МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНБАСЬКА ДЕРЖАВНА МАШИНОБУДІВНА АКАДЕМІЯ

# АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ СКЛАДНИХ ОБ'ЄКТІВ І СИСТЕМ

## КОНСПЕКТ ЛЕКЦІЙ

ДЛЯ СТУДЕНТІВ СПЕЦІАЛЬНОСТІ 151
«АВТОМАТИЗАЦІЯ ТА КОМП'ЮТЕРНО-ІНТЕГРОВАНІ ТЕХНОЛОГІЇ»

**Краматорськ 2020**

ББК 32.965

УДК 004.415.26

**Автоматизоване проектування складних об'єктів і систем**: Конспект лекцій для студентів спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології». / Укладач:А.В. Люта. - Краматорськ: ДДМА, 2020. – 124 с.

Розглянуто концепції, базові принципи, методи поетапного ана-лізу і синтезу складних систем, виділення аспектів їх розгляду, діа-програмних методики документування проектів програмних систем (ПС). Розглядаються питання проектування як машинобудівних, так і програмних виробів, щоб виділити загальні підходи до реалізації процесу проектування складних систем і показати особливості, пов'язані з об'єктом проектування. ПС як будь-які технічні системи постійно розвиваються, тому в процесі проектування і нової техніки, та програмного забезпечення важливе значення має розгляд тенденцій розвитку і зміни технологій проектування, способів вирішення виникаючих завдань і проблемних ситуацій. Показано застосування діаграмних методик (DFD, STD, ERD та інших) при структурному підході до проектування ПС. Розглянуто також окремі питання організації інфраструктури, в якій функціонують системи автоматизованого проектування (САПР) та інші програмні системи.

Укладачі: А. В. Люта, к.т.н., доц.

Відп. за випуск: А. В. Люта, к.т.н., доц.

ЗМІСТ

Умовні позначення

ПК - персональний комп'ютер;
САПР - системи автоматизованого проектування;
ІАСУ - інформаційна автоматизована система управління виробництвом;
АСОУ - автоматизована система організаційного управління;
ЖЦ - життєвий цикл системи (об'єкта);
CALS (Continuous Acquisition and Life cycle Support) - система безперервної інформаційної підтримки всіх стадій життєвого циклу продукту;
ООП - об'єктно-орієнтований підхід;
БД і БЗ - бази даних і знань;
ТС - технічна система;
ТО - технічний об'єкт;
ШІ - штучний інтелект;
ЄСКД - єдина система конструкторської документації;
ЄСТД - єдина система технологічної документації;
ЄСТПВ - єдина система технологічної підготовки виробництва;
ЄСПД - єдина система програмної документації;
ТЗ - технічне завдання;
CASE (Computer-Aided Software Engineering) - засіб проектування програмних і автоматизованих систем;
SSADM (Structured Systems Analysis and Design Method) - технологія розробки автоматизованих систем;
CAD (Computer-Aided Design) - системи автоматизованого проектування призначені для створення об'ємних моделей виро-лій;
CAE (Computer-Aided Engineering) - системи забезпечують виконання інженерних розрахунків (міцність, кінематичні, динамічні, тепло-ші і інші процеси);
CAM (Computer-Aided) - системи для розробки технологічних процесів обробки деталей на верстатах з ЧПУ, візуалізації процесів обробки;
СРС - спеціалізовані підсистеми для розрахунку процесів лиття, формозміни листових деталей, проектування пружин, зварювання, розводки трубопроводів;
PDM - системи для вирішення організаційних питань при проектуванні виробів;
CRM - системи для взаємодії зі споживачами продукції;
MES (Manufacturing Execution Systems) - виробничі виконавчі системи для оперативного планування і управління виробництвом.
ERP - системи управління ресурсами підприємства.

# ВСТУП

Автоматизація проектування стала можливою не тільки в зв'язку з розвитком комп'ютерної техніки, а й внаслідок появи нових інформаційних технологій, що забезпечують спільну роботу співробітників підприємства по забезпеченню діяльності творців нових технічних систем. Тому сьогодні системи автоматизованого проектування (САПР) перетворилися в організаційно-технічні системи, що включають функції організації колективної роботи над проектами, створення електронних архівів, баз стандартизованих елементів конструкції та інші. Для ефективного застосування САПР необхідно створення певної інфраструктури, що включає необхідні ресурси, матеріально - технічні компоненти і організаційні рішення щодо підтримки функціонування САПР.

САПР в свою чергу входить до складу інтегрованої автоматизованої-ванній системи управління виробництвом (ІАСУ) і активно взаємодіє з іншими компонентами цієї системи. Сучасний розвиток САПР відбувається в рамках підтримки частини етапів життєвого циклу проектованої системи (продукту). Життєвий цикл (ЖЦ) продукту - це сукупність процесів, які виконуються від моменту виявлення потреб суспільства в певної продукції до задоволення цих потреб і утилізації продукту (ISO 9004-1). Безперервна інформаційна підтримка всіх стадій ЖЦ продукту здійснюється в рамках CALS-систем (Continuous Acquisition and Life cycle Support) і являє собою стратегію переходу на безпаперову електронну технологію. Метою такого переходу є підвищення ефективності бізнес-процесів, які виконуються в ході ЖЦ продукту, за рахунок спільного використання інформації на всіх етапах ЖЦ. Інформаційна інтеграція проводиться на основі єдиної моделі створюваної технічної системи.

З розвитком інформаційних технологій пов'язують найголовніше завдання - суттєве скорочення термінів створення нової техніки, що реалізує прогресивну технологію, яка підвищує продуктивність праці, високонадійну, з мінімальним споживанням матеріальних і енергетичних ресурсів. Автоматизація проектування - природний процес розвитку інтелектуальної та матеріальної основи процесу автоматизації в цілому. Хоча проектування відноситься до виробничої діяльності людини в ньому істотні творчі початку, що надає йому певні риси мистецтва. Завдання САПР не в заміні людини програмної системою, а в наданні інструментів для інформаційного, методичного та інших видів забезпечення проектування, включаючи прийоми і методи пошуку варіантів оптимальних технічних рішень. Проектування складних систем - поетапний, варіативний, ітераційний процес, що вимагає аналізу технічних систем - аналогів, синтезу конструктивних рішень, порівняння і вибору найкращого варіанту, визначення оптимальних параметрів елементів системи і реалізації інших функцій. Процес проектування здійснюється за ряд стадій (етапів), на кожному з яких вирішуються свої важливі завдання: технічне завдання, технічна пропозиція, ескізний проект, технічний проект, робоча документація. САПР в змозі допомогти конструктору у виконанні багатьох процедур і операцій цих етапів - проектувальні і перевірочні розрахунки, пошук необхідної інформації, проведення кінематичного і динамічного аналізу об'єктів проектування, оптимізації його

параметрів, математичного та геометричного моделювання. Створюються САПР і для пошукового конструювання, включаючи синтез варіантів технічних рішень і вибір з них оптимального. Широко застосовується автоматизований розрахунок і конструювання деталей машин, деяких вузлів і інших виробів. Однак, незважаючи на успіхи в розвитку САПР, системи, що охоплюють всі стадії розробки від технічного завдання до робочої документації, є тільки в окремих областях техніки. Це пов'язано зі складністю формального опису процесу проектування, яке дозволило б використовувати кошти САПР на всіх стадіях розробки проекту і документації.

Основні концепції автоматизації проектування:

1. При будь-якому рівні автоматизації проектування людина завжди буде основним елементом системи - за ним залишається остаточний вибір проектних рішень.
2. В процесі розвитку САПР форми проектування будуть змінюватися, але етапність проектування, що відповідає процесу розпізнавання об'єкта в середовищі його оточення і заснована на концепції життєвого циклу вироби, залишиться незмінною.
3. Проектування повинно здійснюватися в єдиному інформаційному просторі. В основі цієї концепції лежить використання відкритих архітектур програмних систем і міжнародних стандартів обміну даними.

Методичне забезпечення САПР об'єднує логіку процесу проектування, методи і алгоритми виконання проектних процедур і операцій, орієнтованих на використання комп'ютерів, методи описів об'єкта проектування на послідовних стадіях розробки.

САПР в свою чергу також є системами, які вимагають проектування, тому в даному навчальному посібнику розглядаються питання проектування як машинобудівних, так і програмних виробів, щоб виділити загальні підходи до реалізації процесу проектування складних систем і показати особливості, пов'язані з об'єктом проектування. САПР як будь-яка технічна система постійно розвивається, тому в процесі проектування і нової техніки, та програмного забезпечення важливе значення має розгляд тенденцій розвитку і зміни технологій проектування, способів вирішення виникаючих завдань і проблемних ситуацій.

*Мета дисципліни* - вивчити базові принципи, методи, підходи до поетапного аналізу і синтезу складних систем, виділенню аспектів їх рас-перегляду, діаграмні методики документування проектів програмних систем. Важливе значення мають також питання створення певної інфраструктури, в якій функціонує САПР і інші програмні системи.

# 1. АНАЛІЗ ОБ'ЄКТІВ ПРОЕКТУВАННЯ ЯК СИСТЕМ

## 1.1 Загальні поняття і принципи подання інформації про системи

Поняття *системи* включає *цілісне* безліч *пов'язаних* між собою *елементів* системи, які *узгоджено взаємодіють* для досягнення певної (заданої) *мети системи в навколишньому середовищі*.

Система щодо навколишнього середовища виділяється і сприймається як щось ціле (цілісність). Ступінь взаємозв'язку між елементами систе-ми може мати відчутні відмінності (різна ступінь цілісності) [1 - 5].

*Складні системи* - характеризуються розвиненими внутрішніми зв'язками, наявність і форми яких істотно впливають на функціонування системи. У складній системі сукупність зв'язків між блоками розвинена в такій мірі, що зміна одного з факторів (вхідних параметрів) призводить до зміни декількох інших (вихідних).

Взаємозв'язку між елементами системи можуть мати різну фізичну природу.

Система може взаємодіяти з навколишнім середовищем системи, в которую входить безліч об'єктів (з їх істотними властивостями). Об'єкти навколишнього середовища, з якими система взаємодіє, можуть змінювати стан системи в разі зміни своїх властивостей. У свою чергу система впливає на своє навколишнє середовище. Система відокремлена від навколишнього середовища *кордоном системи*. Виділення об'єктів навколишнього середовища для системи залежить від *аспектів її вивчення,* постановки *мети і завдань дослідження*.

## 1.2 Системний підхід до декомпозиції і розробці класифікацій об'єктів проектування

Першим етапом вивчення *об'єктів проектування* є аналіз *предметної області* до якої вони належать. У процесі вивчення предметної області з метою автоматизації діяльності по створенню *нових об'єктів (виробів, продуктів виробництва)* можна виділити **два основні завдання**:

1. Аналіз стану та перспектив розвитку, виділення аспектів розгляду, пошук аналогів, *декомпозиція, уявлення, процедури синтезу* виробів в предметної області (рис.1).

2. Аналіз стану та перспектив розвитку автоматизації робіт зі створення нових об'єктів, декомпозиція, уявлення і т. д. інформаційних технологій і програмних продуктів, за допомогою яких моделюють (розраховують, візуалізують, проектують) об'єкти або вироби предметної області.


Рисунок 1 - Загальносистемні принципи проектування

Вирішення першого завдання спрямоване на вивчення перспектив предметної області та виявлення тенденцій її розвитку, а друге завдання дозволяє правильно вибрати засоби автоматизації, правильно розробити модель об'єкта проектування і визначити функції, які необхідно автоматизувати. Таким чином, рішення задач даного рівня вивчення систем пов'язано з формалізацією інформації не тільки про них, але і про оточення, в якому системи функціонують. Виділення системи з навколишнього світу пов'язано з класифікацією їх *ознак, функцій, складу, структури,* конструктивного виконання і інших показників їх функціонування.

Складність подання інформації про систему пов'язана з неоднорідністю даних, існуванням декількох *аспектів* розгляду властивостей об'єктів: *функціональний* аспект, *конструктивний, технологічний, антропологічний, економічний* та інші, специфічні для досліджуваної області, відповідно до процесів різної фізичної природи в технічних системах: *фізичні* процеси , хімічні, *механічні, геометричні* та ін. [1].

У процесі вивчення системи і її елементів виникає необхідність їх класифікації, щоб зіставити отриману інформацію з уже наявними відомими даними про існуючі системи. Класифікація є одним з перших етапів дослідження систем. Основна мета класифікації - представити наявну інформацію про систему в структурованому вигляді, виявити елементи системи і їх зв'язку, поставити у відповідність елементам системи певні поняття предметної області (терміни). Завдання класифікації: аналіз властивостей системи, виділення головних елементів, виявлення шляхів розвитку і вдосконалення системи.

При вивченні різних систем можливе застосування загальних принципів і алгоритмів їх розгляду, як сукупності елементів і зв'язків між ними. Тому послідовність виконання класифікації містить ряд загальних етапів, пов'язаних з розробкою списків (специфікацій) різних видів. За умови дотримання вимоги формально-логічного побудови виділимо наступні принципи і етапи класифікації та використання її результатів:

а) виділення ознак класифікації (видів підмножин для групування елементів системи) з урахуванням рівня абстракції, які є підставою для групування;

б) виділення елементів, відповідних ознаками класифікації (що входять в підмножини, що виключають одна одну), вибір для цієї мети відповідних показників і критеріїв якості;

в) виділення видів *функцій* (цілей) системи і її елементів, а також характеристик функцій (показників);

г) зв'язування елементів і виконуваних функцій;

д) надання інформації в структурованому вигляді, зазвичай у вигляді графів або дерев;

е) використання матриць суміжності (або інцидентності) для формалізації інформації про зв'язки елементів системи;

ж) вивчення отриманих матриць з використанням теорії множин;

з) побудова критеріїв якості і моделі для порівняння систем в цілому, наприклад, реалізація цільового методу [6];

і) використання різних методів параметричної оптимізації і оптимізації елементного складу системи.

Особливе значення при цьому має графічне представлення інформації про системи, основним засобом для цього є дерева, графи. Дерева, зокрема, використовуються для декомпозиції і структурованого представлення елементів системи на різних рівнях абстракції, а графи - для показу структури системи. Для подання інформації про різні системи застосовують також різні методики побудови діаграм, таблиць, структурних карт та інших видів графічного зображення властивостей систем [3, 6].

