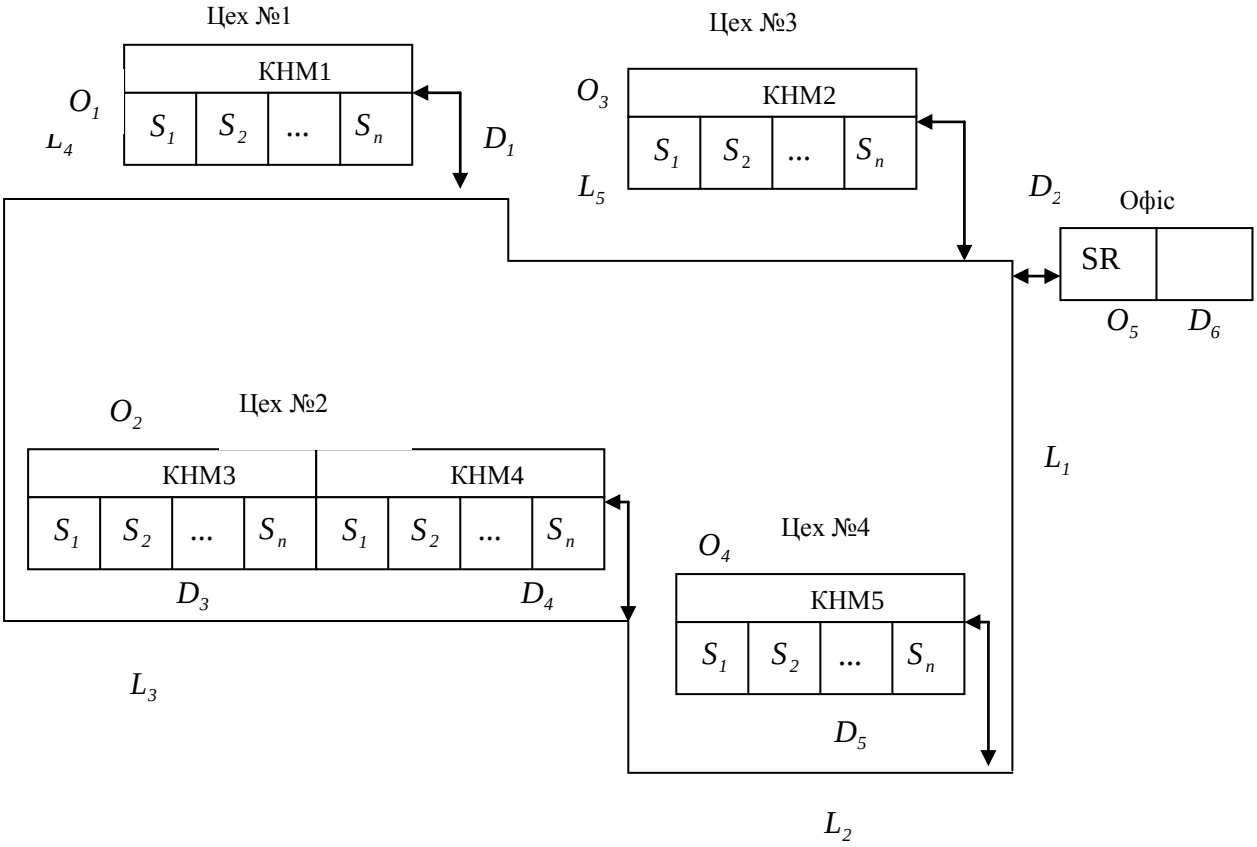


Рис.3.2. Топологія комп’ютерної мережі на підприємстві

SR–сервер, КНМ–контролер низової мережі, S_1– сенсори , L_1– довжини комунікаційних ліній мережі між окремими цехами.

З рис.3.2 видно, що модель руху даних повинна мати розмірність 5×6. Тобто цехи і офіс визначають об’єкти ММ $O_1, O_2 \ldots O_5$. У цехах підприємства та офісі виникають відповідні класи даних, які формуються цеховими КНМ

$D_1, D_2 \ldots D_6$. Таким чином представлена топологія КС може бути описана двовимірною матричною моделлю (рис.3.3).

Слід зауважити, що на графі ММ точну прив'язку до об'єктів та даних мають джерела інформації та пункти затвердження і архівізації даних, а пункти обробки даних мають точну прив'язку до об'єктів.

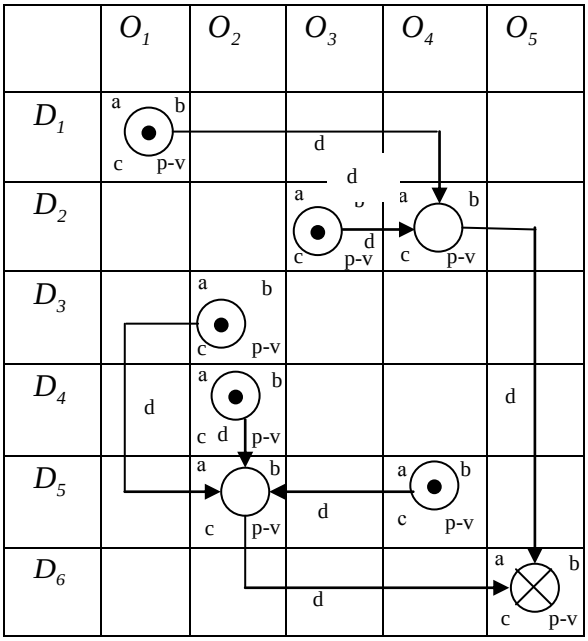


Рис. 3.3 Двовимірна ММ руху даних на підприємстві

На основі формальних параметрів двовимірної ММ будується наступна сукупність моделей руху даних [6]:

– модель граф-розгалужене дерево;

– параметрична часова модель;

– структурно–часова модель;

– мережевий графік;

– суміщений часовий граф;

– блок–схема алгоритму обробки та контролю руху даних.

Крім того формалізуються епюри собівартості циклів руху даних (ЕРД):

– сигнальна прибутково-затратна ЕРД, диференціальна (ΔЕРД), інтегральна ($\int \Delta ЕРД$), сумарна інтегральна ($\sum \int \Delta$ЕРД), глобальна GЕРД (G).

В той же час, незважаючи на достатньо детальне відображення руху даних в КМ у вигляді проекцій двовимірних ММ на основі сімейства перечислених моделей, двомірна ММ не відображає наявні обчислювальні ресурси та степінь їх використання, що охоплюється трьовимірними ММ.

3.3. Розробка похідних моделей на основі матричних моделей руху даних

В основі побудови матричної моделі покладений двовимірний граф, приклад якого поданий на рис.3.4.

Матрична модель утворює мережу направлених зв'язків між джерелами інформації (ДІ), місцями обробки і приймачами даних, відображає маршрути документопотоків, що передаються по каналах зв'язку згідно топології інформаційної системи реальних промислових об'єктів системи передачі даних або локальної обчислювальної мережі ПЕОМ.

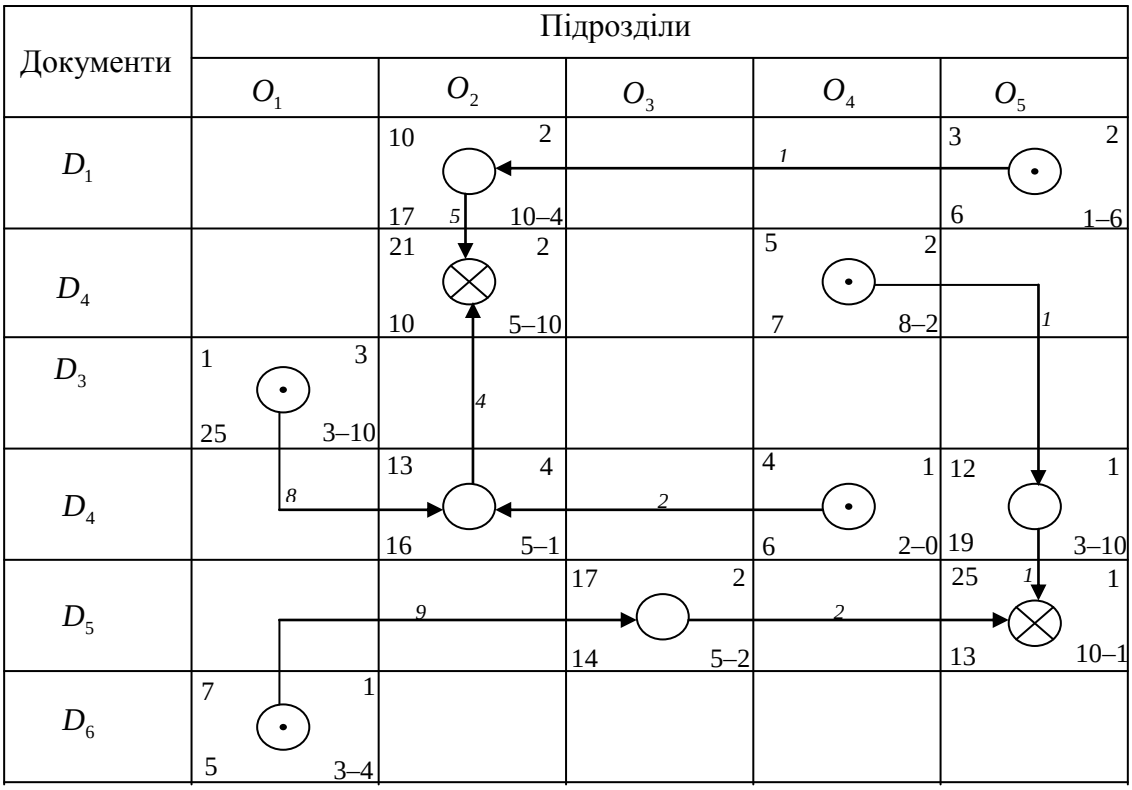


Рисунок 3.4. Матрична модель руху даних РКС

Формалізацію матричної моделі проводять наступним чином:

$$D_iO_j \cdot X \cdot a,b,c,p-v \;\; i \in \overline{1,n} \;\; , j \in \overline{1,m} \quad D_iO_j : D_iO_j \, ,$$

$D_3O_1 \cdot 1 \cdot 1 \cdot 3 \cdot 25 \cdot 3 - 10;$	$D_3O_1 : D_4O_2$
$D_6O_2 \cdot 1 \cdot 7 \cdot 1 \cdot 5 \cdot 3 - 4; O_1;$	$D_6O_1 : D_5O_3$
$D_1O_2 \cdot 2 \cdot 10 \cdot 2 \cdot 17 \cdot 10 - 4;$	$D_1O_2 : D_2O_2$
$D_2O_2 \cdot 3 \cdot 21 \cdot 2 \cdot 10 \cdot 5 - 10;$	$D_4O_2 : D_2O_2$
$D_4O_2 \cdot 2 \cdot 13 \cdot 4 \cdot 16 \cdot 5 - 1;$	$D_5O_3 : D_5O_5$.
$D_5O_3 \cdot 2 \cdot 17 \cdot 2 \cdot 14 \cdot 5 - 2;$	$D_2O_4 : D_4O_5$
$D_2O_4 \cdot 1 \cdot 5 \cdot 2 \cdot 7 \cdot 8 - 2;$	$D_4O_4 : D_4O_2$
$D_4O_4 \cdot 1 \cdot 4 \cdot 1 \cdot 6 \cdot 2 - 0;$	$D_1O_5 : D_1O_2$
$D_1O_5 \cdot 1 \cdot 3 \cdot 2 \cdot 6 \cdot 1 - 6;$	$D_4O_5 : D_5O_5$

ММ є базою для побудови всієї сукупності інформаційних моделей ОУ та МРД КС складних ДІ.

3.3.1. Граф розгалужене дерево

Побудова моделі типу «граф – розгалужене дерево» реалізує наявну мережеву структуру ММ у відповідний набір ієрархічних моделей з приймачем в якості кореня. При цьому досягається можливість просторового подання необхідних структур даної моделі, представлення всіх джерел та відображення шляхів руху даних, установлених на рівні конкретного приймача.

Формалізація моделі граф–розгалужене дерево

$$\left.\begin{array}{l} D_3O_1 : D_4O_2 : D_2O_2 \\ D_4O_4 : D_4O_2 : D_2O_2 \\ D_1O_5 : D_1O_2 : D_2O_2 \end{array}\right\} \in D_2O_2 \cdot 3; \qquad \left.\begin{array}{l} D_6O_1 : D_5O_3 : D_5O_5 \\ D_2O_4 : D_4O_5 : D_5O_5 \end{array}\right\} \in D_5O_5 \cdot 3.$$

Приклад такої моделі побудованої на базі ММ (див. рис.3.4) поданий на рис.3.5.

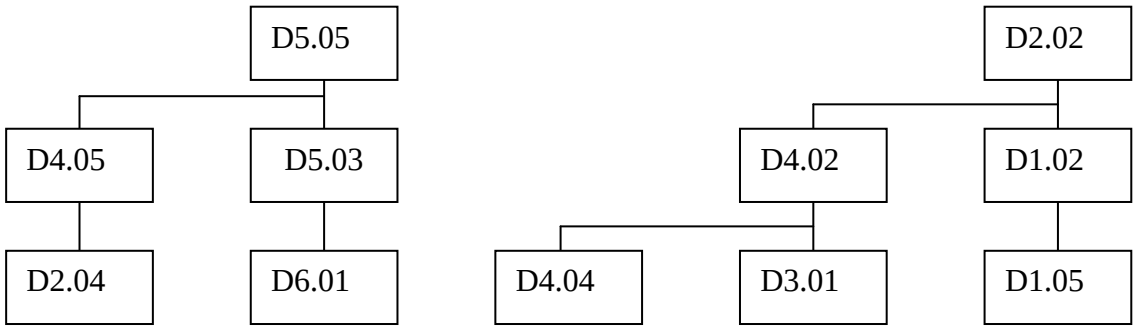


Рисунок 3.5. Модель “граф–розгалужене дерево”

З рис. 3.5. видно, що методологічно модель типу “граф– розгалужене дерево ” будується розширенням графа вліво і вниз.

3.3.2 Часові інформаційні моделі

Виділяють наступні часові інформаційні моделі:

- параметрична часова модель;
- структурно–часова модель;
- мережевий графік руху даних;
- суміщений часовий граф виконання системних функцій.

3.3.2.1. Параметрична часова модель

Параметрична часова модель (рис.3.6) структурно показує необхідний час виконання системних операцій, згрупованих по ознаках джерел, пунктів обробки та приймачів інформації. Розглянута модель дозволяє оцінити і розрахувати необхідні часові, апаратурні або людино–машинні ресурси для реалізації конкретних системних операцій в РКС.

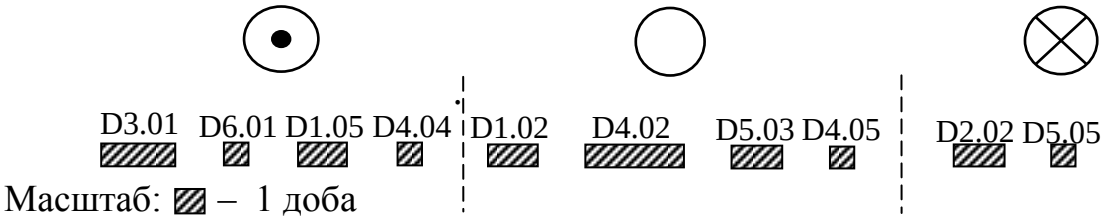


Рис.3.6. Параметрична часова модель

Формалізація параметричної часової моделі має наступний вигляд:

Таблиця ідентифікаторів активного вузла МРД

$$D_3O_1 \cdot 1 \cdot b_{31};\quad D_6O_1 \cdot 1 \cdot b_{61};\quad D_1O_2 \cdot 2 \cdot b_{21};\quad D_2O_2 \cdot 3 \cdot b_{22};\quad D_4O_2 \cdot 2 \cdot b_{42};$$

$$D_5O_3 \cdot 2 \cdot b_{53}\quad D_2O_4 \cdot 1 \cdot b_{24}\quad D_4O_4 \cdot 1 \cdot b_{44}\quad D_1O_5 \cdot 1 \cdot b_{15}\quad D_4O_5 \cdot 2 \cdot b_{45}\quad D_5O_5 \cdot 3 \cdot b_{55}.$$

Впорядкована таблиця активних атрибутів МРД з ознакою тривалості виконання операції b_{ij}

$$D_3O_1 \cdot 1 \cdot b_{31};\quad D_1O_2 \cdot 2 \cdot b_{12}$$
$$D_6O_1 \cdot 1 \cdot b_{61};\quad D_4O_2 \cdot 2 \cdot b_{42}$$
$$D_2O_4 \cdot 1 \cdot b_{24}\quad D_5O_3 \cdot 2 \cdot b_{53}$$
$$D_4O_4 \cdot 1 \cdot b_{44};\quad D_4O_5 \cdot 2 \cdot b_{45}$$
$$D_1O_5 \cdot 1 \cdot b_{15}\quad D_2O_2 \cdot 3 \cdot b_{22}$$
$$D_5O_5 \cdot 3 \cdot b_{55}$$

3.3.2.2. Структурно–часова модель

Структурно–часова модель (рис.3.7) також визначає групування системних операцій по джерелах, пунктах обробки та приймачах, з прив'язкою по горизонталі до початку виконання системних процедур і зберіганням структури руху даних, а по вертикалі – з прив'язкою до конкретних джерел і приймачів даних. Дана модель відображає часову послідовність (причинність) системних процедур і дозволяє на стадії проектування або модернізації розкрити часові неузгодження руху даних і узгодити ресурси мережі з структурою руху даних.

Формалізація параметрів структурно–часової моделі має вигляд:

$$D_1O_5 \cdot 1 : D_2O_2 \cdot 2 \wedge (D_3O_1 \cdot 1 \wedge D_4O_4 \cdot 1 : D_4O_2 \cdot 2) : D_2O_2 \cdot 3;$$
$$(D_2O_4 \cdot 1 : D_4O_5 \cdot 2 \wedge D_6O_1 \cdot 1 : D_5O_3 \cdot 2) : D_5O_5 \cdot 3;$$

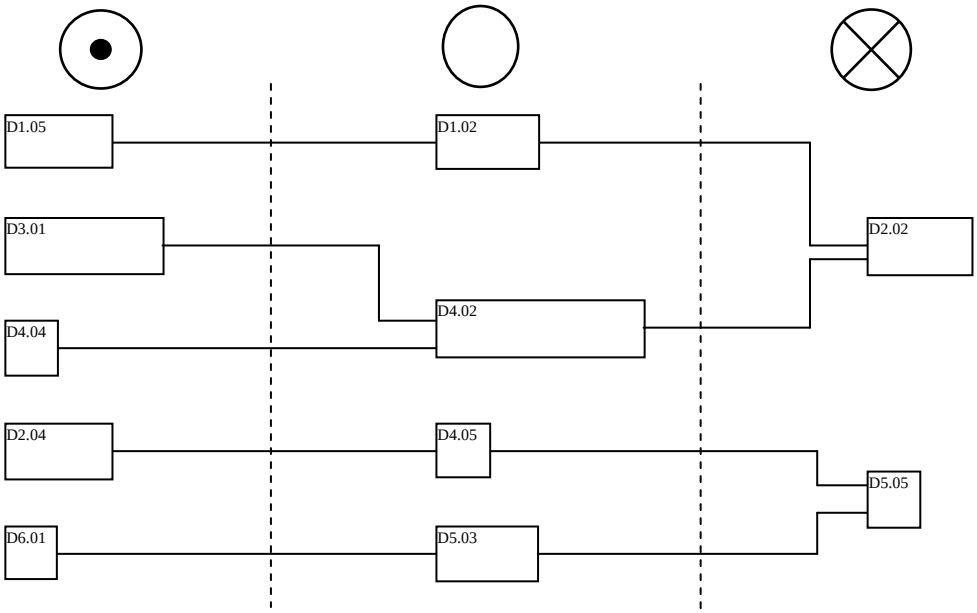


Рис. 3.7. Структурно–часова модель

3.3.2.3 Модель “мережевий графік”

Модель типу ”мережевий графік” (рис.3.8.) будується по типовій методиці побудови мережевих графіків і мереж [115]. В даній моделі з часових параметрів враховується тільки час початку виконання системної операції (a).

Формалізація моделі мережевий графік передбачає формування:

Матриця ідентифікаторів активних вузлів початку виконання операції (a_{ij})

$$\begin{array}{ll} D_3O_1 \cdot 1 \cdot a_{31}; & D_5O_3 \cdot 2 \cdot a_{53} \\ D_6O_1 \cdot 1 \cdot a_{61}; & D_2O_4 \cdot 1 \cdot a_{24} \\ D_1O_2 \cdot 2 \cdot a_{21}; & D_4O_4 \cdot 1 \cdot a_{44} \\ D_2O_2 \cdot 3 \cdot a_{22}; & D_1O_5 \cdot 1 \cdot a_{15} \\ D_4O_2 \cdot 2 \cdot a_{42}; & D_4O_5 \cdot 2 \cdot a_{45} \\ & D_5O_5 \cdot 3 \cdot a_{55} \end{array}.$$

Впорядкована таблиця активних атрибутів з ознакою a_{ij}.

$$\begin{array}{ll} D_3O_1 \cdot 1 \cdot a_{31}; & D_1O_2 \cdot 2 \cdot a_{12} \\ D_6O_1 \cdot 1 \cdot a_{61}; & D_4O_2 \cdot 2 \cdot a_{42} \\ D_2O_4 \cdot 1 \cdot a_{24} & D_5O_3 \cdot 2 \cdot a_{53} \\ D_4O_4 \cdot 1 \cdot a_{44}; & D_4O_5 \cdot 2 \cdot a_{45} \\ D_1O_5 \cdot 1 \cdot a_{15}; & D_2O_2 \cdot 3 \cdot a_{22} \\ & D_5O_5 \cdot 3 \cdot a_{55} \end{array}.$$

Таблиця направлених зв’язків від джерел до приймачів

$$\begin{array}{l} D_3O_1 \cdot 1 \cdot a_{31} : D_4O_2 \cdot 2 \cdot a_{42} : D_2O_2 \cdot 3 \cdot a_{22}; \\ D_6O_1 \cdot 1 \cdot a_{61} : D_5O_3 \cdot 2 \cdot a_{53} : D_5O_5 \cdot 3 \cdot a_{55} \\ D_4O_4 \cdot 1 \cdot a_{44} : D_4O_2 \cdot 2 \cdot a_{42} : D_2O_2 \cdot 3 \cdot a_{22} \\ D_1O_5 \cdot 1 \cdot a_{15} : D_1O_2 \cdot 2 \cdot a_{12} : D_2O_2 \cdot 3 \cdot a_{22} \\ D_2O_4 \cdot 1 \cdot a_{24} : D_4O_5 \cdot 2 \cdot a_{45} : D_5O_5 \cdot 3 \cdot a_{55} \end{array}.$$

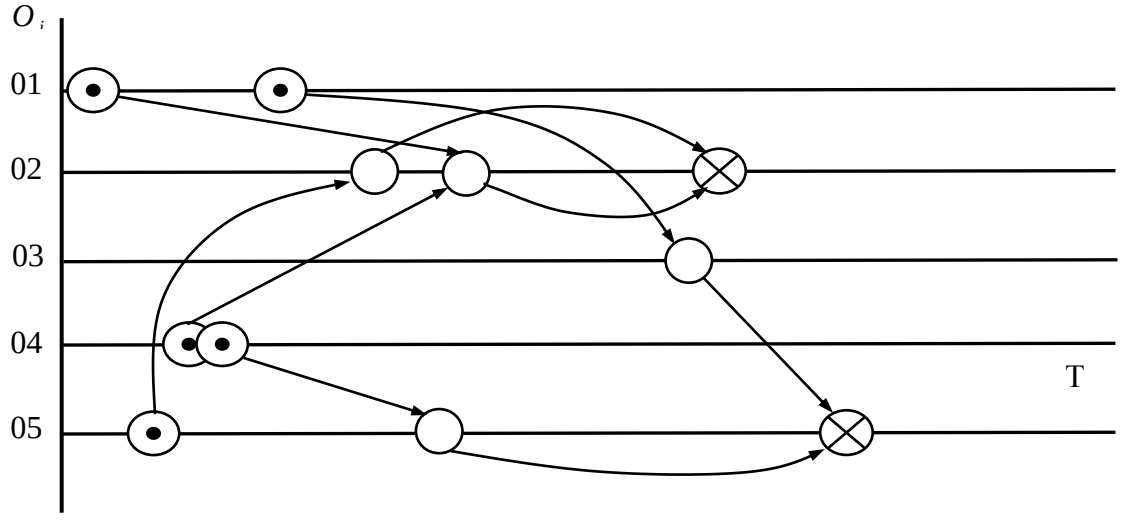


Рис.3.8. Мережевий графік

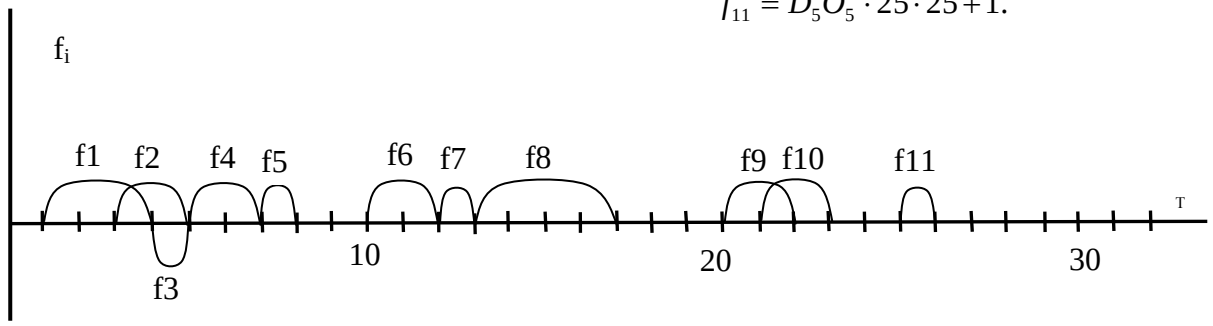
3.3.2.4. Модель суміщений часовий граф

Модель типу “суміщений часовий граф” (рис.3.9) показує асоціацію системних функцій з заданими часовими параметрами без структури руху даних. Розглянута модель дозволяє розрахувати поточні навантаження на обчислювальні ресурси людино–машинних засобів обробки даних в кожному елементі ММ або розподілення пікових навантажень всією мережею обчислювальної системи в цілому. Призначена для центрального сервера КС, який контролює регламентність виконання функцій підрозділами КС і координує їх роботу в часі.

Формалізація моделі суміщений часовий граф:

$$f_i - D_i O_j \cdot a_{ij} \cdot a_{ij} + b_{ij};$$

$$f_i = D_3 O_1 \cdot 1 \cdot 1 + 3; \qquad f_6 = D_1 O_2 \cdot 10 \cdot 10 + 2;$$
$$f_2 = D_1 O_5 \cdot 3 \cdot 3 = 2; \qquad f_7 = D_4 O_5 \cdot 12 \cdot 12 + 1;$$
$$f_3 = D_4 O_4 \cdot 4 \cdot 4 + 1; \qquad f_8 = D_4 O_2 \cdot 13 \cdot 13 + 4;$$
$$f_4 = D_2 O_4 \cdot 5 \cdot 5 + 2; \qquad f_9 = D_5 O_3 \cdot 17 \cdot 17 + 2;$$
$$f_5 = D_6 O \cdot 7 \cdot 7 + 1; \qquad f_{10} = D_2 O_2 \cdot 21 \cdot 21 + 2;$$
$$f_{11} = D_5 O_5 \cdot 25 \cdot 25 + 1.$$



f1 – D3.O1 f4 – D2.O4 f7 – D4.O5 f10 – D2.O2
f2 – D1.O5 f5 – D6.O1 f8 – D4.O2 f11 – D5.O5
f3 – D4.O4 f6 – D1.O2 f9 – D5.O3

Рис. 3.9. Модель суміщений часовий граф.

3.3.3. Модель блок–схема алгоритму оброблення даних

Модель ”блок–схема алгоритму оброблення даних” (рис.3.12) є діагностичною моделлю КС, яка будується на основі суміщеної часової моделі у відповідності з використанням типових позначень блок–схем алгоритмів і

програм. Дана модель використовується центральним сервером РКС для діагностування виконання процесів руху даних в системі згідно регламентного алгоритму її функціонування на основі моделі „суміщений часовий граф”.

Суть побудови даної моделі методично виконується формалізацією процедур побудови оптимального несуперечного змістовного графа розгалуженого алгоритму у наступному порядку:

– формалізація умови задачі;
– побудова суміщеного часового графа;
– побудова логічного розгалуженого графа;
– покриття логічного графа блок–схемою;
– нумерація операторів блок–схеми.

Формалізація моделі „блок–схема алгоритму”:

$$f_i = \begin{cases} f_1, T_1 \le T \le T_4; \\ f_2, T_3 \le T \le T_5; \\ f_3, T_4 \le T \le T_5; \\ f_4, T_5 \le T \le T_7; \\ f_5, T_7 \le T \le T_8; \\ f_6, T_{10} \le T \le T_{12}; \\ f_7, T_{12} \le T \le T_{13}; \\ f_8, T_{13} \le T \le T_{17}; \\ f_9, T_{20} \le T \le T_{22}; \\ f_{10}, T_{21} \le T \le T_{23}; \\ f_{11}, T_{25} \le T \le T_{26}. \end{cases}$$

Формалізація умови задачі полягає у відповідності до нумерації системних функцій $f_i(r)$, які в реальній процедурі описуються системою аналітичних виразів, які виконуються при заданих часових організаціях.

Наприклад, для умови задачі $f_i(r)$ – багатоканальне аналого–цифрове перетворення технологічних параметрів; $f_2(t_2)$ – ковзне усереднення формуючих відліків процесів; $f_3(t_3)$ – обчислення матриці коефіцієнтів кореляції; $f_4(t_4)$ – індикація параметрів.

Формалізація умови задачі побудови моделі „блок-схеми алгоритму”:

$$Y(t)=\begin{cases} f_1(t) & d <= t < b; \\ f_2(t) & c <= t <= d; \\ f_3(t) & e < t. \end{cases},$$

де a,b,c,d,e – часові обмеження.

Нехай a=1, b=2, c=4, d=6, e=5. Тоді суміщений часовий граф (СЧГ) має вигляд, приведений на рис.3.10 – 3.11, де стрілки вказують, що відповідні системні процедури виконуються до настання часу $t=b$ або пізніше часу $t=e$.

Змінюючи значення часових обмежень a,b,c,d,e і здійснюючи розпаралелювання операцій виконання системних процедур $f_i(t)$ одержимо відповідно різні реалізації суміщеного графа (див. рис. 3.10).

Для визначення формалізованої методики побудови розгалуженого логічного графа (РЛГ) сформулюємо ряд стверджень:

1. Системними атрибутами логічного графа є 5 вершин: початок, ввід–вивід, оператор системної функції, умова і кінець;

2. Основним атрибутом розгалуженого логічного графа є умова, при чому:

2.1) якщо умова виконується, ЛГ розширюється вправо, в протилежному випадку – вниз;

2.2) якщо умова невиконується, то вимагається його уточнення, граф розширюється зліва вниз;

3. Вид суміщеного часового графа одночасно визначає структуру логічного графа, при чому:

3.1) якщо системні функції на СЧГ не пересікаються і не накладаються, то ЛГ розширюється вправо і вниз, а для виводу використовується одна загальна вершина;

3.2) в протилежному випадку ЛГ розширюється тільки вниз, а для виводу використовується автономна вершина після кожного оператора системної функції.

Приклади побудови розгалужених ЛГ для суміщених часових моделей ілюструє рис.3.10.

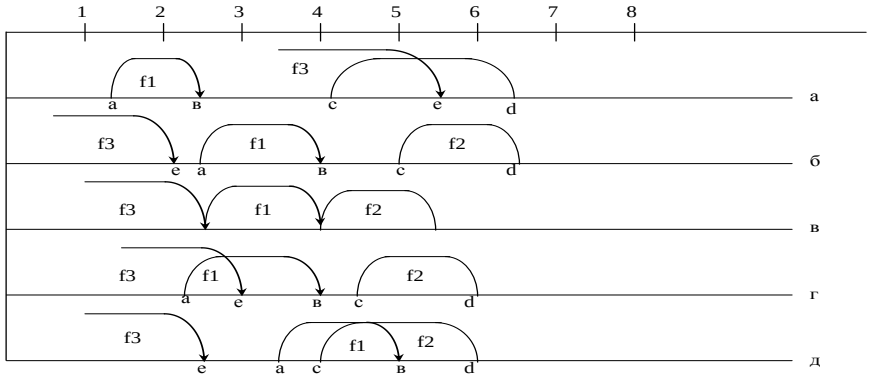


Рисунок.3.10. Приклади суміщених часових графів.

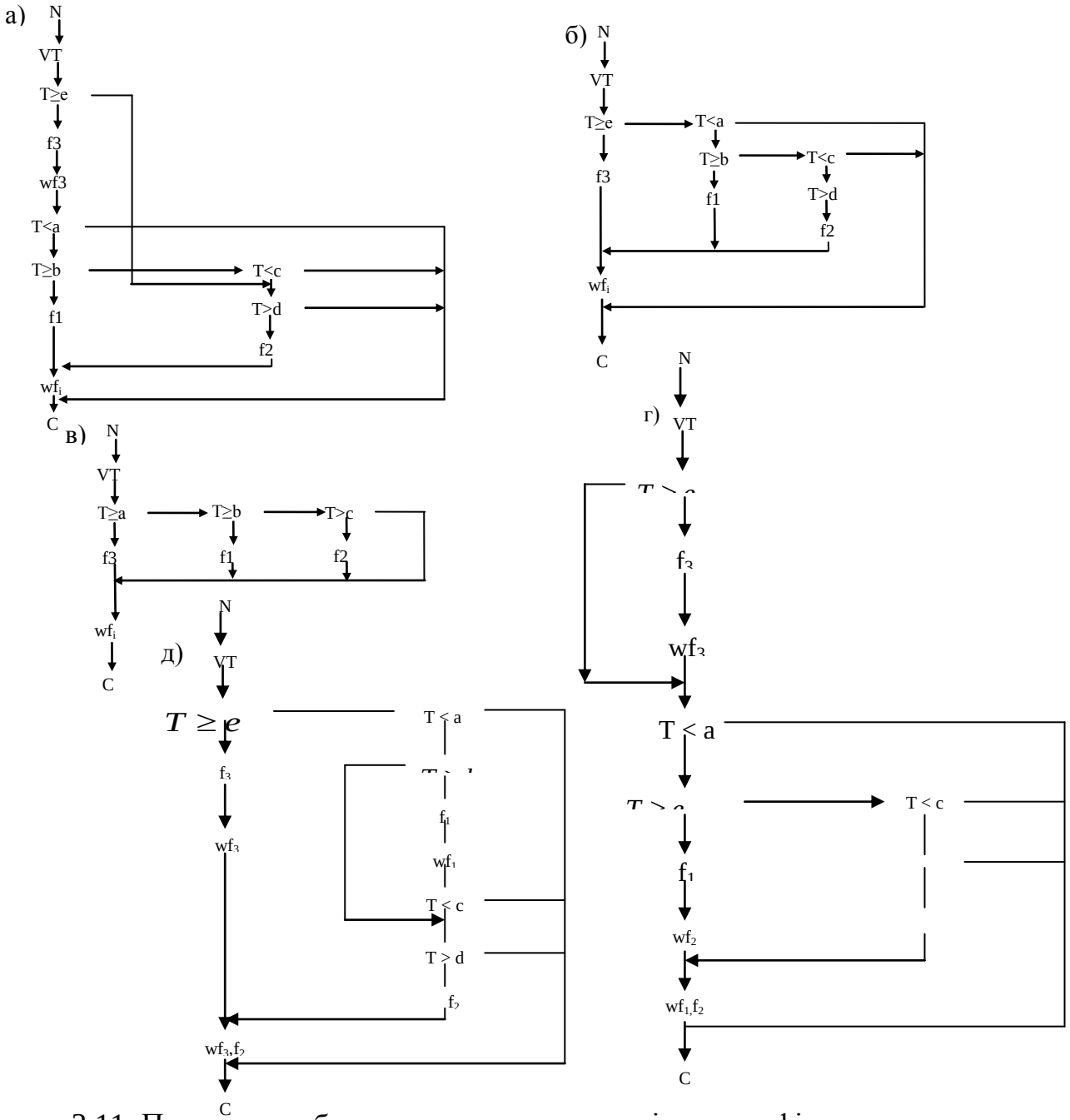


Рисунок 3.11. Приклади побудови розгалужених логічних графів.

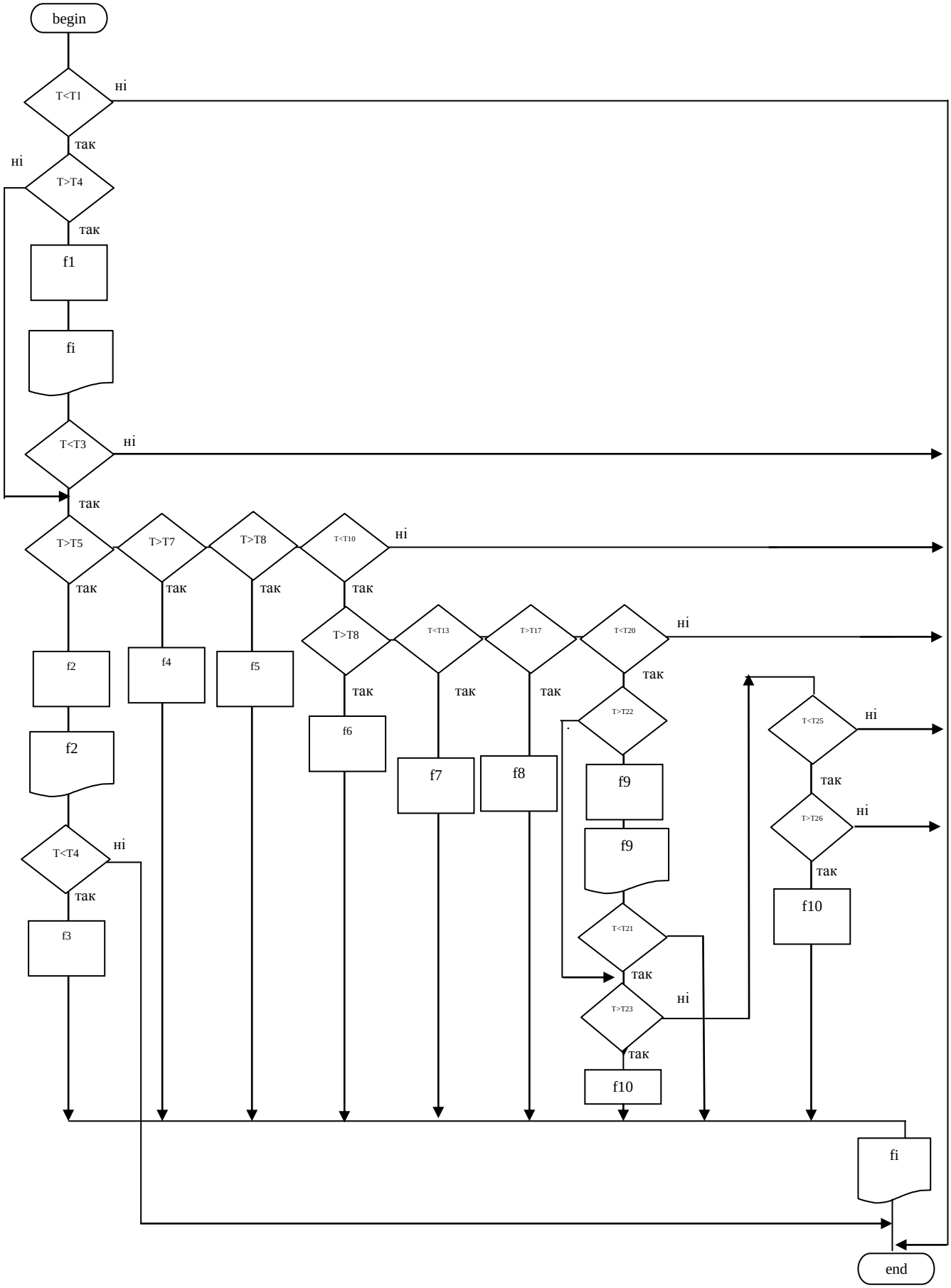


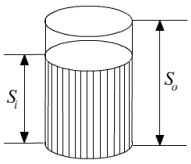
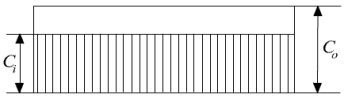
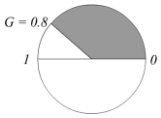
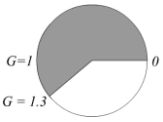
Рисунок 3.12 Блок–схема алгоритму оброблення даних

3.4. Тривимірні матричні моделі

В роботі [6] представлена інформаційна технологія побудови та теоретичне обгрунтування параметрів тривимірної матричної моделі руху даних (ТММРД), яка базується на наступних принципах формалізації (табл.3.3).

Символіка тривимірних матричних моделей руху даних, яка враховує коефіцієнт використання ресурсів елементів ММ.

Таблиця 3.3

Символ	Пояснення
	**S_0** – максимальне число записів **S_i** – реальне число записів
	**C_0** – швидкість створення та передавання даних **C_i** – проектна швидкість створення та передавання даних
	**G=1** – завантаженість 100%
	Завантаженість 80% **G=0.8**
	**G=1.3** 0.3 перевантаження по ресурсах читань

Введена оцінка коефіцієнта ефективності руху даних (3.2) та символіка матричних моделей, яка представлена в табл.3.3, дозволяє розрахувати характеристики швидкості створення повідомлень у вузлах матричної моделі і шляхом їх представлення в трьохвимірному просторі перейти до побудови трьохвимірних матричних моделей .

На рисунку 3.13 показаний приклад тривимірної проекції двовимірної матричної моделі, поданої на рисунку 3.1.

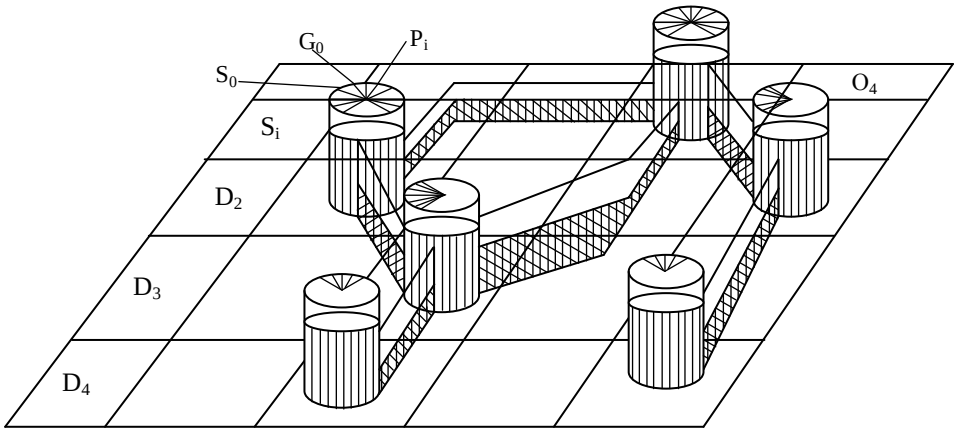
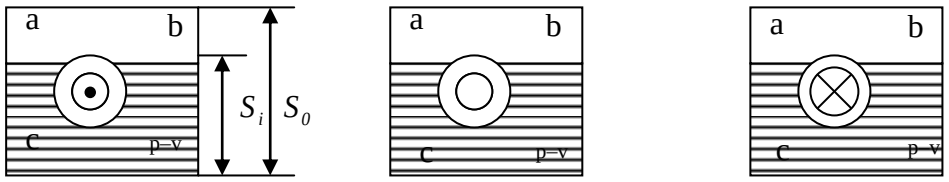


Рис.3.13. Тривимірна проекція двовимірної матричної моделі руху даних

ТММРД є ефективним способом представлення руху даних в КС, враховує ресурси та ступінь використання ресурсів КМ. Однак ТММРД не в повній мірі відображає архітектуру та параметри руху даних в КМ, оскільки представлена досить громіздко в тривимірному просторі і не дозволяє чітко представити часові атрибути матричної моделі.

3.5. Модифіковані двовимірні матричні моделі

Досвід розробки та використання ТММРД дозволяє представити її формальні параметри в двомірному варіанті ММ шляхом додаткового інформаційного насичення елементу ММ. На рис.3.14 показаний приклад інформаційного насичення елементу ДММ параметрами ТММРД [31].



а) б) с)

Рис.3.14. Інформаційне насичення елементу ДММ параметрами ТММРД

На основі введених символів формуємо модифіковану двовимірну ММ, яка враховує формальні параметри ТММРД (рис.3.15).

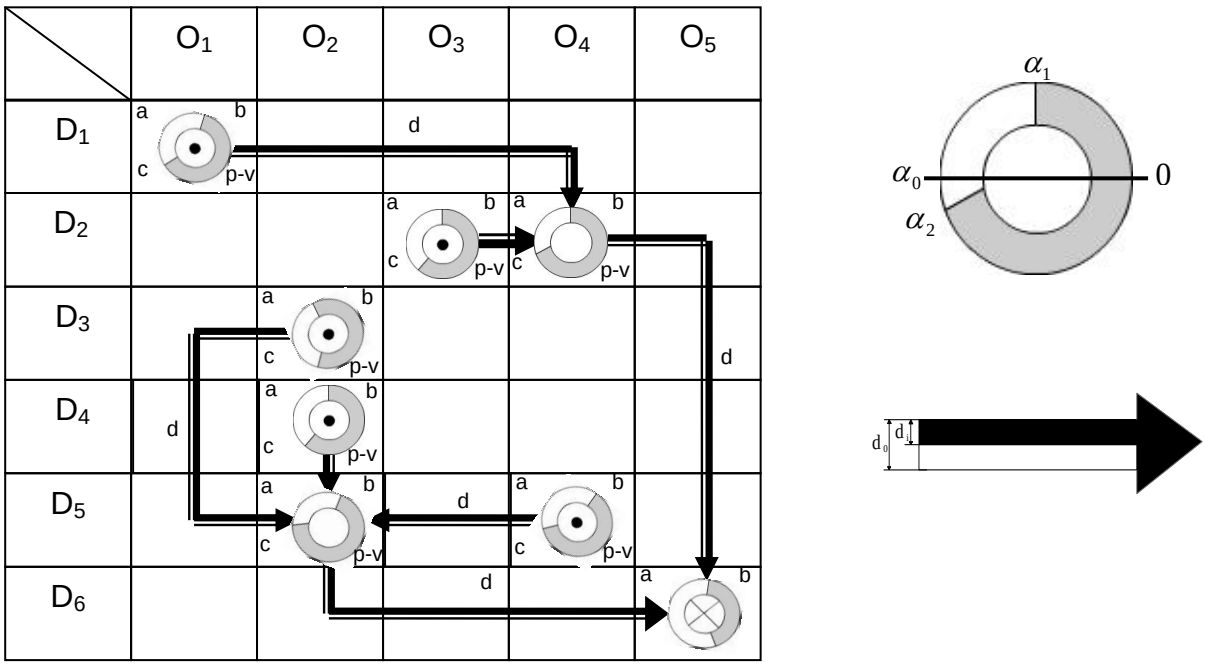


Рис. 3.15. Модифікована двовимірна ММ.

Запропонована система атрибутів та інформаційна технологія побудови модифікованих матричних моделей, приклад яких показаний на рис.3.15, де $\alpha_0 = 180^\circ$ - стовідсоткове використання ресурсів; $0 \le \alpha_1 \le 180^\circ$-ступінь використання ресурсів зчитування даних; $180^\circ \le \alpha_2 \le 360^\circ$-ступінь використання ресурсів записів; d_0, d_i - відповідно ресурси та ступінь використання швидкості передавання даних в каналі зв'язку.

Проведений аналіз інформаційних технологій побудови інформаційних моделей руху даних показує, що технологію проектування та аналіз КМ промислового підприємства в кожному конкретному випадку можна однозначно формалізувати на основі символіки ДММРД.

В той же час дана модель не відображає ресурсів КМ при виконанні операцій в кожному елементі ММ. Тому розроблена технологія побудови ТММРД, яка враховує наявність ресурсів в кожному елементі ММ, а також ступінь їх використання шляхом введення спеціальної символіки і трьохмірного зображення руху даних, компенсує недоліки ДММРД. Досвід проектування КМ та побудови ТММРД продемонстрував деяку громіздкість даного представлення руху даних у вигляді ТММРД. Розроблена технологія представлення ресурсів та їх ступені

використання у вузлах ММ, шляхом модифікації ДММ привела до спрощення моделі представлення формалізованого руху даних без втрати інформативності.

Очевидно, що в окремих випадках можуть використовуватись всі типи описаних інформаційних моделей руху даних, на основі яких однозначно у формалізованій формі можуть бути побудовані інші проекції моделей руху даних, які входять в їх базове сімейство.

3.6. Розробка моделей руху даних для комп'ютерних систем з різною архітектурою

В даний час існує широка різноманітність архітектур інформаційних систем, до яких належать концентровані та розподілені системи обробки даних. До систем першого класу можна віднести монопольні архітектури, архітектури з розподіленим часом, архітектури з мультипрограмною та мультипроцесорною обробкою даних. Другий клас представлений значним числом однорівневих архітектур сучасних комп'ютерних мереж, в тому числі: магістральні, зіркові, кільцеві, систолічні [54,55]. До класу багаторівневих розподілених архітектур інформаційних систем сляд віднести ієрархічні, багаторівневі–магістральні та зірково–магістральні архітектури [56].

Окремим класом архітектур представлені безпровідні радіотехнічні інформаційні системи та комп'ютерні мережі наступного типу:

– безретрансляторні;

– з пасивними ретрансляторами;

– з активними ретрансляторами, в тому числі сотові мережі.

Комп'ютерні системи з оптичними каналами зв'язку охоплюються архітектурами на основі:

– дуплексних оптичних ретрансляторів;

– оптичних активних ретрансляторів;

– оптичних сканерів;

– волоконно–оптичних ліній зв'язку.

Значною оригінальністю архітектур характеризуються спеціалізовані комп'ютерні системи (СКС), які часто можуть базуватися на об'єднанні окремих елементів різних типових архітектур [16, 133, 134, 135]. До такого класу інформаційних систем, наприклад, належать:

– системи обліку витрат енергоносіїв з глибоким розпаралеленням потоків даних;

– комп'ютерні розподілені системи екологічного моніторингу;

– спеціалізовані охоронні системи;

– проблемно–орієнтовані корпоративні системи промислових та адміністративних організацій.

Така велика кількість архітектур інформаційних систем в значній мірі ускладнює вирішення задач оптимізації проектних рішень при побудові інформаційних систем, що потребує розробки відповідних моделей архітектур, які б дозволили шляхом формалізації структурних елементів різних мереж з єдиних позицій провести дослідження та порівняння їх системних характеристик. Одним з перспективних підходів до вирішення такої задачі є використання теорії та технології побудови одномірних та багатомірних матричних моделей руху даних [57], що визначає актуальність таких досліджень.

На рис.3.16 подана класифікація архітектур комп'ютерних систем з фізичними лініями зв'язку.

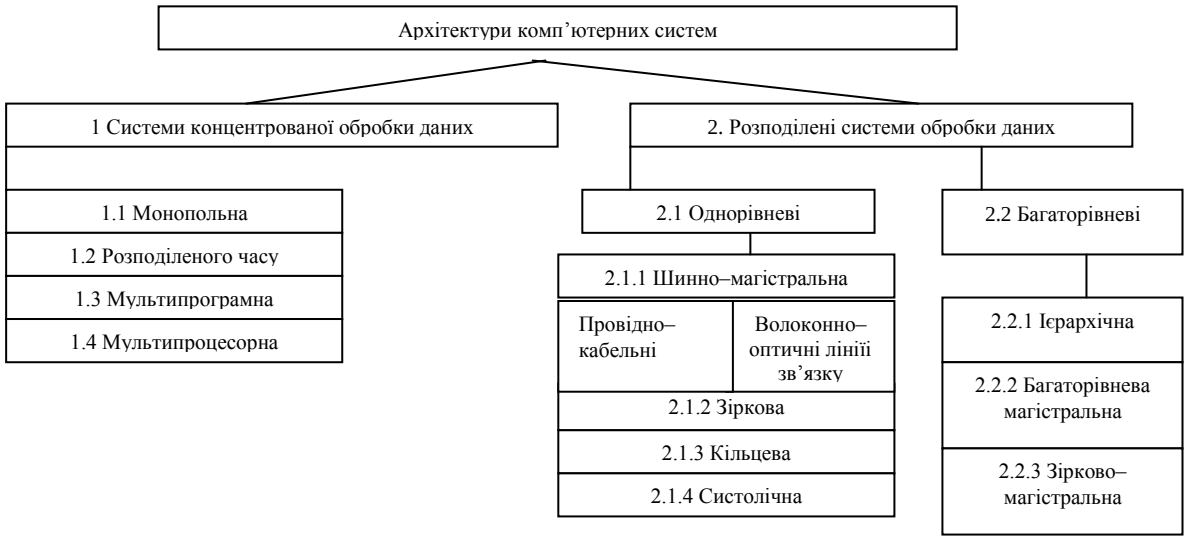


Рис.3.16. Класифікація архітектур КС з фізичними лініями зв'язку

На рис. 3.17 подана класифікація архітектур КС з безпровідними лініями зв’язку.

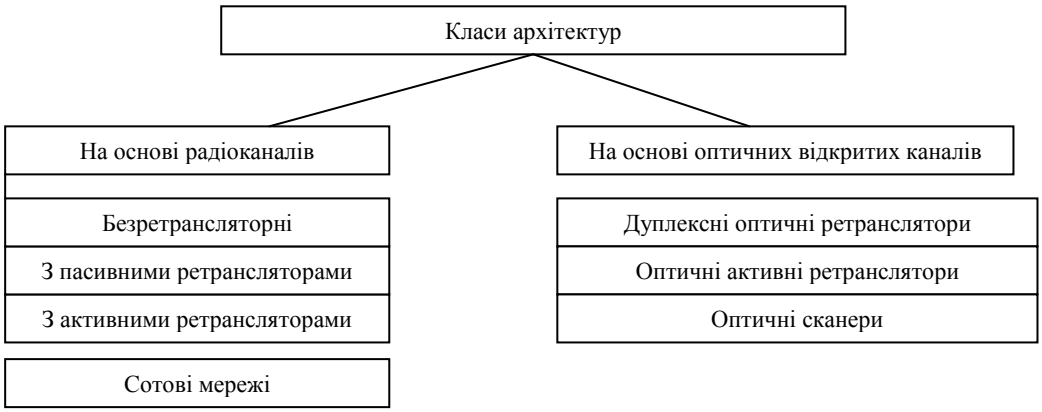


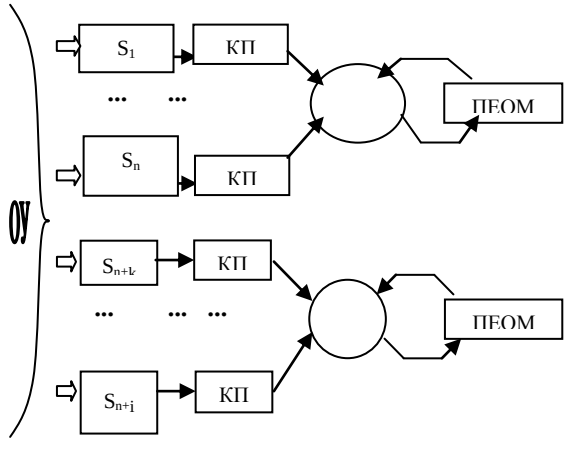
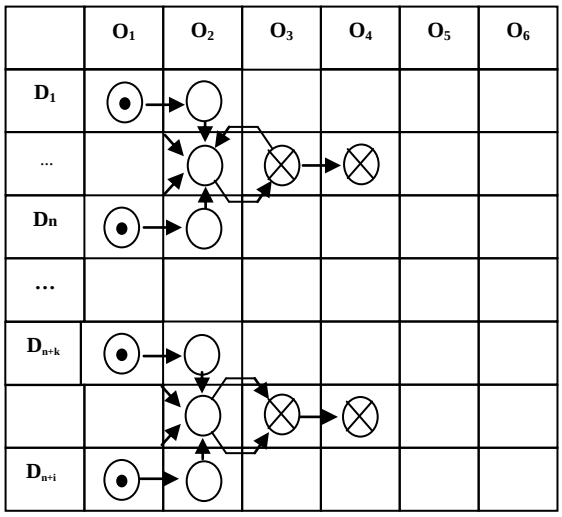
Рис.3.17. Класифікація архітектур КС з безпровідними радіоканалами.

Подані класифікації архітектур КС дозволяють виконати формалізований вибір відповідного класу архітектури КС в залежності від їх проблемної орієнтації та необхідних системних характеристик.

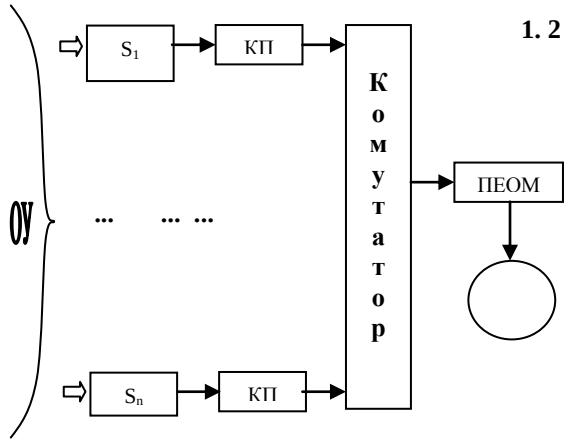
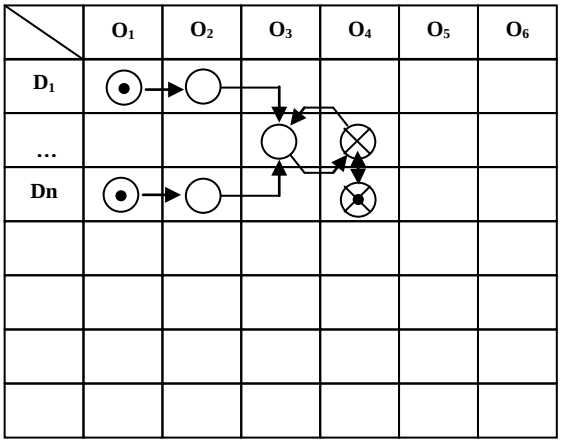
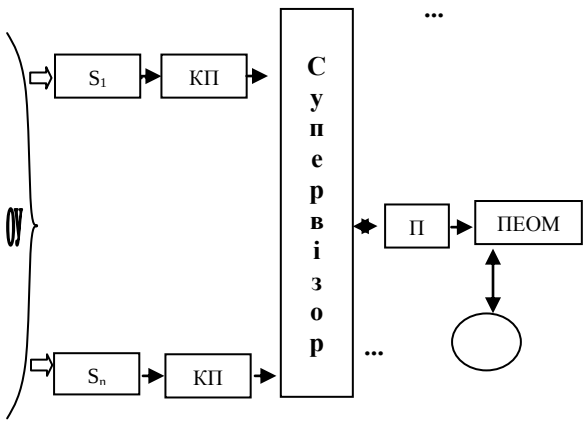
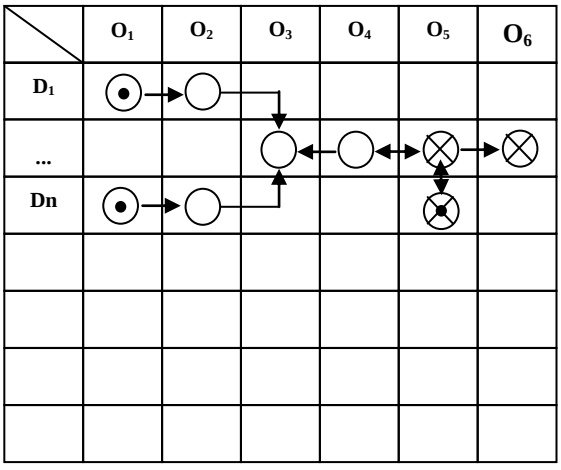
Формалізацію архітектур інформаційних мережевих систем на першому етапі доцільно виконати на основі технології побудови та символіки двомірних матричних моделей [4, 145].

У табл.3.4. подані формалізовані моделі архітектур систем концентрованої обробки даних.

Моделі архітектур КС *Таблиця 3.4*

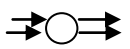
Архітектура КС	Модель руху даних
1.1 	

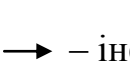
Продовж. табл.3.4

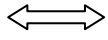
1. 2  	
1. 3 ...	
1. 4  	

У таблиці 3.4 використовуються наступні позначення: *ОУ* – об'єкт управління; S1, Sn – датчики; КП – контрольні пункти; ПЕОМ – персональна електронно-обчислювальна машина; D1, Dn – дії; O1, O2, O3, O4, O5, O6 – операції.

равління, $S_1 \ldots S_n$ –сенсори, КП–кодовий перетворювач, – джерело інформації,

 –пункт обробки інформації,  –пункт обробки даних,

 – пункт затвердження та архівації,  – оператор (джерело–приймач) інформації  фізичний зв'язок,  – інформаційний зв'язок,

 –паралельна шина даних,  – послідовна шина даних (мережева магістраль), ПЕОМ –персональний комп'ютер, СТ – станція мережі,

П –процесор, ПКД –пам'ять колективного доступу,

К – комутатор, СВ – супервізор, ВМ – виконавчий механізм,

D – документ, O –об'єкт, $D_i O_j$ –пункт моделі руху даних.

Аналіз архітектур комп’ютерних систем [50, 136, 140] концентрованої обробки даних дозволяє зробити наступні висновки:

1). Переваги та недоліки.

Монопольна архітектура (див.позицію 1.1 табл.3.4) характеризується максимальним паралелізмом руху даних, внаслідок цього має максимальну подільність та живучість. Крім того, кожен оператор володіє всіма ресурсами персональних ЕОМ, включаючи повний об’єм пам’яті, швидкодію, операційне та прикладне програмне забезпечення, час.

Основними недоліками такої архітектури є:

– відсутність інформаційних зв’язків між комп’ютерами, що не дозволяє операторам використовувати масиви даних та прикладні програми інших операторів без фізичного переміщення носіїв даних;

– надзвичайно висока собівартість обробки даних, обумовлена невідповідністю ресурсів операторів і ПЕОМ. Наприклад, швидкість формування даних оператором з клавіатури 5–12 операцій/с, а швидкість обробки даних ПЕОМ $10^6 – 10^9$ операцій/с;

– недостатнє використання часового ресурсу (4–6 год./добу, що складає 25% потенційних можливостей ПК);

– непрофесійність, недостатня кваліфікація та низька математична підготовка операторів, які не використовують всі можливості операційної системи

та прикладного програмного забезпечення. Тому експертна оцінка ефективності монопольної архітектури складає 0,01 % ККД.

Архітектура розподіленого часу (див.позицію 1.2 табл.3.4) не забезпечує високопаралельного режиму роботи, що обумовлено наявністю комутатора інформаційних потоків. При цьому також виникають ефекти старіння інформації. Позитивними характеристиками такої архітектури є зниження собівартості обробки даних за рахунок більш ефективного завантаження КС в часі та колективного користування ПЗ та масивами даних.

Основними недоліками даної архітектури є суттєве зниження надійності, яка обумовлена наявністю комутатора та одного колективного процесора, а також обмеження ресурсів часу для кожного оператора та монопрограмне рішення задач, що може призводити до створення черг на рівні операторів (ККД = 0,1 %) .

Мультипрограмна архітектура (див.позицію 1.3 табл.3.4) дозволяє розпаралелити інформаційні потоки та організувати одночасне виконання всіх задач в мультипрограмному режимі .

Недоліками такої архітектури є значна вартість обробки даних внаслідок використання одного потужного процесора та складного ПЗ з багаторівневою системою переривань та пріоритетів. Дана структура характеризується невисокою надійністю, великою ймовірністю зависання та невизначеністю часу завершення конкретних задач, що породжує невизначеність часу очікування окремих операторів. Крім того, завантаженість системи залежить від активності операторів (ККД= 1,0–1,2%).

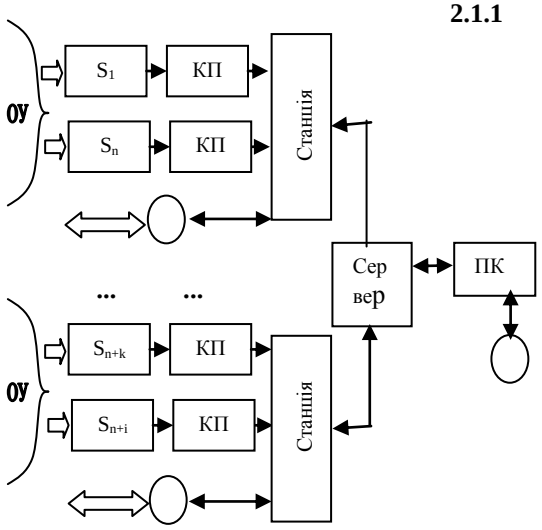
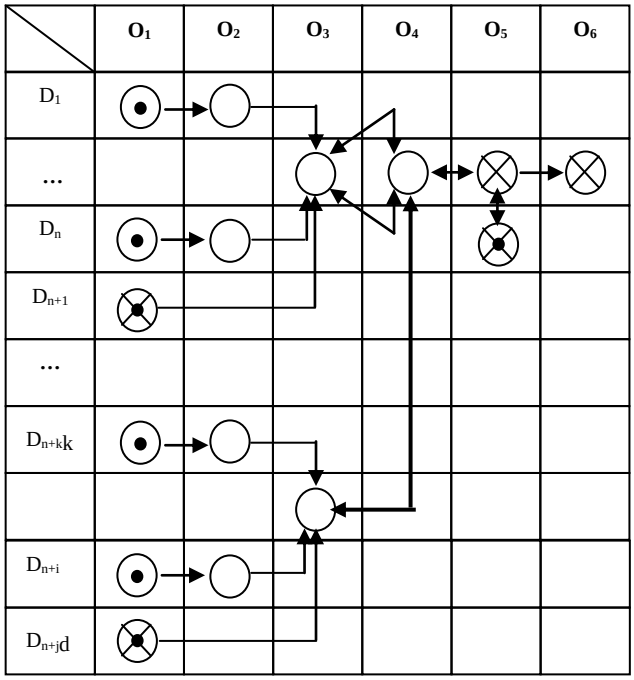
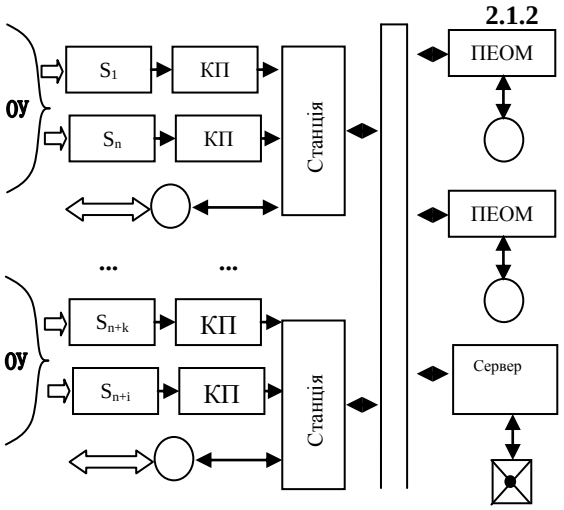
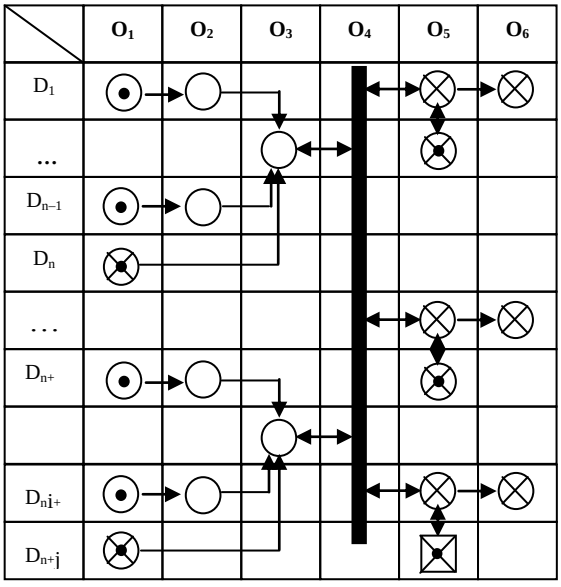
Мультипроцесорна архітектура (див.позицію 1.4 табл.3.4) забезпечує суттєве підвищення надійності системи, можливості розпаралелення інформаційних потоків та суттєве зниження собівартості обробки даних за рахунок одночасного використання супервізора та групи процесорів різної потужності. При цьому супервізор не тільки аналізує активність та характер задач, які виконуються операторами, але й розподіл потужностей процесорів та їх головних ресурсів. Незалежність від активності сенсорних даних забезпечується цілодобовим рішенням фонових задач, які завантажуються в супервізор операторами через комутатор. Позитивною характеристикою даної архітектури є

наявність прямих інформаційних зв’язків між процесорами на основі високошвидкісної паралельної шини (ККД= 5–10%).

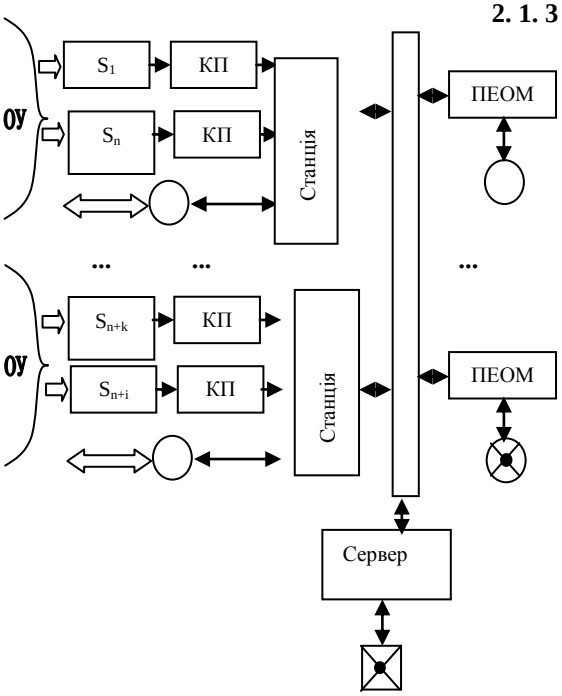
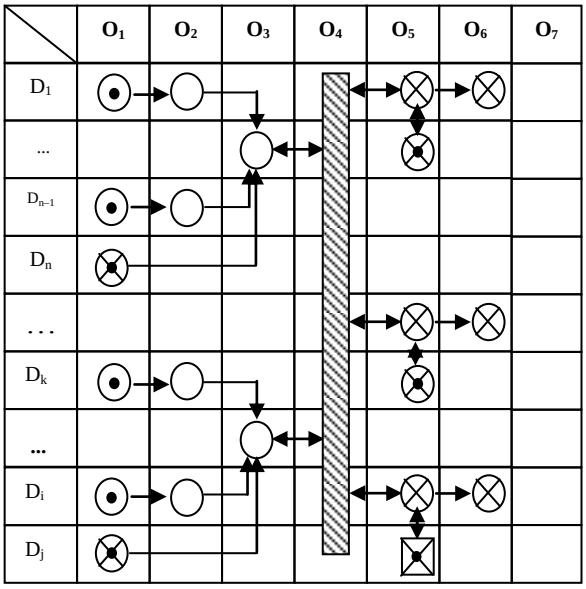
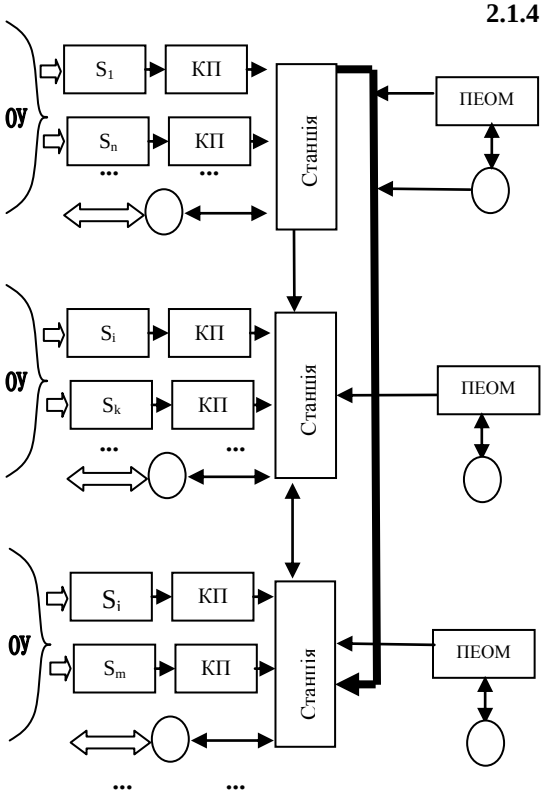
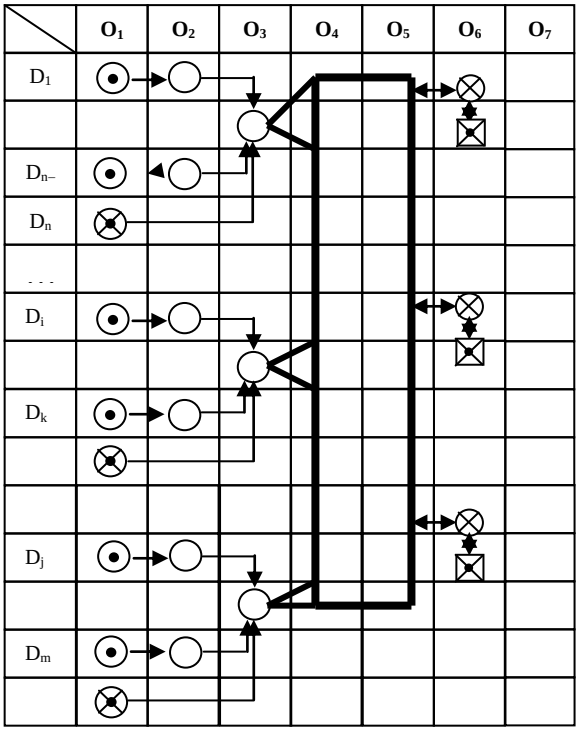
Таким чином, з даного класу архітектур принципам паралелізму формування та цифрової обробки даних у матричних моделях відповідають 1, 3, 4 моделі.

У табл. 3.5 подані формалізовані моделі руху даних розподілених однорівневих комп’ютерних систем. *Таблиця 3.5*

Моделі руху даних однорівневих КС

Архітектура КС	Модель руху даних
2.1.1 	
2.1.2 	

Продовження табл.3.5

2. 1. 3 	
2.1.4 	

Аналіз системних архітектур розподілених систем обробки даних та світової практики їх розробки та експлуатації [4] дозволяє зробити наступні висновки:

Шинно–магістральна архітектура (див.позицію 2.1.1 табл.3.5) характеризується зниженням собівартості обробки розподілених даних за рахунок суттєвого зниження вартості магістрального фізичного каналу зв’язку на основі провідних ліній (витої пари та коаксіального кабеля) або волоконно–оптичних ліній. Головними перевагами такої архітектури є можливість встановлення безпосередніх інформаційних зв’язків між станціями, а також колективне використання ресурсів сенсорних даних та сервера. В той же час дана архітектура має ряд суттєвих недоліків, обумовлених можливістю колізій та конфліктів, які ліквідуються на основі спеціальних складних протоколів доступу, а також низьку надійність обумовлену одним каналом зв’язку. Незважаючи на ці недоліки, ККД становить 20–30%.

Зіркова архітектура (див.позицію 2.1.2 табл.3.5) найчастіше використовується для створення ЛОМ з концентрованою БД на рівні сервера. В даній архітектурі відсутні колізії та організовуються прямі інформаційні зв’язки між станціями за рахунок високошвидкісного матричного комутатора. Перевагою даної архітектури є колективне користування ресурсами потужного сервера. Основним недоліком зіркової архітектури є відносно висока вартість системи каналів зв’язків та недостатня надійність обумовлена комунікаціями через один сервер. В той же час ККД становить 20–30%.

Кільцева архітектура (див.позицію 2.1.3 табл.3.5) найбільш ефективна з даного класу архітектур. Характеризується низькою вартістю каналу зв’язку, що представляє собою шинну магістраль, замкнуту в кільце. Швидкісний маркер, який циркулює в кільці, виключає колізії та спрощує протоколи доступу до каналу. При однократному розриві каналу зв’язку робота мережі не порушується, а по часу поширення сигналів між станціями сервер може локалізувати місце розриву та видати повідомлення про необхідність його профілактики. Головним недоліком такої архітектури є можливість довготривалої затримки маркера на станції, яка вийшла з ладу, що потребує додаткового тестування мережі з метою вилучення таких станцій. ККД становить 30–40%. В окремих випадках

організовується реалізація каналу на основі подвійного кільця, що суттєво підвищує надійність такої мережі. Тоді ККД може становити 50%.