При вивченні, створенні (проектуванні) систем для опису функціональних причинно - наслідкових зв'язків між її елементами, функціями, параметрами, широко використовуються *блок-схеми*. На верхньому *рівні абстракції* систему подають як єдиного блоку і показують зв'язку системи із *зовнішнім середовищем* (рис. 2).

Системи характеризуються рядом властивостей. *Фактори* (параметри, що характеризують фактори), які впливають на систему внаслідок наявності зовнішнього середовища, називають *входами* системи (Ui). Система реалізує свої цілі функціонування і впливає на навколишнє середовище за допомогою *виходів* (Yk).

Якщо не розглядається внутрішній устрій системи, а модель будується на підставі вивчення зв'язків входів і виходів, то така модель системи називається *«чорним ящиком»*.

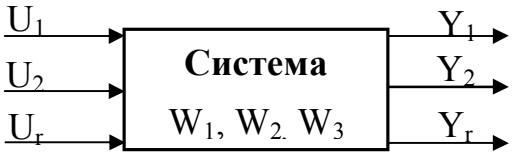



Рисунок 2 - Схема зв'язків системи з навколишнім середовищем

Безліч значень істотних характеристик (властивостей, факторів) системи, якими вона володіє в даний момент часу утворює *стан системи*. Внутрішні параметри системи W1, W2, W3 - параметри внутрішнього стану.

Складні системи, що складаються з ряду підсистем або об'єктів, можуть за допомогою *декомпозиції* бути представлені таким чином, що входи одних

об'єктів є виходами інших (рис. 3). Слід зазначити, що при декомпозиції системи на блоки відбувається така ж декомпозиція параметрів системи (входів, виходів, параметрів внутрішнього стану). Види блоків (конструктивні елементи, функціональні блоки та ін.) Залежать від виду декомпозиції і завдань вивчення системи. Тому для подання структурних схем (*потокових діаграм*) системи, необхідно виділити аспекти розгляду: механічний, електричний, тепловий і т. д., щоб спростити уявлення системи на кожній діаграмі. Таким чином, виділення різних аспектів роботи системи, завдань уявлення, аналізу і синтезу показують складність питань, пов'язаних з вивченням системи.

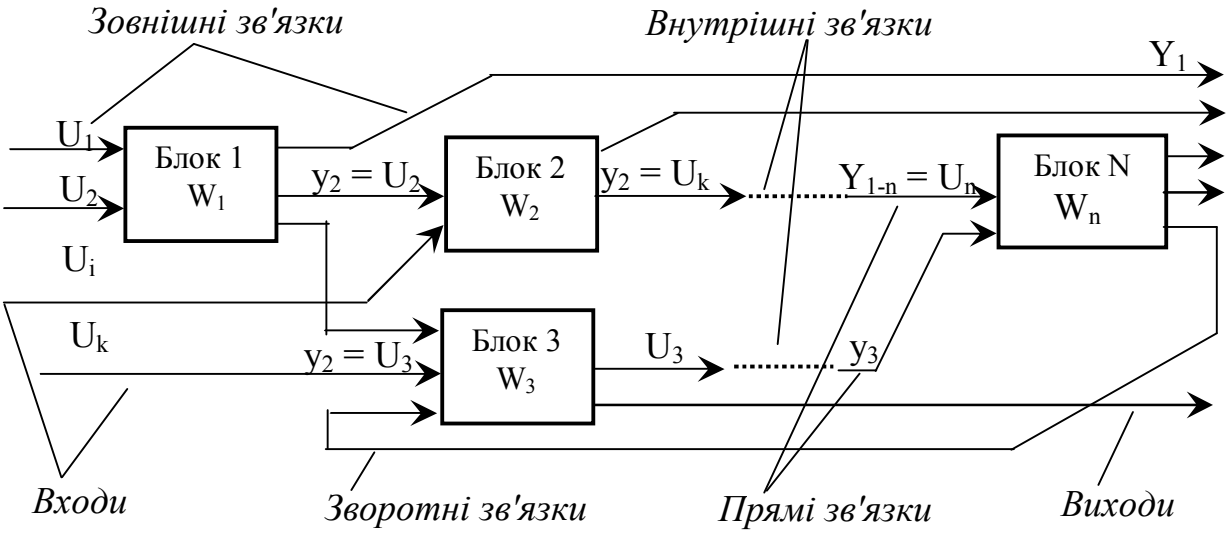



Рисунок 3 - Схема зовнішніх і внутрішніх зв'язків системи
в результаті декомпозиції на блоки

В цілому, структура і елементний склад системи характеризують її *морфологіческіе властивості*. Процес виділення структури системи називають морфологічним аналізом [7].

Зв'язки теж можуть бути класифіковані (див. рис 3, рис4). З'єднання блоків може бути *послідовне* і *паралельне*. Можливо також і *паралельно-послідовне* з'єднання блоків об'єктів, *ієрархічне* і інші. Застосовують і інші класифікаційні ознаки для аналізу видів зв'язків. Наприклад, зв'язку можуть бути *технологічними*: потоки речовини, енергії, дії механічних сил, моментів, фізичних і хімічних факторів і т. д. Зв'язки можуть бути *інформаційними*: потоки даних, документів, сигнали і т. д. Відповідно до цілеспрямованої діяльністю системи зв'язку із зовнішнім середовищем (входи і виходи) можуть бути *корисними* і *шкідливими* (вугілля і порода), *контрольованими* і *неконтрольованими*.

Спільність і наочність класифікації пов'язана з тим, чи вдалося представити різнорідну інформацію про систему, що має багатовимірний простір ознак, на площині. Подання інформації виконується зазвичай у вигляді таблиці або графа, з відповідною матрицею суміжності або інцидентності [1]. Велике значення має також вдале поєднання різних аспектів розгляду системи.

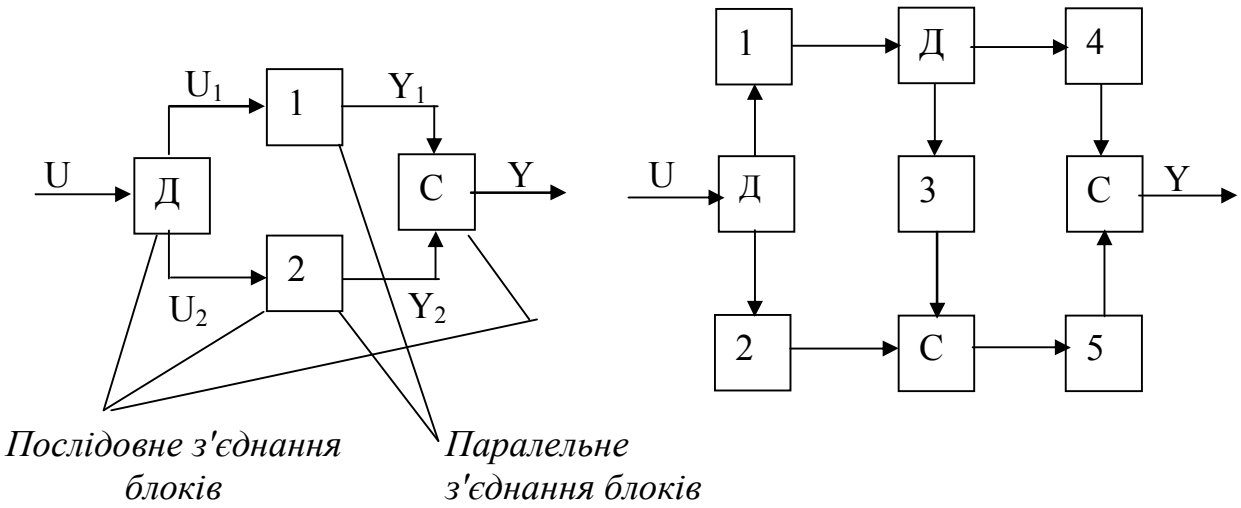



Рисунок 4 - Схема зв'язків блоків системи між собою і з навколишнім середовищем
Д - дільник (блок розподілу);
З - змішувач (блок з'єднання процесів)

На практиці застосовують два основних підходи до розгляду та аналізу систем: *процедурно-орієнтований (структурний, активносних)* і *об'єк-проектної-орієнтований* (ООП) [8,9]. Різниця методів подання інформації про систему в тому, які елементи застосовуються в якості вузлів при побудові схем, діаграм.

Розглянемо приклад початкового етапу вивчення системи, в якості якої приймемо «Праска». Основний (головної) функцією системи є можливість прасувати білизну за рахунок високої температури на робочій поверхні праски. Праска підключається до побутової електричної мережі напругою 220 В і частотою 50 Гц, яка забезпечує нагрівання робочої поверхні. У процесі роботи до ручки, проводу прикладаються певні зусилля рукою людини, тому елементи корпусу не повинні бути гарячими. Крім того, елементи праски повинні бути ізольовані від контакту з струмоведучими частинами, що забезпечується ізоляцією проводи і корпусом праски. Можна виділити також зовнішні елементи управління (регулятор) і контролю (лампа підключення до мережі). Зовнішні фактори, що визначають роботу праски, наведені на рис 5. На даному етапі опису внутрішні чинники (параметри) не розглядаються. При побудові моделі системи на даному етапі є вектор входів - X, вектор виходів - Y і будується залежність параметрів виходів від входів $Y = f(X)$.

Входи і виходи системи в даному випадку можуть бути розділені на корисної, які забезпечують головну функцію (температура, t [°C], механічні навантаження, F [Н]) і шкідливі (деформації: температурні, $\varepsilon_t$ [%], від механічних навантажень, $\varepsilon$ f [%]). Фактори можуть бути розділені на залежні (температура залежить від напруги, U [В], струму, I [А], положення регулятора, $\alpha$ град.) І незалежні (температура, t [°C] не залежить від механічних навантажень,

F [Н]). Параметри мережі, температура, час роботи - контрольовані чинники, а параметри механічних навантажень, деформації - неконтрольовані.

При декомпозиції системи проявляються внутрішні зв'язки між її елементами. Якщо розглядаються параметри внутрішнього стану W1, W2, ... Wn, то відбувається декомпозиція і параметрів внутрішнього стану (рис. 6).

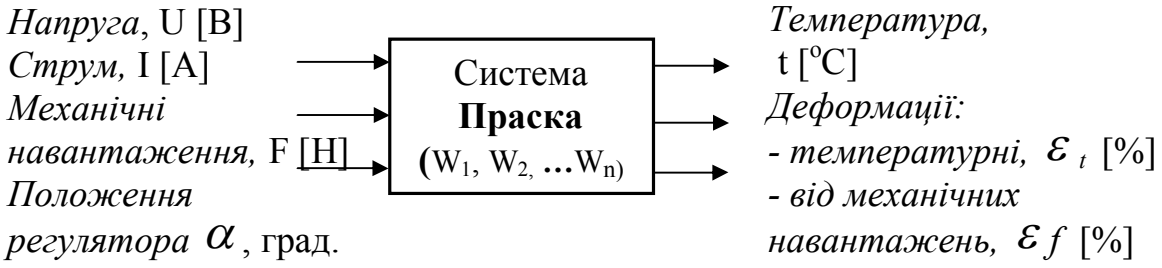



Рисунок 5 - Схема зв'язків системи з навколишнім середовищем

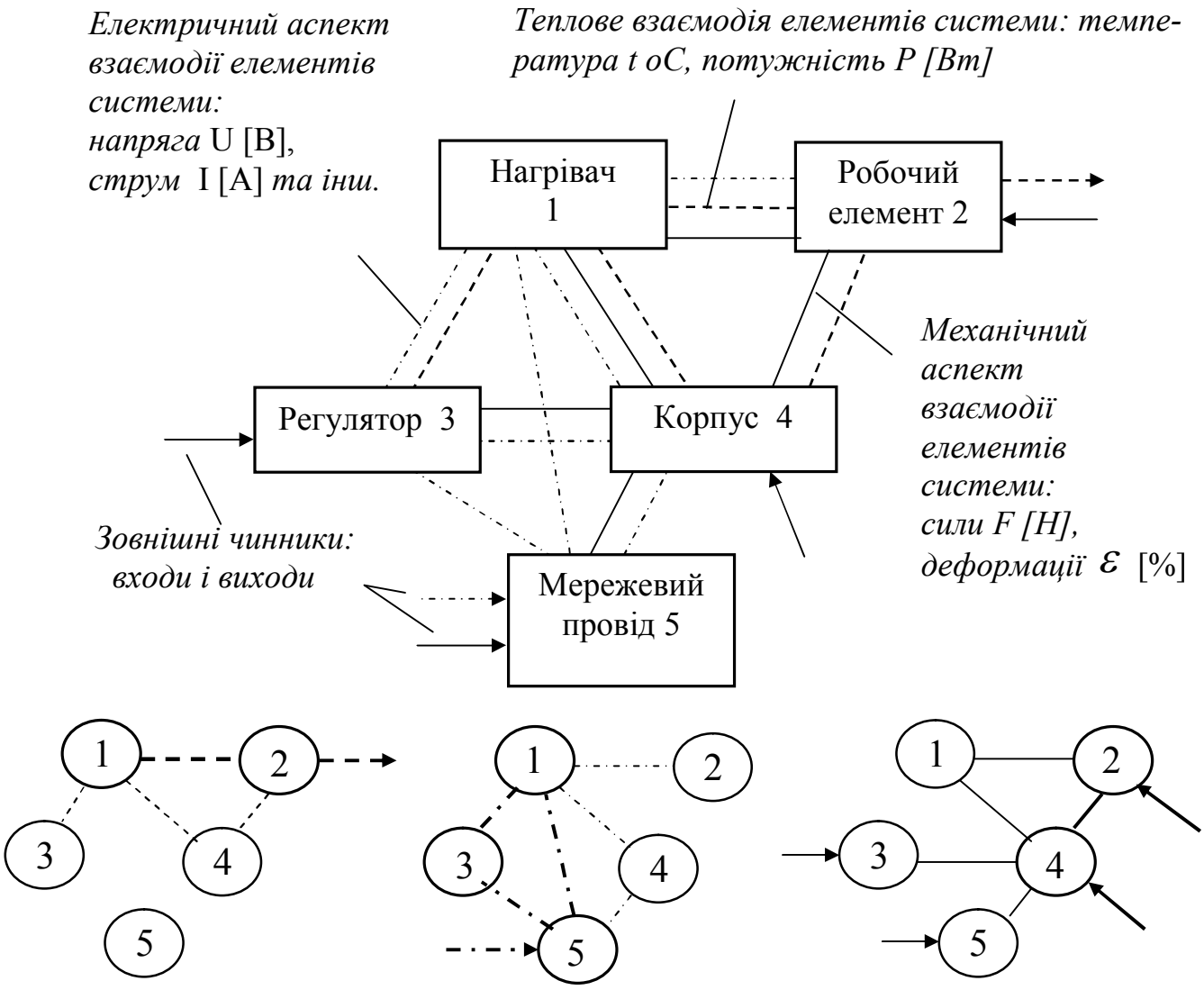



Рисунок 6 - Граф зв'язків елементів системи «Праска» і підграфи в залежності від аспектів її розгляду

а – граф теплового взаємодії елементів системи;
б – граф електричних зв'язків елементів системи;
в – граф механічних взаємодій елементів системи

На загальному графі, що містить формалізовану інформацію, можна виділити підграфи або дерева з точки зору аспектів вивчення системи. Наприклад, дерево складання елементів в ціле, дерево елементів різної технологічної складності (за сукупністю технологічних операцій для кожного елемента), що дозволить оцінити складність елементів, виконати мережеве планування виготовлення об'єкта в цілому. Таким чином, проводиться виділення підмножин за певними ознаками, а потім визначення їх перетину. Наприклад, деталь або вузол, що знаходяться в області перетину максимальної кількості підмножин технологічних операцій, як правило, вимагає найбільшого часу обробки в різних спеціалізованих цехах, найбільш різноманітного інструменту і т.д.