Систолічна архітектура (див.позицію 2.1.4 табл.3.5) базується на організації всіх прямих інформаційних зв'язків між будь–якою парою станцій, що забезпечує максимальну надійність телекомунікаційної системи. До класу таких мереж належать СПД системи з комутацією каналів. ККД становить 50–60%, але в той же час ця архітектурахарактеризується двома основними недоліками: максимальною вартістю каналів зв'язку та можливістю кластерного збудження мережі. Систолічні архітектури є високоемерджентні $K_e > 2$.

Загальним недоліком описаних архітектур та їх моделей руху даних є однорівневість, що значно звужує сферу їх застосування. Принципам паралелізму формування та обробки даних відповідає дві архітектури – зіркова та систолічна.

У табл. 3.6 подана формалізована модель багаторівневої ієрархічної архітектури розподілених систем.

Ієрархічна архітектура (див.позицію 2.2.1 табл.3.6) розподілених систем обробки даних, як видно з табл.3.6, багаторазово повторює зіркову архітектуру, тобто являється її розширенням. Головною перевагою ієрархічної архітектури є зниження собівартості обробки даних за рахунок адаптації ресурсів КМ на кожному рівні або в окремих ієрархічних вітках. Такі архітектури характеризуються максимальною стійкістю. В той же час кожна станція або окремі абоненти мають можливість користуватися потужними обчислювальними ресурсами інших рівнів. ККД таких архітектур становить 40–60%.

Головними недоліками даної архітектури є:

– відсутність прямих інформаційних зв'язків між станціями одного рівня;

– низька емерджентність (інтелектуальність), яка визначається відношенням числа зв'язків до числа станцій, $K_e < 1$;

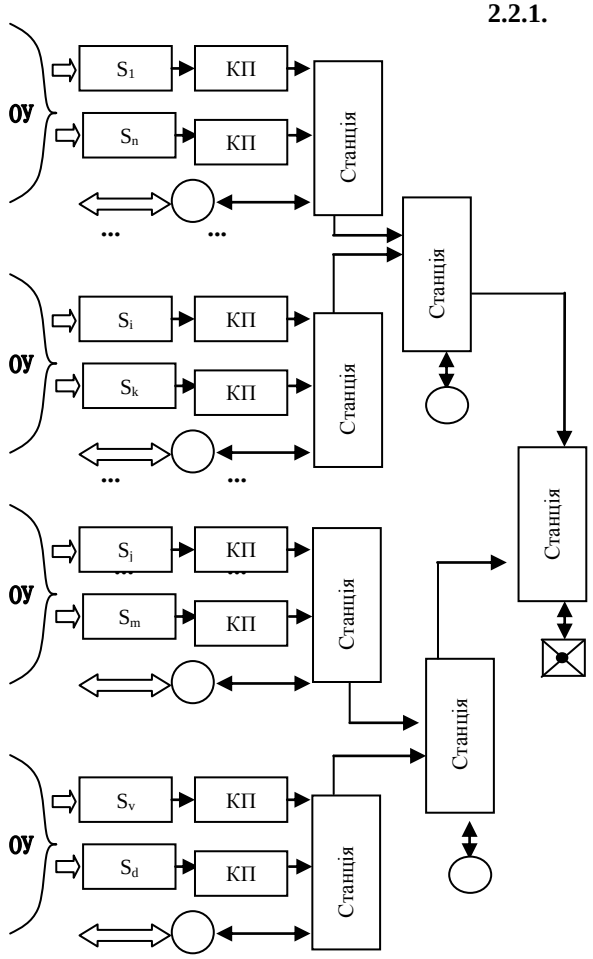
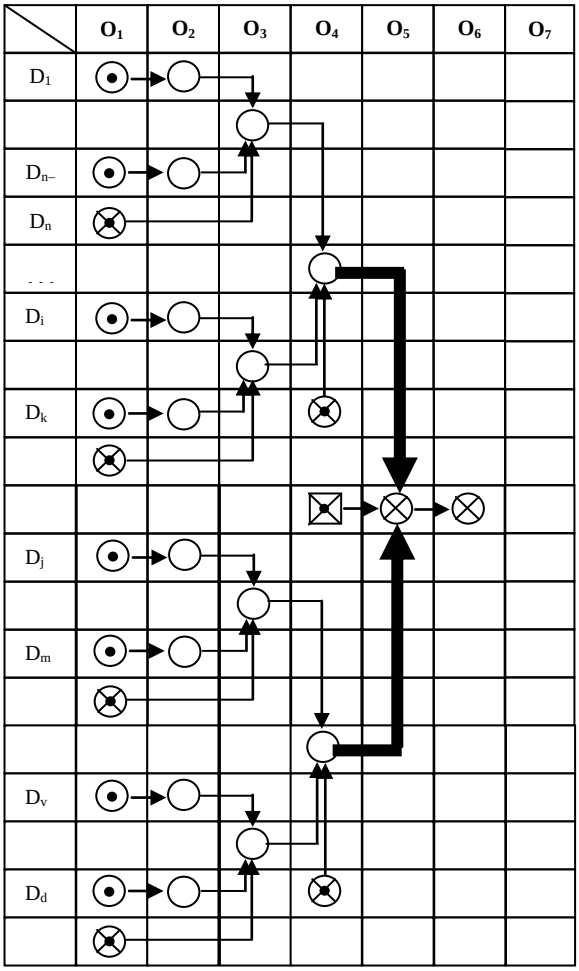
– можливість катастрофічного розмноження помилок при переміщенні даних з верхніх рівнів до нижніх;

– зростання ваги помилок при переміщенні даних з нижніх рівнів до верхніх.

Формалізовані моделі руху даних ієрархічних архітектур КС найчастіше можуть зустрічатися на підприємствах технологічного виробництва та адміністративних установах.

Таблиця 3.6.

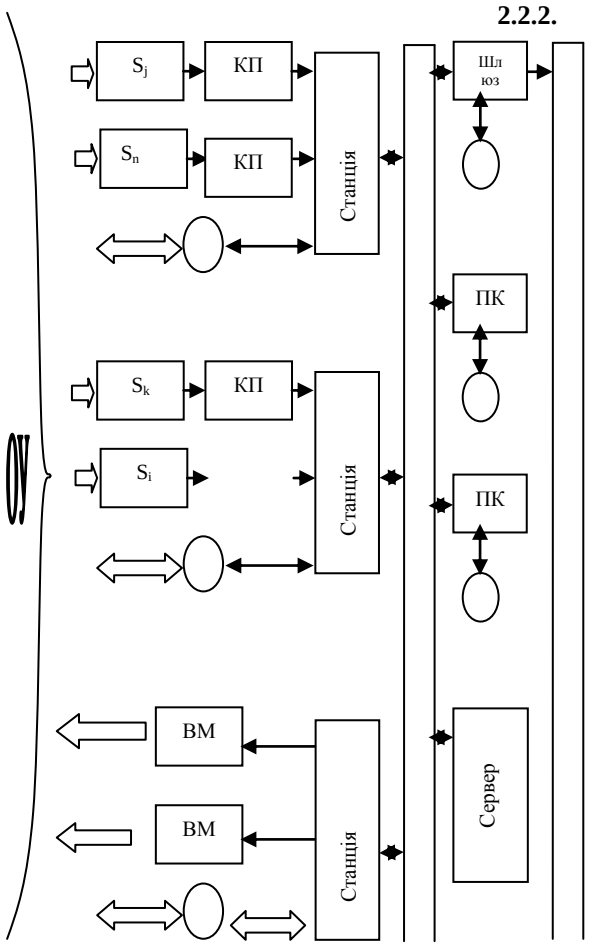
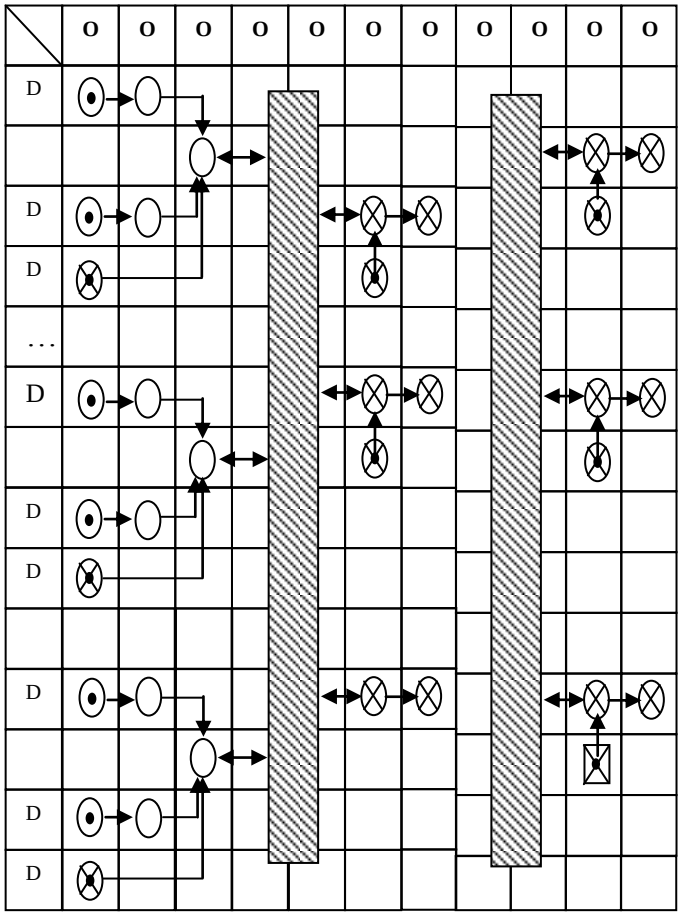
Моделі архітектур КС

Архітектура КС	Модель руху даних
2.2.1. 	

У табл. 3.7 подана формалізована модель багаторівневої магістральної архітектури комп’ютерних систем.

Таблиця 3.7.

Моделі архітектур КС

Архітектура КС	Модель руху даних
2.2.2. 	

2.2.2. Багаторівнева магістральна архітектура

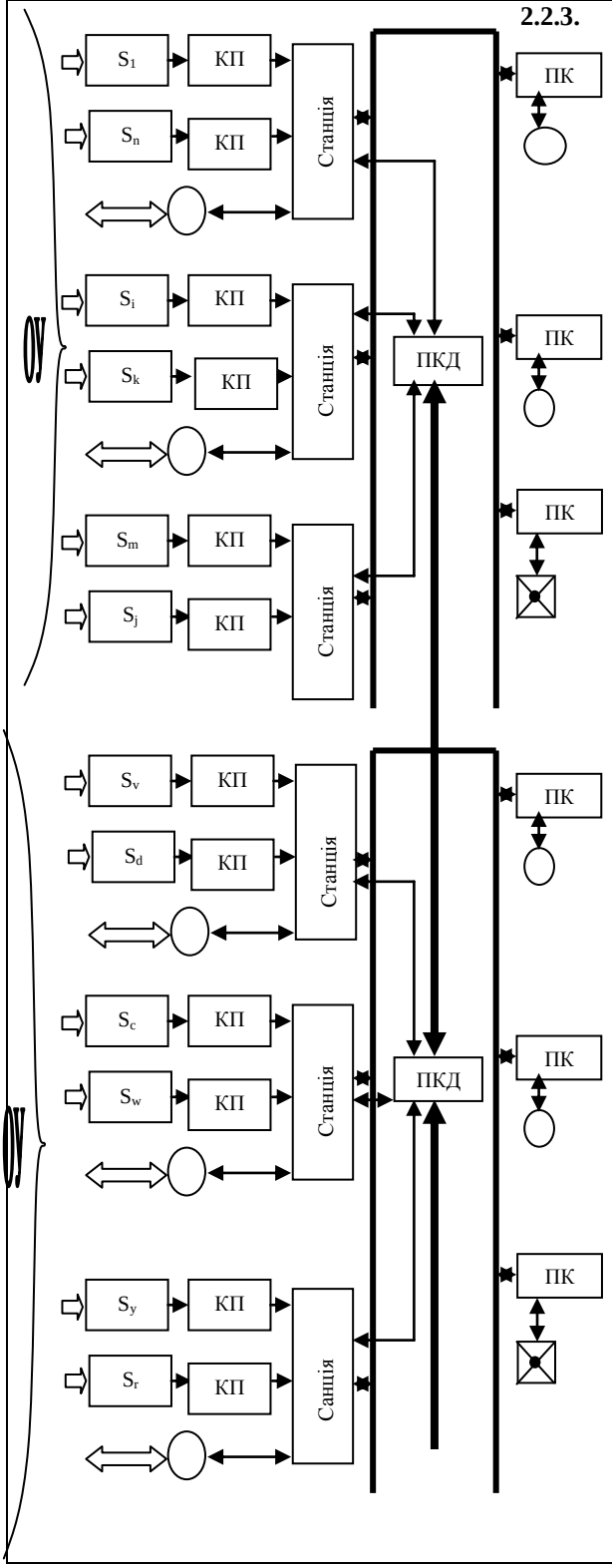
Ця архітектура максимально адаптована до структури промислових виробництв і охоплює три рівні: технологічний (сенсори, виконавчі механізми, оператори, технологи та станції, кодери), цеховий (ЕОМ типу сервер та цехові ПК), адміністративний (ПЕОМ адміністративного апарату та системний сервер з адміністратором БД). ККД становить 50–65%.

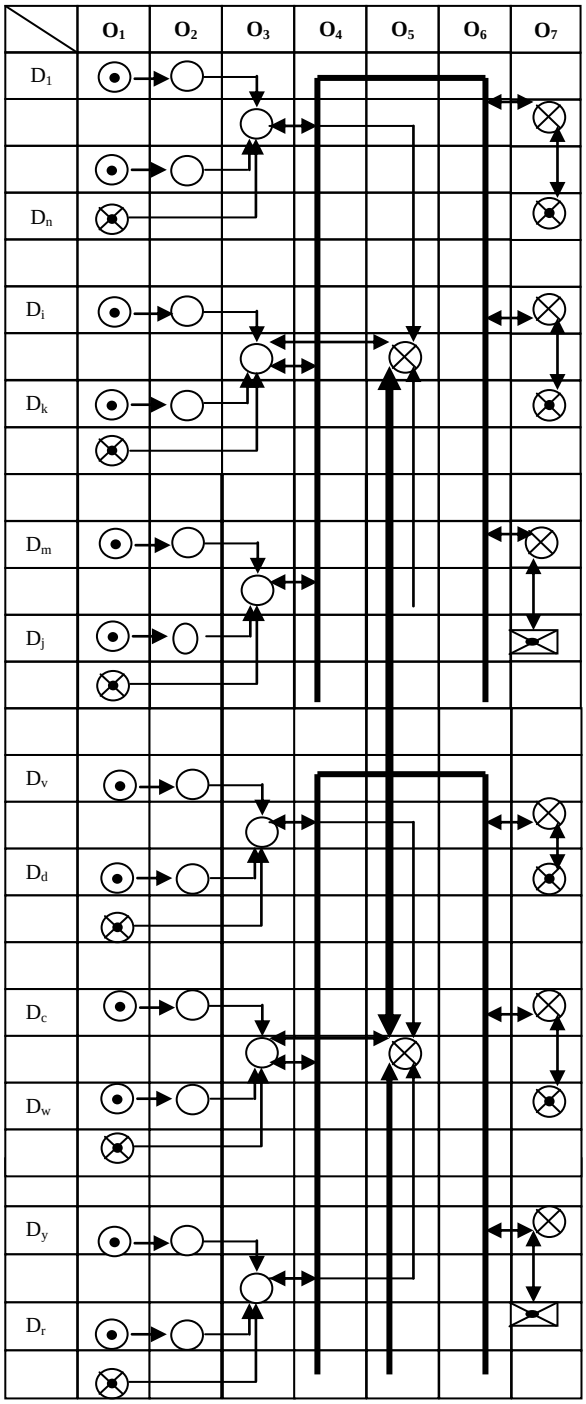
2.2.3. Зірково–магістральна архітектура розподілених систем обробки даних

У таблиці 3.8 подана формалізована модель розподіленої системи обробки даних на основі зірково–магістральної архітектури.

Моделі архітектур КС *Таблиця 3.8.*

Архітектура КМ	Модель руху даних
2.2.3.	





Дана архітектура характеризується суттєвими перевагами по відношенню до інших описаних архітектур, що виражається в наступному:

– багаторівнева розподілена обробка даних;

– відсутність міжрівневих шлюзів, які значно обмежують ресурси багаторівневих систем;

– висока надійність і живучість за рахунок реалізації розпаралелення обробки даних, в тому числі при черезрівневих обмінах;

– висока емерджентність;

– реалізація принципів паралелізму на рівні сенсорів та ПКД.

Особливістю даної архітектури є організація паралельного запису даних в поштові скриньки асоціативних елементів ПКД та можливість паралельного зчитування всіма абонентами будь–яких даних всіх елементів ПКД.

Швидкісний канал міжрівневого зв'язку між ПКД, як правило, реалізується на основі волоконно–оптичних ліній зв'язку, що дозволяє вирівняти трафіки магістральних ліній кожного рівня та розпаралелити міжрівневі зв'язки.

Оцінка ККД зірково–магістральної архітектури сягає 60–80%. Тобто можна вважати дані архітектури найбільш перспективними в застосуванні у КС збору, обробки даних та управління складними об'єктами в реальному масштабі часу.

З приведених таблиць формалізованих моделей архітектур КС видно, що їх представлення в двомірних координатах вузлів матричної моделі (ММ) – $D_i : O_j$ дозволяє спростити формалізацію опису функції кожного вузла. На основі ММ визначають характеристики ліній зв'язку та інформаційних потоків між окремими вузлами. Такий підхід дозволяє спростити та конкретизувати побудову складних ММ, в тому числі, тривимірних для реальних підприємств та адміністративних установ. При цьому ефективно може бути використана технологія побудови інших моделей руху даних, які характеризують часові характеристики формування, передавання, обробки та архівації даних, а також конкретизувати типи функцій у вузлах ММ згідно стандартної символіки.

Розроблена технологія побудови економічних епюр та стратегій проектування і впровадження комп'ютерних систем на основі приведених формалізованих моделей дозволяє оптимізувати процеси проектування та відбір оптимальних проектів.

Запропонована формалізація архітектур інформаційних систем на основі матричних моделей руху даних дозволяє системно представити та формалізувати процеси формування, передавання, цифрової обробки, затвердження та архівації даних в розподілених комп'ютерних системах. Дана формалізація дозволяє ефективно використати апарат побудови сімейства інших моделей руху даних на основі характеристик двомірної матричної моделі, а також використати технологію побудови трьохмірних матричних моделей складних об'єктів управління.

Аналіз формалізованих моделей архітектур існуючих КС дозволяє виділити класи мереж, які забезпечують високий паралелізм формування та цифрової обробки даних, особливо перспективних при побудові систем реального часу.

При цьому очевидні переваги по надійності, можливості глибокого розпаралелення обробки даних та мінімізації собівартості руху даних забезпечують багаторівневі провідні та безпровідні зірково–магістральні архітектури.

Розроблена інформаційна технологія побудови тривимірних моделей руху даних та економічних епюр руху даних показує, що останні є важливим інструментом оптимізації проектних рішень КС, що визначає актуальність досліджень в цьому напрямку.

ВИСНОВКИ ПО ТРЕТЬОМУ РОЗДІЛУ

1. Визначені атрибути матричної моделі руху даних та розроблена їх символіка, яка враховує вхідні та вихідні інформаційні та матеріальні потоки, що дозволило створити базову основу побудови сукупності похідних моделей, які відображають проекції руху даних КС.

2. Запропонований коефіцієнт ефективності руху даних та умови несуперечливості спрацювання переходів у вузлах комп'ютерної системи, що дозволило розробити інформаційну технологію побудови модифікованих моделей руху даних, які є більш досконалими по відношенню до МРД за рахунок врахування ступеня використання ресурсів в активних вузлах КС.

3. Викладена методологія аналізу топології промислового об'єкта управління та формалізація на її основі параметрів руху даних КС, що дозволило довести до інженерного рівня процеси формалізації побудови сукупності моделей руху даних.

4. Розроблена інформаційна технологія побудови сукупності похідних моделей руху даних на основі матричної моделі, включаючи моделі: граф–розгалужене дерево, параметрична часова, структурно–часова, мережевий графік, суміщений часовий граф та блок–схема алгоритму руху даних, що дозволило формалізувати процедури побудови сукупності похідних моделей руху даних, які адекватно відображають структурні, часові, алгоритмічні та діагностичні характеристики КС.

5. Вперше розроблені атрибути та формалізовані технології побудови трьохвимірної та модифікованої двохвимірної моделі руху даних в КС, які дозволили врахувати ступінь використання ресурсів в активних вузлах КС, а також спростити розрахунки ресурсних запасів при послідовно–паралельній реалізації руху даних через активні вузли КС.

6. Розроблена класифікація та проведений аналіз системних характеристик архітектур КС з фізичними та безпровідними лініями зв'язку, для яких вперше побудовані матричні моделі руху даних. Показано, що найбільш перспективними архітектурами розподілених КС є зірково–магістральні архітектури, які

забезпечують високі характеристики трафіків черезрівневих зв’язків та високий рівень розпаралелення руху даних, їх зберігання та транспортування через багатопортову пам’ять колективного доступу.

РОЗДІЛ 4
РОЗРОБКА ТЕОРЕТИЧНИХ ОСНОВ ТА МЕТОДИКИ ПОБУДОВИ СУКУПНОСТІ ЕПЮР СОБІВАРТОСТІ РУХУ ДАНИХ

4.1 Стратегія проектування розподілених комп'ютерних систем

Теорія проектування комп'ютерних систем та мереж базується на ряді фундаментальних підходів та теоретичних положень, які охоплюють теорію мереж Петрі, теорію масового обслуговування, теорію ймовірностей та теорію великих систем [18, 93, 94, 115, 117, 121] . При цьому активно використовується методологія організації та проектування архітектури комп'ютерних систем, а також інформаційна технологія організації руху даних та побудови сукупності моделей руху даних КС.

Значний вклад в розвиток теоретичних положень проектування комп'ютерних систем та мереж внесли відомі зарубіжні вчені, зокрема, Уільямс Столлінс [1] , Джеймс Мартін та Ендрю Таненбаум [5].

Розвитком теорії проектування розподілених комп'ютерних систем в Україні активно займається наукова школа під керівництвом О.В. Палагіна [17, 118, 122].

В той же час надзвичайна складність сучасних комп'ютерних мереж особливо їх просторово–розподілених та проблемно–орієнтованих модифікацій приводить до великого числа інваріантних рішень з слабою випуклістю максимумів оптимізаційних критеріїв, що ускладнює вибір оптимального варіанта на практиці [116, 119].

Розроблена методологія та інформаційна технологія проектування комп'ютерних мереж на основі двохвимірних та трьохвимірних матричних моделей [3, 6] дозволяє диференціювати етапи проектування на два рівні :

1) оптимізаційне моделювання руху даних у вузлах матричної моделі ;

2) стратегічне проектування комп'ютерної мережі на основі евристичних підходів оптимізації параметрів двохвимірних та трьохвимірних моделей руху даних.

Даний підхід дозволяє суттєво звузити число інваріантних рішень, підвищити випуклість функцій оптимізаційних критеріїв, а також суттєво спростити інформаційну технологію проектування комп'ютерних мереж при врахуванні багатьох оптимізаційних факторів.

Кожен вузол ММ ідентифікується номером документа і об'єкта $D_i \bullet O_j$, де i –номер документа, j –номер об'єкта . При проектуванні комп'ютерної мережі на основі ММ на рівні окремих вузлів необхідно класифікувати типи задач технічного, інформаційного, організаційного та іншого забезпечення. Наприклад, наявність: 1) приміщень для розміщення операторів та комп'ютерної техніки, які виконують функції вузла ММ; 2) комп'ютерного мережевого обладнання відповідної потужності (сервери, ПК, сканери, принтери, модеми та інше); 3) телекомунікаційного обладнання (кабельні лінії, репітори, ретранслятори, трансівери та інше); 4) обладнання низових рівнів мереж (інтелектуальні сенсори, АЦП, кодери, декодери та інше); 5) відповідного рівня підготовки операторів, в тому числі оснащених мобільними ПК; 6) стабільного енергоживлення (UPS, автономні джерела живлення).

Запропонована еврістична стратегія проектування вузла комп'ютерної мережі включає три масиви вихідних даних, які отримуються шляхом обстеження готовності підприємства до впровадження окремих задач, які повинна реалізувати проектована мережа [31,32] :

1) готовність підприємства до впровадження та реалізації конкретних задач (рис.4.1).

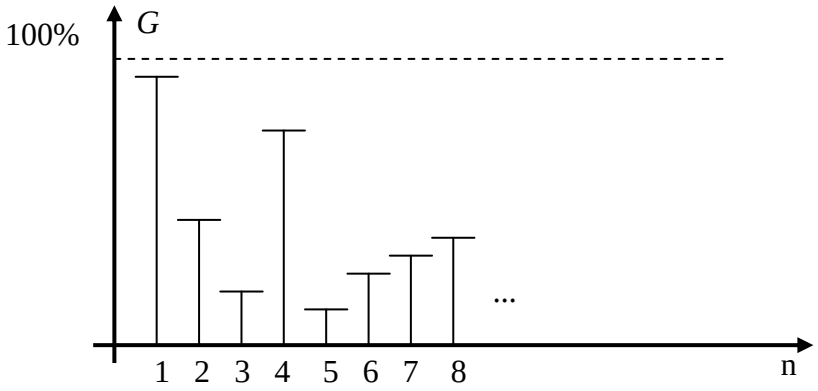


Рис.4.1. Готовність підприємства до реалізації функцій у вузлі матричної моделі

2) вартість постановки та запуску функцій вузла ММ (рис.4.2).

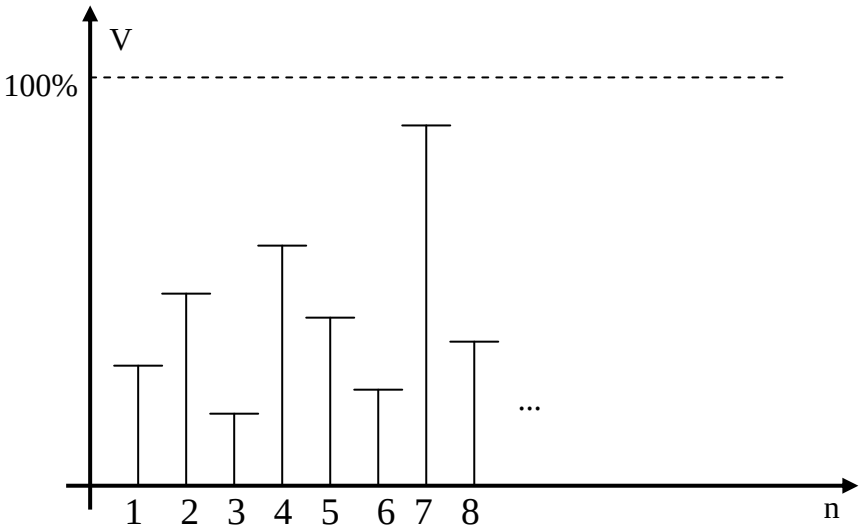


Рис.4.2. Вартість постановки та запуску функцій у вузлі ММ.

3) ефект від впровадження функцій вузла ММ (рис.4.3)

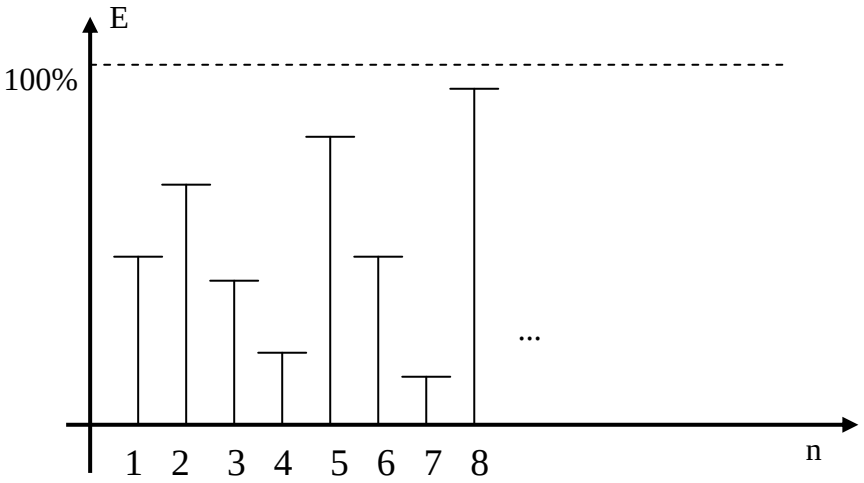


Рис.4.3. Економічний ефект від впровадження функцій вузла ММ.

4) затрати часу на реалізацію функцій вузла ММ (рис.4.4).

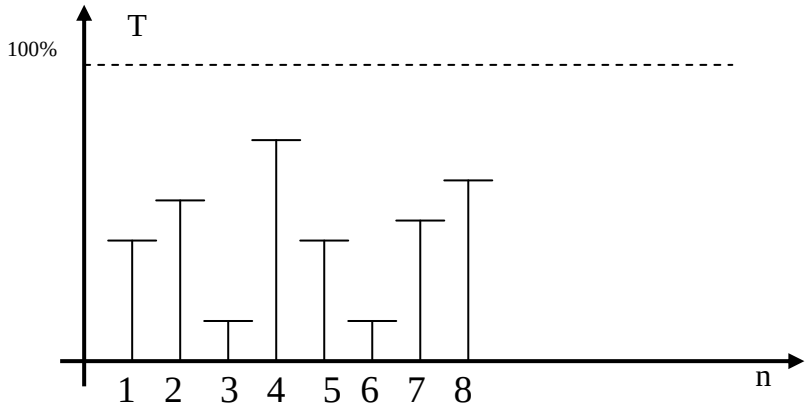


Рис. 4.4. Затрати часу на реалізацію функцій вузла ММ.

На основі вказаних масивів вихідних даних реалізується наступна стратегія проектування економічних характеристик вузлів матричної моделі:

1) визначається номер функції з максимальною готовністю підприємства до його реалізації згідно рис.4.2.;

2) визначається функція з максимальним економічним ефектом для досліджуваного підприємства ;

3) виконується пункт 1 і рекурентно процедури 1, 2 до моменту реалізації всіх функцій комп'ютерної мережі .

Формалізація даної стратегії має наступний вид:

$$\begin{cases} \max e.G_i;\ e.V_i = \text{var};\quad e.P_i = \text{var}; \\ \max e.P_j;\quad e.V_j = \text{var};\ e.G_j = \text{var}; \end{cases}$$

В результаті реалізації описаної стратегії оптимізації проектування вузла ММ в комп'ютерній мережі (див. рис.3.1, 3.2, 3.3) отримаємо послідовність проектування запуску функцій вузла $D_i \bullet O_j$ комп'ютерної мережі у вигляді наступної послідовності виконання задач 1,8,4,5,2,6,7,3

e.G	1		4		2		7	
e.P		8		5		6		3

На основі отриманого алгоритму виконується побудова епюри собівартості затрат і прибутків при реалізації функцій в конкретному вузлі ММ, при цьому необхідно враховувати проектовані, регламентовані часові затрати на реалізацію конкретних функцій у вузлах ММ.

4.2.Теоретичні основи побудови, класифікація та елементи епюр собівартості руху даних

В процесі розробки, впровадження та аналізу експлуатаційних характеристик комп’ютерних мереж виникає проблема економічного обґрунтування варіантів проектів та оцінки оптимізаційних характеристик потоків руху даних в реальних промислових умовах роботи мереж. Одним з перспективних напрямків вирішення даної проблеми є використання моделей та інформаційної технології побудови епюр руху даних [139]. В той же час існуючі методології та інформаційні

технології проектування КМ на основі теорії графів не містять теоретичної бази побудови епюр собівартості руху даних. Слід зауважити, що теорія епюр відома в галузі нарисної геометрії, опору матеріалів, моделювання літаючих апаратів та руху рідин в трубопроводах [137] і може бути розвинена в напрямку вирішення економіко–математичних задач та інформаційних технологій оптимізації руху даних в КМ. Такий розвиток теорії епюр у вказаному напрямку, на наш погляд, проводиться нами вперше. Очевидно, що застосування таких теоретичних основ може бути значно ширше при вирішенні аналогічних задач в галузі комп'ютерних технологій та інформаційних систем.

Поняття епюри в класичному розумінні нарисної геометрії визначає проекцію сліду ліній в ортогональному декартовому просторі (рис. 4.5).

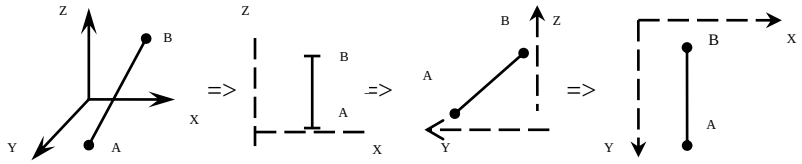


Рис. 4.5. Епюри лінії в декартових координатах

Технологія проектування складних трьохмірних об'єктів на основі теорії епюр дозволяє суттєво спростити САПР та знизити рівень помилок і неточностей при виготовленні деталей машин та механізмів. В опорі матеріалів поняття епюри безпосередньо пов'язане з проектуванням навантажень на конструкції механізмів та машин, а також гідравлічними розрахунками ємностей резервуарів та трубопроводів, які працюють під тиском [137,138].

На рис.4.6 показані приклади епюр в галузі застосування опору матеріалів при проектуванні конструкцій та машин.

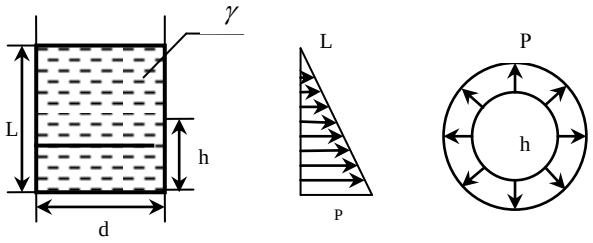


Рис 4.6. Епюри тиску в циліндричному резервуарі

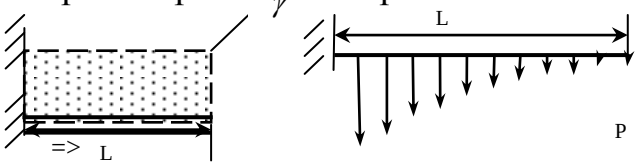
L – висота резервуару;

d – діаметр;

γ – питома вага рідини;

h – рівень, для якого побудовані епюри тиску P=L·γ.

Епюри навантаження на защемлену консоль з рівномірно розподіленою вагою згідно теорії епюр опо матеріалів показано на рис. 4.7.



$$M = -\frac{\gamma \cdot L^2}{2},$$

Рис 4.7. Епюра моменту навантаження на защемлену балку

Приклад використання теоретичних основ епюр в теорії руху рідин та газів показані на рис.4.8.

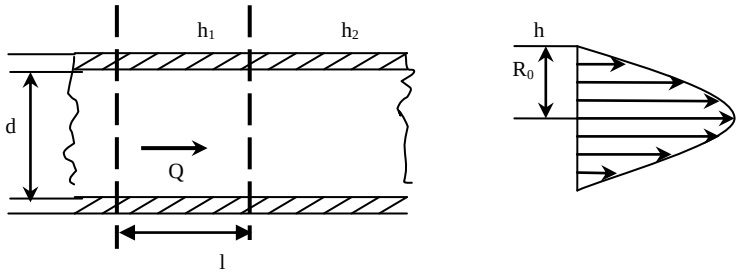


Рис 4.8. Епюра швидкості руху рідин в трубопроводі

Дана епюра аналітично описується рівнянням параболи другої степені

$$Q = F(R) = \frac{P_T(R_0^2 - d^2)}{4\mu \cdot l},$$

де P_T –втрати напору на тертя по довжині;

$l = h_2 - h_1$; μ –в'язкість рідини ;

R_0 – внутрішній діаметр трубопроводу; $0 \le d \le R_0$;

Викладене показує практичну значимість теорії епюр в різних сферах науки і техніки.

Виходячи з аналогії моделей руху потоків рідин та динаміки руху потоків інформаційних даних в комп'ютерних системах, є доцільним використання теорії основ побудови моделей епюр при проектуванні та аналізі руху даних в комп'ютерних мережах, що становить актуальну науково–технічну задачу.

4.3. Систематизація моделей суміщений часовий граф та їх епюри

У роботі [9] викладена теорія проектування спеціалізованих комп'ютерних систем на базі аналогії системних об'єктів енергетики та комп'ютерних мереж, приведена сукупність моделей руху даних, яка включає матричну інформаційну модель СКС, мережевий граф інформаційних потоків та суміщений часовий граф (СЧГ) виконання функціональних операцій в пунктах мережі СКС.

Приклад моделі СЧГ, яка відповідає циклу руху даних від джерела інформації до конкретного приймача показано на рис.4.9.

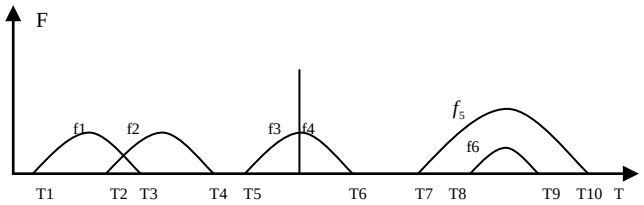


Рис.4.9. Суміщений часовий граф циклу руху даних КМ

Модель СЧГ циклу руху даних використовується для побудови блок–схеми алгоритму обробки даних, який відпрацьовує центральний сервер, реалізуючи алгоритм паралельної обробки даних в реальному часі. При цьому, часові інтервали T_i, T_j, які вказують час виконання функції f_i (формування, обробки, затвердження та зберігання), та їх суміщеність визначають структуру блок–схеми алгоритму цифрової обробки даних, а також можуть бути використані для розрахунку техніко–економічних показників собівартості руху даних в комп'ютерних системах.

На основі даної моделі, як показано в [139], можна побудувати найпростіші ЕРД, виходячи з положення, що затрати на проектування і впровадження засобів КС на рівні пунктів формування, обробки та затвердження даних, та прибутки від функціонування мережі є постійними в часі, тобто описати кусково–постійними функціями Радемахера (рис.4.10).

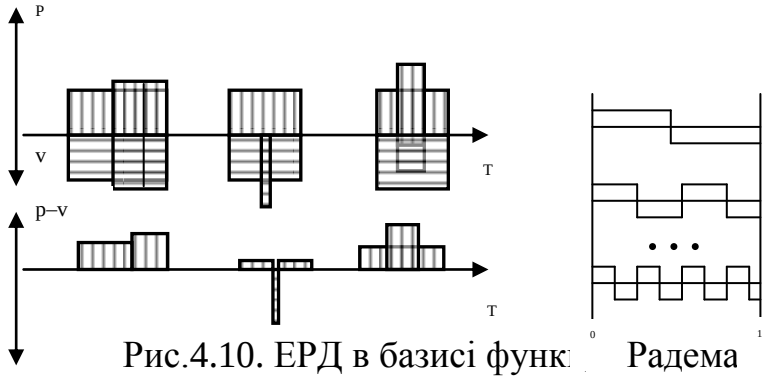


Рис.4.10. ЕРД в базисі функ Радема ра.