На графі (див. рис 6) виділені основні потоки (тепло, енергія і навантаження). На малюнку видно, що виділення аспектів спрощує графи, а отже і вивчення системи. Зокрема, на мал. 6а видно, що нагрівач 1 є джерелом тепла, яке передається на робочий елемент 2 (корисний потік). Інші деталі праски 3 - 5 необхідно ізолювати від впливу тепла (шкідливі потоки).

Граф електричних зв'язків елементів системи (рис 6б) показує, що крім основного потоку електроенергії від мережі до нагрівача 1 через мережевий провід 5 (корисний силовий потік), необхідний ще потік до регулятора температури 3 для управління системою (корисний потік управління). Елементи 2, 4 повинні бути ізольовані і виконані з непровідних матеріалів.

На графі механічних взаємодій елементів системи (рис 6в) показано, що основні навантаження прикладаються до ручки корпусу 2 і робочого елементу 4 і замикаються між ними, щоб гладити тканини (корисні потоки), однак і інші елементи 3, 5 також навантажуються зовнішніми зусиллями. Зовнішні навантаження до нагрівача 1, який працює в несприятливих умовах високих температур, повинні бути виключені (шкідливі потоки).

Входи і виходи кожного блоку і системи в цілому в процесі роботи змінюються, вони є характеристиками відбуваються в складній системі процесів. Для того, щоб дізнатися реакцію системи на вхідні впливу, потрібно перетворити входи і виходи системи в сигнали, документи і т. д., які і будуть носіями інформації про значення входів і виходів. Перетворення значень параметрів входів і виходів системи в сигнали для реєстрації, контролю і управління проводиться *датчиками інформації.*

Інформаційна модель технічної системи показана на рис 7. Досконалість технічних систем описується групою *критеріїв якості*, тобто спостережуваних змінних вихідних величин, які залежать від факторів (входів) технічної системи. Для отримання математичної моделі необхідно з великої кількості входів (параметрів, факторів), що впливають на виходи (*критерії якості*), вибрати найбільш важливі, тобто *статистично значущі*. Для цього існують спеціальні математичні методи.

Фактори можуть взаємодіяти між собою, при цьому вони створюють *системний ефект взаємного впливу факторів* [10], який може суттєво впливати на критерії якості системи (виходи). Тому взаємний вплив чинників в складних системах необхідно враховувати при виборі виду моделі системи. При створенні математичних моделей технічних систем виявлення видів взаємодії факторів і оцінка ступеня їх залежності є визначальними для якості моделювання. Система у зовнішньому середовищі проявляється як щось цілісне, при цьому об'єднання елементів з допомогою зв'язків створює ще один *системний ефект*: надає системі *таку сукупність властивостей, яку не мають елементи системи окремо*.

Для побудови моделі технічної системи на основі подання її у вигляді «чорного ящика», тобто без аналізу фізичних процесів в системі використовують теорію планування експерименту [11]. Для моделювання зв'язків входів і виходів будують *рівняння регресії*, зазвичай у вигляді поліномів 1-го ступеня (*лінійна модель*) або 2-го ступеня (*квадратична модель*).

## 1.3 Графічне представлення ієрархічної структури системи

### 1.3.1 Графи і дерева. Основні поняття, формалізація інформації у вигляді матриць суміжності і інцидентності.

Матеріальна система, яка складається з двох множин: *точок (вершин, вузлів, блоків, місць)* і *ліній (зв'язків, ребер)*, які знаходяться між собою в будь-якому відношенні називається *графом*. Безліч точок відповідає безлічі вершин

$X=\{X_1, X_2, ..., X_n\}$,
$|X| = n$ - потужність графа.

Безліч ліній, що з'єднують пари вершин, називається безліччю ребер або дуг

$U=\{U_1, U_2, ..., U_m\}$,
$|U| = m$.

Граф, що містить тільки орієнтовані лінії, називається *орієнтованим графом* або *орграфом*. Граф, що містить тільки неорієнтовані лінії, називається *неорієнтованим графом*. Граф, у якого існує хоча б одна пара вершин, що з'єднуються m ребрами ($m> = 2$), називається *мультиграфом*, а найбільше число m називається *мультічислом* графа.

Ребра, що з'єднують одну і ту ж пару вершин, називаються *кратними*. Ребро, що з'єднує дві вершини називається інцидентним, а вершини *інцидентні* ребру. *Суміжними* називають дві вершини, якщо є що з'єднує їх ребро; два ребра інцидентні одній вершині.

*Кінцевий граф* - граф, що включає кінцеве безліч вузлів і зв'язків (X і U). *Нульовий граф* - граф, в якому вершини не з'єднані ребрами, при цьому безліч U пусте $U = \{\}$. *Повний граф* - граф, в якому кожна пара вершин з'єднана ребром.

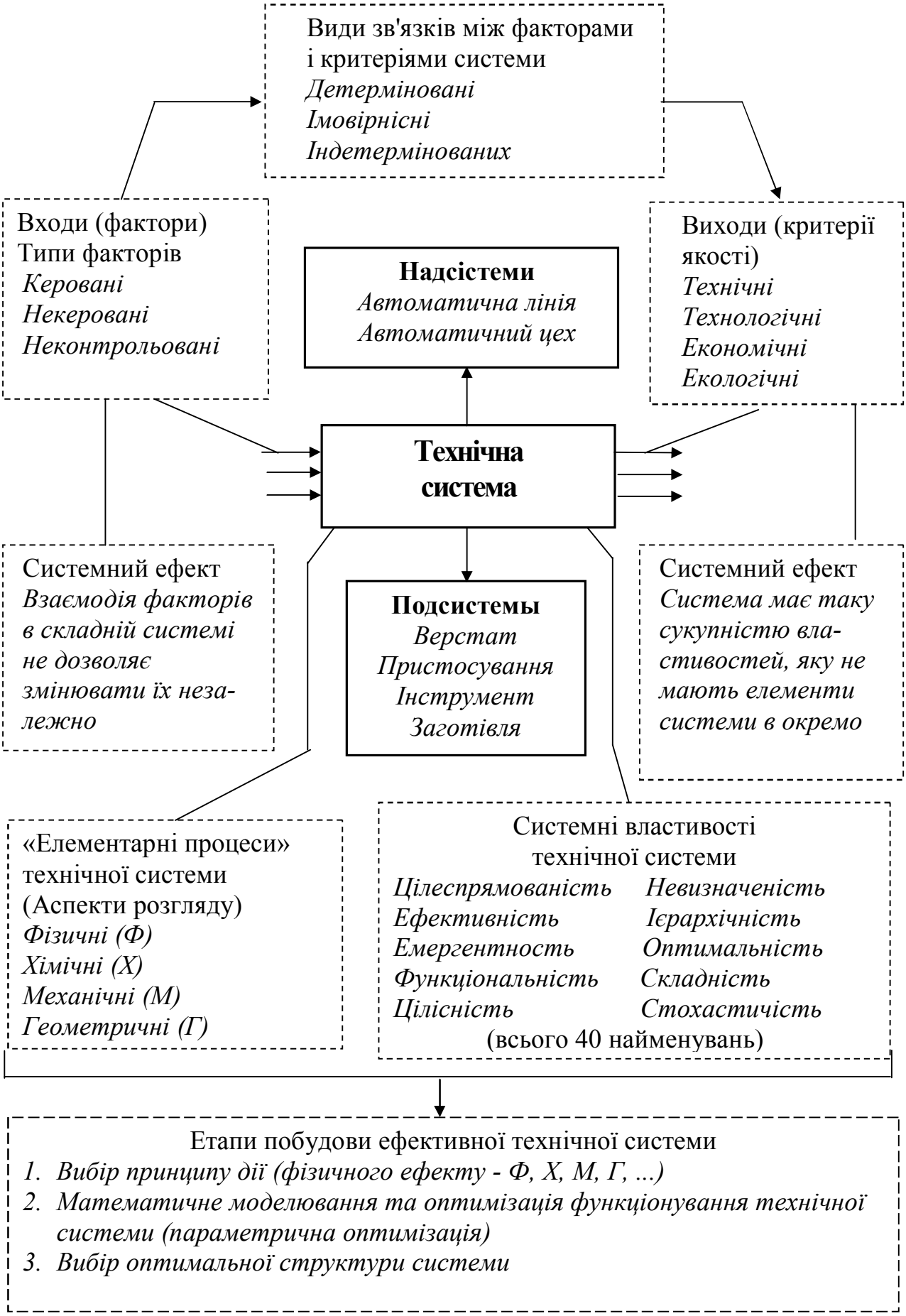



Рисунок 7 - Інформаційна модель технічної системи

*Ступінь вершини* - число ребер, інцидентних даній вершині. Граф, ступеня всіх вершин якого рівні К, називається *однорідним* або *регулярним* графом ступеня К. *Подграф* - частина графа, утворена деяким підмножиною ребер графа і всіма інцидентними їм вершинами. *Суграф* - частина графа, утвореного видаленням з вихідного графа деяких ребер.

Квадратна матриця R [n, n] називається матрицею суміжності, якщо її стовпці і рядки утворені вершинами графа (i = 1, ... i = n; j = 1, ... j = n), а значення в осередках таблиці R [i, j ] показують наявність зв'язків між вершинами. Якщо R [i, j] = 1 то Xi суміжно з Xj а якщо вершини не з'єднані R [i, j] = 0. Якщо граф неорієнтований то R [i, j] = R [j, i].

Прямокутна матриця S [n, m] називається матрицею інцидентності, якщо її рядки утворені вершинами графа (i = 1, ... i = n), а стовпці - зв'язками (j = 1, ... j = m) або навпаки. Елементи в осередках утворюються за правилом Sij = 1, якщо зв'язок (ребро) Uj виходить з Xi; Sij = - 1, якщо зв'язок Uj входить в Xi .. Якщо ребро Uj не пов'язане з вершиною Xi, то Sij = 0.

### 1.3.2 Особливості виділення рівнів ієрархії

Для опису ієрархічної структури конкретних систем (виробів) зазвичай застосовуються I - *дерева*. Ці дерева є сукупність-ність вершин і зв'язують їх ребер, згрупованих на різних ієрархічних рівнях (*ярусах*). Кожен ієрархічний рівень являє собою проектовану систему з різним ступенем деталізації. Нульовий ієрархічний рівень (система) є найбільш абстрактним, а останній - найбільш деталізованим.

Кількість рівнів декомпозиції повинно відповідати постав-ленній задачі. Мінімальна неподільна частина в рамках завдання називається *елементом*. Дерево не завжди відображає чисто конструктивну декомпозицію об'єкта.

Застосування ієрархічної декомпозиції допомагає формувати *словник предметної області*, в якій існує розглянута система, і дозволяє виділити базові поняття (*абстракції*) предметної області, які необхідні при її аналізі, в тому числі і при створенні САПР [3]. Крім того, на кожному рівні абстракції застосовуються свої принципи і методики опису системи, що відповідають ступені її декомпозиції. Таким чином, розбиття об'єкта знаходиться в рамках *блочно-ієрархічного* підходу до структурної опису об'єктів. Вершини відображають складові частини проектованого об'єкта. Ребра відображають різноманітні зв'язки між вершинами (механічні, теплові, електричні). При створенні програмних продуктів вказують, наприклад, потоки даних між блоками, модулями. Вершини (елементи) самого нижнього ярусу (ієрархічного рівня) в рамках прийнятого уявлення складної системи - (наприклад автомобіля) називаються *базовими елементами* або *листям*. Таким чином I - дерево є поданням конкретної системи, в якому елементи об'єднані зв'язками в вузлах I, відповідно до ієрархічної декомпозицией системи (рис. 8). Оскільки всі вузли однакові (мають тип I), їх іноді на ри-ках не показують.

Технічне рішення для системи, представленої I-деревом, на каж-будинок рівні абстракції включає в себе інформацію про всі елементи системи. Ці еле-

менти являють собою безліч конструктивних рішень (деталей, вузлів), призначених для виконання функцій системи. Дерево І може включати також інформацію про особливості конструктивного виконання елементів: про геометричній формі, основні ознаки і параметри.

## 1.4 Використання І - АБО - дерев для узагальнення інформації про групи об'єктів

При розгляді завдань в САПР необхідно вирішувати не тільки питання представлення інформації про ієрархічну структуру системи, які пов'язані з різним ступенем декомпозиції системи, а також питання, пов'язані з пошуком засобів поліпшення технічних рішень для елементів і системи в цілому. Найбільш наочним і ефективним способом для вирішення цього завдання є графи і дерева, зокрема І-АБО-дерева [1, 2]. З використанням І-АБО-*дерев* здійснюється відображення структури цілого класу подібних *систем* - *аналогів*, які включають накопичений досвід в розробці систем даного типу. Побудова загального І-АБО-дерева технічних рішень зазвичай виробляють шляхом об'єднання *І-дерев*, побудованих для кожної з декількох найбільш перспективних конкретних систем (рис. 9).