P,V – сигнальна ЕРД; $P-V$ – диференціальна ЕРД; P – прибутки від впровадження циклу руху даних КМ; V – витрати на проектування та функціонування КМ.

Теорія економічних вчень та практика використання інвестицій для розробки проектування та вдосконалення засобів комп'ютерної техніки, а також інформаційних мереж, дозволяє стверджувати, що такі випадки нехарактерні і на практиці можуть описуватися функціями інших теоретико–числових базисів. Наприклад, функціями базису Крестенсона (рис.4.11), функціями базису Фур'є (рис.4.12), функціями логарифмічного базису (рис.4.13):

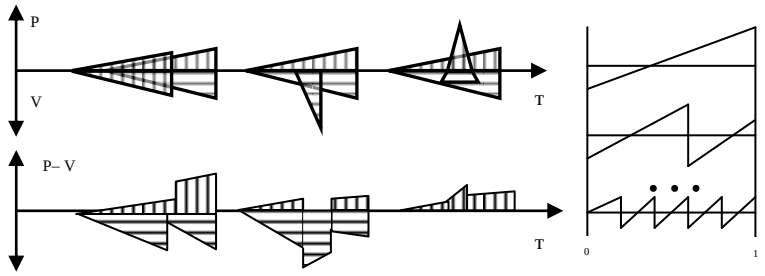


Рис.4.11. ЕРД в базисі Крестенсона

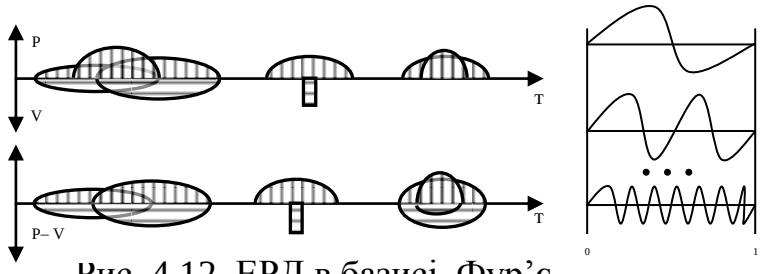


Рис. 4.12. ЕРД в базисі Фур'є

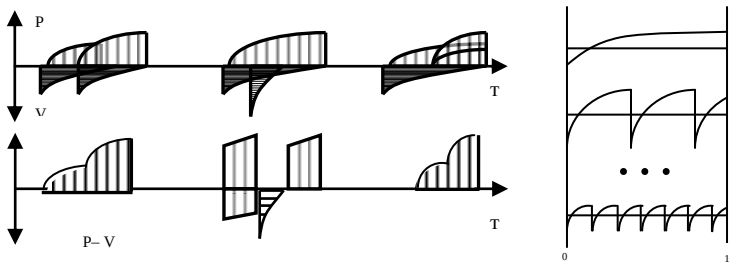


Рис.4.13. ЕРД в базисі логарифмічних функцій

У загальному випадку функції затрат і прибутку можуть також описуватися ортогональними базисами Бесселя, Крейга, Хаара, Уолша та ін. Стратегія вибору конкретних функцій для побудови економічних епюр потребує окремого дослідження економістів, математиків та спеціалістів по проектуванню КМ.

Викладені теоретичні основи інформаційних технологій побудови ЕРД дозволяють суттєво розширити можливості оптимізації проектів КМ та можуть бути добрим інструментом розвитку самої теорії проектування складних інформаційних систем. В конкретних випадках оптимізаційні задачі можуть вирішуватися шляхом перебору варіантів і вибору оптимального по критерію мінімальних затрат і максимальних прибутків, а також виходячи з теорії інтервальних моделей [10].

4.4. Методи побудови ЕРД на основі продукційних моделей подання знань

В даний час стало очевидним, що розвиток теорії проектування комп'ютерних мереж, особливо таких, які характеризуються паралельними інформаційними потоками, потребує відповідного розвитку теорії економічних епюр собівартості для вузлів матричних моделей таких мереж, а також створення інтегральних епюр собівартості витрат і прибутків, при проектуванні комп'ютерних систем представлених тривимірними матричними моделями [139].

Для визначення поняття економічних епюр (ЕЕ) розглянемо елемент двовимірної матричної моделі руху даних в комп'ютерній мережі (рис.4.14.)

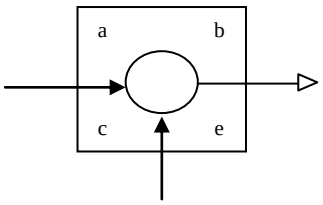


Рис.4.14. Елемент вузла матричної моделі руху даних.

a,b–відповідно час початку та тривалість виконання операції формування, обробки та затвердження даних;

⊙– символ джерела ,◯–пункту обробки, ⊗– затвердження даних ;

c –тип операції у вузлі ММ;

e –собівартість виконання операції, формування, обробки та затвердження даних.

Під поняттям оцінки значення ЕЕ будемо розуміти відношення:

$$K_p = \frac{(P_{ij} - V_{ij})}{T_{ij}} \quad , \qquad (4.1.)$$

де P_{ij} – сумарний дохід від впровадження функцій вузла ММ;

V_{ij} – затрати на постановку та запуск функцій вузла ММ;

T_{ij} – регламентні затрати часу на реалізацію функцій вузла ММ;

$i = 1,2,..n$ –число документів ММ;

$j = 1,2,..m$ –число підрозділів ММ;

Таким чином K_p представляє собою питому оцінку прибутків отриманих у вузлі ММ на інтервалі регламентного часу T_{ij}. На основі формули (4.1) для різних значень ij, які відповідають активним вузлам ММ (рис.4.15)

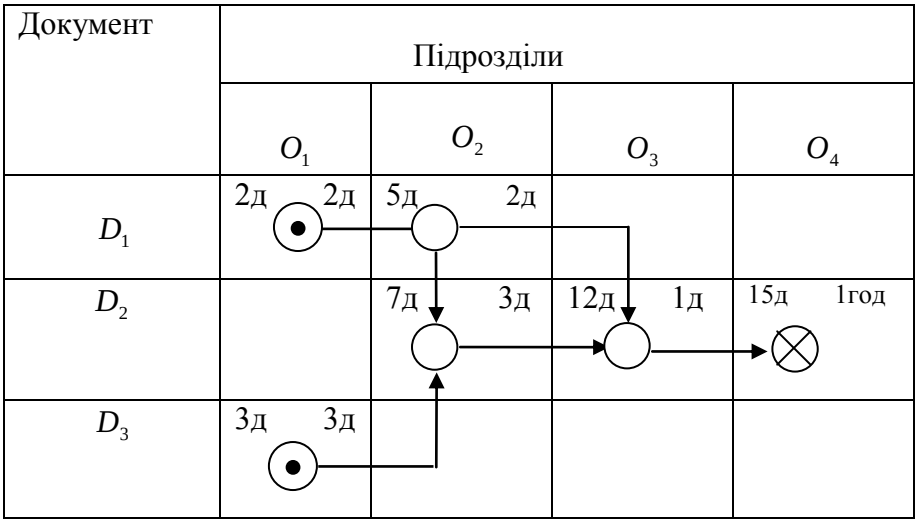


Рис 4.15 Матрична модель руху даних

На основі ММ виконується формальна побудова моделі суміщений часовий граф, яка представлена на рис.4.16.

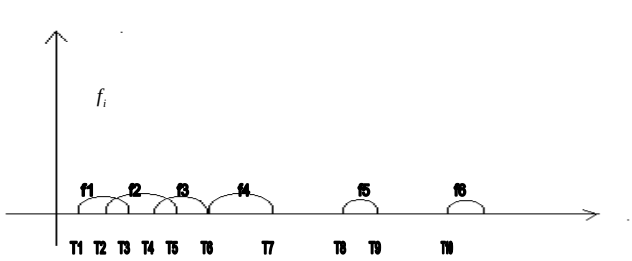


Рис.4.16 Суміщений часовий граф реалізації функцій (f_i) ММ, представленої на рис.4.15.

Модель суміщений часовий граф ілюструє послідовні та паралельні процедури f_i, які виконуються комп'ютерною системою з архітектурою визначеною ММ. При цьому реалізація функцій f_i виконується на основі розгалуженого алгоритму обробки даних, який описується виразом:

$$f_i(T) = \begin{cases} f_1 & T_1 \le T \le T_3; \\ f_2 & T_2 \le T \le T_5; \\ f_3 & T_4 \le T \le T_6; \\ f_4 & T_6 \le T \le T_7; \\ f_5 & T_8 \le T \le T_9; \\ f_6 & T = T_{10}. \end{cases} \qquad (4.2)$$

Враховуючи характеристики економічних затрат і собівартості руху даних в активних вузлах ММ, виконується побудова функції собівартості епюри руху даних у вузлах ММ (рис.4.17.).

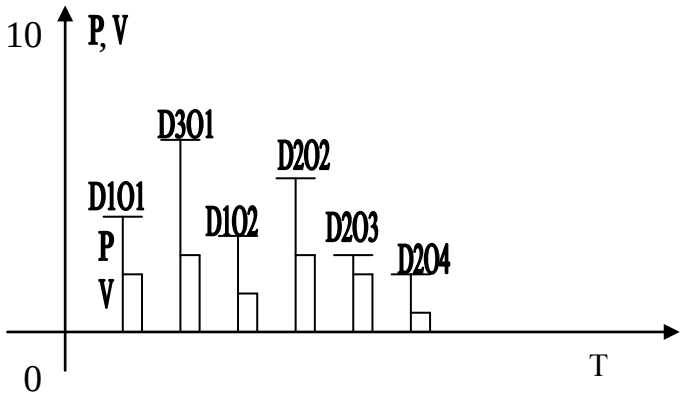


Рис.4.17. Функції собівартості РД в активних вузлах ММ.

Для побудови економічної епюри необхідно врахувати взаємозв'язок між коефіцієнтом K_p та регламентною оцінкою собівартості руху даних (e) у вузлі ММ. Традиційне визначення собівартості в економіці задається рівнянням [21]:

$$C = \frac{\sum_{i=1}^{N} P_i - \sum_{i=1}^{N} V_i}{N}, \qquad (4.3)$$

де N - число активних вузлів ММ.

Крім цього, оцінку (4.3) необхідно нормувати відносно регламентного часу T_{ij}, звідси нормована по часу оцінка собівартості

$$e = \frac{C}{T_{ij}} = K_p. \qquad (4.4)$$

Тобто оцінка e визначається відношенням об'ємів прибутків до об’ємів затрат в умовних одиницях. Наприклад, $e = 2$ може відповідати відношенням 2/1, 20/10, 512/256, причому фізичними об'єктами відношень можуть бути затрати капітальних вкладень, електроенергії, комп'ютерних засобів, об’ємів пам’яті , об’ємів переданих даних та інше. Таким чином, технологія побудови економічної епюри руху даних ММ згідно рис. 4.15 та рис.4.16 буде мати вигляд, приведений на рис.4.18.

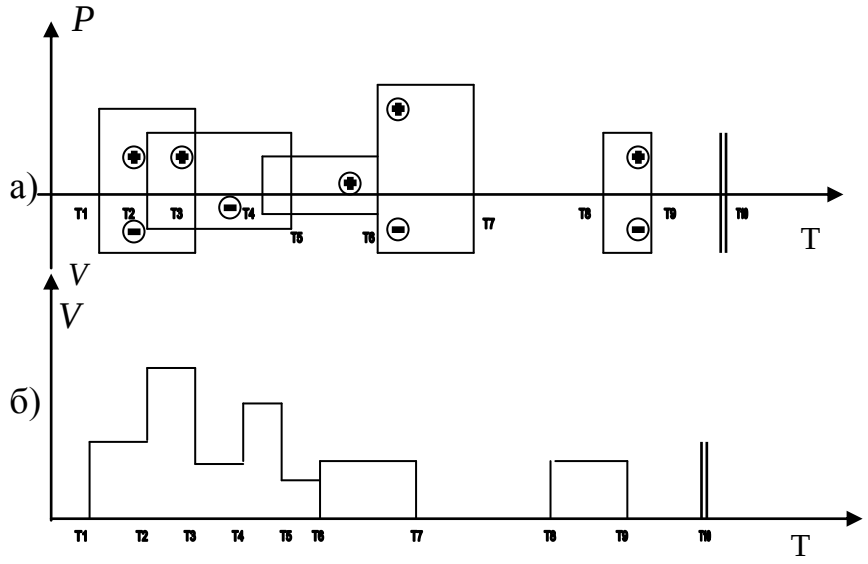


Рис.4.18. Графіки епюр експлуатації комп'ютерної мережі з паралельними інформаційними потоками

На основі формули (4.3) розраховується характеристика нормованих собівартостей для реалізації функцій кожного активного вузла ММ (рис.4.19.)

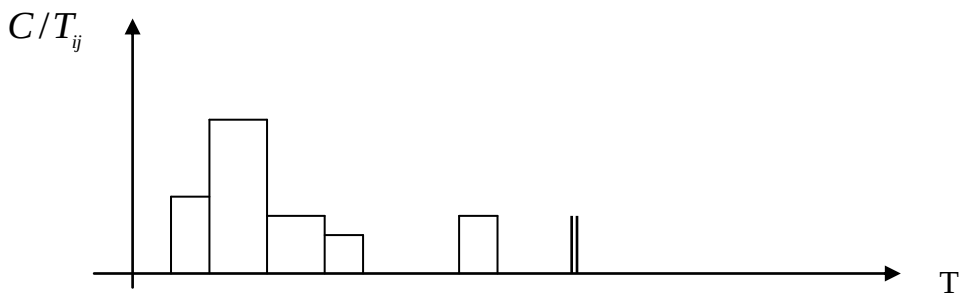


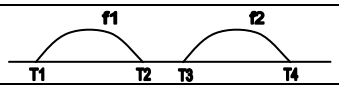
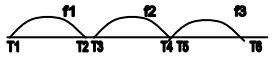
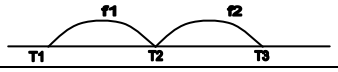
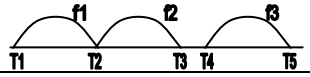
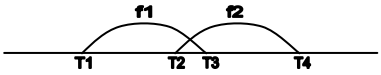
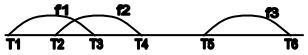
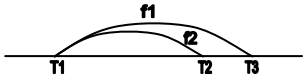
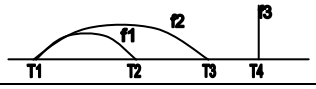
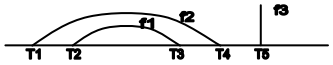
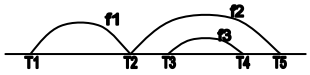
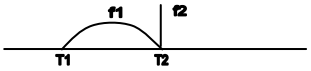
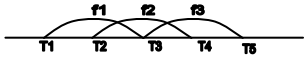
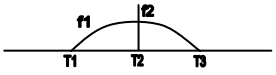
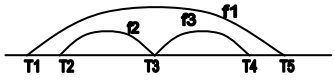
Рис.4.19. Розраховані характеристики нормованих собівартостей реалізації функцій активного вузла ММ.

Таким чином, представлена інформаційна технологія побудови економічних епюр затрат та прибутків при організації руху даних в комп'ютерних системах з паралельними інформаційними потоками базується на використанні системних характеристик вузлів матричної моделі та суміщеного часового графа реалізації функцій формування, обробки та затвердження даних в комп'ютерних мережах. Розроблена технологія дозволяє реалізувати побудову таких епюр формалізованим способом на основі відповідного програмного забезпечення. Ефективність застосування розробленої інформаційної технології значно зростає при збільшенні розмірності ММ, які на практиці для типових підприємств можуть характеризуватися числом документів від 500 до 1000 і числом підрозділів від 20 до 100.

У табл. 4.1 систематизовані типові характеристики моделі суміщеного часового графу на прикладах двох та трьох обчислювальних функцій, які виконуються в активних вузлах ММ.

З табл. 4.1 видно, що розпаралелення обробки даних повинно реалізуватися при пересіченні або накладанні функцій формування, обробки або затвердження даних у вузлах двовимірних ММ, тобто в моделях № 1, 2, 6, 9, 10 не передбачається процедур розпаралелення обчислювальних процесів, що приводить до спрощених форм ЕРД, наприклад, представлених на рис.4.20.

Систематизовані характеристики моделі СЧГ *Таблиця 4.1*

	f1 * f2		f1 * f2 * f3
1		9	
2		10	
3		11	
4		12	
5		13	
6		14	
7		15	
8		16	

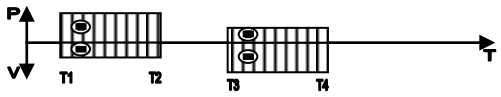
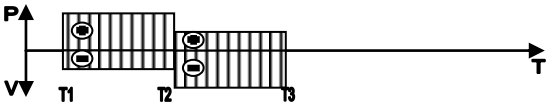
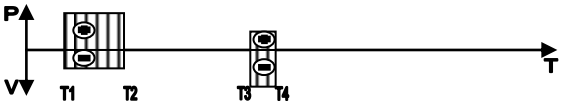
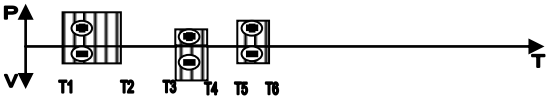
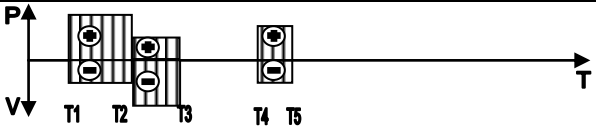
№	
1	
2	
6	
9	
10	

Рис. 4.20. ЕРД операцій в вузлах ММ, які не потребують розпаралелення обробки даних.

З рис.4.20 видно, що в даному випадку число ЕРД строго відповідає числу моделей суміщений часовий граф.

Особливістю можуть бути такі характеристики затрат і прибутків при реалізації функцій вузла ММ, коли понесені фактичні затрати виявилися більшими від отриманих прибутків. При цьому диференціальні епюри собівартості руху даних можуть мати характеристики, представлені на рис.4.21.

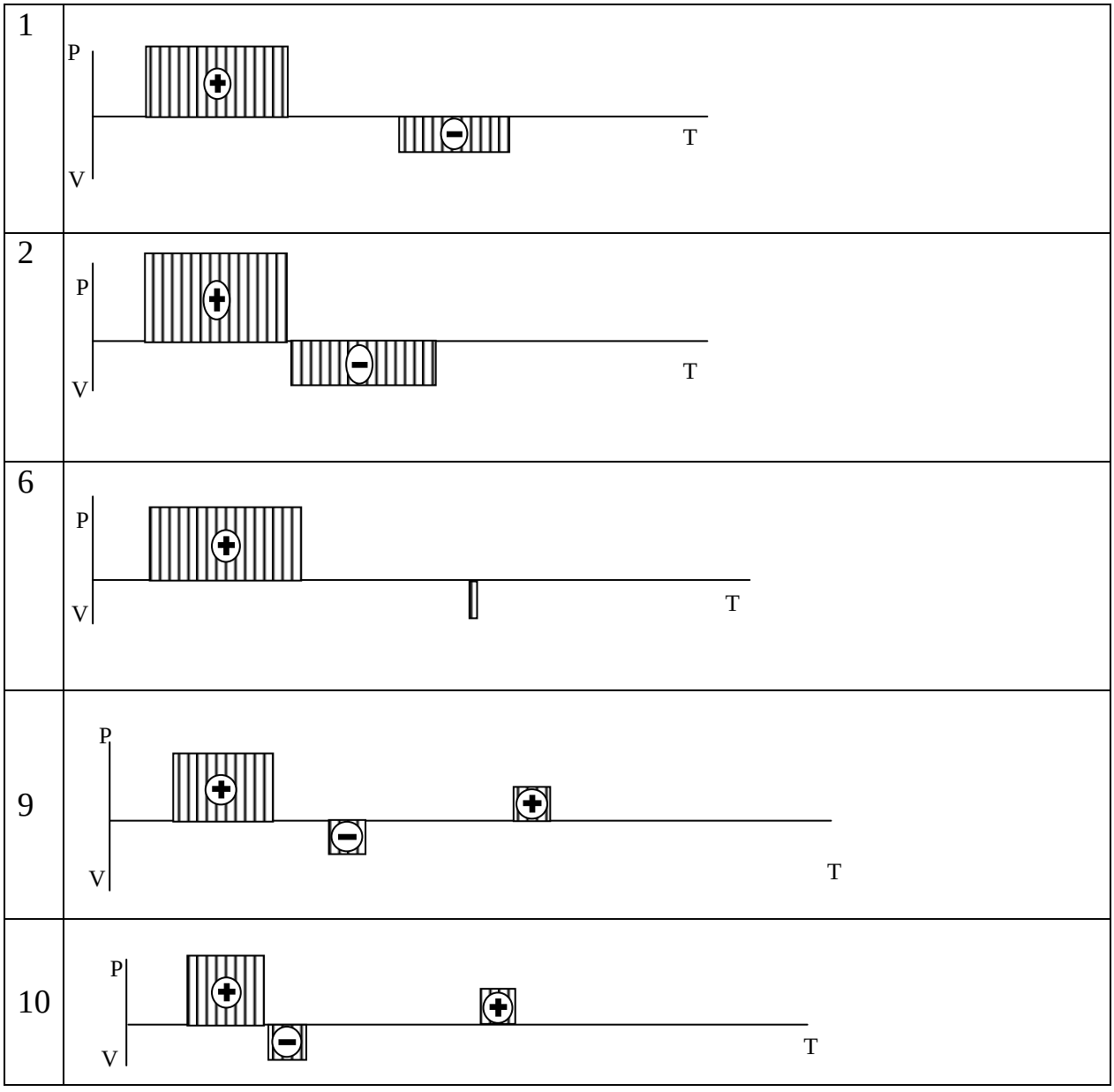


Рис.4.21. Епюри собівартості руху даних зі збитками.

У випадку, коли модель суміщений граф містить пересічення або накладання функцій руху даних у вузлах ММ, в тому числі, з наявністю від’ємних

характеристик, ЕРД моделей, представлених в табл.4.1, будуть мати вигляд, поданий на рис.4.22.

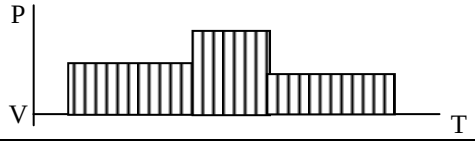
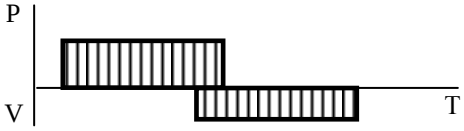
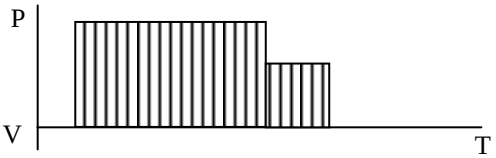
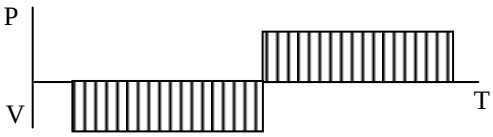
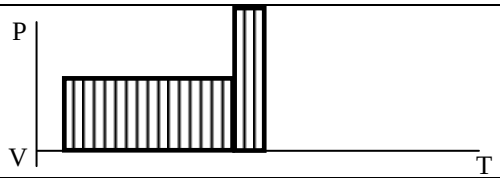
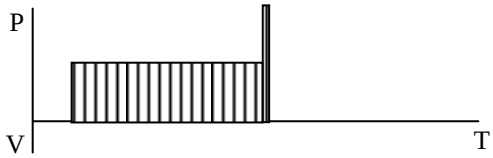
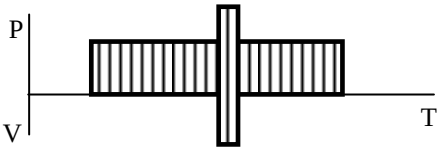
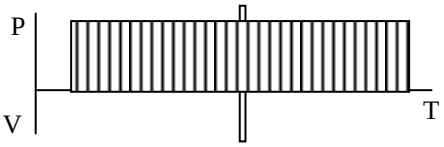
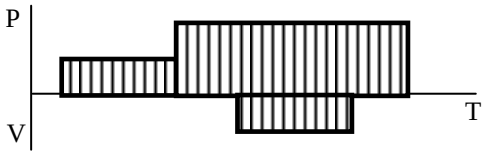
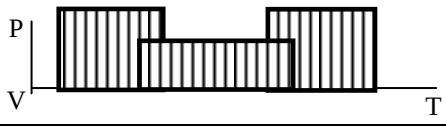
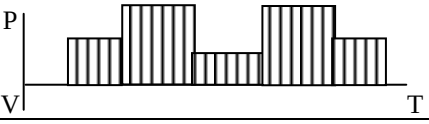
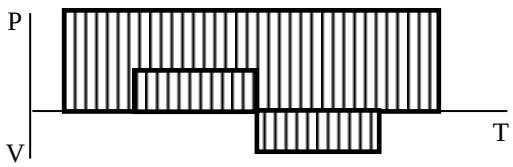
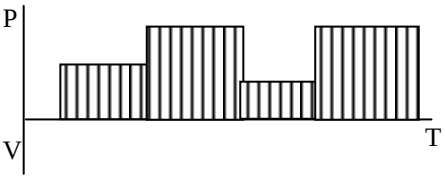
	$f_i * f_j$	e_{ij}
3		
3		
4		
4		
7		
8		
14		
15		
16		

Рис.4.22. ЕРД при розпаралеленні функцій в вузлах ММ.

З рис.4.21 видно, що при миттєвому виконанні окремих операцій у вузлах ММ, ЕРД розпаралеленої обробки інформації однозначно відображаються епюрами собівартості руху даних в заданих вузлах ММ.

Приведені результати систематизації моделей суміщений граф та економічних епюр, які їм відповідають, базувалися на гіпотезі, що економічні затрати та прибутки при реалізації функцій у вузлах ММ описуються функціями нульового порядку. Тобто V_i , P_i і, відповідно, e_{ij} задовільняють систему рівнянь:

$$\begin{cases} V_{ij} = const \\ P_{ij} = const \\ e_{ij} = const \end{cases}, \quad \text{при} \quad \Delta T_{ij} = T_{i+1,j} - T_{ij},$$

де ΔT_{ij} – інтервал часу виконання функції f_{ij} в ММ;

i – число пунктів руху даних ММ;

j – число документів ММ.

Таким чином, результуючі епюри собівартості e_{ij} описуються квазістаціонарними стрибкоподібними функціями нульового порядку.

В результаті результуюча економічна ефективність або собівартість реалізації процедур формування, обробки, архівації та затвердження даних в комп'ютерних мережах з глибоким розпаралеленням інформаційних потоків може бути обчислена на основі інтегральної оцінки.

В загальному випадку функції економічних затрат V_{ij} та прибутків P_{ij} можуть описуватися функціями першого та другого порядку. Відповідно сумарна функція епюри собівартості руху даних буде описуватись стрибкоподібними квазістаціонарними функціями відповідного порядку (рис.4.23, 4.24).

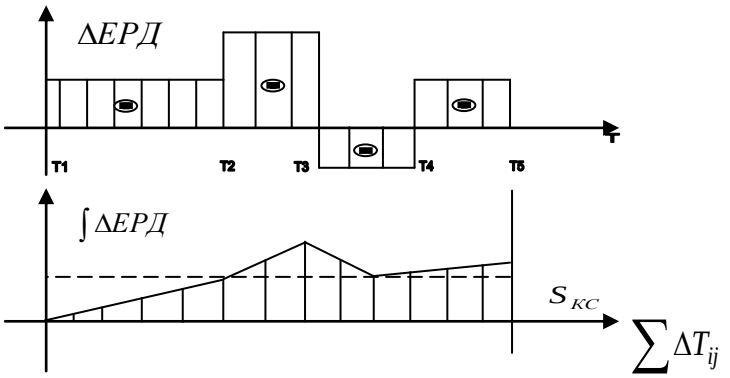


Рис.4.23. Інтегральні оцінки собівартості руху даних в циклі ММ

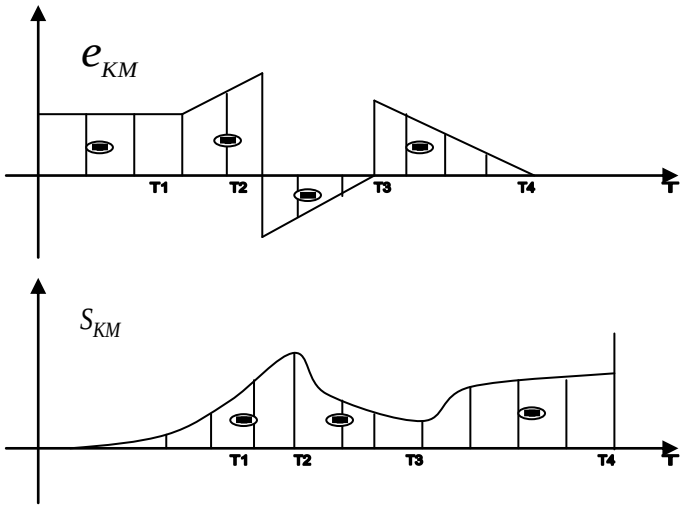


Рис.4.24. Характеристики затрат і прибутків

Для побудови таких епюр доцільно скористатись поняттям циклу руху даних. Дане поняття представляється сумарною послідовністю виконання функцій у вузлах ММ, починаючи від конкретного джерела інформації і закінчуючи пунктом затвердження та архівації даних.

Аналітично сумарну собівартість S_{KL} руху даних на інтервалі одного циклу руху даних та питому собівартість S_{KC} можна розрахувати згідно формули:

$$S_{KL} = \sum_{i=1}^{N} \sum_{j=1}^{\Delta T_{ij}} (P_{ij} - V_{ij}) \ ;$$

$$S_{KC} = \frac{S_{KL}}{\sum_{i=1}^{N} \Delta T_{ij}} \ .$$

В загальному випадку оцінка S_{KL} може бути розрахована на основі аналітичного виразу [37]:

$$S_{KL} = \sum_{i=1}^{M} \sum_{j=1}^{\Delta T_{ij}} \left[F_1(P_{ij}) - F_2(V_{ij}) \right] ,$$

де $F_1(P_{ij})$ і $F_2(V_{ij})$ – відповідні аналітичні вирази функцій, що описують часові характеристики затрат та отриманих прибутків при реалізації функцій в активних вузлах ММ.

Для ММ, яка представлена на рис.3.13 виконується побудова ЕРД наступного типу:

– цикли руху даних (рис.4.25) ґрунтуються на описі всіх трафіків руху даних від джерел до пунктів затвердження та зберігання даних.

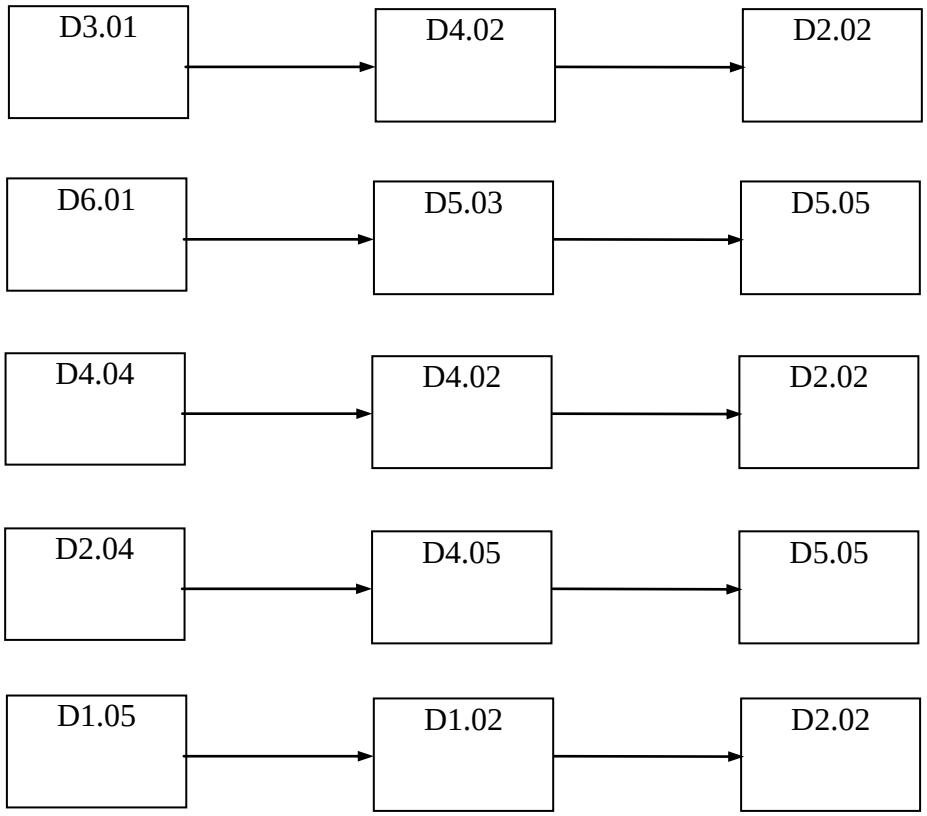


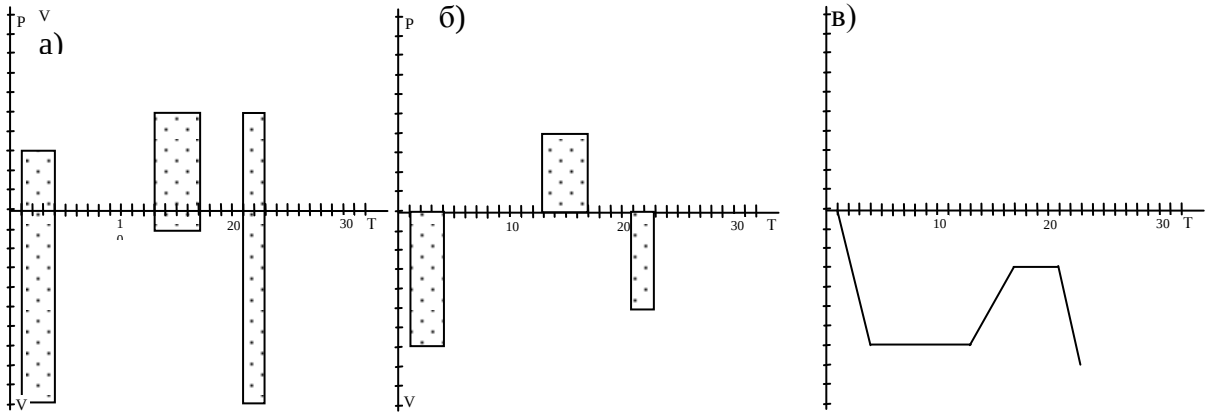
Рис. 4.25. Цикли руху даних

– епюра собівартості циклу ЕРД будується для кожного циклу руху даних на основі часових параметрів (a,b) та параметрів собівартості $(p-v)$ матричної моделі;

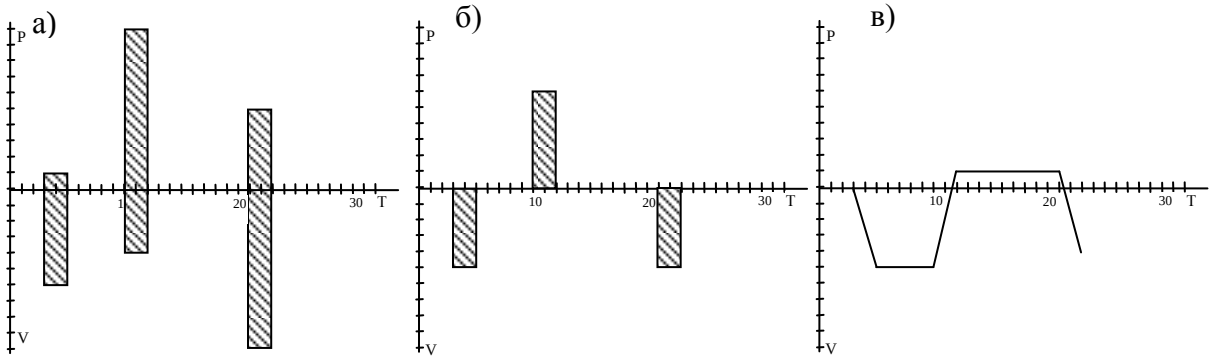
–диференціальна епюра $\Delta ЕРД$ руху даних представляє собою різницевий граф прибутків та витрат для кожного циклу руху даних;

–інтегральна епюра $(\int \Delta ЕРД)$ циклу руху даних будується на основі моделі $\Delta ЕРД$ шляхом інтегрування характеристик витрат та прибутків в кожному пункті руху даних матричної моделі (рис.4.26).

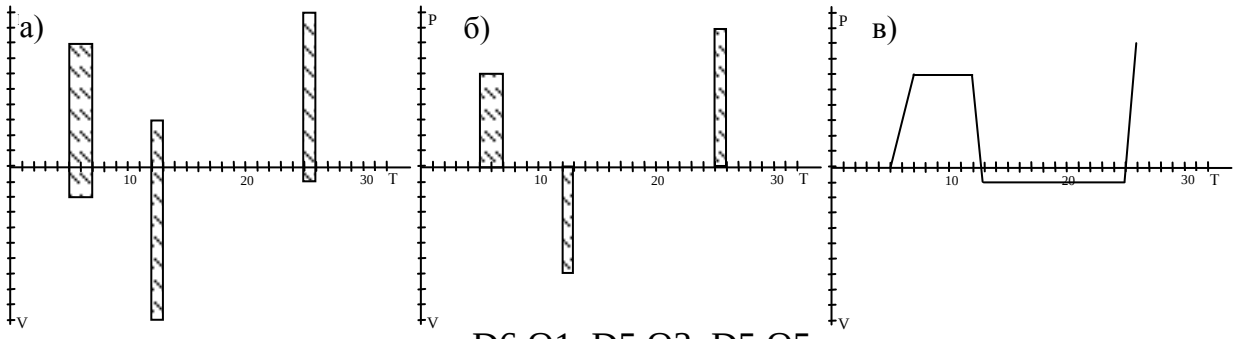
цикл D3.O1–D4.O2–D2.O2



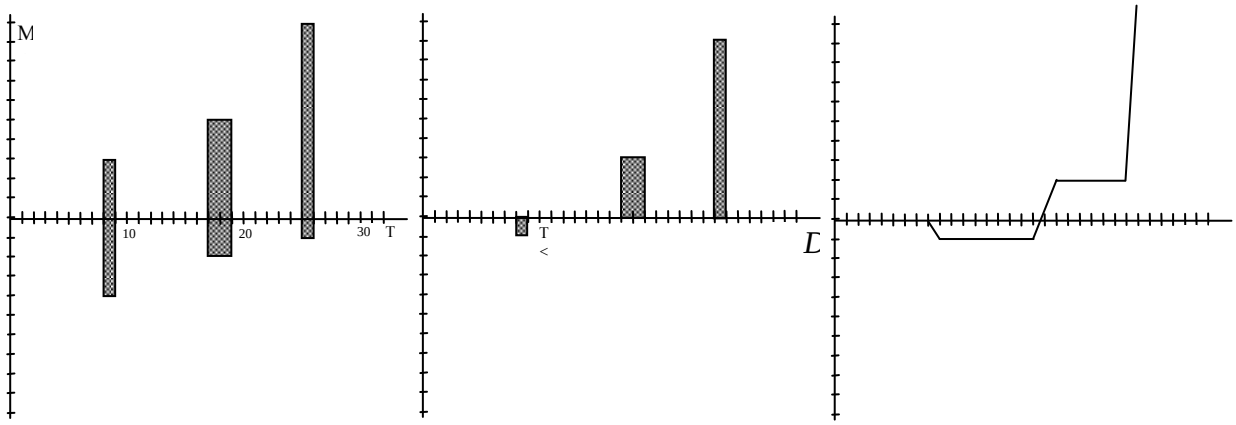
цикл D1.O5–D1.O2–D2.O2



цикл D2.O4–D4.O5–D5.O5



цикл D6.O1–D5.O3–D5.O5



цикл D4.O4–D4.O2–D2.O2

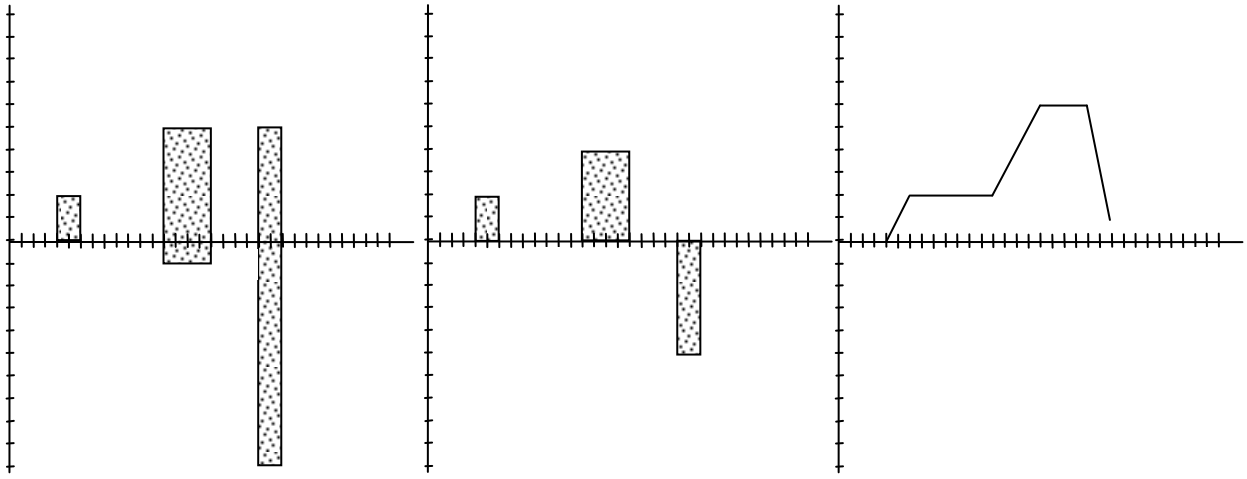


Рис.4.26. Епюри собівартості (а), диференціальні епюри (б), інтегральні епюри (в)

– сумарна інтегральна епюра ($\sum \Delta ЕРД$) циклу руху даних будується шляхом сумування інтегральних епюр (рис 4.27);

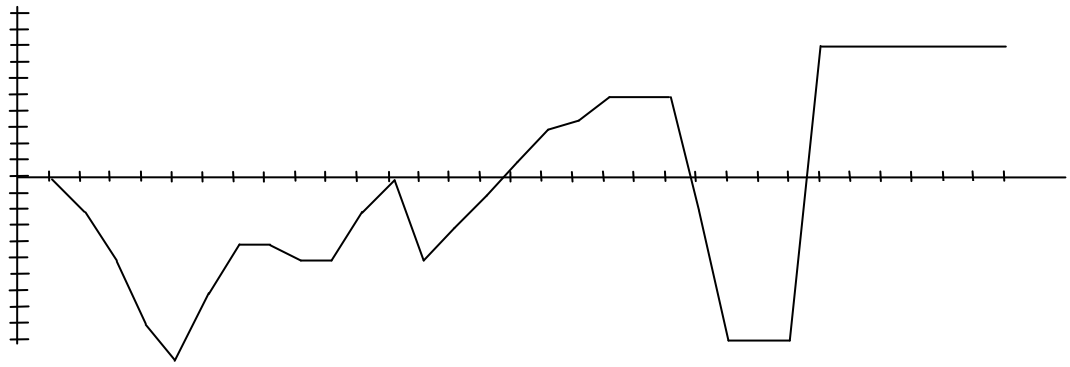


Рис. 4.27. Сумарна інтегральна епюра собівартості циклів руху даних.

На основі вищевказаних епюр будуємо глобальну характеристику ефективності КС (рис.4.28).

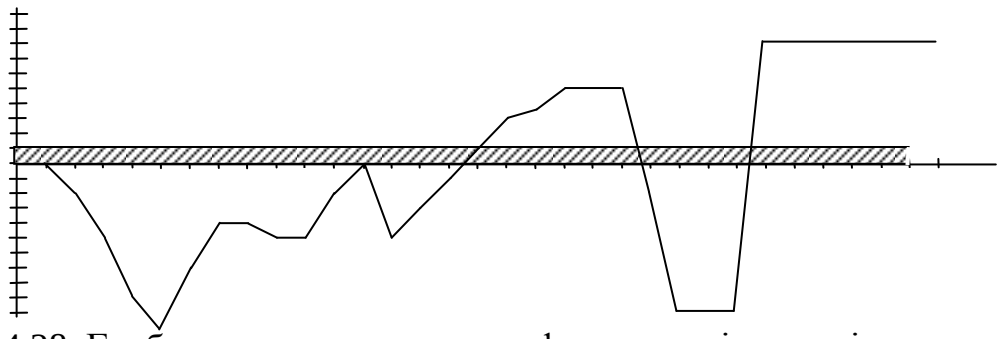


Рис. 4.28. Глобальна характеристика ефективності розподілен

системи.

Глобальна характеристика ефективності проектованої комп

розраховується на основі обчислення вибіркового математи

характеристики сумарної інтегральної економічної епюри згідно формули [37]:

$$G_{KC} = \frac{\sum_{i=1}^{n} \int_{0}^{T} \Delta EPД_i(T)dT}{N},$$

де n – число часових інтервалів ковзної вибірки,

N – загальне число часових інтервалів.

Викладені теоретичні основи, методи та інформаційні технології побудови епюр собівартості руху даних в РКС є ефективним інструментом аналізу та діагностики діючих систем, а також засобом проектування складних РКС з високим рівнем розпаралелення системних операцій та основою реалізації стратегії вибору оптимізованих варіантів їх архітектури, трафіків інформаційних потоків, функціональних можливостей та ступеня використання ресурсів в активних вузлах ММ.

ВИСНОВКИ ПО ЧЕТВЕРТОМУ РОЗДІЛУ

1. Розроблена методологія та стратегія проектування вузла КМ на основі 4 масивів вихідних даних: готовність підприємства до реалізації, вартість постановки та запуску, ефект від впровадження та затрати часу на запуск, впровадження та реалізацію функцій вузла ММ.

2. Вперше розроблена класифікація та викладені теоретичні основи побудови епюр руху даних в КС, яка дозволила суттєво спростити вирішення задач розрахунку економічних характеристик проектованої або діагностованої КС.

3. Розроблені інформаційні технології побудови ЕРД на основі суміщеного часового графу циклу руху даних з використанням функцій теоретико–числових базисів Радемахера, Крестенсона та логарифмічних функцій, які можуть бути адаптовані до собівартісних стратегій вкладення коштів у розробку та модифікацію КС з можливістю оцінки глобального ефекту їх функціонування в реальному часі.

4. Систематизовані моделі суміщений часовий граф та викладена методологія побудови епюр собівартості руху даних при виконанні функцій у вузлах ММ, які передбачають відсутність або наявність процедур розпаралелення виконання функцій КС, а також епюр собівартості руху даних зі збитками, що дозволило розробити технологію діагностики КС на основі диференціальних та інтегральних епюр собівартості руху даних.

5. Отримані аналітичні вирази оцінок сумарної, питомої та глобальної оцінок ефективності руху даних в розподілених КС та розроблена інформаційна технологія їх побудови на основі циклу руху даних, що дозволило використати методи оптимізації прийняття рішень щодо удосконалення системи та порівняння альтернативних варіантів системи.

РОЗДІЛ 5
РЕАЛІЗАЦІЯ В ПРОМИСЛОВОСТІ РОЗРОБЛЕНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ОРГАНІЗАЦІЇ РУХУ ДАНИХ В РКС

5.1. Розробка структури та реалізація програмного забезпечення формалізованої побудови сукупності моделей руху даних в РКС на основі матричної моделі

Станом на сьогоднішній день робота в Delphi являється одним із найбільш продуктивних методів створення прикладних програм для Windows [149, 150, 151]. Загальна продуктивність будь-яких інструментів створення програмного забезпечення визначається наступними важливими аспектами:

- якістю візуального середовища розробки;
- швидкістю роботи компілятора та швидкодією відкомпільованих програм;
- потужністю мови програмування, її складністю;
- гнучкістю та масштабованістю архітектури баз даних, яка використовується;
- наявністю шаблонів проектування та використання, які підтримуються середовищем розробки.

Безумовно існує ще немало важливих факторів – наприклад, питання впровадження, наявна документація, підтримка третіх фірм і т.д. Тим не менше, наведених вище факторів повністю достатньо для того, щоб пояснити, чому саме Delphi вибрано в якості інструменту реалізації програмного забезпечення [153, 154].

Візуальне середовище розробки в загальному випадку складається із трьох взаємопов’язаних компонентів: редактора, відладчика та конструктора форм. В будь-якому із сучасних інструментів розробки прикладних програм (Rapid application Development - RAD) ці три компоненти повинні гармонійно взаємодіяти один з одним в процесі створення програми. При роботі в конструкторі форм Delphi неявно генерує програмний код всіх тих візуальних компонентів, які розміщаються та обробляються у формах. У вікні редактора в

текст автоматично створеної програми можна внести необхідні доповнення та коректури, які будуть визначати специфічну поведінку даної програми. Тут же, у вікні редактора, програми можуть відлагоджуватись з допомогою внесення точок зупинки програми (breakpoints), визначення контрольованих змінних і т.д.

Швидкий компілятор дає можливість розробляти програмне забезпечення поетапно, оскільки допускає багатократне внесення у початкову програму незначних змін, з наступним перекомпілюванням та тестуванням. В результаті отримуємо досить ефективний цикл розробки. Більш повільні компілятори заставляють розробника одночасно вносити досить великі обсяги змін, комбінуючи кілька окремих допрацювань в одному циклі компілювання та відлагодження. Це, безумовно, знижує ефективність окремих циклів розробки. Переваги, досягнуті за рахунок підвищеної ефективності роботи відкомпільованих програм очевидні. В будь-якому випадку, чим швидше працює програма і чим менший її об’єктний код, тим краще [155, 157].

Потужність і складність мови – це є відносні поняття. Найбільш оптимальній варіант - коли досягається баланс між складністю та потужністю.

Одна з таких мов – Object Pascal, яка й використовується в середовищі Delphi. Object Pascal не є такою потужною мовою, як наприклад, асемблер, проте володіє достатніми можливостями для розробки програм практично любого рівня складності. В той же час мова не є настільки складною, як наприклад C++, проте надає можливість використання об’єктно-орієнтованого проектування.

У фірмі Borland (Inprise) відсутня власна лінія продуктів управління базами даних. Зате до складу Delphi входить досить потужний інструментарій, який забезпечує надзвичайно гнучку архітектуру підтримки баз даних. Механізм BDE (Borland Database Engine) успішно працює і забезпечує хорошу продуктивність для взаємодії з широким діапазоном локальних, розподілених баз даних та баз даних, сконфігурованих на базі драйверів ODBC. Крім того, для доступу до баз даних можна використати спеціалізовані ADO-компоненти. Якщо ж вони не задовольняють потреб проекту, то завжди можна використати

компоненти сторонніх розробників, які підтримують практично всі існуючі на сьогодні СУБД [156, 158].

В основу розробки ПЗ поставлена технологія об'єктно–орієнтованого програмування засобами Inprise Delphi [85–88]. Базова структура ПЗ, яка реалізує формалізовані процедури сукупності похідних моделей, описаних в розділі 3 показана на рис. 5.1.

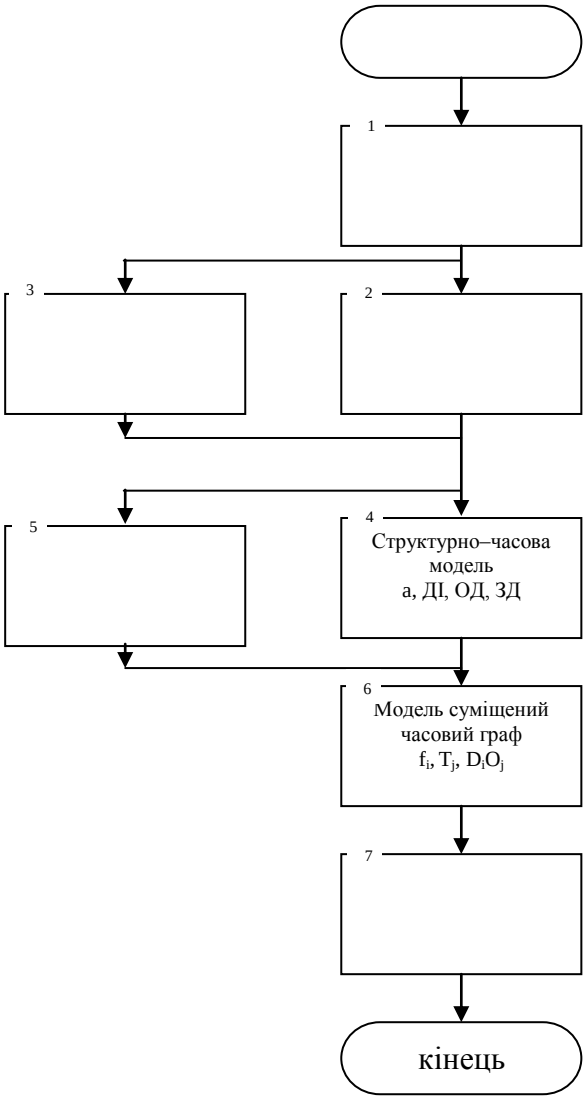


Рис. 5.1. Модулі ПЗ

Модулі ПЗ (див. рис. 5.1) реалізують наступні функції:

Блок 1 – інтерфейс користувача для вводу початкових даних

Блоки 2, 3 – реалізують побудову структурних моделей граф–розгалужене дерево та структурно–часову, які враховують структуру руху даних згідно МРД та її атрибути типу джерело, пункт обробки і пункт приймання та архівації даних.

Блок 4 – реалізує побудову параметрично–часової моделі на основі атрибутів a і b МРД, а також причинності в реальному масштабі часу згідно атрибутів ДІ, ОД, ЗД, МРД.

Блок 5 – реалізує побудову моделі мережевий графік на основі параметрів МРД: a – початок виконання операції, O_j – номер об'єкта, в якому розміщений активний вузол, $O_i - O_j$ – ребра мережевих інформаційних зв'язків.

Блок 6 – реалізує побудову моделі суміщений часовий граф на основі атрибутів a, b, f_i ,T, МРД.

Блок 7 – реалізує побудову діагностичної для центрального сервера КС моделі „Блок–схема алгоритму руху даних" на основі моделі суміщений часовий граф та атрибутів a, b, T, МРД.

5.2. Програмні засоби побудови епюр собівартості руху даних та оцінки глобальної ефективності комп'ютерних систем

Програмне забезпечення виконане на основі програмного середовища Delphi та інформаційної технології побудови сукупності моделей собівартості руху даних, описаних в розділі 4 [131, 132]. На рис.5.2 представлена блок–схема даного ПЗ.

Програма реалізована з використанням об'єктно-орієнтованого підходу. Структурно вона складається з головного файлу проекту («model.dpr» – ініціалізація та запуск екранних форм) та файлів модулів. У файлах модулів реалізовано механізм роботи з базою даних BIMA MySQL, в якій зберігаються дані по проекту. Структура бази даних зображена на рис.5.3. Базу даних реалізовано в середовищі MySQL. На сьогоднішній день це одна з найбільш потужних СУБД з відкритим кодом, вона є безплатною, надійною, має широкий набір функцій доступу до даних та підтримується багатьма розробниками компонентів доступу до даних.

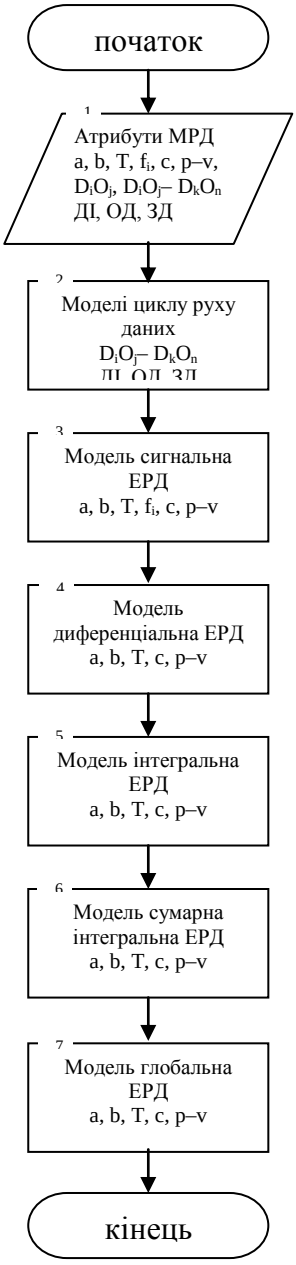


Рис.5.2. Блок –схема програмного забезпечення побудови циклів руху даних

Працює програма згідно наступного алгоритму:

1. Кожна нова задача – це проект, який характеризується такими атрибутами: номер проекту, назва проекту, кількість задіяних в проекті документів, кількість підрозділів, одиниця оцінки часу проекту, одиниця оцінки вартості проекту та коментар (даний атрибут не є обов’язковим) – див. поля таблиці PROJECTS на рис. 5.3. Для кожного нового проекту вводяться ці дані. Крім того, в програмі реалізовано роботу з довідниками одиниць оцінки часу (таблиця TIME_OUM) та одиниць оцінки вартості (таблиця PRICE_UOM). Таблиці DOCS та DEPTS містять, відповідно, описові дані по назвах документів та назвах підрозділів. Всі дані (номери документів та підрозділів) прив’язані до

номера проекту (ключове поле). Тому спочатку потрібно ввести атрибути проекту, а потім – дані по проекту:

1.1. Документи по даному проекту (назва документу).

1.2. Структурні підрозділи по даному проекту (назва підрозділу).

1.3. Інформація по кожному елементу матричної моделі (у таблицях DI_INFO, OD_INFO, DZ_INFO та TRANSFER_INFO містяться дані по кожному елементу матричної моделі (джерела даних, обробка даних, затвердження даних та цикли руху між елементами)). Для ДІ, ОД та ДЗ заносяться такі дані: номер документу; номер підрозділу; час початку; час формування (обробки, затвердження); тип операції; величина прибутку; величина витрат.

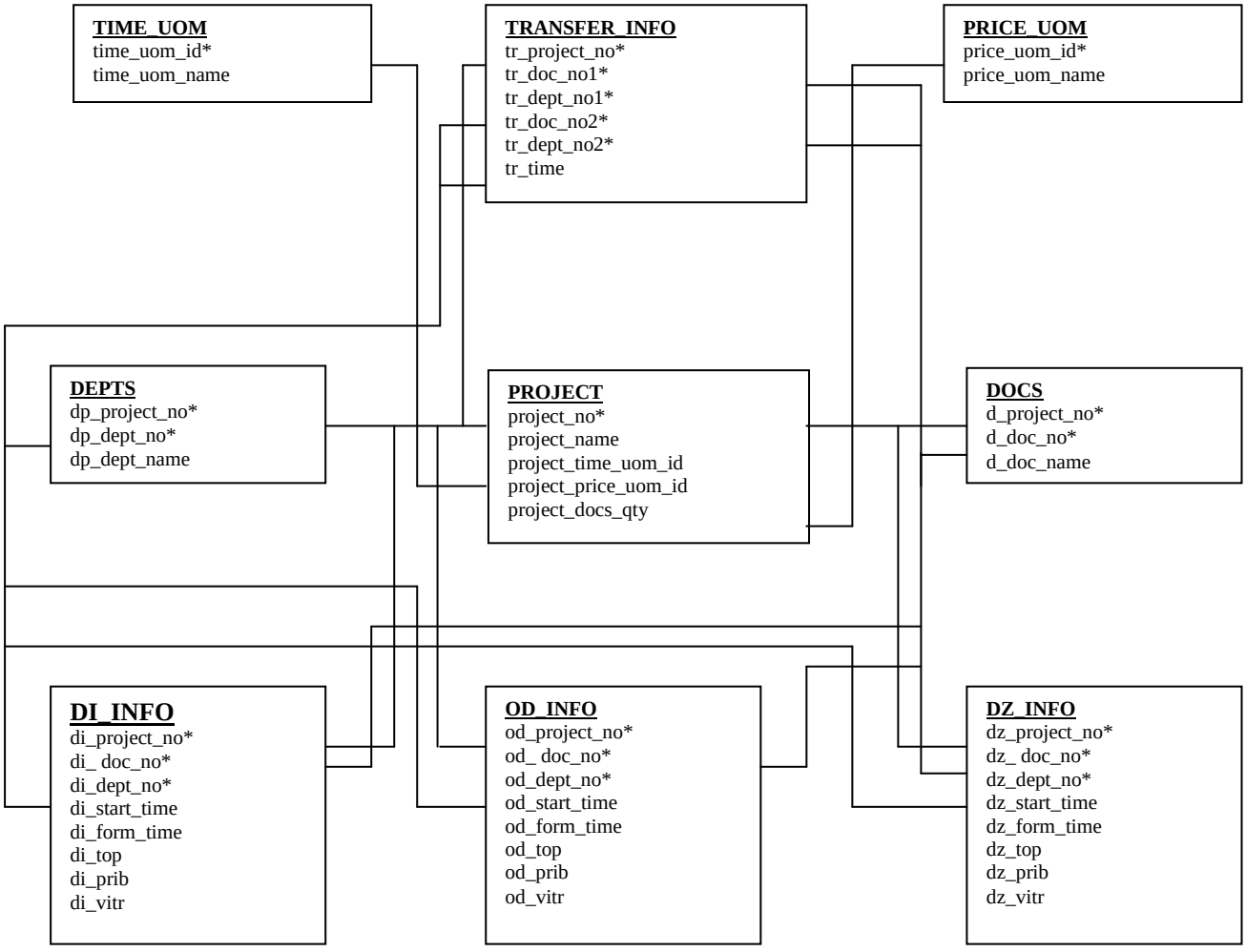


Рис. 5.3. Структура бази даних

Для елементів циклів руху даних вводяться:

- вихідний документ;

- вихідний підрозділ;

- кінцевий документ;

- кінцевий підрозділ;

- час руху.

2. Якщо для побудови моделі потрібно сформувати цикли руху даних, то формуємо їх (якщо потрібно). Для цього в програмі передбачено відповідний елемент інтерфейсу (кнопка «Формувати цикли руху даних»), який активується, коли вибрано модель, яка потребує наявність циклів.

3. Проводиться вибір моделі. Якщо для якоїсь моделі не заведено всіх необхідних даних, то появиться відповідне повідомлення і модель не буде побудовано.

4. Формування самої моделі у вигляді графічного малюнка в форматі BMP. Є можливість зберегти модель у файлі, якщо передбачається подальша її обробка та використання.

Лістінг програми приведено в додатку А.

Для побудови моделей:

- матрична,

- граф-розгалужене дерево,

- параметрично часова,

- структурно-часова,

- епюра собівартості,

- диференційна епюра,

- інтегральна епюра,

- сумарна епюра собівартості,

- сумарна диференційна епюра,

- сумарна інтегральна епюра

викликаються відповідно процедури:

DrawMatrixModel, DrawGraphTree, DrawParamTime, DrawStructTime, DrawEpureSb, DrawEpureDif, DrawEpureInt, DrawTotalEpureSb, DrawTotalEpureDif, DrawTotalEpureInt.

5.3. Програмні засоби побудови інтерфейсу користувача

Інтерфейс користувача реалізовано з допомогою бібліотеки компонентів VCL середовища Delphi. На рис.5.4 показано загальний вигляд інтерфейсу користувача.

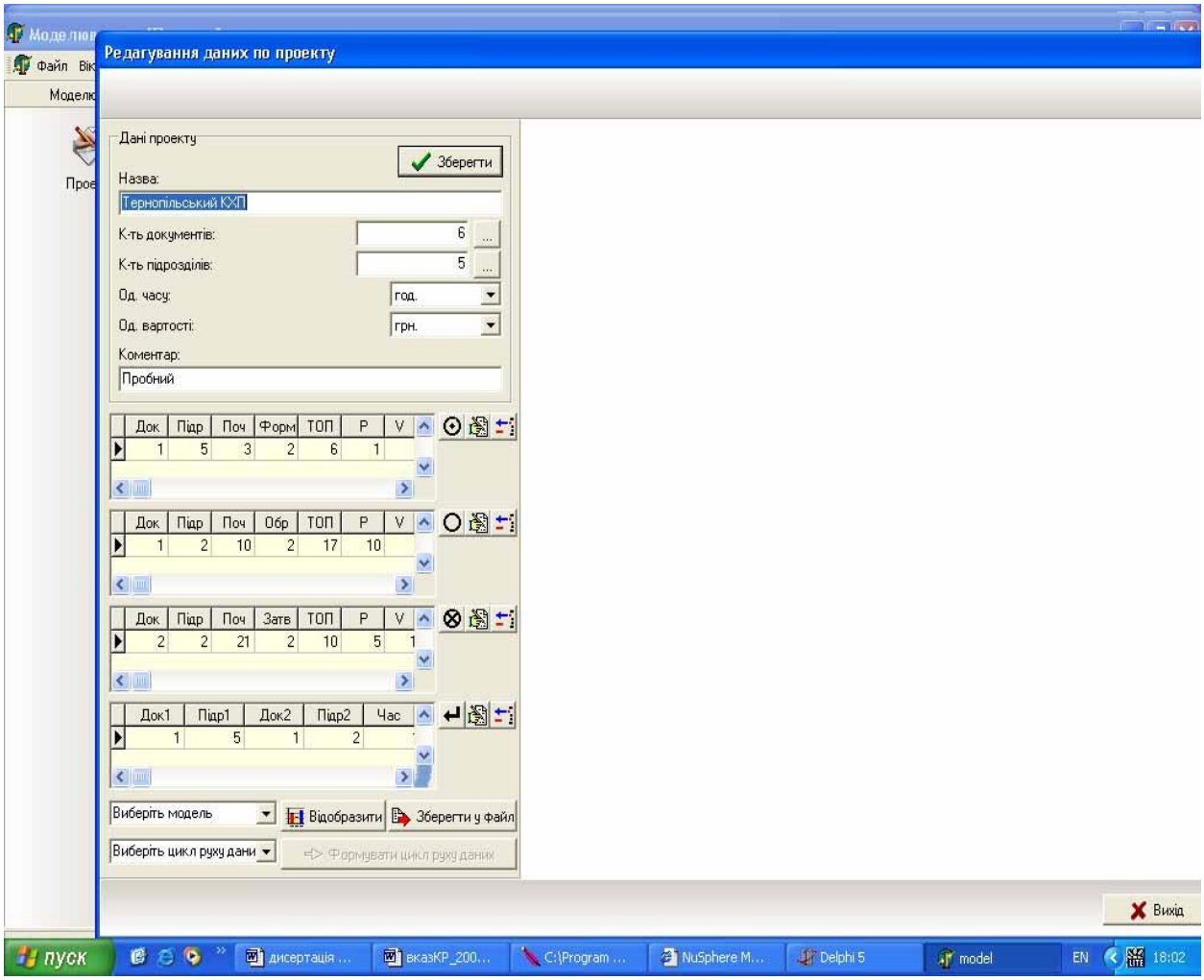


Рис. 5.4. Загальний вигляд інтерфейсу користувача.

На рис.5.5. показано програмну реалізацію формування матричної моделі

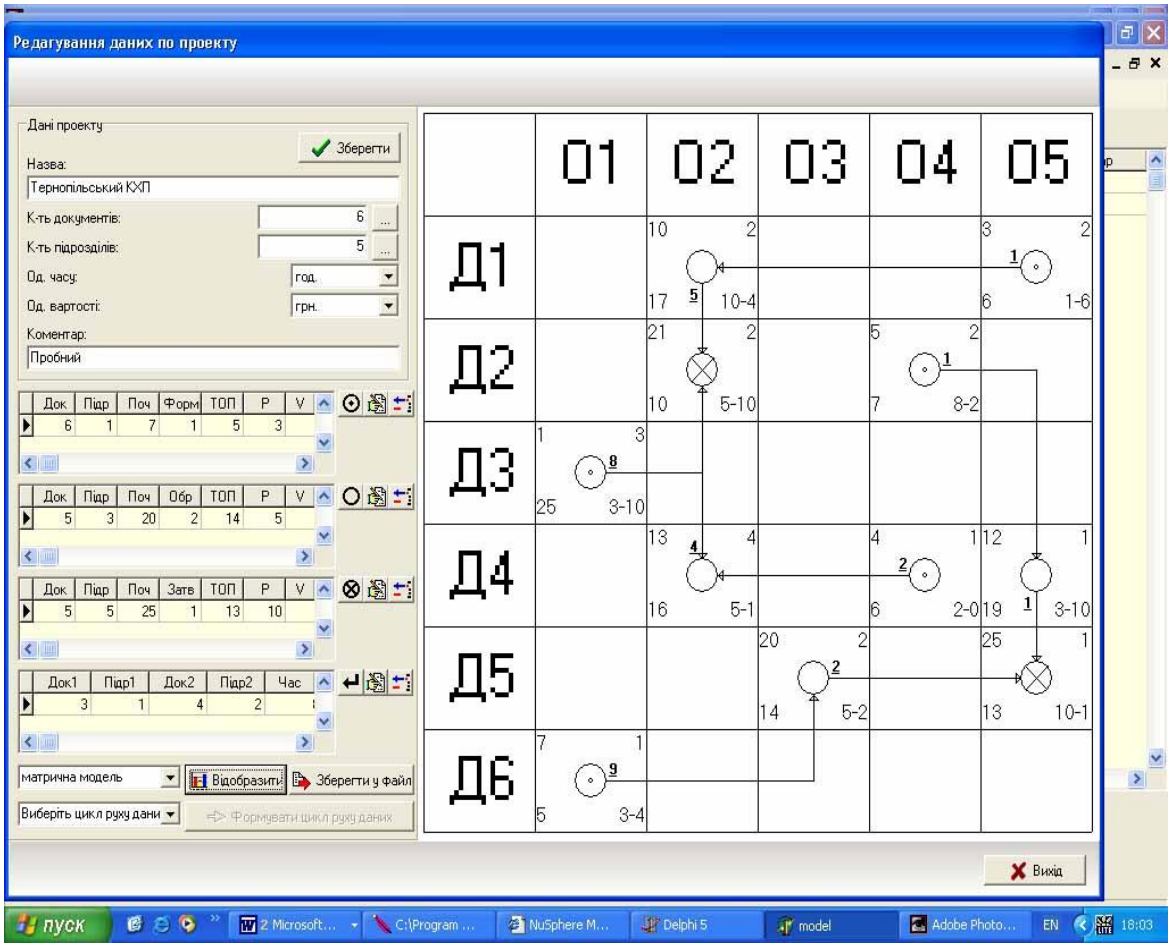
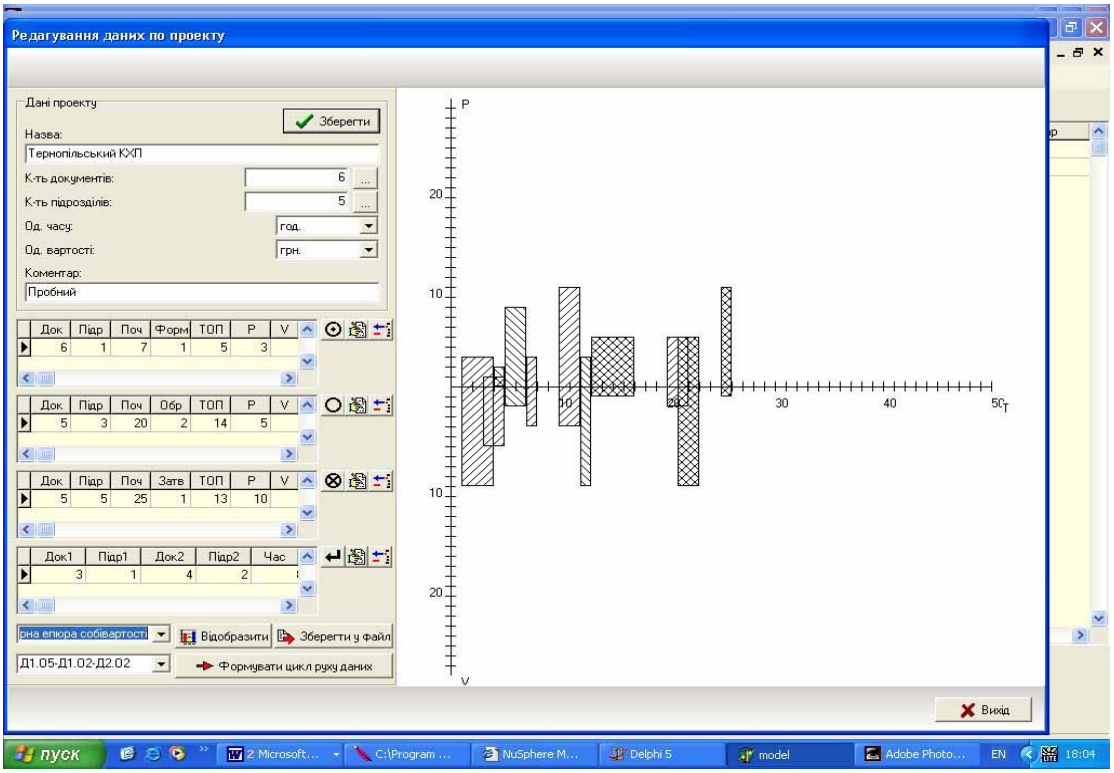
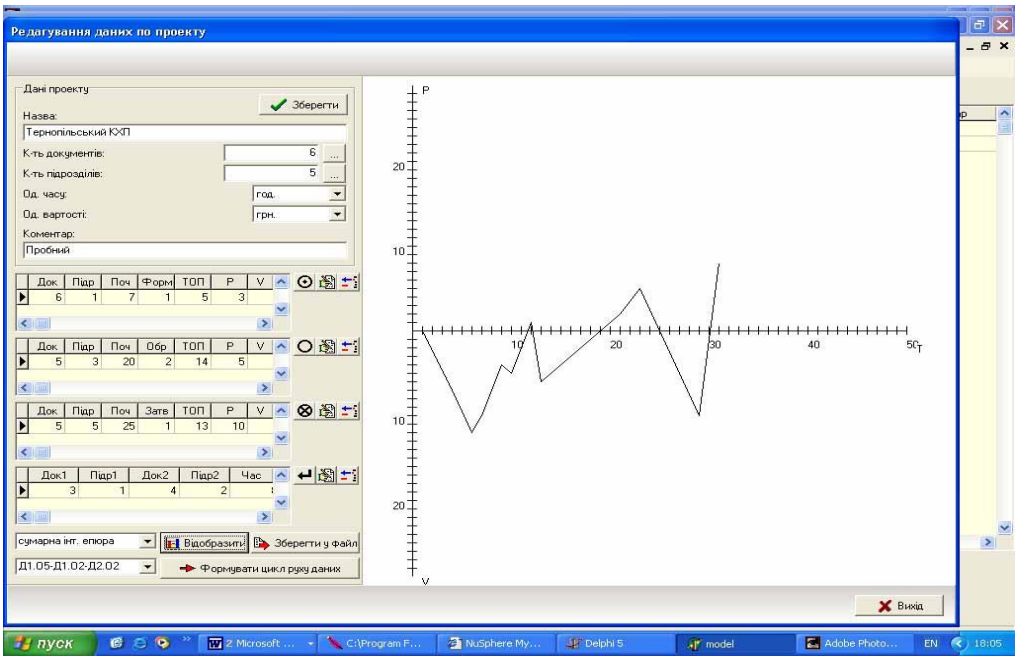


Рис.5.5**.** Формування матричної моделі

На рис 5.6. показані результати виконання програми





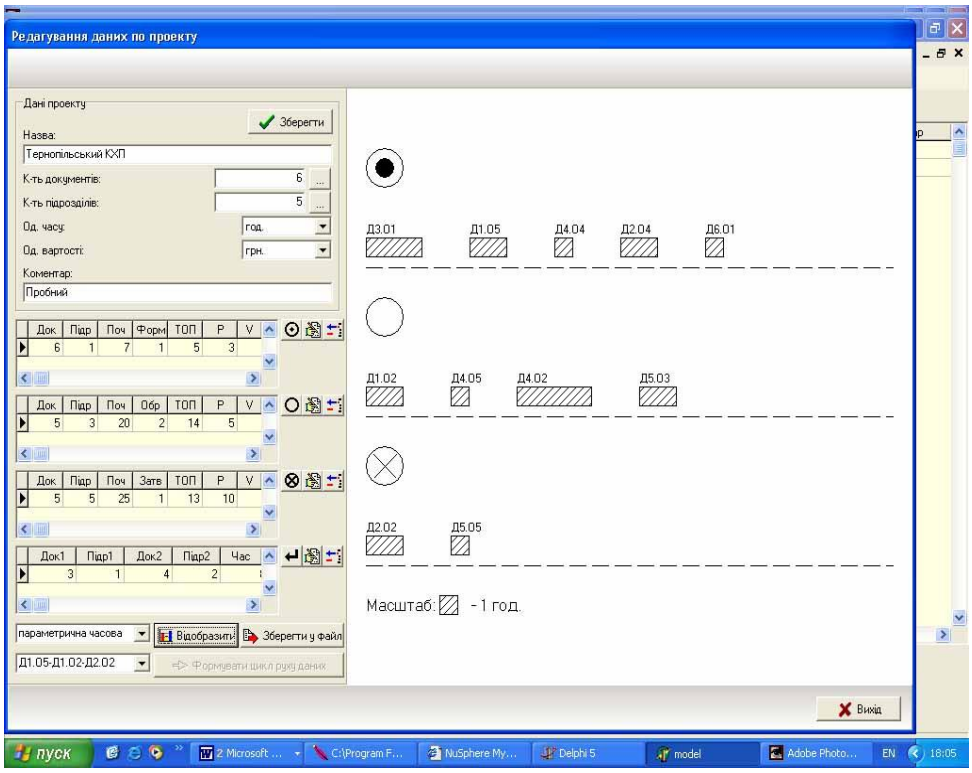


Рис. 5.6. Результати виконання програми

ВИСНОВКИ ПО ПЯТОМУ РОЗДІЛУ

1. Розроблено в середовищі Delphi ПЗ побудови сукупності похідних моделей руху даних на основі МРД, які є інженерними засобами автоматизованого проектування.