І-АБО - дерево являє собою односпрямований граф з однією *кореневою вершиною*. Кожна вершина, крім кореневої, підпорядкована будь-якої однієї вершині, розташованої на більш високому рівні абстракції (ярусі). Вершини бувають двох типів: І, АБО. Вершини, незалежно від типу, можуть мати дві або кілька *підлеглих вершин*. Вершини, які не мають підлеглих, називаються *висячими*. До вузлів можна приписати *дискримінатори* (принципи поділу або об'єднання елементів), зокрема, «І» - дискримінатор *агрегування* елементів в систему.

На І-АБО - дереві вузли І пов'язують узагальнені (*функціональні*) елементи розглянутого класу систем, які в сукупності реалізують необхідні для неї функції, а вузли АБО з'єднують альтернативні варіанти відомих конструктивних рішень для кожного функціонального елемента. Таким чином, у вершин типу І підлеглі вершини є її *складовими частинами*. У вершини типу АБО, підлеглі елементи є її можливими *альтернативними варіантами* (альтернативами), які зустрічаються у відомих систем, тому сукупність елементів, об'єднаних в вершині АБО, називають *альтернативної лінійкою*.

Функціональні елементи і їх конструктивні рішення розділені на кожному ієрархічному рівні декомпозиції системи і представляються на різних ярусах. Яруси, на яких розташовані вершини І і АБО, повинні чергуватися.

Доцільно обмежити вихідна безліч систем, які використовуються для побудови І - АБО - дерева і подальшого аналізу, тільки найбільш перспективними рішеннями, що володіють високими техніко-економічними показниками. Надалі розширення безлічі варіантів рішень на І - АБО - дереві здійснюється не тільки за альтернативними варіантами технічних рішень (об'єднаних в вузлах АБО), але і по окремим її функціональних елементів (об'єднаним в вузлах І).

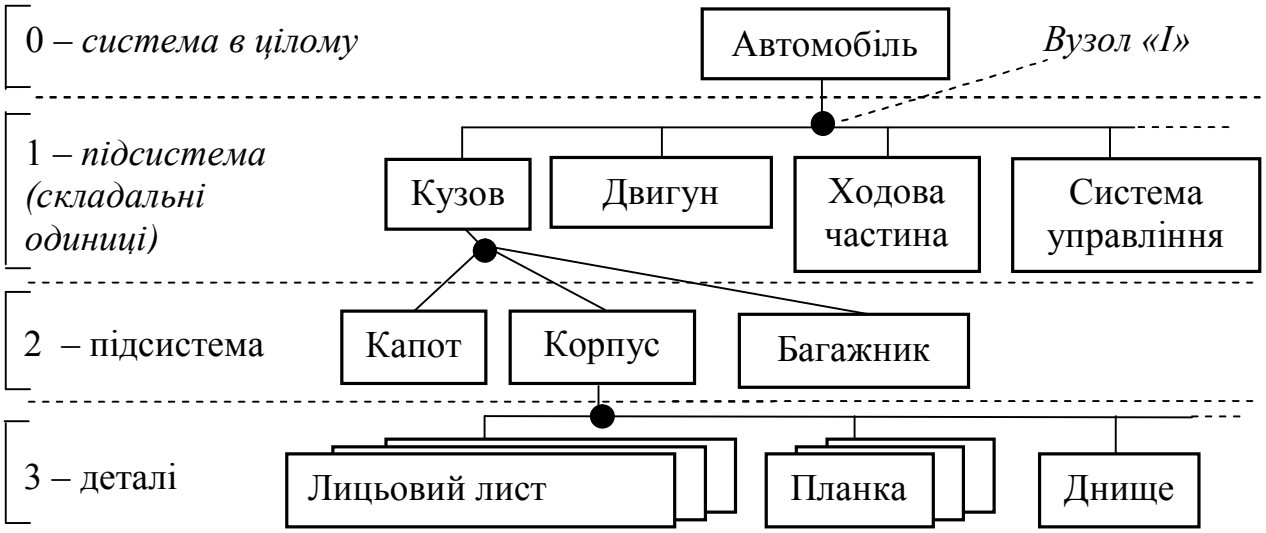



Рисунок 8 - Декомпозиція системи «Автомобіль», представлена у вигляді І - дерева

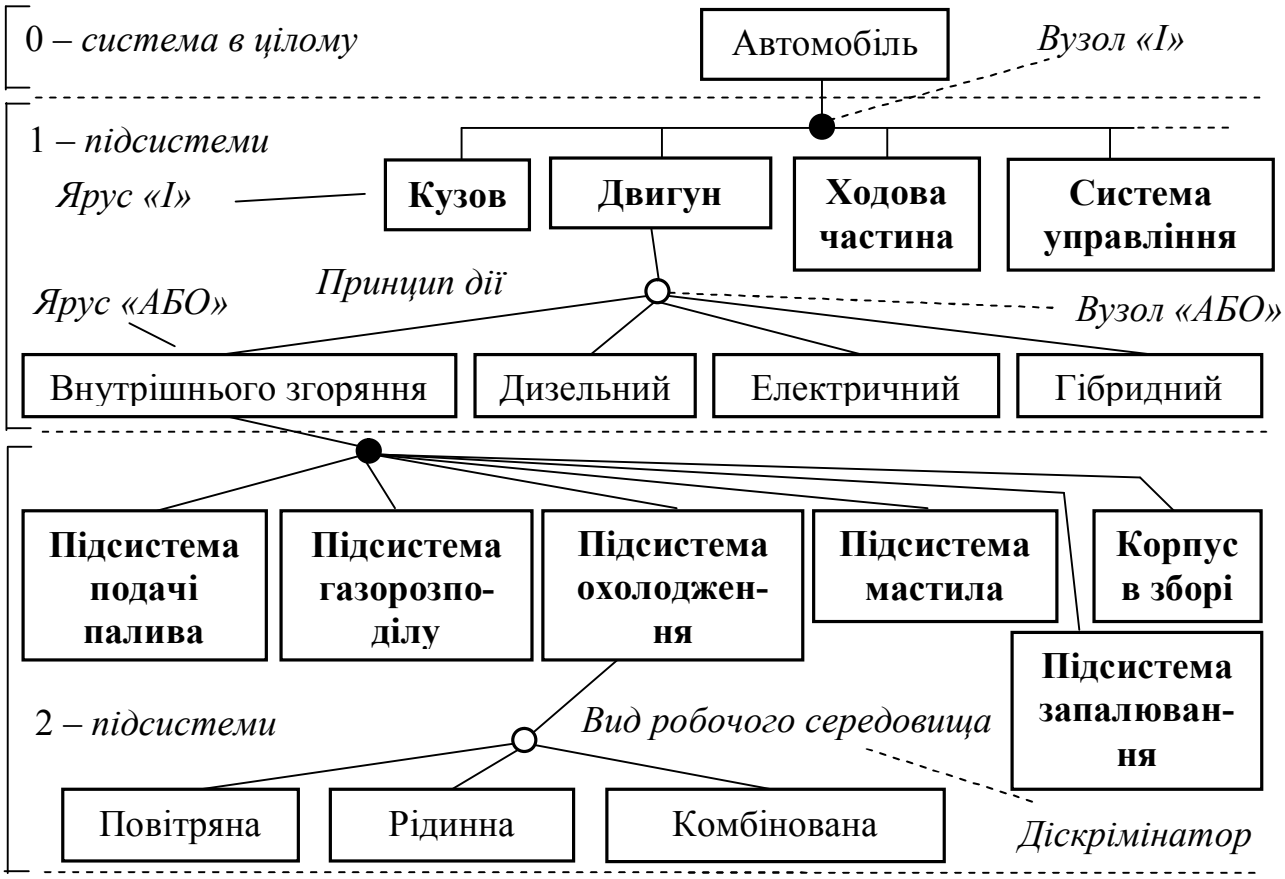



Рисунок 9 - Декомпозиція та узагальнення конструктивних рішень для різних автомобілів, представлених у вигляді І-АБО - дерева

Потужність безлічі рішень, що надаються деревом І-АБО, являє собою кількість варіантів конструкції вузла. Потужність висячої вершини дорівнює 1, потужність вершини типу І дорівнює добутку потужностей підлеглих вершин, а вершини типу АБО сумі. Відповідно, потужність дерева дорівнює потужності кореневої вершини.

Розширення безлічі технічних рішень дозволяє відшукувати на ньому не тільки перспективні існуючі системи, а й нові аналогічно методу морфологічного аналізу. Після проведення декомпозиції по всім функціональним елементам можна перетворювати отримане І-АБО-дерево в різні І-дерева, що відповідають різним конструкціям системи з подальшою оцінкою їх з точки зору поставлених вимог.

Ухвалення проектних рішень, представлених на І-АБО дереві, вимагає врахування великої кількості взаємодіючих чинників різного характеру. При цьому технічні, економічні та інші чинники настільки сильно впливають один на одного, що їх облік доцільно проводити тільки в комплексі. Для обгрунтованого вибору найкращого варіанту технічного рішення необхідна модель, яка забезпечує оцінку якості обраної конструкції. Рішення полягає в декомпозиції самого процесу проектування, використання технічних і економічних критеріїв для оцінки ефективності різних варіантів систем, одержуваних на І-АБО дереві.

І-АБО дерева не є єдиною формою подання ін-формації про систему. Наприклад, SADT-технологія також дозволяє виконувати функціональну декомпозицію системи, а потім розглянути технічні рішення для кожної активності (функції).

Як приклад наведемо також конструктивну декомпозицію редуктора і основні функції виділених елементів конструкції (рис. 10). Основою для конструктивної декомпозиції системи є забезпечення конструктивними елементами функцій і підфункцій, які виконуються системою. В цьому випадку функціональні ознаки об'єкта служать для контролю повноти конструктивних рішень для даної системи.

Розробимо граф зв'язків деталей муфти (рис.11), що відображає конструктивні зв'язки її елементів. Для цього на І-дереві, побудованому для муфти, виберемо рівень абстракції (в даному випадку "Деталі") і розглянемо зв'язку деталей муфти, які виконують її функції (див. Рис. 10). Муфта зчеплення включає дві базові деталі (напівмуфти), які пов'язані з валами електродвигуна і редуктора за допомогою шпонок. Крутний момент між ними передається шпильками (розміщені в одній напівмуфті) і гумовими втулками (розміщені на шпильках в інший напівмуфті). Шпильки закріплені в напівмуфті гайками. Після такого аналізу взаємозв'язку елементів муфти можна показати на діаграмі (графі) (рис. 12). Такі діаграми називають потоковими, так як на них можна показати передачу навантажень (сил, моментів) між елементами муфти [10].

Розглянемо декомпозицію основних функцій муфти (рис. 13), наведених на рис. 10, щоб встановити відповідність функцій і конструктивних елементів, які забезпечують їх виконання при роботі вузла.


Рисунок 10 - Конструктивна декомпозиція редуктора із зазначенням функціонального призначення конструктивних елементів