2. Розроблено програмне забезпечення, яке може бути використано для аналізу та діагностики ефективності КС.

3. Розроблений інтерфейс користувача, який дозволяє практично реалізувати формування вхідних даних для побудованої моделі на основі атрибутів МРД.

4. Згідно розроблених теоретичних засад та моделювання методів організації руху даних в КС побудовано сімейство МРД для діючого промислового підприємства, що дозволило формалізувати структуру руху даних, оптимізувати інформаційні потоки КС, розрахувати ступінь використання ресурсів, розробити рекомендації по вдосконаленню інформаційної системи та ефективності використання інтелектуальних ресурсів КМ, з метою підвищення ефективності контролю відхилень технологічних параметрів від норми, підвищення стабільності та зниження собівартості виробництва.

ЗАГАЛЬНІ ВИСНОВКИ

1. Отримані аналітичні вирази та розроблена методологія обґрунтування критеріїв якості розподілених комп’ютерних систем на основі оцінки ентропії сукупності факторів, математичного сподівання та середньоквадратичного відхилення експертних оцінок якості.

2. Запропоновані функціонали характеристик системних об’єктів КС на основі глобальної моделі, що включає процесори, дані, систему передавання даних, об’єкти управління та оператори, які об’єднані між собою через інтерфейсні зв’язки. Розроблені функціонали включають параметри часу функціонування, об’єм пам’яті, ресурси зчитування та запам’ятовування даних, швидкість обміну даними та імовірності помилок в каналах зв’язку, а також характеристики моделей об’єктів управління, що дозволило суттєво спростити процеси проектування та оптимізації КС на основі моделей руху даних.

3. Розвинуто теорію графів в частині розширення та конкретизації атрибутів направлених графів та матриць інциденцій шляхом розробки продукційних моделей подання знань на основі матричних моделей руху даних, що дозволило розробити сукупність похідних моделей руху даних і суттєво спростило інженерні розрахунки по проектуванню, діагностиці та оптимізації комп’ютерних систем.

4. Розроблено метод побудови тривимірних та двовимірних модифікованих моделей руху даних на основі додатково введених атрибутів оцінки ступеня використання ресурсів в активних вузлах комп’ютерної системи, що дозволило підвищити якість моделювання комп’ютерних систем та практичну результативність діагностики їх ефективності та оптимізації характеристик.

5. Побудовані матричні моделі руху даних адекватні архітектурам КС з концентрованою, розподіленою однорівневою, багаторівневою організацією з використанням провідних, безпровідних та оптичних ліній зв'язку, які дозволили спростити процеси проектування складних РКС та оптимізувати їх характеристики.

6. Розроблено методи проектування КС на основі характеристик: готовності підприємства та затрат часу на впровадження інформаційних задач, витрат на

постановку та оцінку економічної ефективності від впровадження інформаційних задач.

7. Розроблені теоретичні основи побудови епюр собівартості руху даних в активних вузлах КС, що дозволило спростити вирішення інженерних задач розрахунку характеристик циклів руху даних та глобальної оцінки ефективності проектованої або діагностованої РКС.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Столлингс В. Структурная организация и архитектура компьютерных систем: Пер. с англ.–М.: Издательский дом "Вильямс", 2002.–896с.

2. Буров Є. Комп'ютерні мережі. – Львів: БаК, 1999. – 468 с.

3. Коуров Л.В. Информационные системы и сети. Мн.: Издание НИУП, 1997.

4. Бунин С. Г., Войтер А.П. Вычислительные сети с пакетной радиосвязью. – Киев: Техніка, 1989. – 223с.

5.Таненбаум Є. Современные компьютерные сети.–СПб.: Питер, 2003.–992с.

6. I. Pitukh, Y. Nykolaychuk, N. Vozna. Principles of computer networks construction with deep paralleling of information flows on the basis of matrix models of data movement // Proc. of International Conf. "Modern Problems of Radio Engineering, Telecommunicatioons and Computer Science". (TCSET'2004). Lviv-Slavske (Ukraine).-2004.- P.417 – 419.

7. Пітух І. Проектування характеристик системних об'єктів комп'ютерних мереж з глибоким розпаралеленням інформаційних потоків // Вісник Технологічного університету Поділля. Технічні науки. – Хмельницький. – 2005. – Т.2, Ч.1, №4. – С.133-136.

8. Конюховский П.В., Колесов Д.Н. Экономическая информатика.- СПб,: 2001.-560 с.

9. Я.М. Николайчук, А.І. Сегін, Н.Д. Круцкевич, Н.Я. Возна. Теорія проектування спеціалізованих комп'ютерних систем на базі аналогії системних об'єктів енергетики// Вісник НУ „Львівська політехніка". Комп'ютерні системи проектування. –2002. № 470.- С. 48–57.

10. Дивак М.П. Властивості інтервальних моделей при інтервальній формі їх параметрів // Сб. науч. тр. международного науч.–учеб. центра информ. технологий и систем, науч. совет НАН Украины по пробл. „Кибернетика". Моделирование и управление состоянием эколого–экономических систем региона.– К.–2001.–С.58–63.

11. Стеклов В.К., БеркманЛ.Н. Проектування телекомунікаційних мереж.–К.: Техніка, 2002.–792с.

12. Майника Э. Алгоритмы оптимизации на сетях и графах.–М.: Мир, 1981.–323с.

13. Оре О. Графы и их применение.– М.: Мир, 1965.–173с.

14. Мартин Дж. Вычислительные сети и распределенная обработка данных.–М.:Финансы и статистика, 1985.–256с.

15. Пуртов С.Т. Автоматизированные системы управления предприятием.–М.: Высшая школа, 1989.–396с.

16. Николайчук Я.М. Низові обчислювальні мережі: Учбовий посібник.–К:УМК ВО, 1990.– 64с.

17. Пономаренко В.С. Проектування інформаційних систем.–К:“Академія”, 2002.– 488с.

18. Локазюк В.М., Поморова О.В., Домінов А.О. Інтелектуальне діагностування мікропроцесорних пристроїв та систем: Навч. посібник для вузів. Хмельницьк 2001.–286с.

19. Бабич М.П., Жуков І.А. Комп’ютерна схемотехніка: Навчальний посібник.–К.: „МК–Прес”, 2004.– 312с.

20. Олифер В.Г., Олифер Н.А., Компьютерные сети. Принципы, технологии, протоколы. –СПб: Питер, 2000.– 672с.

21. Точчи, Рональд, Дж., Уидмер, Нил, С. Цифровые системы. Теория и практика.: Пер. с англ.–М. : Издательский дом „Вильямс”, 2004.– 1024 с. 22. Новиков Ю., Новиков Д., Черепанов А., Чуркин В. Компьютеры, сети, Интернет. Энциклопедия.– СПб.: Питер, 2002.– 928с.

23. Олифер В.Г., Олифер Н.А., Сетевые операционные системы. – СПб: Питер, 2002.– 544с.

24. Катренко А.В. Системний аналіз об’єктів та процесів комп’ютеризації: Навч. посібник. –Львів: «Новий світ – 2000», 2001. – 424 с.

25. Дунець Р.Б. Аналіз та синтез топологій комп’ютерних видавничо–поліграфічних систем. – Львів: НВФ „Українські технології”, 2003. – 192с.

26. Макс Ж. Методы и техника обработки сигналов при физических измерениях: Пер.с франц. – М.: Мир, 1983. –Т.1. – 311 с.; Т.2. – 256с.

27. Кузьмин И.В., Кедрус В.А. Основы теории информации и кодирования. – К.: «Вища школа», 1986.– 238с.

28. Сегін А.І., Николайчук Я.М., Сабадаш І.О. Теорія побудови ентропійних моделей складних об’єктів управління на базі кореляційних функцій. Оптико–електронні інформаційно–енергетичні технології. – Вінниця: ВДТУ. – 2002.- №1(3). –С.69–79.

29. Грибанов Ю.И., Веселова Г.П., Андреев В.Н. Автоматические цифровые корреляторы. –М.: Энергия, 1971.–240с.

30. Молчанов А.А. Моделирование и проектирование сложных систем. – К.: Высш. шк.,–1988. – 359 с.

31. Пітух І., Николайчук Я., Возна Н. Принципи побудови комп’ютерних мереж з глибоким розпаралелюванням інформаційних потоків на основі матричних моделей руху даних // Вісник НУ „Львівська політехніка”. Радіоелектроніка та телекомунікації. – 2004.- № 508. - С. 263–268.

32. Y. Nykolaychuk, I. Pitukh, N. Vozna, L. Nykolaychuk. Information technologies of models formalization and designing for data movement in computer networks of automatic control system // Proc. of the third IEEE workshop on Intelligent data acquisition and advanced computing systems: Technology and applications (IDAACS 2005). – Sofia, (Bulgaria).-2005. – P.253–259.

33. Пітух І.Р. Системні характеристики формальних об’єктів моделей руху даних в комп’ютерних мережах // Тези доповідей ІІІ Міжнар. конф. „PHOTONICS–ODS 2005”. – Вінниця. – 2005. – С. 60–61.

34. M. Dyvak, Yu.Franko, I.Pituh, V. Tsymbaliy. Algorithm of technological process interval modeling// Proc. International Conf. on Modern Problems of Telecommunications, Computer Science and Engineers Training.- Lviv–Slavsko, (Ukraine)-2000. – P. 31.

35. М. П. Дивак, І.Р. Пітух, Н.П. Шкляренко, Ю.П. Франко. Використання властивостей інтервальних похибок при моделюванні технологічних процесів//

Вимірювальна та обчислювальна техніка в технологічних процесах: Збірник наукових праць.–Хмельницький: ТУП, 2000.– С. 272.

36. M. Dyvak, Yu.Franko, I.Pituh, S. Voloshchyk. The full combination algorithm modification in the task of technological process interval modeling// Proc. the VI–th International Conf. CADSM 2001.- Lviv–Slavsko (Ukraine).-2001. – P. 133.

37. Пітух І., Николайчук Я., Возна Н. Моделювання руху даних та методологія проектування комп'ютерної мережі з паралельними інформаційними потоками // Вісник Технологічного університету Поділля. Технічні науки. – Хмельницький. – 2004. – Т.2, Ч.1, №2.– С. 33-35.

38. Пітух І. Особливості структурної організації фреймів в комп'ютерних мережах з глибоким розпаралеленням інформаційних потоків// Вісник Технологічного університету Поділля . –Хмельницький.- 2005.- №4,Ч.1, Т.2. С.7–10.

39. I. Pitukh, Y. Nykolaychuk, N. Vozna. Information technologies of models data movement construction in the automatic management systems // Proc. of the VIIII–th International Conf. CADSM 2005. – Lviv–Polyana (Ukraine). – 2005.- P.427–428.

40. Пітух І. Критерії ефективності використання ресурсів архітектури інформаційних систем, які реалізують моделі руху даних // Вісник Технологічного університету Поділля. Технічні науки. – Хмельницький. – 2006. – №5.– С.106-109.

41. Пітух І. Кореляційні та ентропійні моделі об'єктів управління розподілених комп'ютерних мереж// Наукові вісті інституту менеджменту та економіки "Галицька академія". Ів. Франківськ.–2006. – № 2 (10).– С.117 – 120.

42. Николайчук Я.М., Лучук М.А., Жуган Л.И., Шевчук Б.М Идентификация информационных состояний объектов исследования на основе системы логико–статистических информационных моделей: Препринт/ АНУССР. Ин–т кибернетики им.В.М.Глушкова; 88–45. К.: 1988.- 86 с.

43. Y.Nikolaychuk, I.Andrushko. Theoretical Bases of Logical Statistic Informative Models and Prospect of Their Application in the Distributed Computer System// Матеріали VIII Міжнар. наук.–техн. конференції CADSM 2005.- Львів-Поляна.- 2005.-265с.

44. Петришин Л.Б. Теоретично–числові основи перетворення форми інформації// Матеріали 3–ї Міжнар. НТК “Контроль і управління в технічних системах”. – Вінниця. – 1995.- 76 с.

45. Пасічник В.В., Резніченко В.А. Організація баз даних та знань.–К,: Видавнича група BHV, 2006.–384с.

46. Палагін А.В., Николайчук Я.М. Опыт разработки микропроцессорных распределенных систем реального времени.–К : Знание, 1988.– 19 с.

47. Пасічник В.А., Галайда Р.В., Ковальова Ю.В. Моделювання машинобудівного виробництва у середовищі „GALAXY”// Вісник Хмельницького національного університету.–2006 №5 с.14–23.

48.Томашевський В.М. Моделювання систем.– К.: Видавнича група BHV, 2005.– 352с.

49. Pening P.J., Buzen J.P. Operetional analysis of queueing network models //Computing Serveys.– 1978.– Vol. 10, № 3.– P.225–261.

50. Шахнович И. Современные технологии беспроводной святи. М., Техносфера, 2006 – 166с.

51 Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем : Учебник для вузов .–СПб.: Питер, 2006.– 668 с.

52. Мельничук С.І. Основи автоматизованого проектування елементів та засобів обчислювальної техніки : Навчально–методичний посібник.– Івано–Франківськ : Видавництво ІМЕ, 2004.–180 с.

53. Кузьмин С.З. Основы проектирования систем цифровой обработки радиолокационной информации.–М.: Радио и связь, 1986.– 352 с.

54. Цикритзис Д., Лоховски Ф. Модели данных.- М.: Финансы и статистика, 1985.-343 с.

55. Столлингс В . Современные компьютерные сети .– СПб Питер, 2003.– 783 с.

56. Столлингс В. Передача данных.– СПб.: Питер, 2004.– 750 с.

57. Хаусли Т. Системы передачи и телеобработки данных: Пер. с англ.– М.: Радио и связь, 1994.–456 с.

58. Гук М. Апаратные средства IBM PC. Энциклопедия .– СПб : Питер, 2001. – 816 с.

59. Сеньо П.С. Теорія ймовірностей та математична статистика : Підручник.–К : Центр навчальної літератури , 2004. – 448 с.

60. Журавський В.С. Україна на шляху до інформаційного суспільства .– К: ІВЦ „Видавництво „Політехніка”, 2004.– 484 с.

61. Бусленко Е.А. и др. Лекции по теории сложных систем.– М.: Сов. радио, 1973.–283 с.

62. Основы моделирования сложных систем /Л.И. Дыхненко, В.Ф. Кабаненко, И.В. Кузьмин и др.– К: Вища шк., 1981.– 246 с.

63. Месарович М., Такахара Я. Общая теория систем: математические основы: Пер.с англ.– М.: Мир, 1978.– 457 с.

64. Николаев В.И., Брук В. М. Системотехніка: методы и приложения .–Л.: Машиностроение, 1985.–684 с.

65. Флейшман Б.С. Основы системологии. – М.: Радио и связь, 1982.– 342 с.

66. Шнейдер Ю.А., Шаров А.А. Системы и модели.– М.:Радио и связь, 1982.– 348 с.

67. Веников В.А Теория подобия и моделирования .– М.: Высш. шк., 1976.– 384 с.

68. Молчанов А.А. Моделирование и проектирование сложных систем .– К.: Высш. шк., 1988.–359 с.

69. Пранявичюс Г. Модели и методы исследования вычислительных систем.– Вильнюс: Мокслас, 1982.–228 с.

70. Железов И.Г. Сложные технические системы (оценка характеристик). – М .: Высш.шк., 1984.– 119 с.

71. Основы теории вычислительных систем /С.А. Майоров и др.. – М.: Высш. шк..,1978.–408 с.

72. Локазюк В.М. Контроль і діагностування обчислювальних пристроїв та систем : Навч. посібник для вузів .– Хмельницький : ТУП, 2001.–242 с.

73. Локазюк В.М. Проблеми та методологія контролю і діагностування сучасних мікропроцесорних пристроїв та систем // Вимірювальна та обчислювальна техніка в технологічних процесах.– 2000.- № 2.–с.10–17.

74. Справочник проектировщика АСУ ТП / Г.Л. Смилянский и др.-М.: Машиностроение, 1983, 527 с.

75. О создании квазинатурной модели комплекса технических средств АСУ / В.А. Бункин и др./ Под ред. В.И. Николаева.- Л.: 1980.- 218 с.

76. Автоматизовані системи обробки економічної інформації: Підручник / Г.В. Лавінський і ін.- К.: Вища школа, 1985.- 287 с.

77. Точчи Р., Уиядмер Н. Цифровые системы. Теория и практика. – М.: Вильямс, 2004. – 1024 с.

78. Литвинов В.В. Математическое обеспечение проектирования вычислительных систем и сетей. – К.: Техніка, 1982. – 216 с.

79. Основи системного аналізу і проектування АСУ: Учеб.пособие / А.А. Павлов, С.Н.Гриша, В.Н.Томашевский и др./ Под ред. А.А.Павлова. – К.: Вища школа, 1991. – 367 с.

80. Питерсон Дж. Теория сетей Петри и моделирование систем. – М.: Мир, 1984. – 264 с.

81. Феррари Д. Оценка производительности вычислительных систем. – М.: Мир, 1981. – 576 с.

82. http://www.siemens.com - офіційний сайт компанії Siemens

83. http://www.motorola.com - офіційний сайт компанії Motorola

84. http://www.philips.com - офіційний сайт компанії Philips

85. Культин Н.Б. Delphi в задачах и примерах. – М.: Вильямс, 2004. – 288 с.

86. Пестриков В., Маслобоев А. Delphi на примерах. – Л.: КБП, 2005. – 496 с.

87. Фаронов В.В. Delphi: Программирование на языке высокого уровня. – К.: Вища школа, 2006 – 640 с.

88. Осипов Д. Delphi Профессиональное программирование. – М.: Россия, 2004. – 1056 с.

89. Андрушко І.В., Пітух І.Р., Николайчук Я.М. Теоретичні основи та інформаційні технології побудови логіко–статистичної інформаційної моделі (ЛСІМ–4) на основі контролю спектральних характеристик об'єктів управління // Оптико–електронні інформаційно–енергетичні технології.– Вінниця : ВНТУ. – 2006. –№ 2 (12). –С.110–118.

90. Пітух І. Інформаційна технологія побудови миттєвих та інтегральних економічних епюр руху даних на основі циклів матричних моделей комп'ютерних систем //Вісник Технологічного університету Поділля. Технічні науки. – Хмельницький. – 2007. – Т.1, №3. – С.130-134.

91. Пітух І.Р. Теоретичні основи побудови моделей економічних епюр руху даних в комп'ютерних мережах з використанням різних теоретико – числових базисів // Збірник наукових праць. Інститут проблем моделювання в енергетиці НАН України. – 2006.– № 37. – С.42–46 .

92. Henry Ott. Noise Reduction Techniques in Electronic System.- John Wiley & Sons, 1988.

93. Patrick J Zabinski. Surface roughness of pcb tracks…// Correspondence to the SI–LIST.- 11 Jun 2001.

94. Svensson C., Dermer G. Time Domain Modeling of Lossy Interconnect//, IEEE Trans. Advanced Packaging.-2001.- Vol. 24, №. 2.

95. Edwards T.C. Foundations of Interconnect and Microstrip Design, .- John Wiley and Sons, 2000.

96. ISO/IEC 11801:2002 Information Technology. Generic Cabling for Customer Premises/ 2002.

97. Cunningham D., Lane W.G. Gigabit Ethernet Networking.- Macmillan Technical Publishing, 1999.

98. Управляющие вычислительные машины в АСУ технологическими процессами /Под. ред. Т. Харрисона.- Т.1. – М.: Мир, 1975.– 230с.

99. Управляющие вычислительные машины в АСУ технологическими процессами /Под. ред. Т. Харрисона/- Т.2. – М.: Мир, 1975.– 530с.

100. Малиновський Б.М., Боюн В.П., Козлов Л.Г. Введение в кибернетическую технику. Параллельные структуры и методы. – К: Наукова думка, 1989. – 272с.

101. Рабин М. Основы современной системотехники. – М.: Мир, 1975. –528 с.

102. Глушков В.М., Иванов В.В., Яненко В.М. Моделирование развивающихся систем. – М.: Наука, 1983.- 352 с.

103. Дуж Я. Организация системы информации на предприятии.– М.: Прогресс, 1972.– 352 с.

104. Фритч В. Применение микропроцессоров в системах управления : Пер. с нем.– М.: Мир, 1984.– 464 с.

105. Краус М., Кучбах Э., Вошни О. Сбор данных в управляющих вычислительных системах : Пер с нем.–М .: Мир, 1987.– 294 с.

106. Блейхаут Р. Быстрые алгоритмы цифровой обработки сигналов. – М.: Мир, 1989. – 448 с.

107. Левицький А.О. Метод формування повідомлень на основі інтегрально-імпульсних моделей. // BISTRO/96/052 Матеріали 2-ї Міжнар. наук.-практ. конф. «Управління енерговикористанням». – Львів. – 1997. – С.36-39.

108. Литвин А.И., Май А.И., Писаренко Л.А. Организация векторных вычислений спектральных коэффициентов преобразования Хаара // Тезисы докладов междунар. конф. по вычислительной математике (МКВМ-2002).- Новосибирск.- 2002.

109. Корнилов А.И., Семенов М.Ю., Ласточкин О.В. Принципы построения модулярных индексных умножителей // Известия ВУЗов. Электроника. – 2004. - №2. – С.48-55.

110. Я. Николайчук, Р. Король. Вертикальна інформаційна технологія в базисі Галуа – новий напрямок у розвитку комп’ютерних машин. – Львів: CCU’2000, 2000.

111. Петришин Л.Б. Теоретичні основи перетворення форми та цифрової обробки інформації в базисі Галуа: Навч.посібник. – К.: ІзіМН МОУ, 1997. – 237 с.

112. Палагин А.В., Журавский С.Н., Павлишин С.В. Организация и принципы построения локальной сети управления технологическими обьектами.- Киев: ИНК.АН УССР, 1987-23с.

113. Джонсон, Говард В. Высокоскоростная передача цифровых данных.: Пер. с англ.. – М.: Издательский дом «Вильямс», 2005. – 1024 с.

114. Мартин Дж. Организация баз данных в вычеслительных системах.-М.: Мир, 1980.- 662 с.

115. Таненбаум Э., Ван Стеен М. Распределенные системы. Принципы и парадигмы. - СПб.: Питер, 2003. - 880с.

116. Гулевич Д. С. Сети связи следующего поколения.: - М.: БИНОМ, 2007. – 673 с.

117. Корнеев И.К., Ксандопуло Г.Н., Машурцев В.А. Информационные технологии.: - М.: Проспект, 2007.- 854 с.

118. Никульский И. Оптические интерфейсы цифровых коммутационных станций и сети доступа.М.: Техносфера, 2006.-563 с.

119. Логунова О. С., Ячиков И. М., Ильина Е. А. Человеко-машинное взаимодействие: теория и практика.- Ростов-на-Дону.: ФЕНИКС, 2006.- 612 с.

120. Гаврилов М.В. Информатика и информационные технологии.: СПб.: ГАРДАРИКА, 2006.-453 с.

121. Портнов Э. Л. Оптические кабели связи и пассивные компоненты волоконно-оптических линий связи -М.: Горячая Линия - Телеком, 2006.-876 с.

122. Брянцев И.Н Data Mining. Теория и практика - М.: БДЦ-Пресс, 2006. – 598 с.

123. Ralph D., Graham P. MMS: Technologies, Usage and Business Models.: Пер. с англ.. – М.: Издательский дом «Вильямс», 2003. – 986 с.

124. Алексеев В.Е. Основы информационных технологий. Графы и алгоритмы. Структуры данных. Модели вычислений.: М.: BHV, 2006. – 320 с.

125. Могилев А.В., Листрова Л.В. Информация и информационные процессы. Социальная информатика - М.: BHV, 2006. – 240 с.

126. Дейтел Х.М., Дейтел П.Дж., Чофнес Д.Р. Операционные системы. Распределенные системы, сети, безопасность - М.: БИНОМ, 2006. – 704 с.

127. Кормен Т. Х., Лейзерсон Ч.И., Ривест Р.Л., Штайн К. Алгоритмы: построение и анализ. - СПб.: Вильямс, 2005.-1296 с.

128. Раскин Д. Интерфейс: Новые направления в проектировании компьютерных систем - СПб.: Символ 2005.-272с.

129. Гофф Макс К. Сетевые распределенные вычисления: достижения и проблемы - М.: „Кудиц-образ”, 2006. – 320 с.

130. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие .- М.- Высш. шк., 2005.- 295 с.

131. Михайлова Е.Е., Рыбак К.С., Тюкачев Н.А. Программирование в Delphi для начинающих.- М.: BHV, 2007. – 672 с.

132. Шпак Ю.А. Разработка приложений в Delphi 2005/2006.- М.: МК-Пресс, 2006.- 544 с.

133. Гриценко В.И., Урсатьев А.А. Распределенные информационные системы. Состояния. Перспективы развития // Управляющие системы и машины. №4. 2003. с 11-21.

134. Модель распределенной информационной системы широкого применения / В.И. Гриценко, Е.А. Котиков, А.А. Урсатьев и др.// УСиМ.-1999.- №5- С.32.-42.

135. Ахо Альфред, В. Хопкрофт, Джон, Ульман, Джеффри Структуры данных и алгоритмы.: Пер.с англ.: М.: Издательский дом “Вильямс”, 2001.- 384с.

136. Гук. М. Аппаратные средства локальных сетей. Энциклопедия – СПб: Питер, 2000.-576 с.

137. Андріїшин М.П. та інш. Гідравліка. Навчальний посібник.– Івано–Франківськ: Факел, 2000.–253с.

138. Посацький С. Опір матеріалів.– Львів: Вид.ЛДУ,1973.–404с.

139. Пітух І.Р. Моделі комп’ютерних мереж на основі інтегральних економічних епюр // Збірник наукових праць, Інститут проблем моделювання в енергетиці НАН України. – Київ. – 2004.– № 27. – С.81–86 .

140. Васильев В.В., Кузьмук В.В. Сети Петри, параллельные алгоритмы и моделирование мультипроцессорных систем.- К.: Наукова думка, 1990.

141. Волкова В.Н., Денисов А.А. Основы теории систем и системного анализа: Учебник, издание 2. – СПб.: Изд-во СПбГТУ, 1999.

142. Джонсон Дж . Методы проектирования.- М.: Мир, 1986.

143. Кузьмина Е.А., Кузьмин А.М. Функционально- стоимостный анализ. Концепции и перспективы // „Методы менеджмента качества”, №8, 2002.

144. Литвак Б.Г. Экспертная информация: методы получения и анализа. – М: Радио и связь, 1981.

145. Николайчук Я.Н. Разработка теории и комплекса технических средств формирования, передачи и обработки цифровых сообщений в низовых вычислительных сетях автоматизированных систем: Дис. д-ра техн. наук: 05.13.05 – К., 1991.- 573с.

146. Коутс Л., Олейник И. Интерфейс „человек - компьютер”.- М.: Мир, 1993.- 400 с.

147. Мартин Дж. Планирование развития автоматизированных систем.- М.: Финансы и статистика, 1984.- 169 с.

148. Культин Н.Б. Основы программирования в Turbo Delphi.: М.- BHV.2007.-384 с.

149. Бобровский С. И Технологии Delphi. Разработка приложений для бизнеса. Учебный курс.: СПб.- Питер. 2006.-720 с.

150. Архангельский А. Я. Программирование в Delphi. Учебник по классическим версиям Delphi.: М.- Бином. 2006.- 1152 с.Начало формы

151. Хетагуров Я.А. Проектирование автоматизированных систем обработки информации и управления.: М.-Висш.шк.-2006.-223 с.

152. Николайчук Я.М., Яцків Н.Г. Методи стиснення даних в багатоканальних системах на основі кодів Галуа // Вісник національного університету “Львівська політехніка”. Радіоелектроніка та телекомунікації. – Львів. – 2002. – №443. – С.135–138.

153. Барышников А.А. Кузьмин А.М. Формы применения функционально-стоимостного анализа // „Машиностроитель”, № 6, 2001.

154. Коваленко А.М. САПР: Методология и формализированные методы Л., 1988.-120 с.

155. www.dalsemi.com – офіційний сайт компанії Dallas Semiconductor.

156. www.maxim-ic.com – офіційний сайт компанії Maxim.

157. Горев А., Макашарипов С., Владимиров Ю., Microsoft SQL Server 6.5 для профессионалов.- СПб : Питер, 1998.-464 с.