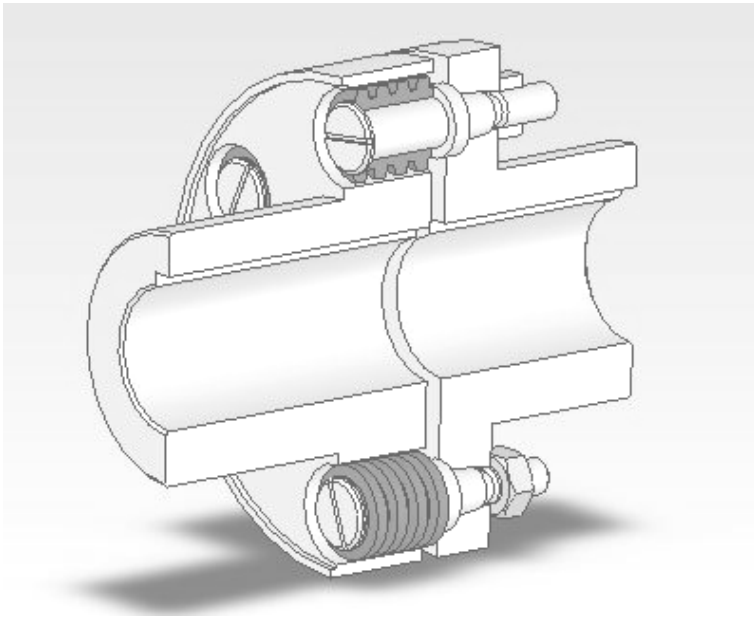


Рисунок 11 – Втулочно-пальцева муфта, що з'єднує двигун і редуктор

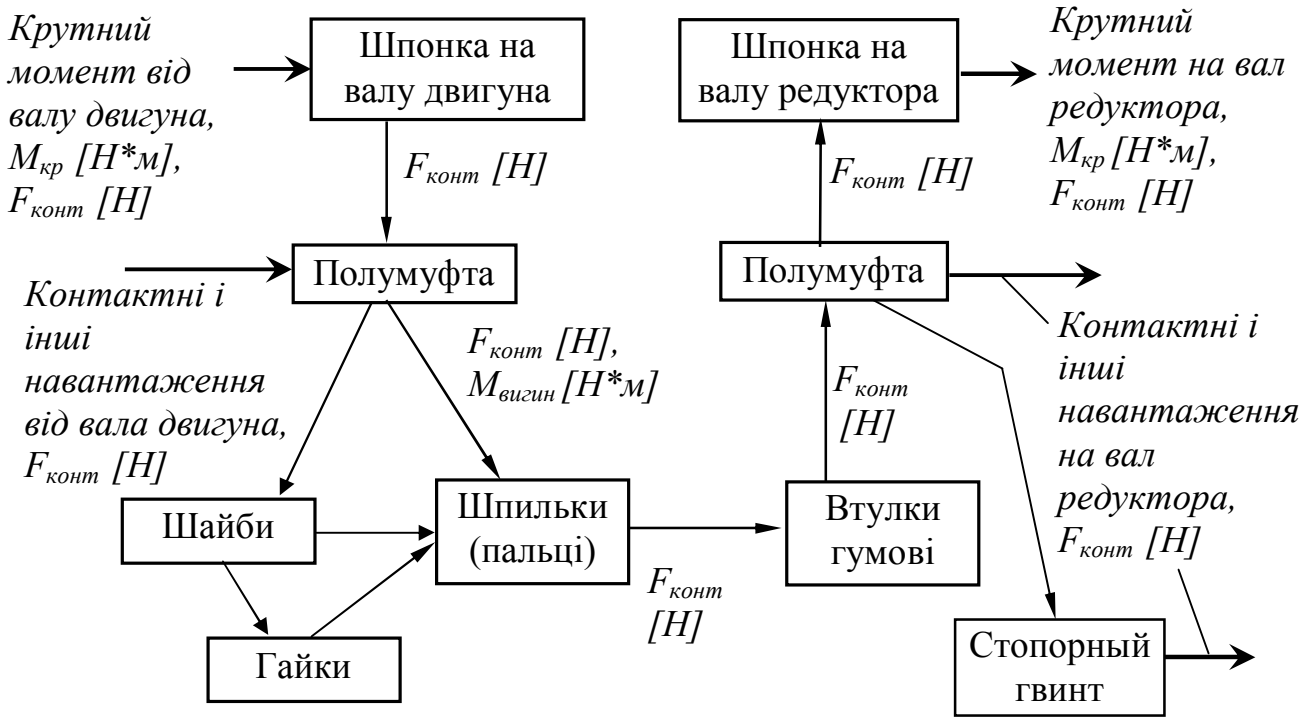



Рисунок 12 - Граф связей и передача нагрузок между элементами муфты

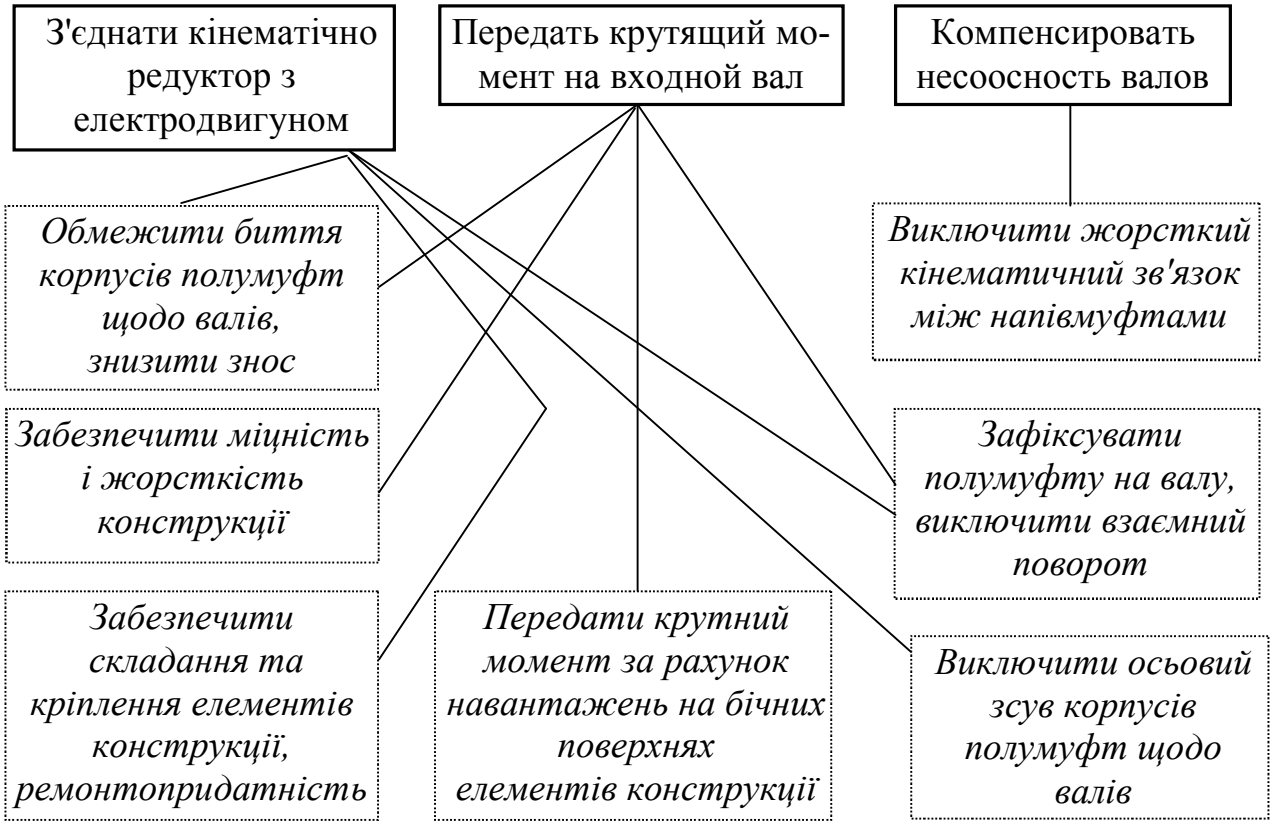



Рисунок 13 - Декомпозиция функций муфты, обеспечивающей соединение вала электродвигателя с входным валом редуктора

Для виконання функції з'єднання вала редуктора з електродвігателем на валах встановлені напівмуфти, які розташовані співвісно валів по посадці, що обмежує биття корпусів полумуфт щодо них. У пазах кожного вала і напівмуфти розташована шпонка, яка за рахунок навантажень на бічних поверхнях, що контактують з валом і напівмуфтою передає крутний момент з валу на полумуфту або навпаки на вал. У масивної напівмуфті, що забезпечує жорсткість конструкції, консольно встановлені і закріплені гайками пальці (шпильки) з гумовими втулками. Це дозволяє виключити жорсткий кінематичний зв'язок між напівмуфтами і компенсувати несоосность валів за рахунок піддатливості гумових втулок, що передають момент на другу полумуфту.

Наведені вище діаграми (див. Рис. 12, 13) можна поєднати, при цьому елементи конструкції можуть розташовуватися у вузлах (див. Рис. 12), а функції (див. Рис. 13) відносяться до зв'язків між ними (об'єктний підхід до декомпозиції ) або навпаки - функції в вузлах, а елементи конструкції - це зв'язок між функціями (процедурно-орієнтований підхід). Як правило, на першому етапі дослідження вироби застосовують об'єктний підхід до аналізу його складу і функцій і конструктивну декомпозицію, хоча метою дослідження може бути і вдосконалення або створення нової сукупності функцій для цього виробу.

Виконати декомпозицію можна не тільки для вироби в цілому, але і для окремих деталей. В якості першого рівня абстракції (ярусу) в цьому випадку використовують "частини (елементи) деталі", що мають певне функ-нальні призначення для деталі в цілому. На другому рівні абстракції розглядають "елементи частин", які забезпечують працездатність елементів деталі першого рівня декомпозиції. Потім на третьому рівні абстракції розглядають поверхні, ребра, точки (графічні примітиви), що утворюють геометрію елементів другого рівня. Таким чином паралельно з конструктивної декомпозицией відбувається і декомпозиція функцій виділених елементів конструкції.

Розглянемо приклад конструктивної декомпозиції і узагальнення конструкцій кришок підшипників редуктора. Кришки підшипників бувають занакладними та прітичного виконання, які кріпляться гвинтами. Розробимо І-АБО - дерево для кришок прітичного виконання, які кріпляться гвинтами. Розділимо базову конструкцію на три функціональних елементи (перший рівень абстракції), які мають конструктивні варіанти:

- тіло кришки - центральна частина кришки, яка герметизує розточення під підшипник;
- кільцева частина - забезпечує центрування кришки в корпусі ре-редуктора;
- фланець - служить для кріплення кришки до корпусу редуктора.

Конструктивні варіанти цих елементів є елементами - наповнювачами для кришки при узагальненні конструкції деталі. Наприклад, вал може виходити з корпусу редуктора, тоді кришка повинна мати отвір і проточку для установки ущільнення. У більшості конструкцій тіло кришки має плоску форму, але застосовують і купольну конструкцію тіла кришки, що дозволяє підвищити міцність конструкції. Конструкції фланця кришки також бувають різної форми (з

Бонк, расточками під гвинти з круглою головкою і ін.) При різних варіантах кріпильних виробів.

Розробка узагальненої конструкції деталі дозволяє автоматизувати синтез конструкції кришки при заданих умовах експлуатації, а так само розробляти технологічний процес виготовлення деталі за допомогою ЕОМ.

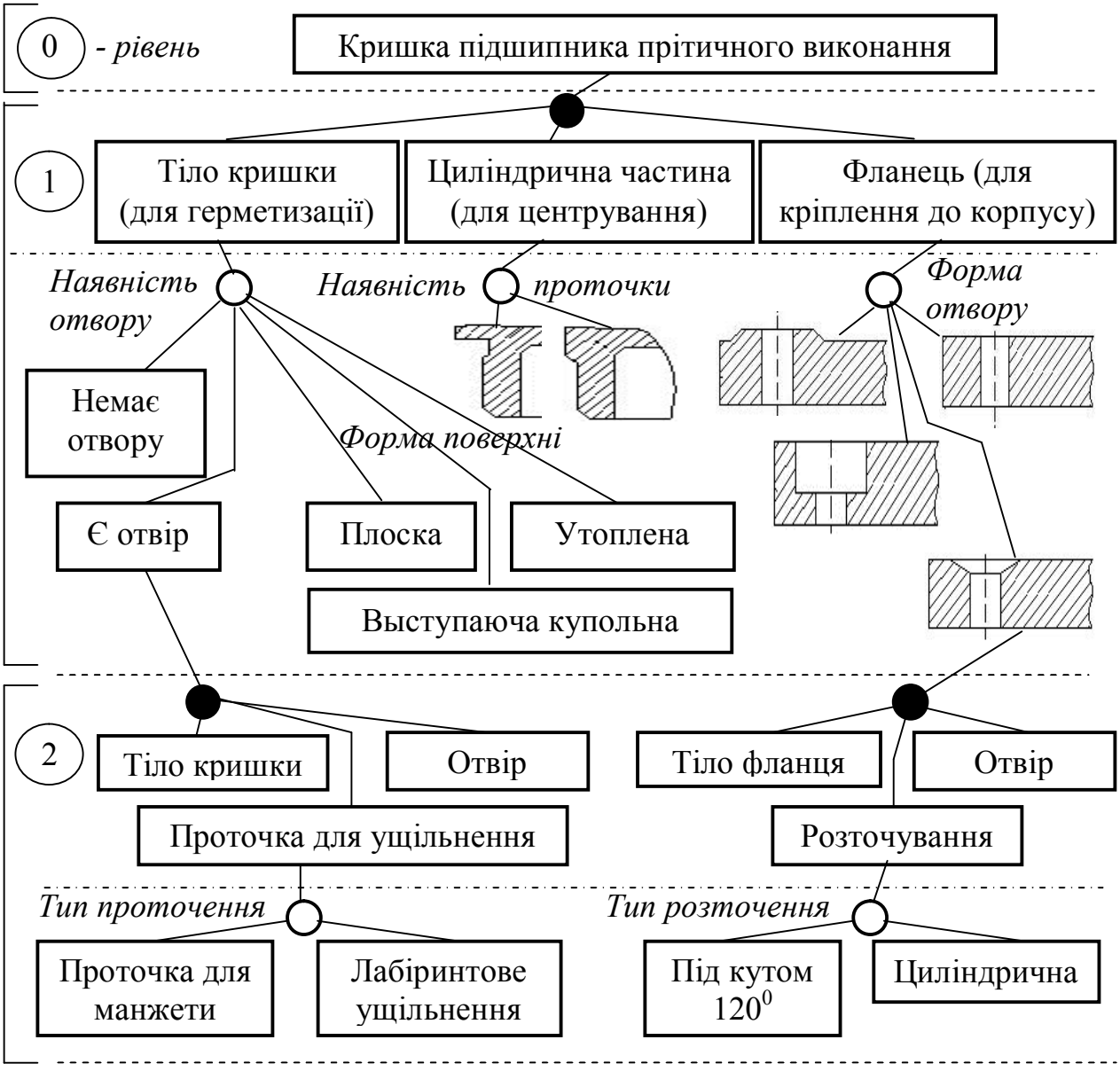



Рисунок 14 - Декомпозиція та узагальнення конструкції деталі «Кришка підшипника», представлені у вигляді І - АБО - дерева

В якості ще одного прикладу об'єктного підходу до декомпозиції розглянемо взаємодію інструменту і заготовки при її обробці рис. 15. Для ефективного вибору методу обробки вивчимо систему «інструмент-заготовка» з використанням методів системного аналізу. Виявлення загальних процесів, що відбуваються в таких системах, дозволяє більш широко і цілеспрямовано використовувати досвід, накопичений в аналогічних технічних системах для вдосконалення технологічних процесів обробки, а також конструкції інструменту.

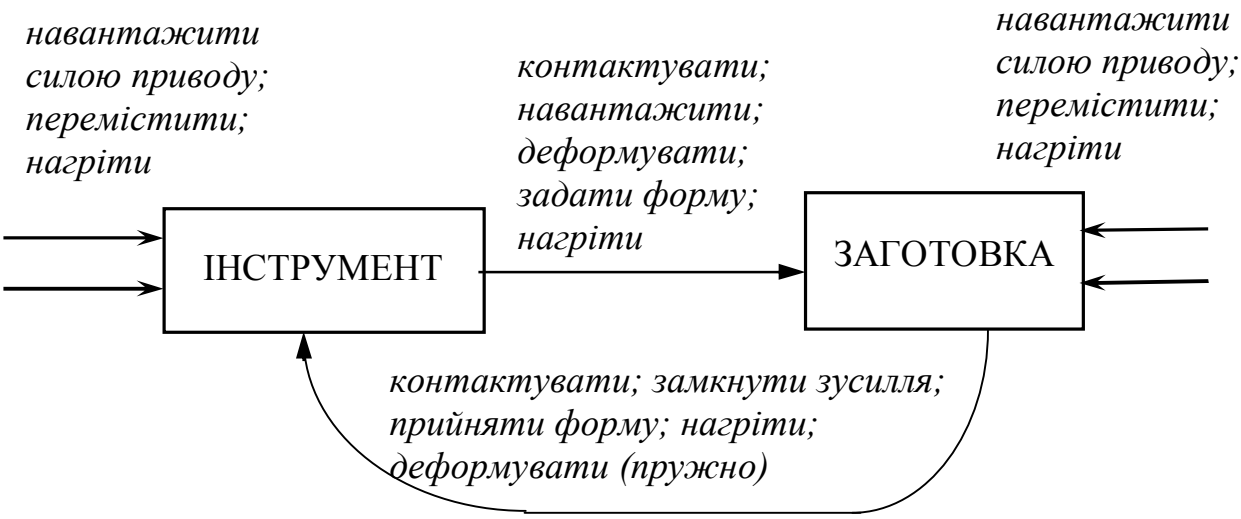



Рисунок 15 - Об'єктна декомпозиція системи «інструмент-заготовка»; позначення і характеристики параметрів наведені у табл. 1

У процесі впливу інструменту на заготовку реалізується визна-ленна послідовність операцій (функцій), що супроводжуються низкою фізичних явищ. Тому виділимо фактори (назва, позначення, розмірність і інші характеристики, табл.1), які характеризують взаємодію інструменту і заготовки і розглянемо ступінь їх впливу на процес обробки. Обмеженням для застосування нових методів обробки є остаточна форма деталі, що визначається, як правило, поза технологічного процесу.