158. Шумаков П.В. Delphi 3 и разработка приложений баз данных.- М.: НОЛИДЖ, 1998.- 704 с.

```
program model;
uses
  Forms,
  UMain in 'UMain.pas' {frmMain},
  UProjects in 'UProjects.pas' {fmProjects},
  UNewProject in 'UNewProject.pas' {fmNewProject},
  UData in 'UData.pas' {BimaDataModule: TDataModule},
  UNewDI in 'UNewDI.pas' {fmNewDI},
  UNewOD in 'UNewOD.pas' {fmNewOD},
  UNewDZ in 'UNewDZ.pas' {fmNewDZ},
  UNewTransfer in 'UNewTransfer.pas' {fmNewTransfer},
  UAddDocs in 'UAddDocs.pas' {fmAddDocs},
  UAddDepts in 'UAddDepts.pas' {fmAddDepts};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TfrmMain, frmMain);
  Application.CreateForm(TBimaDataModule, BimaDataModule);
  Application.Run;
end.
```

```
unit UAddDepts;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
```

```
  ActnList, DBCtrls, Grids, DBGrids, StdCtrls, Buttons, ExtCtrls, Db;

type
  TfmAddDepts = class(TForm)
    pnlBottom: TPanel;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    pnlClient: TPanel;
    DBGrid1: TDBGrid;
    pnlTop: TPanel;
    DBNavigator1: TDBNavigator;
    ActionList1: TActionList;
    acSave: TAction;
    acCancel: TAction;
    procedure acSaveExecute(Sender: TObject);
    procedure acCancelExecute(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  fmAddDepts: TfmAddDepts;

implementation

uses UData;

{$R *.DFM}

procedure TfmAddDepts.acSaveExecute(Sender: TObject);
begin
  if BimaDataModule.dsDepts.State in [dsEdit,dsInsert] then BimaDataModule.mytblDepts.Post;
end;

procedure TfmAddDepts.acCancelExecute(Sender: TObject);
begin
  if BimaDataModule.dsDepts.State in [dsEdit,dsInsert] then
BimaDataModule.mytblDepts.Cancel;
end;

end.



unit UAddDocs;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, StdCtrls, Buttons, Grids, DBGrids, DBCtrls, ActnList, Db;

type
  TfmAddDocs = class(TForm)
    pnlBottom: TPanel;
    pnlClient: TPanel;
    pnlTop: TPanel;
    DBGrid1: TDBGrid;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    DBNavigator1: TDBNavigator;
    ActionList1: TActionList;
    acSave: TAction;
    acCancel: TAction;
    procedure acSaveExecute(Sender: TObject);
    procedure acCancelExecute(Sender: TObject);
  private
    { Private declarations }
  public
```

```
    { Public declarations }
  end;

var
  fmAddDocs: TfmAddDocs;

implementation

uses UData;

{$R *.DFM}

procedure TfmAddDocs.acSaveExecute(Sender: TObject);
begin
  if BimaDataModule.dsDocs.State in [dsEdit,dsInsert] then BimaDataModule.mytblDocs.Post;
end;

procedure TfmAddDocs.acCancelExecute(Sender: TObject);
begin
  if BimaDataModule.dsDocs.State in [dsEdit,dsInsert] then BimaDataModule.mytblDocs.Cancel;
end;

end.


unit UData;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, mySQLDbTables;

type
  TBimaDataModule = class(TDataModule)
    dsProject: TDataSource;
    dsTimeuom: TDataSource;
    dsPriceuom: TDataSource;
    dsqProject: TDataSource;
    mytblProject: TmySQLTable;
    mytblTimeuom: TmySQLTable;
    mytblPriceuom: TmySQLTable;
    myqrProject: TmySQLQuery;
    bima: TmySQLDatabase;
    mytblDIinfodi_start_time: TIntegerField;
    mytblDataCycledc_form_dz: TIntegerField;
    mytblDataCycledc_prib_dz: TIntegerField;
    mytblDataCycledc_vitr_dz: TIntegerField;
    dsDepts: TDataSource;
    dsDocs: TDataSource;
    mytblDocs: TmySQLTable;
    mytblDepts: TmySQLTable;
    mytblDocsd_project_no: TIntegerField;
    mytblDocsd_doc_no: TIntegerField;
    mytblDocsd_doc_name: TStringField;
    mytblDeptsdp_project_no: TIntegerField;
    mytblDeptsdp_dept_no: TIntegerField;
    mytblDeptsdp_dept_name: TStringField;
    dsqDataCycle: TDataSource;
    myqrDataCycle: TmySQLQuery;
    myqrDataCycledc_project_no: TIntegerField;
    myqrDataCycledc_doc_di: TIntegerField;
    myqrDataCycledc_dept_di: TIntegerField;
    myqrDataCycledc_start_di: TIntegerField;
    myqrDataCycledc_form_di: TIntegerField;
    myqrDataCycledc_prib_di: TIntegerField;
    myqrDataCycledc_vitr_di: TIntegerField;
    myqrDataCycledc_doc_od: TIntegerField;
    myqrDataCycledc_dept_od: TIntegerField;
    myqrDataCycledc_start_od: TIntegerField;
    myqrDataCycledc_form_od: TIntegerField;
```

```
    myqrDataCycledc_prib_od: TIntegerField;
    myqrDataCycledc_vitr_od: TIntegerField;
    myqrDataCycledc_doc_dz: TIntegerField;
    myqrDataCycledc_dept_dz: TIntegerField;
    myqrDataCycledc_start_dz: TIntegerField;
    myqrDataCycledc_form_dz: TIntegerField;
    myqrDataCycledc_prib_dz: TIntegerField;
    myqrDataCycledc_vitr_dz: TIntegerField;
    procedure DataModuleCreate(Sender: TObject);
    procedure DataModuleDestroy(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  BimaDataModule: TBimaDataModule;

implementation

{$R *.DFM}

procedure TBimaDataModule.DataModuleCreate(Sender: TObject);
begin
  try
    mytblProject.Open;
    mytblTimeuom.Open;
    mytblPriceuom.Open;
    mytblDIinfo.Open;
    mytblODinfo.Open;
    mytblDZinfo.Open;
    mytblTransfer.Open;
    mytblDataCycle.Open;
    mytblDocs.Open;
    mytblDepts.Open;
  except
    raise Exception.Create('Âèíèêëà ïîìèëêà ïðè â³äêðèò³ ôàéë³â áàçè äàíèõ.');
    mytblProject.Close;
    mytblTimeuom.Close;
    mytblPriceuom.Close;
    mytblDIinfo.Close;
    mytblODinfo.Close;
    mytblDZinfo.Close;
    mytblTransfer.Close;
    mytblDataCycle.Close;
    mytblDocs.Close;
    mytblDepts.Close;
  end;

end;

procedure TBimaDataModule.DataModuleDestroy(Sender: TObject);
begin
  try
    mytblProject.Close;
    mytblTimeuom.Close;
    mytblPriceuom.Close;
    mytblDIinfo.Close;
    mytblODinfo.Close;
    mytblDZinfo.Close;
    mytblTransfer.Close;
    mytblDataCycle.Close;
    mytblDocs.Close;
    mytblDepts.Close;
  except
    raise Exception.Create('Âèíèêëà ïîìèëêà ïðè çàêðèò³ ôàéë³â áàçè äàíèõ.');
  end;

end;
```

```
end.



uses UProjects;

{$R *.DFM}

procedure TfrmMain.acQuitExecute(Sender: TObject);
begin
  Close;
end;

procedure TfrmMain.sbMainChangeSelectedItem(Sender: TObject);
var
  i : integer;
begin
  if sbMain.SelectedItem = nil then
    exit;
  try
    if Assigned(sbMain.SelectedItem) and Assigned(sbMain.ActiveGroup) then
    begin
    try
      i := sbMain.SelectedItem.Index;
      sbMain.SelectedItem := nil;
      case sbMain.ActiveGroup.Index of
        0:
        case i of
          0: acProjects.Execute;
        end;
      end;

    finally
      sbMain.UpdateControlState;
    end;
  end;
  finally
    //FChangingItem := false;
  end;

end;

procedure TfrmMain.acProjectsExecute(Sender: TObject);
begin
  if not Assigned(fmProjects) then
    fmProjects := TfmProjects.Create(Application);
  ShowWindow(fmProjects.Handle, SW_SHOWMAXIMIZED);

end;

end.


unit UNewDI;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, Mask, DBCtrls, ActnList, Db;

type
  TfmNewDI = class(TForm)
    DBEdit1: TDBEdit;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit5: TDBEdit;
    DBEdit6: TDBEdit;
    DBEdit7: TDBEdit;
```

```
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    ActionList1: TActionList;
    acSave: TAction;
    acCancel: TAction;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormDestroy(Sender: TObject);
    procedure acSaveExecute(Sender: TObject);
    procedure acCancelExecute(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  fmNewDI: TfmNewDI;

implementation

uses UData;

{$R *.DFM}

procedure TfmNewDI.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action := caFree;
end;

procedure TfmNewDI.FormDestroy(Sender: TObject);
begin
  fmNewDI := nil;
end;

procedure TfmNewDI.acSaveExecute(Sender: TObject);
begin
  if BimaDataModule.dsDIinfo.State in [dsEdit,dsInsert] then
BimaDataModule.mytblDIinfo.Post;
end;

procedure TfmNewDI.acCancelExecute(Sender: TObject);
begin
  if BimaDataModule.dsDIinfo.State in [dsEdit,dsInsert] then
BimaDataModule.mytblDIinfo.Cancel;
end;

end.



unit UNewDZ;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, Mask, DBCtrls, ActnList, Db;

type
  TfmNewDZ = class(TForm)
    ActionList1: TActionList;
    acSave: TAction;
    acCancel: TAction;
    Label1: TLabel;
```

```
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    DBEdit7: TDBEdit;
    DBEdit6: TDBEdit;
    DBEdit5: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit2: TDBEdit;
    DBEdit1: TDBEdit;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    procedure acSaveExecute(Sender: TObject);
    procedure acCancelExecute(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  fmNewDZ: TfmNewDZ;

implementation

uses UData;

{$R *.DFM}

procedure TfmNewDZ.acSaveExecute(Sender: TObject);
begin
  if BimaDataModule.dsDZinfo.State in [dsEdit,dsInsert] then
BimaDataModule.mytblDZinfo.Post;
end;

procedure TfmNewDZ.acCancelExecute(Sender: TObject);
begin
  if BimaDataModule.dsDZinfo.State in [dsEdit,dsInsert] then
BimaDataModule.mytblDZinfo.Cancel;
end;

end.
unit UNewOD;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ActnList, StdCtrls, Buttons, Mask, DBCtrls, Db;

type
  TfmNewOD = class(TForm)
    Label1: TLabel;
    DBEdit1: TDBEdit;
    Label2: TLabel;
    DBEdit2: TDBEdit;
    Label3: TLabel;
    DBEdit3: TDBEdit;
    Label4: TLabel;
    DBEdit4: TDBEdit;
    Label5: TLabel;
    DBEdit5: TDBEdit;
    Label6: TLabel;
    DBEdit6: TDBEdit;
    Label7: TLabel;
    DBEdit7: TDBEdit;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
```

```
    ActionList1: TActionList;
    acSave: TAction;
    acCancel: TAction;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormDestroy(Sender: TObject);
    procedure acSaveExecute(Sender: TObject);
    procedure acCancelExecute(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  fmNewOD: TfmNewOD;

implementation

uses UData;

{$R *.DFM}

procedure TfmNewOD.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action := caFree;
end;

procedure TfmNewOD.FormDestroy(Sender: TObject);
begin
  fmNewOD := nil;
end;

procedure TfmNewOD.acSaveExecute(Sender: TObject);
begin
  if BimaDataModule.dsODinfo.State in [dsEdit,dsInsert] then
BimaDataModule.mytblODinfo.Post;
end;

procedure TfmNewOD.acCancelExecute(Sender: TObject);
begin
  if BimaDataModule.dsODinfo.State in [dsEdit,dsInsert] then
BimaDataModule.mytblODinfo.Cancel;
end;

end.
```

```
procedure TfmNewProject.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  Action := caFree;
end;

procedure TfmNewProject.FormDestroy(Sender: TObject);
begin
  fmNewProject := nil;
end;

procedure TfmNewProject.acSaveProjectExecute(Sender: TObject);
begin
```

```
  if BimaDataModule.dsProject.State in [dsEdit,dsInsert] then
BimaDataModule.mytblProject.Post;
end;

procedure TfmNewProject.acQuitExecute(Sender: TObject);
begin
  if BimaDataModule.dsProject.State in [dsEdit,dsInsert] then
BimaDataModule.mytblProject.Cancel;
end;

procedure TfmNewProject.acSaveProjectUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.dsProject.State in [dsEdit,dsInsert];
end;

procedure TfmNewProject.acDrawModelGridExecute(Sender: TObject);
begin
  ClearCanvas(Image1.Canvas);
  // ***
  case ComboBox1.ItemIndex of
    0 : DrawMatrixModel;
    1 : DrawGraphTree;
    2 : DrawParamTime;
    3 : DrawStructTime;
    4 : DrawEpureSb;
    5 : DrawEpureDif;
    6 : DrawEpureInt;
    7 : DrawTotalEpureSb;
    8 : DrawTotalEpureDif;
    9 : DrawTotalEpureInt;
  else
    MessageDlg('Áóäü-ëàñêà, âèáåð³òü ìîäåëü.', mtInformation, [mbOk], 0);
  end;
  // ***
end;

procedure TfmNewProject.ClearCanvas(ACanvas: TCanvas);
var
  R : TRect;
begin
  with ACanvas do begin
    Brush.Style := bsSolid;
    Brush.Color := clWhite;
    FillRect(ClientRect);
  end;
end;

procedure TfmNewProject.acAddDIExecute(Sender: TObject);
begin
  BimaDataModule.mytblDIinfo.Insert;
  BimaDataModule.mytblDIinfoDI_PROJECT_NO.Value :=
BimaDataModule.mytblProjectPROJECT_NO.Value;
  fmNewDI:=TfmNewDI.Create(Application);
  try
    fmNewDI.ShowModal;
  finally
    fmNewDI.Free;
    fmNewDI:=nil;
  end;
end;

procedure TfmNewProject.acEditDIExecute(Sender: TObject);
begin
  BimaDataModule.mytblDIinfo.Edit;
  fmNewDI:=TfmNewDI.Create(Application);
  try
    fmNewDI.Caption := 'Ðåäàãóâàííÿ äàíèõ ïî Ä²';
    fmNewDI.ShowModal;
  finally
    fmNewDI.Free;
    fmNewDI:=nil;
```

```
  end;
end;

procedure TfmNewProject.acDeleteDIExecute(Sender: TObject);
begin
  BimaDataModule.mytblDIinfo.Delete;
end;

procedure TfmNewProject.acEditDIUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblDIinfo.RecordCount > 0;
end;

procedure TfmNewProject.acDeleteDIUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblDIinfo.RecordCount > 0;
end;

procedure TfmNewProject.acAddODExecute(Sender: TObject);
begin
  BimaDataModule.mytblODinfo.Insert;
  BimaDataModule.mytblODinfoOD_PROJECT_NO.Value :=
BimaDataModule.mytblProjectPROJECT_NO.Value;
  fmNewOD:=TfmNewOD.Create(Application);
  try
    fmNewOD.ShowModal;
  finally
    fmNewOD.Free;
    fmNewOD:=nil;
  end;
end;

procedure TfmNewProject.acEditODExecute(Sender: TObject);
begin
  BimaDataModule.mytblODinfo.Edit;
  fmNewOD:=TfmNewOD.Create(Application);
  try
    fmNewOD.Caption := 'Ðåäàêòèðîâàíèå ïî ÎÄ';
    fmNewOD.ShowModal;
  finally
    fmNewOD.Free;
    fmNewOD:=nil;
  end;
end;

procedure TfmNewProject.acDeleteODExecute(Sender: TObject);
begin
  BimaDataModule.mytblODinfo.Delete;
end;

procedure TfmNewProject.acEditODUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblODinfo.RecordCount > 0;
end;

procedure TfmNewProject.acDeleteODUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblODinfo.RecordCount > 0;
end;

procedure TfmNewProject.acAddZDExecute(Sender: TObject);
begin
  BimaDataModule.mytblDZinfo.Insert;
  BimaDataModule.mytblDZinfoDZ_PROJECT_NO.Value :=
BimaDataModule.mytblProjectPROJECT_NO.Value;
  fmNewDZ:=TfmNewDZ.Create(Application);
  try
    fmNewDZ.ShowModal;
  finally
    fmNewDZ.Free;
    fmNewDZ:=nil;
```

```
  end;
end;

procedure TfmNewProject.acEditZDExecute(Sender: TObject);
begin
  BimaDataModule.mytblDZinfo.Edit;
  fmNewDZ:=TfmNewDZ.Create(Application);
  try
    fmNewDZ.Caption := 'Ðåäàãóâàíèå äàííûõ ïî ÄÇ';
    fmNewDZ.ShowModal;
  finally
    fmNewDZ.Free;
    fmNewDZ:=nil;
  end;
end;

procedure TfmNewProject.acDeleteZDExecute(Sender: TObject);
begin
  BimaDataModule.mytblDZinfo.Delete;
end;

procedure TfmNewProject.acDeleteZDUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblDZinfo.RecordCount > 0;
end;

procedure TfmNewProject.acEditZDUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblDZinfo.RecordCount > 0;
end;

procedure TfmNewProject.acAddTransferExecute(Sender: TObject);
begin
  BimaDataModule.mytblTransfer.Insert;
  BimaDataModule.mytblTransferTR_PROJECT_NO.Value :=
BimaDataModule.mytblProjectPROJECT_NO.Value;
  fmNewTransfer:=TfmNewTransfer.Create(Application);
  try
    fmNewTransfer.ShowModal;
  finally
    fmNewTransfer.Free;
    fmNewTransfer:=nil;
  end;
end;

procedure TfmNewProject.acEditTransferExecute(Sender: TObject);
begin
  BimaDataModule.mytblTransfer.Edit;
  fmNewTransfer:=TfmNewTransfer.Create(Application);
  try
    fmNewTransfer.Caption := 'Ðåäàãóâàíèå äàííûõ ïî ïåðåäà÷³';
    fmNewTransfer.ShowModal;
  finally
    fmNewTransfer.Free;
    fmNewTransfer:=nil;
  end;
end;

procedure TfmNewProject.acDeleteTransferExecute(Sender: TObject);
begin
  BimaDataModule.mytblTransfer.Delete;
end;

procedure TfmNewProject.acEditTransferUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblTransfer.RecordCount > 0;
end;

procedure TfmNewProject.acDeleteTransferUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := BimaDataModule.mytblTransfer.RecordCount > 0;
```

```
end;

procedure TfmNewProject.DrawMatrixModel;
const
  StartX : integer = 5;
  StartY : integer = 5;
  margin : integer = 10;
var
  cols, rows, cwidth, cheight, twidth, theight : integer;
  i : integer;
  _header : String;
  R : TRect;
  delta : integer;
  text_width : integer;
  x1, x2, y1, y2 : integer;
  circle_size, small_circle, diag : integer;
  _text : String;
begin
  with BimaDataModule do begin
    cols := mytblProjectPROJECT_DEPTS_QTY.Value + 1;
    rows := mytblProjectPROJECT_DOCS_QTY.Value + 1;
    twidth := Image1.Width - margin;
    theight := Image1.Height - margin;
    cwidth := Trunc(twidth / cols);
    cheight := Trunc(theight / rows);
  end;
  // draw rows
  for i := 0 to rows do begin
    Image1.Canvas.MoveTo(StartX, StartY + (i * cheight));
    Image1.Canvas.LineTo(StartX + (cols * cwidth), StartY + (i * cheight));
  end;
  // draw cols
  for i := 0 to cols do begin
    Image1.Canvas.MoveTo(StartX + (i * cwidth), StartY);
    Image1.Canvas.LineTo(StartX + (i * cwidth), StartY + (rows * cheight));
  end;
  // cols header
  for i := 1 to (cols - 1) do begin
    _header := 'O' + IntToStr(i);
    delta := Trunc(cheight / 3);
    Image1.Canvas.Font.Height := cheight - delta;
    text_width := Image1.Canvas.TextWidth(_header);
    while text_width >= cwidth do begin
      Image1.Canvas.Font.Height := Image1.Canvas.Font.Height - 1;
      text_width := Image1.Canvas.TextWidth(_header);
    end;
    Image1.Canvas.TextOut(StartX + (cwidth * i) + Trunc((cwidth-text_width) / 2), StartY +
Trunc((cheight - Image1.Canvas.Font.Height) / 2), _header);
  end;
  // rows header
  for i := 1 to (rows - 1) do begin
    _header := 'Ä' + IntToStr(i);
    delta := Trunc(cheight / 3);
    Image1.Canvas.Font.Height := cheight - delta;
    text_width := Image1.Canvas.TextWidth(_header);
    while text_width >= cwidth do begin
      Image1.Canvas.Font.Height := Image1.Canvas.Font.Height - 1;
      text_width := Image1.Canvas.TextWidth(_header);
    end;
    Image1.Canvas.TextOut(StartX + Trunc((cwidth-text_width) / 2), StartY + (cheight * i) +
Trunc((cheight - Image1.Canvas.Font.Height) / 2), _header);
  end;
  // draw DI elements
  with BimaDataModule do begin
    mytblDIinfo.Filtered := False;
    mytblDIinfo.Filter := 'DI_PROJECT_NO = ' + IntToStr(mytblProjectPROJECT_NO.Value);
    mytblDIinfo.Filtered := True;
    mytblDIinfo.First;
    while not mytblDIinfo.Eof do begin
      x1 := StartX + mytblDIinfoDI_DEPT_NO.Value * cwidth;
      y1 := StartY + mytblDIinfoDI_DOC_NO.Value * cheight;
```

```
      x2 := StartX + (mytblDIinfoDI_DEPT_NO.Value + 1) * cwidth;
      y2 := StartY + (mytblDIinfoDI_DOC_NO.Value + 1) * cheight;
      if cwidth < cheight then circle_size := Trunc(cwidth / 3)
        else circle_size := Trunc(cheight / 3);

      Image1.Canvas.Font.Height := circle_size - 5;
      _text := IntToStr(mytblDIinfoDI_START_TIME.Value);
      Image1.Canvas.TextOut(x1+1, y1+1, _text);
      _text := IntToStr(mytblDIinfoDI_FORM_TIME.Value);
      Image1.Canvas.TextOut(x2-1-Image1.Canvas.TextWidth(_text), y1+1, _text);
      _text := IntToStr(mytblDIinfoDI_TOP.Value);
      Image1.Canvas.TextOut(x1+1, y2-1-Image1.Canvas.Font.Height, _text);
      _text := IntToStr(mytblDIinfoDI_PRIB.Value) + '-' +
IntToStr(mytblDIinfoDI_VITR.Value);
      Image1.Canvas.TextOut(x2-1-Image1.Canvas.TextWidth(_text), y2-1-
Image1.Canvas.Font.Height, _text);

      x1 := x1 + Trunc((x2 - x1 - circle_size) / 2);
      y1 := y1 + Trunc((y2 - y1 - circle_size) / 2);
      x2 := x1 + circle_size;
      y2 := y1 + circle_size;
      Image1.Canvas.Ellipse(x1, y1, x2, y2);
      small_circle := 4; //Trunc(circle_size / 4);
      Image1.Canvas.Ellipse(Trunc(x1+(circle_size - small_circle)/2), Trunc(y1+(circle_size
- small_circle)/2), Trunc(x2-(circle_size - small_circle)/2), Trunc(y2-(circle_size -
small_circle)/2));
      mytblDIinfo.Next;
    end;
    //mytblDIinfo.Filtered := False;
  end;
  // draw OD elements
  with BimaDataModule do begin
    mytblODinfo.Filtered := False;
    mytblODinfo.Filter := 'OD_PROJECT_NO = ' + IntToStr(mytblProjectPROJECT_NO.Value);
    mytblODinfo.Filtered := True;
    mytblODinfo.First;
    while not mytblODinfo.Eof do begin
      x1 := StartX + mytblODinfoOD_DEPT_NO.Value * cwidth;
      y1 := StartY + mytblODinfoOD_DOC_NO.Value * cheight;
      x2 := StartX + (mytblODinfoOD_DEPT_NO.Value + 1) * cwidth;
      y2 := StartY + (mytblODinfoOD_DOC_NO.Value + 1) * cheight;
      if cwidth < cheight then circle_size := Trunc(cwidth / 3)
        else circle_size := Trunc(cheight / 3);

      Image1.Canvas.Font.Height := circle_size - 5;
      _text := IntToStr(mytblODinfoOD_START_TIME.Value);
      Image1.Canvas.TextOut(x1+1, y1+1, _text);
      _text := IntToStr(mytblODinfoOD_FORM_TIME.Value);
      Image1.Canvas.TextOut(x2-1-Image1.Canvas.TextWidth(_text), y1+1, _text);
      _text := IntToStr(mytblODinfoOD_TOP.Value);
      Image1.Canvas.TextOut(x1+1, y2-1-Image1.Canvas.Font.Height, _text);
      _text := IntToStr(mytblODinfoOD_PRIB.Value) + '-' +
IntToStr(mytblODinfoOD_VITR.Value);
      Image1.Canvas.TextOut(x2-1-Image1.Canvas.TextWidth(_text), y2-1-
Image1.Canvas.Font.Height, _text);

      x1 := x1 + Trunc((x2 - x1 - circle_size) / 2);
      y1 := y1 + Trunc((y2 - y1 - circle_size) / 2);
      x2 := x1 + circle_size;
      y2 := y1 + circle_size;
      Image1.Canvas.Ellipse(x1, y1, x2, y2);
      mytblODinfo.Next;
    end;
    //mytblODinfo.Filtered := False;
  end;
  // draw DZ elements
  with BimaDataModule do begin
    mytblDZinfo.Filtered := False;
    mytblDZinfo.Filter := 'DZ_PROJECT_NO = ' + IntToStr(mytblProjectPROJECT_NO.Value);
    mytblDZinfo.Filtered := True;
    mytblDZinfo.First;
```

```
    while not mytblDZinfo.Eof do begin
      x1 := StartX + mytblDZinfoDZ_DEPT_NO.Value * cwidth;
      y1 := StartY + mytblDZinfoDZ_DOC_NO.Value * cheight;
      x2 := StartX + (mytblDZinfoDZ_DEPT_NO.Value + 1) * cwidth;
      y2 := StartY + (mytblDZinfoDZ_DOC_NO.Value + 1) * cheight;
      if cwidth < cheight then circle_size := Trunc(cwidth / 3)
        else circle_size := Trunc(cheight / 3);

      Image1.Canvas.Font.Height := circle_size - 5;
      _text := IntToStr(mytblDZinfoDZ_START_TIME.Value);
      Image1.Canvas.TextOut(x1+1, y1+1, _text);
      _text := IntToStr(mytblDZinfoDZ_FORM_TIME.Value);
      Image1.Canvas.TextOut(x2-1-Image1.Canvas.TextWidth(_text), y1+1, _text);
      _text := IntToStr(mytblDZinfoDZ_TOP.Value);
      Image1.Canvas.TextOut(x1+1, y2-1-Image1.Canvas.Font.Height, _text);
      _text := IntToStr(mytblDZinfoDZ_PRIB.Value) + '-' +
IntToStr(mytblDZinfoDZ_VITR.Value);
      Image1.Canvas.TextOut(x2-1-Image1.Canvas.TextWidth(_text), y2-1-
Image1.Canvas.Font.Height, _text);

      x1 := x1 + Trunc((x2 - x1 - circle_size) / 2);
      y1 := y1 + Trunc((y2 - y1 - circle_size) / 2);
      x2 := x1 + circle_size;
      y2 := y1 + circle_size;
      Image1.Canvas.Ellipse(x1, y1, x2, y2);
      diag := Trunc(((SQRT(2*circle_size*circle_size) - circle_size)/2) / SQRT(2));

      Image1.Canvas.MoveTo(x1+diag, y1+diag);
      Image1.Canvas.LineTo(x2-diag, y2-diag);
      Image1.Canvas.MoveTo(x2-diag, y1+diag);
      Image1.Canvas.LineTo(x1+diag, y2-diag);

      mytblDZinfo.Next;
    end;
    //mytblDZinfo.Filtered := False;
  end;
  // draw transfer elements
  with BimaDataModule do begin
    mytblTransfer.Filtered := False;
    mytblTransfer.Filter := 'TR_PROJECT_NO = ' + IntToStr(mytblProjectPROJECT_NO.Value);
    mytblTransfer.Filtered := True;
    mytblTransfer.First;
    Image1.Canvas.Font.Height := Trunc(circle_size / 2);
    Image1.Canvas.Font.Style := [fsBold, fsUnderline];
    while not mytblTransfer.Eof do begin
      if mytblTransferTR_DOC_NO1.Value = mytblTransferTR_DOC_NO2.Value then begin
        x1 := StartX + Trunc((mytblTransferTR_DEPT_NO1.Value * cwidth) + (cwidth / 2));
        y1 := StartY + Trunc((mytblTransferTR_DOC_NO1.Value * cheight) + (cheight / 2));
        x2 := StartX + Trunc((mytblTransferTR_DEPT_NO2.Value * cwidth) + (cwidth / 2));
        y2 := StartY + Trunc((mytblTransferTR_DOC_NO2.Value * cheight) + (cheight / 2));
        if mytblTransferTR_DEPT_NO1.Value < mytblTransferTR_DEPT_NO2.Value then begin
          x1 := x1 + Trunc(circle_size/2);
          x2 := x2 - Trunc(circle_size/2);
          Image1.Canvas.TextOut(x1+2, y1-2-Image1.Canvas.Font.Height,
IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2-4);
          Image1.Canvas.LineTo(x2-4, y2+4);
          Image1.Canvas.LineTo(x2, y2);
        end else if mytblTransferTR_DEPT_NO1.Value > mytblTransferTR_DEPT_NO2.Value then
begin
          x1 := x1 - Trunc(circle_size/2);
          x2 := x2 + Trunc(circle_size/2);
          Image1.Canvas.TextOut(x1-2-
Image1.Canvas.TextWidth(IntToStr(mytblTransferTR_TIME.Value)), y1-2-
Image1.Canvas.Font.Height, IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2+4, y2-4);
          Image1.Canvas.LineTo(x2+4, y2+4);
```

```
          Image1.Canvas.LineTo(x2, y2);
        end;
        Image1.Canvas.MoveTo(x1, y1);
        Image1.Canvas.LineTo(x2, y2);
      end else if mytblTransferTR_DEPT_NO1.Value = mytblTransferTR_DEPT_NO2.Value then
begin
        x1 := StartX + Trunc((mytblTransferTR_DEPT_NO1.Value * cwidth) + (cwidth / 2));
        y1 := StartY + Trunc((mytblTransferTR_DOC_NO1.Value * cheight) + (cheight / 2));
        x2 := StartX + Trunc((mytblTransferTR_DEPT_NO2.Value * cwidth) + (cwidth / 2));
        y2 := StartY + Trunc((mytblTransferTR_DOC_NO2.Value * cheight) + (cheight / 2));
        if mytblTransferTR_DOC_NO1.Value < mytblTransferTR_DOC_NO2.Value then begin
          y1 := y1 + Trunc(circle_size/2);
          y2 := y2 - Trunc(circle_size/2);
          Image1.Canvas.TextOut(x1-4-
Image1.Canvas.TextWidth(IntToStr(mytblTransferTR_TIME.Value)), y1+2,
IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2-4);
          Image1.Canvas.LineTo(x2+4, y2-4);
          Image1.Canvas.LineTo(x2, y2);
        end else if mytblTransferTR_DOC_NO1.Value > mytblTransferTR_DOC_NO2.Value then
begin
          y1 := y1 - Trunc(circle_size/2);
          y2 := y2 + Trunc(circle_size/2);
          Image1.Canvas.TextOut(x1-4-
Image1.Canvas.TextWidth(IntToStr(mytblTransferTR_TIME.Value)), y1-2-
Image1.Canvas.Font.Height, IntToStr(mytblTransferTR_TIME.Value));          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2+4);
          Image1.Canvas.LineTo(x2+4, y2+4);
          Image1.Canvas.LineTo(x2, y2);
        end;
        Image1.Canvas.MoveTo(x1, y1);
        Image1.Canvas.LineTo(x2, y2);
      end else if ((mytblTransferTR_DOC_NO1.Value < mytblTransferTR_DOC_NO2.Value) and
      (mytblTransferTR_DEPT_NO1.Value <> mytblTransferTR_DEPT_NO2.Value)) then begin
        x1 := StartX + Trunc((mytblTransferTR_DEPT_NO1.Value * cwidth) + (cwidth / 2));
        y1 := StartY + Trunc((mytblTransferTR_DOC_NO1.Value * cheight) + (cheight / 2));
        x2 := StartX + Trunc((mytblTransferTR_DEPT_NO2.Value * cwidth) + (cwidth / 2));
        y2 := StartY + Trunc((mytblTransferTR_DOC_NO2.Value * cheight) + (cheight / 2));
        if mytblTransferTR_DEPT_NO1.Value < mytblTransferTR_DEPT_NO2.Value then begin
          x1 := x1 + Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x1, y1);
          Image1.Canvas.LineTo(x2, y1);
          y2 := y2 - Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x2, y1);
          Image1.Canvas.LineTo(x2, y2);
          Image1.Canvas.TextOut(x1+2, y1-2-Image1.Canvas.Font.Height,
IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2-4);
          Image1.Canvas.LineTo(x2+4, y2-4);
          Image1.Canvas.LineTo(x2, y2);
        end else if mytblTransferTR_DEPT_NO1.Value > mytblTransferTR_DEPT_NO2.Value then
begin
          x1 := x1 - Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x1, y1);
          Image1.Canvas.LineTo(x2, y1);
          y2 := y2 - Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x2, y1);
          Image1.Canvas.LineTo(x2, y2);
          Image1.Canvas.TextOut(x1-2-
Image1.Canvas.TextWidth(IntToStr(mytblTransferTR_TIME.Value)), y1-2-
Image1.Canvas.Font.Height, IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2-4);
          Image1.Canvas.LineTo(x2+4, y2-4);
          Image1.Canvas.LineTo(x2, y2);
```

```
        end;
      end else if ((mytblTransferTR_DOC_NO1.Value > mytblTransferTR_DOC_NO2.Value) and
      (mytblTransferTR_DEPT_NO1.Value <> mytblTransferTR_DEPT_NO2.Value)) then begin
        x1 := StartX + Trunc((mytblTransferTR_DEPT_NO1.Value * cwidth) + (cwidth / 2));
        y1 := StartY + Trunc((mytblTransferTR_DOC_NO1.Value * cheight) + (cheight / 2));
        x2 := StartX + Trunc((mytblTransferTR_DEPT_NO2.Value * cwidth) + (cwidth / 2));
        y2 := StartY + Trunc((mytblTransferTR_DOC_NO2.Value * cheight) + (cheight / 2));
        if mytblTransferTR_DEPT_NO1.Value < mytblTransferTR_DEPT_NO2.Value then begin
          x1 := x1 + Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x1, y1);
          Image1.Canvas.LineTo(x2, y1);
          y2 := y2 + Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x2, y1);
          Image1.Canvas.LineTo(x2, y2);
          Image1.Canvas.TextOut(x1+2, y1-2-Image1.Canvas.Font.Height,
IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2+4);
          Image1.Canvas.LineTo(x2+4, y2+4);
          Image1.Canvas.LineTo(x2, y2);
        end else if mytblTransferTR_DEPT_NO1.Value > mytblTransferTR_DEPT_NO2.Value then
begin
          x1 := x1 - Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x1, y1);

          Image1.Canvas.LineTo(x2, y1);
          y2 := y2 + Trunc(circle_size/2);
          Image1.Canvas.MoveTo(x2, y1);
          Image1.Canvas.LineTo(x2, y2);
          Image1.Canvas.TextOut(x1-2-
Image1.Canvas.TextWidth(IntToStr(mytblTransferTR_TIME.Value)), y1-2-
Image1.Canvas.Font.Height, IntToStr(mytblTransferTR_TIME.Value));
          // arrow
          Image1.Canvas.MoveTo(x2, y2);
          Image1.Canvas.LineTo(x2-4, y2+4);
          Image1.Canvas.LineTo(x2+4, y2+4);
          Image1.Canvas.LineTo(x2, y2);
        end;
      end;
      mytblTransfer.Next;
    end;
    //mytblTransfer.Filtered := False;
  end;
end;

procedure TfmNewProject.acSaveImgToFileExecute(Sender: TObject);
begin
  SaveDialog1.DefaultExt := '.BMP';
  SaveDialog1.Filter := 'Bitmaps (*.bmp)|*.BMP';
  SaveDialog1.Options := [ofOverwritePrompt, ofFileMustExist, ofHideReadOnly ];
  if SaveDialog1.Execute then
  begin
    try
      Image1.Picture.SaveToFile(SaveDialog1.FileName);
    except
      MessageDlg('Íåìîæåòêâî çáåðåãòè ôàéë!', mtError, [mbOk], 0);
    end;
  end;
end;

procedure TfmNewProject.DrawGraphTree;
const
  rect_a : integer = 25;
  rect_b : integer = 50;
  dist_v : integer = 50;
  f_height : integer = 10;
var
  tmp_str: String;
  dz_qty : integer;
  dist_h : integer;
```

```
  rect_text : String;
  i, j, c : integer;
  x1, x2, y1, y2, x0, y0, x02, y02 : integer;
  doc_no, dept_no : integer;
begin
  Image1.Canvas.Font.Style := [];
  with BimaDataModule do begin
    with myqrDZinfo do begin
      Close;
      SQL.Clear;
      tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                  'from dz_info ' +
                  'where dz_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value);
      SQL.Add(tmp_str);
      Open;
      dz_qty := RecordCount;
    end;
    i := 0;
    while not myqrDZinfo.Eof do begin
      // ***
      Inc(i);
      rect_text := 'Ä' + IntToStr(myqrDZinfoDZ_DOC_NO.Value) + '.O' +
IntToStr(myqrDZinfoDZ_DEPT_NO.Value);
      x1 := Trunc(Image1.ClientWidth/dz_qty)*i - rect_b - 10;
      x2 := x1 + rect_b;
      y1 := 10;
      y2 := y1 + rect_a;
      Image1.Canvas.Rectangle(x1, y1, x2, y2);
      Image1.Canvas.Font.Height := f_height;
      Image1.Canvas.TextOut( x1+5, y1+5, rect_text);
      x0 := Trunc((x1+x2)/2);
      y0 := y2;
      // ***
      myqrTransfer.Close;
      myqrTransfer.SQL.Clear;
      tmp_str :=  'select tr_project_no, tr_doc_no1, tr_dept_no1, tr_doc_no2, tr_dept_no2,
tr_time ' +
                  'from transfer_info ' +
                  'where tr_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' and
' +
                  'tr_doc_no2 = ' + IntToStr(myqrDZinfoDZ_DOC_NO.Value) + ' and ' +
                  'tr_dept_no2 = ' + IntToStr(myqrDZinfoDZ_DEPT_NO.Value);
      myqrTransfer.SQL.Add(tmp_str);
      myqrTransfer.Open;
      j := 0;
      c := 0;
      while not myqrTransfer.Eof do begin
        Inc(j);
        rect_text := 'Ä' + IntToStr(myqrTransferTR_DOC_NO1.Value) + '.O' +
IntToStr(myqrTransferTR_DEPT_NO1.Value);
        x1 := Trunc(Image1.ClientWidth/dz_qty)*i - (rect_b*j) - ((j-1)*10) - 10;
        x2 := x1 + rect_b;
        y1 := 10 + rect_a + dist_v;
        y2 := y1 + rect_a;
        Image1.Canvas.Rectangle(x1, y1, x2, y2);
        Image1.Canvas.Font.Height := f_height;
        Image1.Canvas.TextOut( x1+5, y1+5, rect_text);
        Image1.Canvas.MoveTo(Trunc((x1+x2)/2), y1);
        Image1.Canvas.LineTo(x0, y0);
        doc_no := myqrTransferTR_DOC_NO1.Value;
        dept_no := myqrTransferTR_DEPT_NO1.Value;
        x02 := Trunc((x1+x2)/2);
        y02 := y2;
        // ***
        myqrTransfer2.Close;
        myqrTransfer2.SQL.Clear;
        tmp_str :=  'select tr_project_no, tr_doc_no1, tr_dept_no1, tr_doc_no2,
tr_dept_no2, tr_time ' +
                    'from transfer_info ' +
```

```
                    'where tr_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + '
and ' +
                    'tr_doc_no2 = ' + IntToStr(doc_no) + ' and ' +
                    'tr_dept_no2 = ' + IntToStr(dept_no);
        myqrTransfer2.SQL.Add(tmp_str);
        myqrTransfer2.Open;
        //c := 0;
        while not myqrTransfer2.Eof do begin
          Inc(c);
          rect_text := 'Ä' + IntToStr(myqrTransfer2TR_DOC_NO1.Value) + '.O' +
IntToStr(myqrTransfer2TR_DEPT_NO1.Value);
          x1 := Trunc(Image1.ClientWidth/dz_qty)*i - (rect_b*c) - ((c-1)*10) - 10;
          x2 := x1 + rect_b;
          y1 := 10 + (2* rect_a) + (2 * dist_v);
          y2 := y1 + rect_a;
          Image1.Canvas.Rectangle(x1, y1, x2, y2);
          Image1.Canvas.Font.Height := f_height;
          Image1.Canvas.TextOut( x1+5, y1+5, rect_text);
          Image1.Canvas.MoveTo(Trunc((x1+x2)/2), y1);
          Image1.Canvas.LineTo(x02, y02);
          myqrTransfer2.Next;
        end;
        myqrTransfer.Next;
      end;
      myqrDZinfo.Next;
    end;
  end;
end;

procedure TfmNewProject.DrawParamTime;
const
  rect_y : integer = 20;
  rect_x : integer = 20;
  dist_h : integer = 30;
  top_dist : integer = 150;
  left_dist : integer = 20;
  f_height : integer = 10;
  circle_size : integer = 40;
var
  tmp_str : String;
  x1, y1, x2, y2 : integer;
  shift : integer;
  rect_text : String;
  tmp_var : integer;
begin
  Image1.Canvas.Font.Style := [];
  with BimaDataModule do begin
    with myqrDIinfo do begin
      Close;
      SQL.Clear;
      tmp_str :=  'select di_project_no, di_doc_no, di_dept_no, di_start_time,
di_form_time, di_top, di_prib, di_vitr ' +
                  'from di_info ' +
                  'where di_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' ' +
                  'order by di_start_time';
      SQL.Add(tmp_str);
      Open;
    end;
    shift := 0;
    // Draw DI symbol
    Image1.Canvas.Ellipse(left_dist, top_dist - f_height - 40 - Circle_size, left_dist +
circle_size, top_dist - f_height - 40);
    Image1.Canvas.Brush.Color := clBlack;
    Image1.Canvas.Brush.Style := bsSolid;
    Image1.Canvas.Ellipse(left_dist + 10, top_dist - f_height - 30 - Circle_size, left_dist
+ circle_size - 10, top_dist - f_height - 50);
    Image1.Canvas.Brush.Color := clWhite;

    while not myqrDIinfo.Eof do begin
      x1 := left_dist + shift;
      x2 := x1 + Trunc(rect_x * myqrDIinfoDI_FORM_TIME.Value);
```

```
      y1 := top_dist;
      y2 := top_dist + rect_y;
      rect_text := 'Ä' + IntToStr(myqrDIinfoDI_DOC_NO.Value) + '.O' +
IntToStr(myqrDIinfoDI_DEPT_NO.Value);
      Image1.Canvas.Font.Height := f_height;
      Image1.Canvas.TextOut( x1, y1 - f_height - 5, rect_text);
      Image1.Canvas.Brush.Color := clBlack;
      Image1.Canvas.Brush.Style := bsBDiagonal;
      Image1.Canvas.Rectangle(x1, y1, x2, y2);
      shift := x2 + dist_h;
      Image1.Canvas.Brush.Color := clWhite;
      myqrDIinfo.Next;
    end;
    Image1.Canvas.MoveTo(left_dist, y2 + 10);
    Image1.Canvas.Pen.Style := psDash;
    Image1.Canvas.LineTo(Image1.ClientWidth - 20, y2 + 10);
    Image1.Canvas.Pen.Style := psSolid;
    // ***
    with myqrODinfo do begin
      Close;
      SQL.Clear;
      tmp_str :=  'select od_project_no, od_doc_no, od_dept_no, od_start_time,
od_form_time, od_top, od_prib, od_vitr ' +
                  'from od_info ' +
                  'where od_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' ' +
                  'order by od_start_time';
      SQL.Add(tmp_str);
      Open;
    end;
    shift := 0;
    // Draw OD symbol
    Image1.Canvas.Ellipse(left_dist, (2*top_dist) - f_height - 40 - Circle_size, left_dist
+ circle_size, (2*top_dist) - f_height - 40);

    while not myqrODinfo.Eof do begin
      x1 := left_dist + shift;
      x2 := x1 + Trunc(rect_x * myqrODinfoOD_FORM_TIME.Value);
      y1 := 2 * top_dist;
      y2 := (2 * top_dist) + rect_y;
      rect_text := 'Ä' + IntToStr(myqrODinfoOD_DOC_NO.Value) + '.O' +
IntToStr(myqrODinfoOD_DEPT_NO.Value);
      Image1.Canvas.Font.Height := f_height;
      Image1.Canvas.TextOut( x1, y1 - f_height - 5, rect_text);
      Image1.Canvas.Brush.Color := clBlack;
      Image1.Canvas.Brush.Style := bsBDiagonal;
      Image1.Canvas.Rectangle(x1, y1, x2, y2);
      shift := x2 + dist_h;
      Image1.Canvas.Brush.Color := clWhite;
      myqrODinfo.Next;