Таблиця 1. Чисельні показники, що характеризують функції елементів системи «інструмент-заготовка»

| Функції елементів | Чисельні показники, розмірність |
|---|---|
| Навантажити силою приводу | сила $P_1$, Н |
| Контактувати | площа F, $м^2$; напруга $\sigma_2$, Па |
| Навантажити | сила $P_2$, Н; напруга $\sigma_1$, Па |
| Замкнути силу | сила $P_3$, Н |
| Нагріти | температура t, $^0C$ |

| Функції елементів | Чисельні показники, розмірність |
|---|---|
| Перемістити | шлях S, м; коефіцієнт тертя, $\mu$ |
| Задати форму | лінійні розміри $L_1$, м; координати $X_i$, м; кути $\alpha_1$, радіани (градуси) |
| Деформувати | ступінь деформації ε, %; хід h, м; коефіцієнт тертя, $\mu$ |
| Прийняти форму | лінійні розміри $L_2$, м; координати $X_j$, м; кути, $\alpha_2$ радіани (градуси) |

Виконання послідовності операцій, що визначають спосіб впливу на заготовку, супроводжується якісним і кількісним зміною факторів, які характеризують взаємодію інструменту і заготовки. Створення нової сукупності технологічних впливів на заготівлю пов'язана з введенням нових факторів, що інтенсифікують процес обробки (якісна зміна функцій). Удосконалення існуючого методу пов'язано зі зміною ступеня впливу параметрів (кількісне зміна), що характеризують вплив на заготовку.

Наведені вище графічні уявлення системи як правило недостатні для її моделювання в програмній системі. Подальша деталізація розгляду об'єктів проводиться шляхом опису сценарію розвитку предметної області (опису «життя» системи). На основі сценарію розвитку системи створюється словник предметної області (тезаурус), в якому наводяться визначення всіх термінів (понять предметної області). Зокрема, бажано виділити і перерахувати ознаки, характеристики, функції, параметри, стану об'єктів системи, а також чисельні показники властивостей і функцій системи.

Представлені взаємодії в системі (див. рис. 14) не відображають її динамічних властивостей, пов'язаних з тим, що процеси відбуваються в часі. Виділення динамічних аспектів необхідно, оскільки залежності між параметрами і ступінь їх впливу можуть змінюватися в процесі виконання технологічних операцій. Завдяки цьому по-є можливість управління технологічним процесом, наприклад, примусовий зсув інструменту може бути не пов'язане безпосередньо з кінематикою деформування (зворотно-поступальний, вібраційне зміщення, обертання при поступальному русі і т. д.). Прикладом реалізації такого підходу в обробці заготовок тиском є термофрикційної штампування, при якій інструмент впроваджується в заготовку з обертанням. При цьому за рахунок тертя відбувається локальне по-щення температури і пластичності металу, зниження зусилля деформування. У процесах різання тертя і температуру знижують і стабілізують за рахунок застосування емульсії (мастила), яку подають в зону контакту інструменту і заготовки, що є загальним методом зниження тертя в техніці.

В даному прикладі основні методи формозміни заготовок можна представити у вигляді:

- набору елементів системи навантаження заготовок;
- набору функцій і діючих факторів (як характеристик функцій);
- уявлення топології системи, як сукупності елементів і функцій, які знаходяться між собою в певному відношенні (структурне уявлення системи);
- послідовності виконання функцій і дії чинників (тимчасової, поведінковий аспект);
- значень параметрів (факторів, чисельних показників), які характеризують виконання функцій.

У сукупності ці види описів методів формозміни заготовок формують загальну модель даного процесу або предметної області в цілому.

Кожна функція може характеризуватися одним або декількома параметрами. Ступінь впливу функціональних факторів системи на технологічний процес визначається чисельними значеннями їх параметрів. Раціональне управлін-

ня впливом факторів (збільшення, зниження, стабілізація) призводить до модернізації процесу, досягнення необхідного технологічного ефекту. Розглянемо, наприклад, вплив фактора «тертя» (коефіцієнт тертя), що діє на контакті інструменту і заготовки (при реалізації функцій - «Перемістити» і «Деформувати»).

Як видно з представленого дерева (рис. 16) виділення і аналіз впливу на процес факторів дає можливість перейти до узагальнення технологічних ефектів, які вони створюють, і виявлення фізичних явищ, які лежать в основі цих ефектів. Таким чином забезпечується можливість структуризації і узагальнення інформації про системи, наповнення та застосування баз знань в конкретній предметній області.

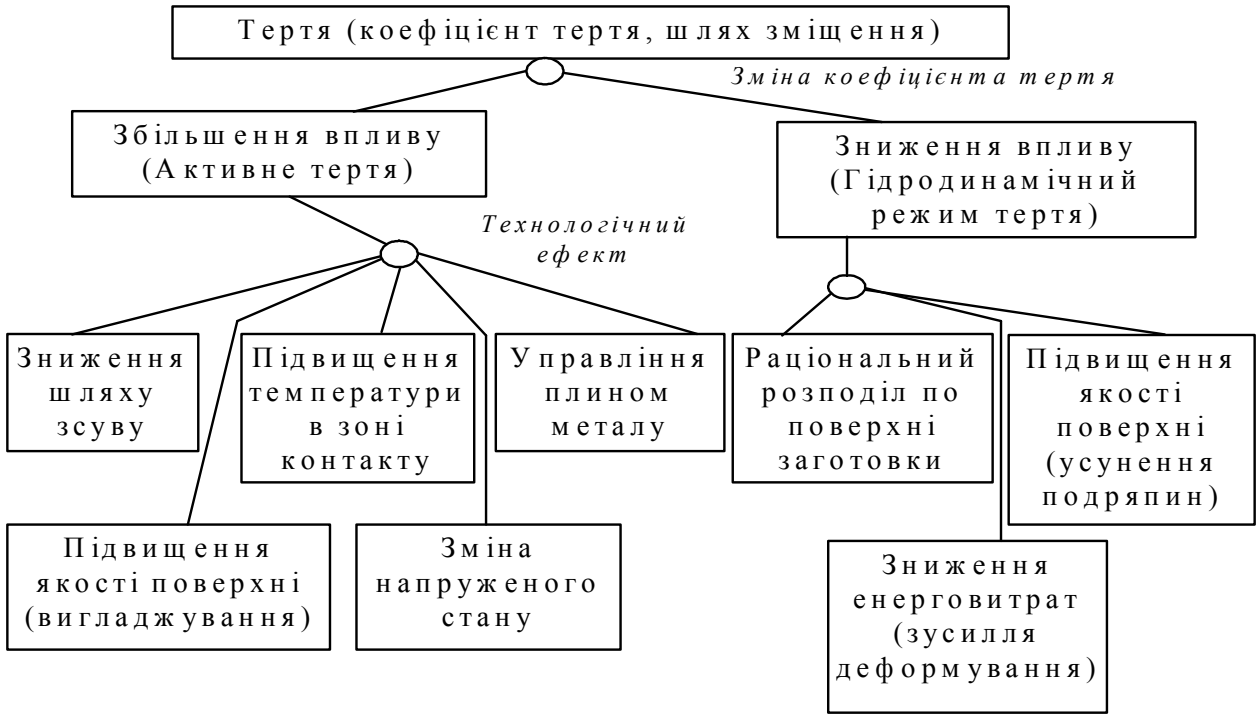



Рисунок 16 - Вплив тертя в технологічних процесах обробки заготовок інструментом

Таким чином, розгляд об'єкта на етапі аналізу його функцій і структури здійснюється на основі наступного загальної схеми (рис. 17).

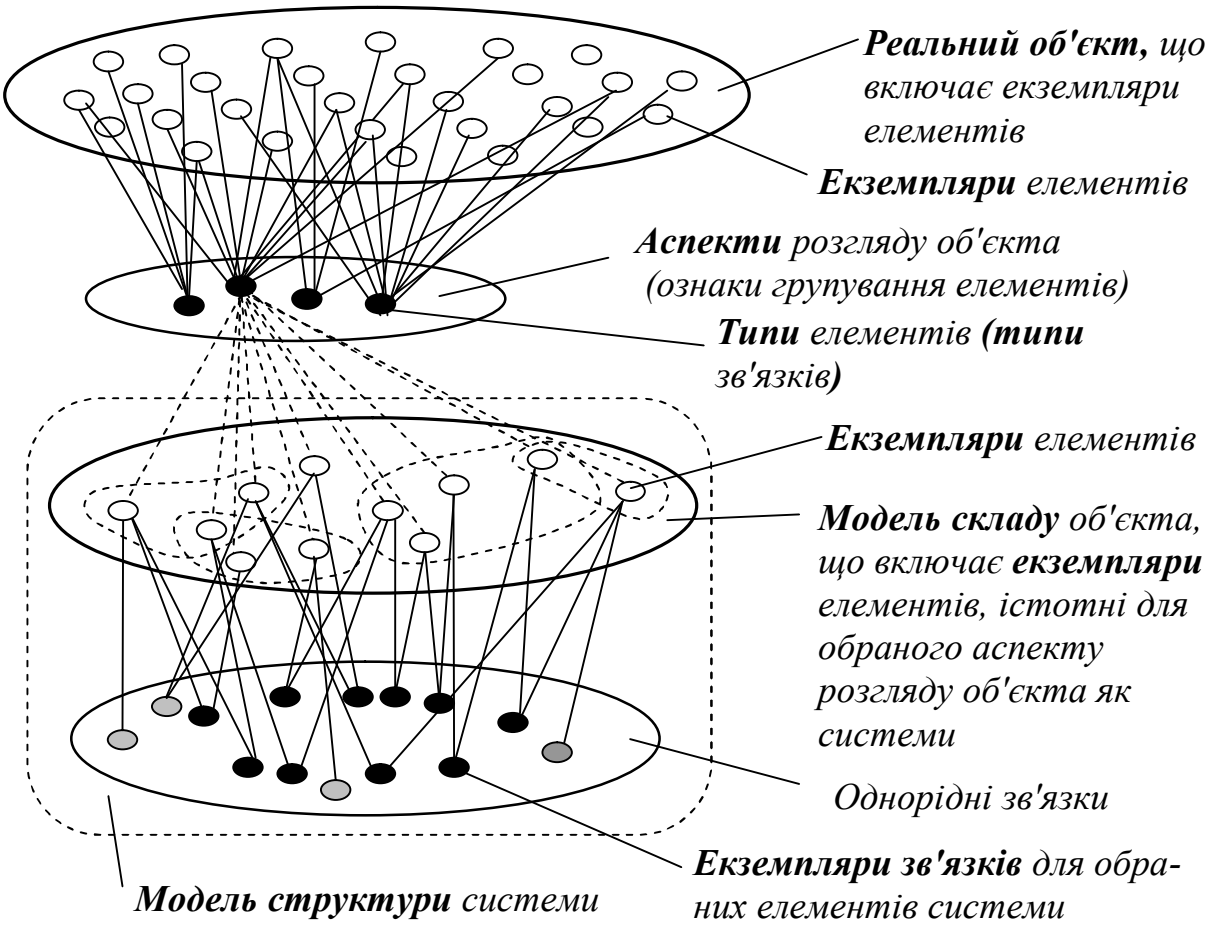



Рисунок 17 – Спрощення об'єкта при вивченні його структури і розробки моделей

1. Реальний об'єкт включає ряд підсистем елементів різної фізичної природи і призначення, які в сукупності забезпечують різні функції об'єкта. Для спрощення розгляду реального об'єкта абстрагуються від ряду його несуттєвих властивостей шляхом виділення різних аспектів розгляду цього об'єкта. Таким чином виділення аспекту розгляду дозволяє сформулювати ознаки групування елементів об'єкта в пересічні підмножини.

2. Аспект розгляду об'єкта практично відповідає виділенню однорідних елементів і зв'язків (одного типу) між елементами.

3. Елементи, які є важливими для обраного аспекту розгляду об'єкта, утворюють модель складу об'єкта, причому аспект розгляду в рамках розв'язуваної задачі визначає і моделі елементів, тому що виділяє важливі для побудови моделі системи параметри кожного елемента. Слід зазначити, що ступінь деталізації системи залежить від виду завдання, тому кількість елементів моделі не завжди збігається з кількістю елементів реальної системи (примірників елементів).

4. Виділення окремої площині (рівня, шару) для демонстрації зв'язків «зрівнює» їх з точки зору уявлення з вузлами, як елементами структури систе-

ми. У загальному випадку ми отримаємо кілька шарів зв'язків для різних аспектів.

5. Сукупність вибраних елементів системи і зв'язків утворюють модель структури системи.

6. Надалі, виділення ознак класифікації і поділ обраних елементів об'єктів на групи необхідно для уточнення їх типів, що потрібно для побудови програмної системи, що моделює об'єкт. Відповідно для уточнення видів зв'язків потрібно і класифікація зв'язків. Важливість зв'язків збільшується зі збільшенням складності системи, тому що росте кількість аспектів, пов'язаних з процесами її функціонування.

Таким чином, декомпозицію системи виробляють на основі блочно-ієрархічного підходу, виділяючи аспекти її розгляду для спрощення моделі і встановлюючи зв'язки між її елементами на обраних рівнях абстракції, що дає можливість в кожному випадку будувати прийнятні за складністю і точності моделі системи і виконувати аналіз її функціонування.

## 1.5 Можливості автоматизації вирішення завдань класифікації і представлення структури систем

Ряд перерахованих дій (етапів вивчення системи) в процесі формалізації інформації про систему може бути автоматизований. Застосування комп'ютерної техніки в процесі вивчення систем, класифікації дозволяє [12]:

а) формалізувати послідовність етапів;

б) формалізувати і візуалізувати уявлення інформації про систему у вигляді графів, дерев;

в) надавати можливість використання баз даних і знань, на-приклад, для доповнення ознак (розширити пошукове поле), забезпечити виведення довідкової інформації;

г) зберігати всю інформацію про систему в електронному вигляді;

д) оперативно візуалізувати і модифікувати уявлення інформації, забезпечити виведення документації по системі;

е) пропонувати варіанти рішень в діалозі або автоматично на основі аналогів, зокрема, виконувати пошук конструкцій (варіантів) на І-АБО- дереві;

ж) оцінювати варіанти рішень з використанням моделі і виконувати оптимізаційні розрахунки, забезпечити контроль на будь-якому етапі роботи і знизити ймовірність помилок;

з) надавати експертам можливість групового (мережевого) режиму роботи.

В цілому автоматизована система повинна забезпечувати підтримку прийняття рішень [1, 2]. Тому корисно виділити ті параметри, які можна автоматично контролювати.

Слід зазначити незалежність алгоритмів обробки даних від про-виходи вихідної інформації, тому можлива, наприклад, експертна оцінка вихідних даних, створення прототипів (списків типових елементів, прикладів їх взаємодії). Усунення помилок при класифікації може бути вироблено за рахунок таких ви-

дів автоматичного контролю:

а) контроль порожніх підмножин;

б) контроль повноти функцій;

в) контроль повноти частин системи (забезпечення реалізації всіх функцій);

г) контроль надмірності елементів;

д) контроль зв'язності графів, виділення фізично неоднорідних підсистем;

е) контроль специфікацій по вузлах і ребрах графа.

Таким чином, розробка автоматизованих засобів формалізації інформації про систему, створення баз даних і знань в потрібних предметних областях, засобів підтримки класифікацій дозволяє здійснити цілеспрямоване вдосконалення різних об'єктів (конструкцій, технологічних процесів та ін.) Шляхом зміни діючих факторів і ступеня їх впливу. Використання таких засобів дає можливість наочно уявити структуру системи, виявити важливі технологічні чинники, загальні властивості, притаманні конструктивних елементів, загальні принципи їх використання.

Формалізація підходу до розгляду системи дає можливість розробити загальні автоматизовані алгоритми її вивчення. Зокрема, при системному підході до моделювання для ряду об'єктів, що становлять механічні, пневматичні, гідравлічні, електричні та інші системи використовується метод розробки еквівалентних схем і полюсних графів заміщення [13]. Загальний підхід заснований на електроаналогіях в описі процесів функціонування цих систем і дозволяє використовувати загальні алгоритми для їх аналізу.

# 2 МОДЕЛІ ПРОЕКТУВАННЯ СКЛАДНИХ ТЕХНІЧНИХ СИСТЕМ

## Сучасне погляд на процес і об'єкт проектування

Створення нових конструкцій відноситься до завдань структурного синтезу. Обчислювальна підтримка прийняття проектних рішень пов'язана з необхідністю визначення статичних, динамічних, вартісних характеристик проектованих об'єктів і систем, вирішення завдань інформаційного забезпечення проектування, оптимізації складу і параметрів проектованої системи. Особливе місце при цьому займають моделі проектованих систем, що дозволяють досліджувати основні особливості реальної поведінки різних варіантів проектованої системи в характерній для неї середовищі. При цьому для різних варіантів конструкції можна проаналізувати експлуатаційні характеристики як окремих елементів, так і всієї системи в цілому. У ряді випадків це призводить до того, що фізичне моделювання системи стає непотрібним.

## 2.1 Становлення науки про проектування

В процесі проектування людина створює в своїй уяві визначенну модель об'єкта проектування, яку він згодом реалізує. В процесі виготовлення виробу люди здійснюють спільну (кооперативну) діяльність, в якій виникла необхідність передавати інформацію про предмет праці всім, хто бере участь в роботі. Необхідність спілкуватися привела до розвитку формалізованих мов опису майбутньої конструкції у вигляді креслень, схем, словесного опису, фізичних моделей-прототипів, які спочатку грали основну роль при конструюванні.

Проектування можна розбити на кілька стадій, які об'єднуються в дві повторювані в процесі розвитку системи метапроцедури:

1. Постановка задач, пошук аналогів і прийняття проектних рішень.

2. Формалізований опис системи та поетапне перетворення описів об'єкта, які дозволяють виконати його виготовлення в заданих умовах.

Виконання першої метапроцедури формалізовано значно менше другий, частина питань відноситься скоріше до області творчості, ніж до науки. Це пов'язано з аналізом неструктурованої або слабоструктурованої інформації в різних областях знань, її класифікацією, аналізом тенденцій та іншої (творчої, евристичної) роботою.

Виконання другої метапроцедури регламентовано набагато повніше, існують: єдина система конструкторської документації (ЄСКД), єдина система технологічної документації (ЄСТД), класифікатори форм деталей, системи нормування та інші. Програмні вироби теж є складними системами, тому їх проектування регламентується технологічними процесами, ГОСТами, ЄСПД.

Розвитку методів проектування йде по шляху послідовної формалізації, алгоритмізації етапів і операцій, однак це не знижує вагомість і важливість творчого підходу в проектуванні.

Проектування йде від створення схеми об'єкта до побудови його конструктівних форми і потім до забезпечення технічних, технологічних, тих-нико-

економічних та інших показників, обмежень, що вимагає розвитку відповідних методів моделювання. Моделювання пов'язано з перетворенням реальної системи в абстрактну, що пов'язано з рішенням окремих задач, в рамках яких допустимо спростити уявлення системи у вигляді окремих розрахункових схем. Зокрема, при вирішенні задач геометричного моделювання розглядаються структурні, кінематичні, динамічні і геометричні (конструктивні, просторові) завдання аналізу і синтезу і враховуються відповідні обмеження.

Приклад: фундаментальні дослідження академіка Артоболевського в області ТММ. Проектування розглядається як складна комплексна про-блема, рішення якої рекомендується розбити на кілька самостійних етапів:

1. Встановити основну кінематичну схему механізму, забезпечую-ного необхідний вид і закон руху.

2. Розробити конструктивну форму механізму, яка забезпечить його міцність, довговічність, високий ККД і т. д.

3. Досягнення технологічних і техніко-економічних показників проектованого механізму, що визначаються експлуатацією у виробництві і ремонтом.

ТММ бере на себе етап 1, при цьому враховуються питання 2 і 3 етапів. Розділ ТММ, присвячений методам проектування, носить назву синтезу механізмів. Основні завдання синтезу:

а) перетворення обертального руху навколо однієї осі під обертальний рух навколо іншої;

б) перетворення обертального руху в поступальний рух;

в) перетворення поступального руху уздовж однієї осі в поступальний рух уздовж іншої;

г) відтворення однієї з точок механізму необхідної траєкторії.

Рушійною силою розвитку технічних систем (ТС) є суперечності, які виникають при спробах удосконалення системи між її елементами (нерівномірність розвитку елементів всередині системи), або між її елементами і зовнішнім середовищем при функціонуванні системи.

Відбуваються якісні зрушення в проектуванні, пов'язані з підвищенням складності технічних систем, застосуванням нових фізичних і хімічних явищ і ефектів, мініатюризацією датчиків і систем обробки інформації. Збільшення кількості підсистем призводить до ускладнення процесу проектування, тому що виникла необхідність у фундаментальних знаннях в різних областях науки.

Істотно зросли темпи зміни елементної бази для створення ТС, що скоротило временя морального старіння ТС, яке стає таким же часом проектування. З одного боку пошук нових перспективних технічних рішень в умовах традиційних методів проектування ускладнюється через постійне зростання обсягу науково-технічної інформації. З іншого боку механістичне створення (МС) з нових елементів і підсистем з високими показниками не приводить до автоматичного підвищення їх ефективності. Вивчення тенденцій розвитку систем показує, що для створення сучасних виробів потрібні знання не тільки в області в області конструювання механічних систем, але і в електромеханіки, електротехніці, в теорії і техніці управління, електроніці, цифровій схемотехніці, програмуванні. Персональні комп'ютери та сучасне програмне забезпечення дозволяють моде-

лювати, проектувати і досліджувати технічні та організаційні системи, обладнані системами автоматизації і управління. Активно розвивається нова область науки і техніки, орієнтована на створення і експлуатацію складних динамічних автоматичних і автоматизованих систем з комп'ютерним (мікропроцесорним) управлінням їх рухом, яка була названа мехатронікою. Областю практичного застосування мехатроніки є створення високоточних, надійних і багатофункціональних систем управління об'єктами починаючи від космічних апаратів до побутової техніки.

Розвиток ТС, інтеграційні процеси в техніці потребують удосконалення методів проектування, що забезпечують якість створення складних ТС. З точки зору автоматизації проектування складних ТС найбільш актуальними є питання інформаційного забезпечення першої метапроцедури проектування, пов'язаної з пошуком і прийняттям технічних рішень. Це вимагає розвитку методів практичного застосування БД та БЗ, інтеграції їх з методами штучного інтелекту (ШІ) і сучасними технологіями розробки програмного забезпечення.

## 2.2 Огляд досліджень в області методології проектування

Завдання пошуку технічних рішень можна віднести до евристичних, так як вони не піддаються формалізації за допомогою відомих математичних і логічних методів [10].

Ідеї комбінаторики знайшли відображення у відомому методі «морфологічного аналізу» американця Цвикки [7]. На думку Ханзена в творчості необхідна систематизація. Область можливих рішень може бути обстежена і будь-яка нова ідея виявиться комбінацією відомих елементів. Дослідження - це рух від явища до сутності, проектування - навпаки. Основним принципом побудови схеми є поділ, завдяки якому процес створення технічного об'єкта розбивається на послідовні етапи. Виділяється 4 етапи проектування:

1. Виділення ядра, в якому знаходиться сукупність всіх можливих рішень.
2. Опис можливих робочих принципів і функцій, що підлягають виконанню.
3. Аналіз недоліків у виконанні функцій, шляхом їх усунення приходять до поліпшеним робочим принципам.
4. Вибір поліпшеного робочого принципу, що має найменше число недоліків.

Хілл каже про інженерне проектування, як про особливу науку, що систематизує знання і приділяє особливу увагу етапам проектування і їх взаємозв'язку. Методика проектування - це не формула і навіть не інструкція, а послідовність подій, що становлять процес проектування, в рамках якого можливо логічний розвиток конструкції. Основні етапи проектування зводяться до наступного:

- визначення потреби (зіткнення конструктора з ситуацією, яка дратує і хвилює, в результаті чого у нього виникає потреба в зміні існуючого положення);
- визначення мети (формулювання в загальних виразах характеристик об'єкта, що проектується, які задовольняють цю потребу);
- наукові дослідження (збір всієї доступної інформації для вирішення завдань, що випливають з поставленої мети);

- формулювання завдання (перелік даних та параметрів, що забезпечують досягнення заданої мети);

- формування ідей (процес народження нових ідей);

- вироблення концепцій (вироблення варіантів для досягнення поставленої мети);

- аналіз (перевірка вибраних концепцій на відповідність фізичним законам);

- експеримент (створення дослідного зразка і лабораторні випробування);

- рішення (опис проектованого об'єкта: робочі креслення, технічні умови);

- виробництво (визначення обсягу виробництва і потреби в виробничому обладнанні, методи виготовлення продукції, календарне планування, контроль якості...);

- розподіл продукції (встановлення конкурентоспроможності цін, реклама, ринки збуту, забезпечення прибутку);

- споживання (контакти зі споживачами, ремонт, обслуговування).

*Методи проектування* покликані організувати процес. Основним з них є:

а) наочне уявлення заданої функції, що служить перехідною ланкою між поставленим завданням і її рішенням і сприяє розширенню інформаційної основи творчості;

б) діаграма ідей, що дає наочне уявлення про розвиток техніки в області, що цікавить;

в) матриця ідей, що представляє собою засіб морфологічного аналізу незалежних змінних і дозволяє виробляти різні поєднання характеристик об'єкта, що проектується, що породжує альтернативні ідеї;

г) мозковий штурм, що дає можливість отримання нових ідей шляхом творчої співпраці групи фахівців;

д) синектика - відрізняється від мозкового штурму тим, що обговорення ведеться в напрямку пошуку невеликого числа ідей (2-3), однак з детальним їх розглядом групою фахівців різних професій.

Особливо потрібно зупинитися на методі, пов'язаному з прийняттям найкращих рішень з сукупності варіантів. Він заснований на побудові матриці рішень. Суть методу полягає у виборі критеріїв для порівняння варіантів, визначенні їх відносної значущості (ваги в частках одиниці) і оцінки варіантів по кожному з критеріїв за десятибальною системою. Найкращим буде той варіант, у якого найбільша сума добутків оцінок на відносну вагу відповідних критеріїв. Метод дає можливість порівнювати варіанти, представлені лише принциповими схемами, що не містять параметричної інформації. Особливе значення Холл надає морфологічному підходу до проектування, пов'язаному з логічною організацією ідей, що відрізняє його від традиційного підходу, заснованого лише на інтуїції і досвіді. Значний арсенал нових методів проектування пропонує Джонс. На його думку проектування слід розуміти як «процес зміни в штучному середовищі». Підкреслюється, що проектування полягає не тільки у виготовленні робочих креслень, а й в плануванні всіх етапів існування майбутнього виробу. Таким чином проектування зачіпає соціальні, політичні, економічні та інші ситуації виникають в результаті впровадження проектованого об'єкта. Загальною відмінною рисою всіх нових методів проектування є прагнення наочно

уявити і систематизувати процес мислення. Досягається це використанням будь-якої схеми, що дозволяє розбити задачу на частини і вказати взаємні зв'язки між цими частинами, залучити до процесу всіх зацікавлених членів суспільства. Методи проектування можна розбити на групи в залежності від основних концепцій:

- чорний ящик;
- система, що самоорганізується;
- прозорий ящик.