    end;
    Image1.Canvas.MoveTo(left_dist, y2 + 10);
    Image1.Canvas.Pen.Style := psDash;
    Image1.Canvas.LineTo(Image1.ClientWidth - 20, y2 + 10);
    Image1.Canvas.Pen.Style := psSolid;
    // ***
    with myqrDZinfo do begin
      Close;
      SQL.Clear;
      tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                  'from dz_info ' +
                  'where dz_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' ' +
                  'order by dz_start_time';
      SQL.Add(tmp_str);
      Open;
    end;
    shift := 0;
    // Draw DZ symbol
    Image1.Canvas.Ellipse(left_dist, (3*top_dist) - f_height - 40 - Circle_size, left_dist
+ circle_size, (3*top_dist) - f_height - 40);
    tmp_var := Trunc((circle_size / 2) * (SQRT(2) - 1));
```

```
    Image1.Canvas.MoveTo(left_dist + tmp_var, (3*top_dist) - f_height - 40 - Circle_size +
tmp_var);
    Image1.Canvas.LineTo(left_dist + circle_size - tmp_var, (3*top_dist) - f_height - 40 -
tmp_var);
    Image1.Canvas.MoveTo(left_dist + circle_size - tmp_var, (3*top_dist) - f_height - 40 -
Circle_size + tmp_var);
    Image1.Canvas.LineTo(left_dist + tmp_var, (3*top_dist) - f_height - 40 - tmp_var);

    while not myqrDZinfo.Eof do begin
      x1 := left_dist + shift;
      x2 := x1 + Trunc(rect_x * myqrDZinfoDZ_FORM_TIME.Value);
      y1 := 3 * top_dist;
      y2 := (3 * top_dist) + rect_y;
      rect_text := 'Ä' + IntToStr(myqrDZinfoDZ_DOC_NO.Value) + '.O' +
IntToStr(myqrDZinfoDZ_DEPT_NO.Value);
      Image1.Canvas.Font.Height := f_height;
      Image1.Canvas.TextOut( x1, y1 - f_height - 5, rect_text);
      Image1.Canvas.Brush.Color := clBlack;
      Image1.Canvas.Brush.Style := bsBDiagonal;
      Image1.Canvas.Rectangle(x1, y1, x2, y2);
      shift := x2 + dist_h;
      Image1.Canvas.Brush.Color := clWhite;
      myqrDZinfo.Next;
    end;
    Image1.Canvas.MoveTo(left_dist, y2 + 10);
    Image1.Canvas.Pen.Style := psDash;
    Image1.Canvas.LineTo(Image1.ClientWidth - 20, y2 + 10);
    Image1.Canvas.Pen.Style := psSolid;
    // ***
    rect_text := 'Ìàñøòàá: ';
    Image1.Canvas.Font.Height := 20;
    Image1.Canvas.TextOut( left_dist, y2 + 40, rect_text);
    x1 := Image1.Canvas.TextWidth(rect_text) + 20;
    x2 := x1 + rect_x;
    y1 := y2 + 40;
    y2 := y1 + rect_y;
    Image1.Canvas.Brush.Color := clBlack;
    Image1.Canvas.Brush.Style := bsBDiagonal;
    Image1.Canvas.Rectangle(x1, y1, x2, y2);
    Image1.Canvas.Brush.Color := clWhite;
    rect_text := ' - 1 ' + mytblProjectPROJECT_TIME_UOM.Value;
    Image1.Canvas.TextOut( x2 + 10, y1, rect_text);


  end;
end;

procedure TfmNewProject.DrawStructTime;
const
  rect_a : integer = 25;
  rect_b : integer = 50;
  dist_v : integer = 125;
  f_height : integer = 8;
  left_dist : integer = 20;
  circle_size : integer = 40;
var
  tmp_str : String;
  tmp_var : integer;
  dz_qty : integer;
  dist_h : integer;
  rect_text : String;
  i, j, c : integer;
  x1, x2, y1, y2, x0, y0, x02, y02 : integer;
  doc_no, dept_no : integer;
  h_qty : integer;
  top_dist : integer;
begin
  Image1.Canvas.Font.Style := [];
  with BimaDataModule do begin
    with myqrDZinfo do begin
      Close;
```

```
      SQL.Clear;
      tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                  'from dz_info ' +
                  'where dz_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value);
      SQL.Add(tmp_str);
      Open;
      dz_qty := RecordCount;
    end;
    i := 0;
    while not myqrDZinfo.Eof do begin
      // ***
      Inc(i);
      rect_text := 'Ä' + IntToStr(myqrDZinfoDZ_DOC_NO.Value) + '.O' +
IntToStr(myqrDZinfoDZ_DEPT_NO.Value);
      x1 := Trunc(Image1.ClientWidth/dz_qty)*i - rect_b - 10;
      x2 := x1 + rect_b;
      y2 := 100;
      y1 := y2 - (rect_a * myqrDZinfoDZ_FORM_TIME.Value);
      Image1.Canvas.Rectangle(x1, y1, x2, y2);
      Image1.Canvas.Font.Height := f_height;
      Image1.Canvas.TextOut( x1+5, y1+5, rect_text);
      x0 := Trunc((x1+x2)/2);
      y0 := y2;
      // Draw DZ symbol
      Image1.Canvas.Ellipse(left_dist, y2 - Circle_size, left_dist + circle_size, y2);
      tmp_var := Trunc((circle_size / 2) * (SQRT(2) - 1));
      Image1.Canvas.MoveTo(left_dist + tmp_var, y2 - Circle_size + tmp_var);
      Image1.Canvas.LineTo(left_dist + circle_size - tmp_var, y2 - tmp_var);
      Image1.Canvas.MoveTo(left_dist + circle_size - tmp_var, y2 - Circle_size + tmp_var);
      Image1.Canvas.LineTo(left_dist + tmp_var, y2 - tmp_var);

      // ***
      myqrTransfer.Close;
      myqrTransfer.SQL.Clear;
      tmp_str :=  'select tr_project_no, tr_doc_no1, tr_dept_no1, tr_doc_no2, tr_dept_no2,
tr_time ' +
                  'from transfer_info ' +
                  'where tr_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' and
' +
                  'tr_doc_no2 = ' + IntToStr(myqrDZinfoDZ_DOC_NO.Value) + ' and ' +
                  'tr_dept_no2 = ' + IntToStr(myqrDZinfoDZ_DEPT_NO.Value);
      myqrTransfer.SQL.Add(tmp_str);
      myqrTransfer.Open;
      j := 0;
      c := 0;
      while not myqrTransfer.Eof do begin
        Inc(j);
        rect_text := 'Ä' + IntToStr(myqrTransferTR_DOC_NO1.Value) + '.O' +
IntToStr(myqrTransferTR_DEPT_NO1.Value);
        x1 := Trunc(Image1.ClientWidth/dz_qty)*i - (rect_b*j) - ((j-1)*10) - 10;

        x2 := x1 + rect_b;
        y2 := 250;
        y1 :=  y2 - (rect_a * ReturnFormTime(mytblProjectPROJECT_NO.Value,
myqrTransferTR_DOC_NO1.Value, myqrTransferTR_DEPT_NO1.Value, 'od')); //dist_v;
//        y2 := y1 + rect_a;
        Image1.Canvas.Rectangle(x1, y1, x2, y2);
        Image1.Canvas.Font.Height := f_height;
        Image1.Canvas.TextOut( x1+5, y1+5, rect_text);
        Image1.Canvas.MoveTo(Trunc((x1+x2)/2), y1);
        Image1.Canvas.LineTo(x0, y0);
        doc_no := myqrTransferTR_DOC_NO1.Value;
        dept_no := myqrTransferTR_DEPT_NO1.Value;
        x02 := Trunc((x1+x2)/2);
        y02 := y2;
        // Draw OD symbol
        Image1.Canvas.Ellipse(left_dist, y2 - Circle_size, left_dist + circle_size, y2);
        // ***
        myqrTransfer2.Close;
        myqrTransfer2.SQL.Clear;
```

```
        tmp_str :=  'select tr_project_no, tr_doc_no1, tr_dept_no1, tr_doc_no2,
tr_dept_no2, tr_time ' +
                    'from transfer_info ' +
                    'where tr_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + '
and ' +
                    'tr_doc_no2 = ' + IntToStr(doc_no) + ' and ' +
                    'tr_dept_no2 = ' + IntToStr(dept_no);
        myqrTransfer2.SQL.Add(tmp_str);
        myqrTransfer2.Open;
        //c := 0;
        while not myqrTransfer2.Eof do begin
          Inc(c);
          rect_text := 'Ä' + IntToStr(myqrTransfer2TR_DOC_NO1.Value) + '.O' +
IntToStr(myqrTransfer2TR_DEPT_NO1.Value);
          x1 := Trunc(Image1.ClientWidth/dz_qty)*i - (rect_b*c) - ((c-1)*10) - 10;
          x2 := x1 + rect_b;
          y2 := 400;
          y1 := y2 - (rect_a * ReturnFormTime(mytblProjectPROJECT_NO.Value,
myqrTransfer2TR_DOC_NO1.Value, myqrTransfer2TR_DEPT_NO1.Value, 'di'));
          //y1 := 10 + (2* rect_a) + (2 * dist_v);
          //y2 := y1 + rect_a;
          Image1.Canvas.Rectangle(x1, y1, x2, y2);
          Image1.Canvas.Font.Height := f_height;
          Image1.Canvas.TextOut( x1+5, y1+5, rect_text);
          Image1.Canvas.MoveTo(Trunc((x1+x2)/2), y1);
          Image1.Canvas.LineTo(x02, y02);
          // Draw DI symbol
          Image1.Canvas.Ellipse(left_dist, y2 - Circle_size, left_dist + circle_size, y2);
          Image1.Canvas.Brush.Color := clBlack;
          Image1.Canvas.Brush.Style := bsSolid;
          Image1.Canvas.Ellipse(left_dist + 10, y2 - Circle_size + 10, left_dist +
circle_size - 10, y2 - 10);
          Image1.Canvas.Brush.Color := clWhite;

          myqrTransfer2.Next;
        end;
        myqrTransfer.Next;
      end;
      myqrDZinfo.Next;
    end;
    // ***
    rect_text := 'Ìàñøòàá: ';
    Image1.Canvas.Font.Height := 20;
    Image1.Canvas.TextOut( left_dist, y2 + 40, rect_text);
    x1 := Image1.Canvas.TextWidth(rect_text) + 20;
    x2 := x1 + rect_b;
    y1 := y2 + 40;
    y2 := y1 + rect_a;
//    Image1.Canvas.Brush.Color := clBlack;
//    Image1.Canvas.Brush.Style := bsBDiagonal;
    Image1.Canvas.Rectangle(x1, y1, x2, y2);
//    Image1.Canvas.Brush.Color := clWhite;
    rect_text := ' - 1 ' + mytblProjectPROJECT_TIME_UOM.Value;
    Image1.Canvas.TextOut( x2 + 10, y1, rect_text);

  end;
end;
function TfmNewProject.ReturnFormTime(AProjectNo, ADocNo, ADeptNo: integer;
  ADb: String): integer;
var
  tmp_str : String;
begin
  with BimaDataModule do begin
    if ADb = 'di' then begin
      with myqrDIinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select di_project_no, di_doc_no, di_dept_no, di_start_time,
di_form_time, di_top, di_prib, di_vitr ' +
                    'from di_info ' +
                    'where di_project_no = ' + IntToStr(AProjectNo) +
```

```
                    ' and di_doc_no = ' + IntToStr(ADocNo) +
                    ' and di_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDIinfo.RecordCount > 0 then result := myqrDIinfoDI_FORM_TIME.Value else result
:= 0;
    end else
    if ADb = 'od' then begin
      with myqrODinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select od_project_no, od_doc_no, od_dept_no, od_start_time,
od_form_time, od_top, od_prib, od_vitr ' +
                    'from od_info ' +
                    'where od_project_no = ' + IntToStr(AProjectNo) +
                    ' and od_doc_no = ' + IntToStr(ADocNo) +
                    ' and od_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrODinfo.RecordCount > 0 then result := myqrODinfoOD_FORM_TIME.Value else result
:= 0;
    end else
    if ADb = 'dz' then begin
      with myqrDZinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                    'from dz_info ' +
                    'where dz_project_no = ' + IntToStr(AProjectNo) +
                    ' and dz_doc_no = ' + IntToStr(ADocNo) +
                    ' and dz_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDZinfo.RecordCount > 0 then result := myqrDZinfoDZ_FORM_TIME.Value else result
:= 0;
    end;
  end;
end;

procedure TfmNewProject.DrawEpureDif;
const
  step : integer = 10;
  x_qty : integer = 29;
  y_qty : integer = 50;
  start_x : integer = 50;
  start_y : integer = 300;
var
  tmp_str, full_str : String;
  delimiter : Char;
  DataEl : TStringList;
  pos_d, pos1, pos2 : integer;
  i : integer;
  doc_no, dept_no : integer;
begin
  delimiter := '-';
  full_str := ComboBox2.Items[ComboBox2.ItemIndex];
  with BimaDataModule do begin
    if ((ComboBox2.Items.Count > 0) and (ComboBox2.ItemIndex > -1)) then begin
    DataEl := TStringList.Create;
    try
      while Length(full_str) > 0 do begin
        pos_d := Pos(delimiter, full_str);
        if pos_d > 0 then begin
          tmp_str := Copy(full_str, 1, pos_d - 1);
          DataEl.Add(tmp_str);
          full_str := Copy(full_str, pos_d+1, Length(full_str) - pos_d);
        end else begin
```

```
          DataEl.Add(full_str);
          full_str := '';
        end;
      end;
      pos1 := Pos('Ä', DataEl[0]);
      pos2 := Pos('.', DataEl[0]);
      doc_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1-1));
      pos1 := Pos('O', DataEl[0]);
      pos2 := Length(DataEl[0]);
      dept_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1));
      mytblDataCycle.Filtered := False;
      mytblDataCycle.Filter := 'DC_PROJECT_NO = ' + IntToStr(mytblProjectPROJECT_NO.Value)
+ ' AND DC_DOC_DI = ' + IntToStr(doc_no) + ' AND DC_DEPT_DI = ' + IntToStr(dept_no);
      mytblDataCycle.Filtered := True;
      // DrawGrid
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x + (step * y_qty), start_y);
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y - (step * x_qty));
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y + (step * x_qty));
      Image1.Canvas.Font.Height := 8;
      Image1.Canvas.Font.Style := [];
      for i := 1 to y_qty do begin
        Image1.Canvas.MoveTo(start_x + (i * step), start_y - 5);
        Image1.Canvas.LineTo(start_x + (i * step), start_y + 5);
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x + (i * step), start_y + 10, IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + (step * y_qty)+10, start_y + 15, 'T');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y - (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y - (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y - (step * i), IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + 10, start_y - (step * x_qty), 'P');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y + (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y + (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y + (step * i), IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + 10, start_y + (step * x_qty), 'V');
      // DrawEpur
      Image1.Canvas.Brush.Color := clBlack;
      Image1.Canvas.Brush.Style := bsBDiagonal;
      Image1.Canvas.Rectangle(start_x + (mytblDataCycleDC_START_DI.Value * step), start_y,
                              start_x + (mytblDataCycleDC_START_DI.Value * step) +
(mytblDataCycleDC_FORM_DI.Value * step), start_y - ((mytblDataCycleDC_PRIB_DI.Value -
mytblDataCycleDC_VITR_DI.Value) * step));
      Image1.Canvas.Rectangle(start_x + (mytblDataCycleDC_START_OD.Value * step), start_y,
                              start_x + (mytblDataCycleDC_START_OD.Value * step) +
(mytblDataCycleDC_FORM_OD.Value * step), start_y - ((mytblDataCycleDC_PRIB_OD.Value -
mytblDataCycleDC_VITR_OD.Value) * step));
      Image1.Canvas.Rectangle(start_x + (mytblDataCycleDC_START_DZ.Value * step), start_y,
                              start_x + (mytblDataCycleDC_START_DZ.Value * step) +
(mytblDataCycleDC_FORM_DZ.Value * step), start_y - ((mytblDataCycleDC_PRIB_DZ.Value -
mytblDataCycleDC_VITR_DZ.Value) * step));
      Image1.Canvas.Brush.Color := clWhite;

      mytblDataCycle.Filtered := False;
    finally
      DataEl.Free;
    end;
    end else MessageDlg('Óâàãà!!! Íå ñôîðìîâàíî öèêëè ðóõó äàíèõ àáî íå âèáðàíî öèêëó.',
mtWarning, [mbOk], 0);
  end;
end;

procedure TfmNewProject.DrawEpureInt;
```

```
const
  step : integer = 10;
  x_qty : integer = 29;
  y_qty : integer = 50;
  start_x : integer = 50;
  start_y : integer = 300;
var
  tmp_str, full_str : String;
  delimiter : Char;
  DataEl : TStringList;
  pos_d, pos1, pos2 : integer;
  i : integer;
  doc_no, dept_no : integer;
  x, y : integer;
begin
  delimiter := '-';
  full_str := ComboBox2.Items[ComboBox2.ItemIndex];
  with BimaDataModule do begin
    if ((ComboBox2.Items.Count > 0) and (ComboBox2.ItemIndex > -1)) then begin
    DataEl := TStringList.Create;
    try
      while Length(full_str) > 0 do begin
        pos_d := Pos(delimiter, full_str);
        if pos_d > 0 then begin
          tmp_str := Copy(full_str, 1, pos_d - 1);
          DataEl.Add(tmp_str);
          full_str := Copy(full_str, pos_d+1, Length(full_str) - pos_d);
        end else begin
          DataEl.Add(full_str);
          full_str := '';
        end;
      end;
      pos1 := Pos('Ä', DataEl[0]);
      pos2 := Pos('.', DataEl[0]);
      doc_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1-1));
      pos1 := Pos('O', DataEl[0]);
      pos2 := Length(DataEl[0]);
      dept_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1));
      mytblDataCycle.Filtered := False;
      mytblDataCycle.Filter := 'DC_PROJECT_NO = ' + IntToStr(mytblProjectPROJECT_NO.Value)
+ ' AND DC_DOC_DI = ' + IntToStr(doc_no) + ' AND DC_DEPT_DI = ' + IntToStr(dept_no);
      mytblDataCycle.Filtered := True;
      // DrawGrid
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x + (step * y_qty), start_y);
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y - (step * x_qty));
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y + (step * x_qty));
      Image1.Canvas.Font.Height := 8;
      Image1.Canvas.Font.Style := [];
      for i := 1 to y_qty do begin
        Image1.Canvas.MoveTo(start_x + (i * step), start_y - 5);
        Image1.Canvas.LineTo(start_x + (i * step), start_y + 5);
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x + (i * step), start_y + 10, IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + (step * y_qty)+10, start_y + 15, 'T');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y - (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y - (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y - (step * i), IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + 10, start_y - (step * x_qty), 'P');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y + (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y + (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y + (step * i), IntToStr(i));
      end;
```

```
      Image1.Canvas.TextOut(start_x + 10, start_y + (step * x_qty), 'V');
      // DrawEpur
      x := start_x + (mytblDataCycleDC_START_DI.Value * step);
      y := start_y;
      Image1.Canvas.MoveTo(x, y);
      x := x + (mytblDataCycleDC_FORM_DI.Value * step);
      y := y - ((mytblDataCycleDC_PRIB_DI.Value - mytblDataCycleDC_VITR_DI.Value) * step);
      Image1.Canvas.LineTo(x, y);
      x := start_x + (mytblDataCycleDC_START_OD.Value * step);
      Image1.Canvas.LineTo(x, y);
      x := x + (mytblDataCycleDC_FORM_OD.Value * step);
      y := y - ((mytblDataCycleDC_PRIB_OD.Value - mytblDataCycleDC_VITR_OD.Value) * step);
      Image1.Canvas.LineTo(x, y);
      x := start_x + (mytblDataCycleDC_START_DZ.Value * step);
      Image1.Canvas.LineTo(x, y);
      x := x + (mytblDataCycleDC_FORM_DZ.Value * step);
      y := y - ((mytblDataCycleDC_PRIB_DZ.Value - mytblDataCycleDC_VITR_DZ.Value) * step);
      Image1.Canvas.LineTo(x, y);

      mytblDataCycle.Filtered := False;
    finally
      DataEl.Free;
    end;
    end else MessageDlg('Óâàãà!!! Íå ñôîðìîâàíî öèêëè ðóõó äàíèõ àáî íå âèáðàíî öèêëó.',
mtWarning, [mbOk], 0);
  end;
end;


procedure TfmNewProject.DrawTotalEpureInt;
const
  step : integer = 10;
  x_qty : integer = 29;
  y_qty : integer = 50;
  start_x : integer = 50;
  start_y : integer = 300;
var
  tmp_str, full_str : String;
  delimiter : Char;
  DataEl : TStringList;
  pos_d, pos1, pos2 : integer;
  i : integer;
  doc_no, dept_no : integer;
  x, y : integer;
  sql_str : String;
begin
  delimiter := '-';
  ComboBox2.ItemIndex := 0;
  full_str := ComboBox2.Items[ComboBox2.ItemIndex];
  with BimaDataModule do begin
    if (ComboBox2.Items.Count > 0) then begin
    DataEl := TStringList.Create;
    try
      while Length(full_str) > 0 do begin
        pos_d := Pos(delimiter, full_str);
        if pos_d > 0 then begin
          tmp_str := Copy(full_str, 1, pos_d - 1);
          DataEl.Add(tmp_str);
          full_str := Copy(full_str, pos_d+1, Length(full_str) - pos_d);
        end else begin
          DataEl.Add(full_str);
          full_str := '';
        end;
      end;
      pos1 := Pos('Ä', DataEl[0]);
      pos2 := Pos('.', DataEl[0]);
      doc_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1-1));
      pos1 := Pos('O', DataEl[0]);
      pos2 := Length(DataEl[0]);
      dept_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1));
      mytblDataCycle.Filtered := False;
```

```
      mytblDataCycle.First;
      // DrawGrid
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x + (step * y_qty), start_y);
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y - (step * x_qty));
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y + (step * x_qty));
      Image1.Canvas.Font.Height := 8;
      Image1.Canvas.Font.Style := [];
      for i := 1 to y_qty do begin
        Image1.Canvas.MoveTo(start_x + (i * step), start_y - 5);
        Image1.Canvas.LineTo(start_x + (i * step), start_y + 5);
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x + (i * step), start_y + 10, IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + (step * y_qty)+10, start_y + 15, 'T');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y - (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y - (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y - (step * i), IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + 10, start_y - (step * x_qty), 'P');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y + (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y + (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y + (step * i), IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + 10, start_y + (step * x_qty), 'V');
      // DrawEpur
      with myqrDataCycle do begin
        Close;
        SQL.Clear;
        sql_str :=  'select dc_project_no, ' +
                    'dc_doc_di, dc_dept_di, dc_start_di, dc_form_di, dc_prib_di,
dc_vitr_di, ' +
                    'dc_doc_od, dc_dept_od, dc_start_od, dc_form_od, dc_prib_od,
dc_vitr_od, ' +
                    'dc_doc_dz, dc_dept_dz, dc_start_dz, dc_form_dz, dc_prib_dz, dc_vitr_dz
' +
                    'from data_cycle ' +
                    'where dc_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' '
+
                    'order by dc_start_di';
        SQL.Add(sql_str);
        Open;
      end;
      x := start_x + (myqrDataCycleDC_START_DI.Value * step);
      y := start_y;
      Image1.Canvas.MoveTo(x, y);
      while not myqrDataCycle.Eof do begin
        x := x + (myqrDataCycleDC_FORM_DI.Value * step);
        y := y - ((myqrDataCycleDC_PRIB_DI.Value - myqrDataCycleDC_VITR_DI.Value) * step);
        Image1.Canvas.LineTo(x, y);
        myqrDataCycle.Next;
      end;
      with myqrDataCycle do begin
        Close;
        SQL.Clear;
        sql_str :=  'select dc_project_no, ' +
                    'dc_doc_di, dc_dept_di, dc_start_di, dc_form_di, dc_prib_di,
dc_vitr_di, ' +
                    'dc_doc_od, dc_dept_od, dc_start_od, dc_form_od, dc_prib_od,
dc_vitr_od, ' +
                    'dc_doc_dz, dc_dept_dz, dc_start_dz, dc_form_dz, dc_prib_dz, dc_vitr_dz
' +
                    'from data_cycle ' +
                    'where dc_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' '
+
```

```
                    'order by dc_start_od';
        SQL.Add(sql_str);
        Open;
      end;
//      x := start_x + (myqrDataCycleDC_START_OD.Value * step);
//      y := start_y;
      Image1.Canvas.MoveTo(x, y);
      while not myqrDataCycle.Eof do begin
        x := x + (myqrDataCycleDC_FORM_OD.Value * step);
        y := y - ((myqrDataCycleDC_PRIB_OD.Value - myqrDataCycleDC_VITR_OD.Value) * step);
        Image1.Canvas.LineTo(x, y);
        myqrDataCycle.Next;
      end;
      with myqrDataCycle do begin
        Close;
        SQL.Clear;
        sql_str :=  'select dc_project_no, ' +
                    'dc_doc_di, dc_dept_di, dc_start_di, dc_form_di, dc_prib_di,
dc_vitr_di, ' +
                    'dc_doc_od, dc_dept_od, dc_start_od, dc_form_od, dc_prib_od,
dc_vitr_od, ' +
                    'dc_doc_dz, dc_dept_dz, dc_start_dz, dc_form_dz, dc_prib_dz, dc_vitr_dz
' +
                    'from data_cycle ' +
                    'where dc_project_no = ' + IntToStr(mytblProjectPROJECT_NO.Value) + ' '
+
                    'order by dc_start_dz';
        SQL.Add(sql_str);
        Open;
      end;
//      x := start_x + (myqrDataCycleDC_START_DZ.Value * step);
//      y := start_y;
      Image1.Canvas.MoveTo(x, y);
      while not myqrDataCycle.Eof do begin
        x := x + (myqrDataCycleDC_FORM_DZ.Value * step);
        y := y - ((myqrDataCycleDC_PRIB_DZ.Value - myqrDataCycleDC_VITR_DZ.Value) * step);
        Image1.Canvas.LineTo(x, y);
        myqrDataCycle.Next;
      end;


    finally
      DataEl.Free;
    end;
    end else MessageDlg(mtWarning, [mbOk], 0);
  end;
end;

procedure TfmNewProject.DrawTotalEpureSb;
const
  step : integer = 10;
  x_qty : integer = 29;
  y_qty : integer = 50;
  start_x : integer = 50;
  start_y : integer = 300;
var
  tmp_str, full_str : String;
  delimiter : Char;
  DataEl : TStringList;
  pos_d, pos1, pos2 : integer;
  i : integer;
  doc_no, dept_no : integer;
begin
  delimiter := '-';
  ComboBox2.ItemIndex := 0;
  full_str := ComboBox2.Items[ComboBox2.ItemIndex];
  with BimaDataModule do begin
    if (ComboBox2.Items.Count > 0) then begin
    DataEl := TStringList.Create;
    try
```

```
      while Length(full_str) > 0 do begin

        pos_d := Pos(delimiter, full_str);
        if pos_d > 0 then begin
          tmp_str := Copy(full_str, 1, pos_d - 1);
          DataEl.Add(tmp_str);
          full_str := Copy(full_str, pos_d+1, Length(full_str) - pos_d);
        end else begin
          DataEl.Add(full_str);
          full_str := '';
        end;
      end;
      pos1 := Pos('Ä', DataEl[0]);
      pos2 := Pos('.', DataEl[0]);
      doc_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1-1));
      pos1 := Pos('O', DataEl[0]);
      pos2 := Length(DataEl[0]);
      dept_no := StrToInt(Copy(DataEl[0], pos1+1, pos2-pos1));
      mytblDataCycle.Filtered := False;
      mytblDataCycle.First;
      // DrawGrid
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x + (step * y_qty), start_y);
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y - (step * x_qty));
      Image1.Canvas.MoveTo(start_x, start_y);
      Image1.Canvas.LineTo(start_x, start_y + (step * x_qty));
      Image1.Canvas.Font.Height := 8;
      Image1.Canvas.Font.Style := [];
      for i := 1 to y_qty do begin
        Image1.Canvas.MoveTo(start_x + (i * step), start_y - 5);
        Image1.Canvas.LineTo(start_x + (i * step), start_y + 5);
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x + (i * step), start_y + 10, IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + (step * y_qty)+10, start_y + 15, 'T');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y - (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y - (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y - (step * i), IntToStr(i));
      end;

      Image1.Canvas.TextOut(start_x + 10, start_y - (step * x_qty), 'P');
      for i := 1 to x_qty - 1 do begin
        Image1.Canvas.MoveTo(start_x - 5, start_y + (i * step));
        Image1.Canvas.LineTo(start_x + 5, start_y + (i * step));
        if (Frac(i/10) = 0) then
          Image1.Canvas.TextOut(start_x - 20, start_y + (step * i), IntToStr(i));
      end;
      Image1.Canvas.TextOut(start_x + 10, start_y + (step * x_qty), 'V');
      // DrawEpur
      while not mytblDataCycle.Eof do begin
        Image1.Canvas.Brush.Color := clBlack;
        if Image1.Canvas.Brush.Style = bsBDiagonal then Image1.Canvas.Brush.Style :=
bsFDiagonal else Image1.Canvas.Brush.Style := bsBDiagonal;
        Image1.Canvas.Rectangle(start_x + (mytblDataCycleDC_START_DI.Value * step), start_y
- (mytblDataCycleDC_PRIB_DI.Value * step),
                                start_x + (mytblDataCycleDC_START_DI.Value * step) +
(mytblDataCycleDC_FORM_DI.Value * step), start_y + (mytblDataCycleDC_VITR_DI.Value *
step));
        Image1.Canvas.Rectangle(start_x + (mytblDataCycleDC_START_OD.Value * step), start_y
- (mytblDataCycleDC_PRIB_OD.Value * step),
                                start_x + (mytblDataCycleDC_START_OD.Value * step) +
(mytblDataCycleDC_FORM_OD.Value * step), start_y + (mytblDataCycleDC_VITR_OD.Value *
step));
        Image1.Canvas.Rectangle(start_x + (mytblDataCycleDC_START_DZ.Value * step), start_y
- (mytblDataCycleDC_PRIB_DZ.Value * step),
                                start_x + (mytblDataCycleDC_START_DZ.Value * step) +
(mytblDataCycleDC_FORM_DZ.Value * step), start_y + (mytblDataCycleDC_VITR_DZ.Value *
step));
```

```
        Image1.Canvas.Brush.Color := clWhite;
        mytblDataCycle.Next;
      end;
    finally
      DataEl.Free;
    end;
    end else MessageDlg( mtWarning, [mbOk], 0);
  end;
end;

function TfmNewProject.CreateDataCycles(AProjectNo: integer): integer;
var
  DataCycles : TStringList;
  sql_str, tmp_str : String;
begin
  tmp_str := '';
//  tmp_str2 := '';
  DataCycles := TStringList.Create;
  try
    with BimaDataModule do begin
      with myqrDatacycleDelete do begin
        Close;
        SQL.Clear;
        sql_str := 'delete from data_cycle where dc_project_no > 0';
        SQL.Add(sql_str);
        ExecSQL;
      end;
      with myqrDIinfo do begin
        Close;
        SQL.Clear;
        sql_str :=  'select di_project_no, di_doc_no, di_dept_no, di_start_time,
di_form_time, di_top, di_prib, di_vitr ' +
                    'from di_info ' +
                    'where di_project_no = ' + IntToStr(AProjectNo);
        SQL.Add(sql_str);
        Open;
      end;
      while not myqrDIinfo.Eof do begin
        tmp_str := 'Ä' + IntToStr(myqrDIinfoDI_DOC_NO.Value) + '.O' +
IntToStr(myqrDIinfoDI_DEPT_NO.Value) + '-';
        with myqrTransfer do begin
          Close;
          SQL.Clear;
          sql_str :=  'select tr_project_no, tr_doc_no1, tr_dept_no1, tr_doc_no2,
tr_dept_no2, tr_time ' +
                      'from transfer_info ' +
                      'where tr_project_no = ' + IntToStr(AProjectNo) + ' and ' +
                      'tr_doc_no1 = ' + IntToStr(myqrDIinfoDI_DOC_NO.Value) + ' and ' +
                      'tr_dept_no1 = ' + IntToStr(myqrDIinfoDI_DEPT_NO.Value);
          SQL.Add(sql_str);
          Open;
        end;
        tmp_str := tmp_str + 'Ä' + IntToStr(myqrTransferTR_DOC_NO2.Value) + '.O' +
IntToStr(myqrTransferTR_DEPT_NO2.Value) + '-';
        with myqrTransfer2 do begin
          Close;
          SQL.Clear;
          sql_str :=  'select tr_project_no, tr_doc_no1, tr_dept_no1, tr_doc_no2,
tr_dept_no2, tr_time ' +
                      'from transfer_info ' +
                      'where tr_project_no = ' + IntToStr(AProjectNo) + ' and ' +
                      'tr_doc_no1 = ' + IntToStr(myqrTransferTR_DOC_NO2.Value) + ' and ' +
                      'tr_dept_no1 = ' + IntToStr(myqrTransferTR_DEPT_NO2.Value);
          SQL.Add(sql_str);
          Open;
        end;
        tmp_str := tmp_str + 'Ä' + IntToStr(myqrTransfer2TR_DOC_NO2.Value) + '.O' +
IntToStr(myqrTransfer2TR_DEPT_NO2.Value);

        DataCycles.Add(tmp_str);
        mytblDataCycle.Insert;
```

```
        myqrTransferTR_DOC_NO2.Value, myqrTransferTR_DEPT_NO2.Value, 'od');
        mytblDataCycleDC_FORM_OD.Value := ReturnFormTime(mytblProjectPROJECT_NO.Value,
myqrTransferTR_DOC_NO2.Value, myqrTransferTR_DEPT_NO2.Value, 'od');
        mytblDataCycleDC_PRIB_OD.Value := ReturnPrib(mytblProjectPROJECT_NO.Value,
myqrTransferTR_DOC_NO2.Value, myqrTransferTR_DEPT_NO2.Value, 'od');
        mytblDataCycleDC_VITR_OD.Value := ReturnVitr(mytblProjectPROJECT_NO.Value,
myqrTransferTR_DOC_NO2.Value, myqrTransferTR_DEPT_NO2.Value, 'od');
        mytblDataCycleDC_DOC_DZ.Value := myqrTransfer2TR_DOC_NO2.Value;
        mytblDataCycleDC_DEPT_DZ.Value := myqrTransfer2TR_DEPT_NO2.Value;
        mytblDataCycleDC_START_DZ.Value := ReturnStartTime(mytblProjectPROJECT_NO.Value,
myqrTransfer2TR_DOC_NO2.Value, myqrTransfer2TR_DEPT_NO2.Value, 'dz');
        mytblDataCycleDC_FORM_DZ.Value := ReturnFormTime(mytblProjectPROJECT_NO.Value,
myqrTransfer2TR_DOC_NO2.Value, myqrTransfer2TR_DEPT_NO2.Value, 'dz');
        mytblDataCycleDC_PRIB_DZ.Value := ReturnPrib(mytblProjectPROJECT_NO.Value,
myqrTransfer2TR_DOC_NO2.Value, myqrTransfer2TR_DEPT_NO2.Value, 'dz');
        mytblDataCycleDC_VITR_DZ.Value := ReturnVitr(mytblProjectPROJECT_NO.Value,
myqrTransfer2TR_DOC_NO2.Value, myqrTransfer2TR_DEPT_NO2.Value, 'dz');
        mytblDataCycle.Post;

//        tmp_str2 := tmp_str2 + #10#13 + tmp_str;
        myqrDIinfo.Next;
      end;
    end;
    ComboBox2.Items.AddStrings(DataCycles);

  finally
    DataCycles.Free;
  end;
//  ShowMessage(tmp_str2);
  result := 0;
end;

procedure TfmNewProject.acFormDataCycleExecute(Sender: TObject);
begin
  CreateDataCycles(BimaDataModule.mytblProjectPROJECT_NO.Value);
end;

procedure TfmNewProject.acFormDataCycleUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := (ComboBox1.ItemIndex in [4,5,6,7,8,9]);
end;

function TfmNewProject.ReturnPrib(AProjectNo, ADocNo, ADeptNo: integer;
  ADb: String): integer;
var
  tmp_str : String;
begin
  with BimaDataModule do begin
    if ADb = 'di' then begin
      with myqrDIinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select di_project_no, di_doc_no, di_dept_no, di_start_time,
di_form_time, di_top, di_prib, di_vitr ' +
                    'from di_info ' +
                    'where di_project_no = ' + IntToStr(AProjectNo) +
                    ' and di_doc_no = ' + IntToStr(ADocNo) +
                    ' and di_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDIinfo.RecordCount > 0 then result := myqrDIinfoDI_PRIB.Value else result :=
0;
    end else
    if ADb = 'od' then begin
      with myqrODinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select od_project_no, od_doc_no, od_dept_no, od_start_time,
od_form_time, od_top, od_prib, od_vitr ' +
                    'from od_info ' +
```

```
                    'where od_project_no = ' + IntToStr(AProjectNo) +
                    ' and od_doc_no = ' + IntToStr(ADocNo) +
                    ' and od_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrODinfo.RecordCount > 0 then result := myqrODinfoOD_PRIB.Value else result :=
0;
    end else
    if ADb = 'dz' then begin
      with myqrDZinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                    'from dz_info ' +
                    'where dz_project_no = ' + IntToStr(AProjectNo) +
                    ' and dz_doc_no = ' + IntToStr(ADocNo) +
                    ' and dz_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDZinfo.RecordCount > 0 then result := myqrDZinfoDZ_PRIB.Value else result :=
0;
    end;
  end;
end;

function TfmNewProject.ReturnVitr(AProjectNo, ADocNo, ADeptNo: integer;
  ADb: String): integer;
var
  tmp_str : String;
begin
  with BimaDataModule do begin
    if ADb = 'di' then begin
      with myqrDIinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select di_project_no, di_doc_no, di_dept_no, di_start_time,
di_form_time, di_top, di_prib, di_vitr ' +
                    'from di_info ' +
                    'where di_project_no = ' + IntToStr(AProjectNo) +
                    ' and di_doc_no = ' + IntToStr(ADocNo) +
                    ' and di_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDIinfo.RecordCount > 0 then result := myqrDIinfoDI_VITR.Value else result :=
0;
    end else
    if ADb = 'od' then begin
      with myqrODinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select od_project_no, od_doc_no, od_dept_no, od_start_time,
od_form_time, od_top, od_prib, od_vitr ' +
                    'from od_info ' +
                    'where od_project_no = ' + IntToStr(AProjectNo) +
                    ' and od_doc_no = ' + IntToStr(ADocNo) +
                    ' and od_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrODinfo.RecordCount > 0 then result := myqrODinfoOD_VITR.Value else result :=
0;
    end else
    if ADb = 'dz' then begin
      with myqrDZinfo do begin
        Close;
        SQL.Clear;
```

```
        tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                    'from dz_info ' +
                    'where dz_project_no = ' + IntToStr(AProjectNo) +
                    ' and dz_doc_no = ' + IntToStr(ADocNo) +
                    ' and dz_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDZinfo.RecordCount > 0 then result := myqrDZinfoDZ_VITR.Value else result :=
0;
    end;
  end;
end;
function TfmNewProject.ReturnStartTime(AProjectNo, ADocNo,
  ADeptNo: integer; ADb: String): integer;
var
  tmp_str : String;
begin
  with BimaDataModule do begin
    if ADb = 'di' then begin
      with myqrDIinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select od_project_no, od_doc_no, od_dept_no, od_start_time,
od_form_time, od_top, od_prib, od_vitr ' +
                    'from od_info ' +
                    'where od_project_no = ' + IntToStr(AProjectNo) +
                    ' and od_doc_no = ' + IntToStr(ADocNo) +
                    ' and od_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrODinfo.RecordCount > 0 then result := myqrODinfoOD_START_TIME.Value else
result := 0;
    end else
    if ADb = 'dz' then begin
      with myqrDZinfo do begin
        Close;
        SQL.Clear;
        tmp_str :=  'select dz_project_no, dz_doc_no, dz_dept_no, dz_start_time,
dz_form_time, dz_top, dz_prib, dz_vitr ' +
                    'from dz_info ' +
                    'where dz_project_no = ' + IntToStr(AProjectNo) +
                    ' and dz_doc_no = ' + IntToStr(ADocNo) +
                    ' and dz_dept_no = ' + IntToStr(ADeptNo);
        SQL.Add(tmp_str);
        Open;
      end;
      if myqrDZinfo.RecordCount > 0 then result := myqrDZinfoDZ_START_TIME.Value else
result := 0;
    end;
  end;
end;


procedure TfmNewProject.acInputDeptsNamesExecute(Sender: TObject);
begin
  BimaDataModule.mytblDepts.Filtered := False;
  BimaDataModule.mytblDepts.Filter := 'DP_PROJECT_NO = ' +
IntToStr(BimaDataModule.mytblProjectPROJECT_NO.Value);
  BimaDataModule.mytblDepts.Filtered := True;
  fmAddDepts:=TfmAddDepts.Create(Application);
  try
    fmAddDepts.ShowModal;
  finally
    fmAddDepts.Free;
    fmAddDepts:=nil;
  end;
  BimaDataModule.mytblDepts.Filtered := False;
end;
```

```
end.
```