У першому випадку проектувальник не може пояснити отримане вдале рішення (мозковий штурм, синектика). Концепція позрачний ящика побудована на припущенні, що проектувальник усвідомлює свої дії і їх причини. Логічна або систематична поведінка проектувальника включає:

- аналіз одержуваної і наявної інформації;
- синтез технічних рішень;
- їх оцінку і повторення циклів до отримання найкращого з можливих результатів.

Загальні риси методів, побудованих на концепції «прозорого ящика»:

1. Цілі, змінні і критерії задаються заздалегідь.
2. Пошуку рішення передує проведення аналізу.
3. Оцінка результатів дається в словесній формі і побудована на логіці.
4. Заздалегідь фіксується стратегія (використовуються послідовні прийоми, умовні і циклічні операції).

Головною умовою застосування цієї концепції є можливість поділу завдання на окремі частини, кожна з яких вирішується самостійно. На думку Джонса мета методології проектування - зменшення циклічності і збільшення лінійності проектування. Циклічність пов'язана з вимушеним повторенням етапів роботи в зв'язку з тим, що деякі, виявленні згодом частини завдання не були враховані. Лінійність припускає, що всі найважливіші проблеми можна виявити з самого початку. Істотним методом забезпечення лінійності Джонс вважає *прогнозування*, що дозволяє визначити вихідні параметри етапів до їх виконання.

Підхід до проектування, як до самоорганізуючої системі викликаний прагненням звузити область пошуку рішень за рахунок обґрунтованого вибору стратегії. Для цього потрібна метамова (набір термінів досить широких за значенням, щоб можна було описати залежності між стратегією і проектної ситуацією, а також проводити оцінку моделі, що дозволяє передбачити можливі результати).

Джонс виділяє три етапи проектування: *дивергенцію, трансформацію, конвергенцію*. *Дивергенція* - розширення меж проектної ситуації з метою забезпечення досить великого простору для пошуку рішень. Дивергенція більше пов'язана з дослідженням, ніж з проектуванням. *Трансформація* - стадія створення принципів і концепцій. На цій стадії виникає загальна концептуальна схема проектованого об'єкта. *Конвергенція* - стадія остаточного вибору варіанта технічного рішення. Саме тут можуть найбільшою мірою використані засоби автоматизації проектування. Стратегії проектування можуть бути лінійними, циклічними, розгалуженими (дозволяють виконувати окремі етапи паралельно),

адаптованими (вибір наступного етапу залежить від результатів виконання попереднього), випадкового пошуку. Не обов'язково дотримуватися однієї стратегії. Одній стратегії слідують до тих пір, поки вона перспективна, потім її замінюють відповідно до обстановки.

Хол в методології виділив 6 процедур - *з'ясування завдання, вибір цілей, синтез систем, вибір найкращих альтернатив, планування дії*. Методами *процедури з'ясування завдання* є - дослідження потреб і оточення, метод входів і виходів. Дослідження потреб пов'язано з визначенням вимог до проектованої системи, на основі яких складається загальна програма розробки. При цьому розглядаються чотири основні напрямки планування проектів:

- розширення і оновлення функцій;
- поліпшення технічних характеристик;
- зниження вартості;
- поліпшення зовнішніх якостей.

Під оточенням Холл розуміє безліч всіх предметів поза системою, зміна ознак яких впливає на систему, а самі ознаки змінюються внаслідок поведінки системи. Основними факторами оточення вважаються: стан технології, природне оточення (клімат, рослинність), пріоритети фірми, економічні умови для нових систем, людський фактор. Успіх проектування вимірюється ступенем інтеграції з оточенням.

*Вибір цілей* Хол вважає однією з найважливіших процедур проектування. Цілі мають відповідати системі цінностей, прийнятої для проектованої системи. Однак ряд елементів системи цінностей виявляється загальним для всіх систем (прибуток, ринок, вартість, якість, технічні характеристики, сумісність з існуючими системами, стійкість проти морального старіння, простоту і витонченість, безпеку в обслуговуванні, час на розробку). В якості *методів синтезу* систем Холл пропонує мобілізацію ідей і функціональне проектування. Перше означає збір всіх відомих альтернатив і вироблення нових (без поспішної критики). Завжди повинна бути більш ніж одна альтернатива рішення задачі. Функціональне проектування представляє найбільш загальний підхід до опису систем. Визначаються граничні умови, бажані входи і виходи, складається докладний перелік функцій або операцій, які повинні виконуватися. Метод в спрощеному вигляді зводиться до складання блок-схеми системи. *Процедура аналізу системи* полягає у виділенні всіх можливих наслідків з альтернативних систем для вибору з них найкращого. При аналізі деякі відомості виходять об'єктивно (шляхом збору досвідчених даних і знаходження розподілу частот), інші суб'єктивно - шляхом інтуїтивного сприйняття відносних частот, неявно відображуючих об'єктивні частоти. Для *вибору оптимальної системи* в умовах визначеності, коли всі слідства визначені за шкалою відносин, можна скористатися апаратом математичного програмування. Справа ускладнюється, коли слідства недостовірні, взаємозалежні і вимагають різних шкал вимірювань.

## 2.3 Схеми вирішення творчих завдань

Існує безліч моделей вирішення творчих завдань, які допомагають вибрати правильні рішення. Процес рішення практично у всіх методах розбивають на етапи. У процесі вирішення питання стоїть не тільки про вибір технічних рішень, параметричної аналізі систем, а й про дисциплінуванні розумових, творчих процесів при вирішенні різних завдань. Аналогічно розглядають і завдання вибору проектних рішень, особливо на перших стадіях, коли постановка завдання ще слабо формалізована.

Розглянемо процес вирішення творчих завдань, який запропонував філософ Джон Дьюї (рис. 18). Він вважав, що людина починає вивчення завдання, коли вступає у взаємодію з навколишнім середовищем, яка є джерелом завдань. Потім він намагається усвідомити для себе ситуацію, виявити її особливості, зробити формулювання, що відображає зміст завдання, а також зробити формулювання цілей рішення. Якщо створюється новий об'єкт, його ділять на складові елементи. Під елементами об'єкта розуміються не деталі, а функціональні елементи. Коли це зроблено, проводять пошук ідеї рішення. Після цього людина оцінює отримане рішення, аналізує його переваги і недоліки. При необхідності зміни знайденого рішення задачі, змінюється і формулювання вихідної задачі. Схема рішення творчих завдань Дьюї передбачає циклічне повторення процедур рішення до отримання задовільного результату.

Основні ланки процедури вирішення (процесу вибору систем) це шість функцій: з'ясування завдання; вибір цілей; синтез систем; аналіз систем; вибір найкращих альтернатив.

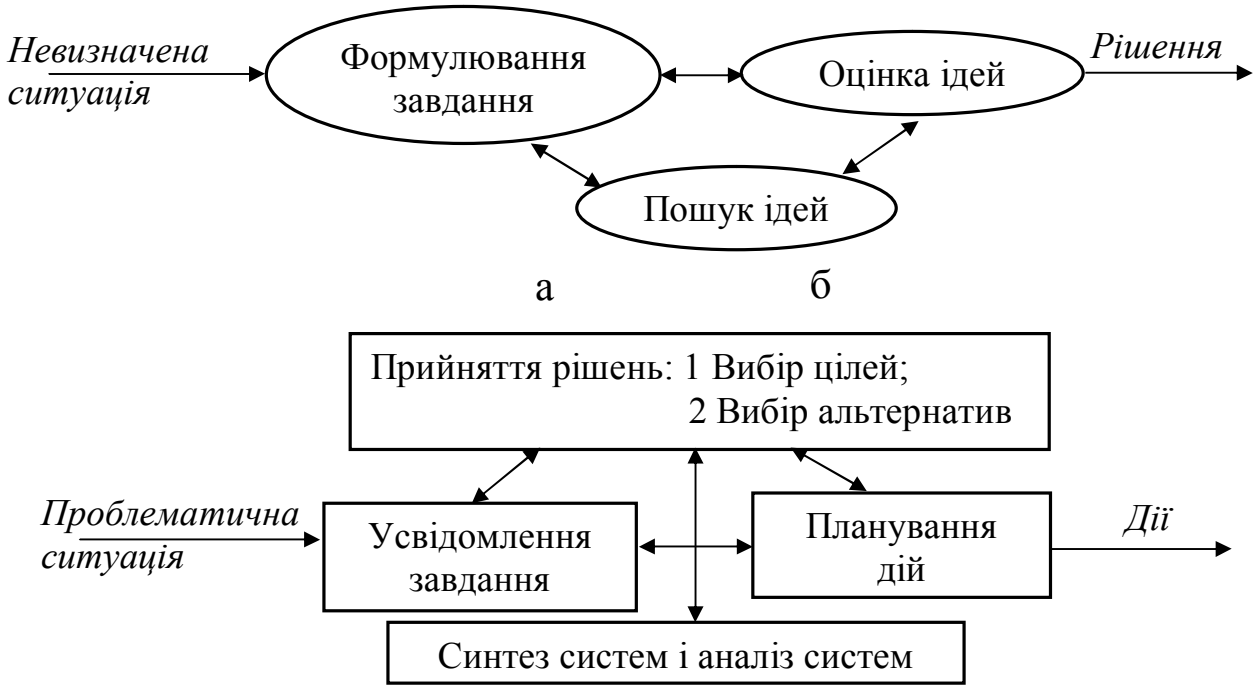



Рисунок 18 - Укрупнена модель процесу, що забезпечує рішенню творчого завдання: а - схема рішення творчих завдань Дьюї; б - укрупнена модель процесу вибору систем

*«Усвідомлення завдання»* – в блоці вирішується вужча завдання, ніж у Дьюї, так як він не містить формулювання цілей. Це опис невизначеній ситуації, в якій визначається потреба вирішення завдання. Блок «Оцінка ідей» в процедурі Дьюї розділений на два етапи за допомогою наступного відмінності: виділення логічних наслідків запропонованих ідей відокремлено від оцінки цінності цих ідей.

Перший етап - аналіз системи (дедукція в математиці). Другий етап - оцінка ідеї - оцінка вибору між альтернативними системами.

«*Планування дій*» – блок показує, що рішення не припиняється з вибором найкращої системи. Необхідно скласти план спостереження за об'єктом (системою). Ця функція має ту ж структуру, що і процес основного рішення задачі, але для іншої задачі.

Виконаємо декомпозицію операцій по визначенню рішення для заданої системи (рис. 19).

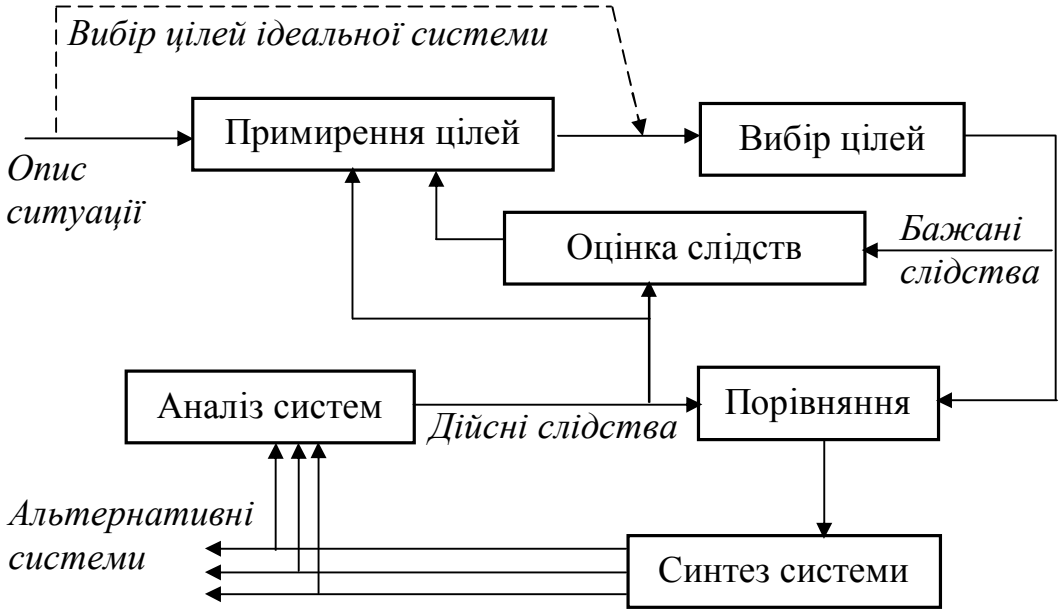



Рисунок 19 – Декомпозиція блоків: «Прийняття рішень», «Аналіз систем» і «Синтез систем»

Операция принятия решений основана на двух понятиях:

*Ідеальна система* – образ потрібної системи, яка виконує необхідні функції. Описується безліччю *бажаних наслідків* (інша назва цілей рішення)

$$\tilde{Y}_q = \frac{Y_U - Y_q}{Y_U},$$

де $\tilde{Y}_q$ - відносне відхилення від ідеального рішення;

$Y_U$ - ідеальне рішення; $Y_q$ - реальне рішення.

*Оптимальна система* – одна з ряду альтернативних систем, яку можна побудувати за даних фізичних, економічних, політичних та ін. обмеженнях. Вона описується безліччю *дійсних наслідків* (досяжних цілей).

Функція *примирення цілей* - полягає в тому, щоб зіставити цілі ідеальної системи і умови задачі. Примирення цілей має три форми:

1) Деякі цілі можуть бути відкинуті (наприклад, порушують закони природи).

2) Аналіз може вказати нові цілі, які потрібно додати до старих.

3) Вирішення конфліктів, які при аналізі несумісні (висока якість і низька вартість). У цьому випадку необхідний компроміс, що забезпечує прийнятний рівень якості системи при виконанні обмежуючих умов.

*Критерій рішення* - показує досягнуто вирішення завдання - це функція, яку ми в підсумку хочемо максі- або мінімізувати. Це правило, яке вказує як комбінувати між собою слідства при виборі оптимальної системи. Критерій рішення може розглядатися як надмета, що враховує всі цінності і невизначеності системи. Критерій рішення дозволяє оцінювати і порівнювати системи з точки зору відповідності ідеальній системі. Можуть бути різні підходи до формування залежностей для об'єднання приватних показників якості в узагальнений, наприклад відповідно до залежності

$$\tilde{Y} = \sum_{i=1}^{n} a_i \frac{Y_{Ui} - Y_{qi}}{Y_{Ui}},$$

де $\tilde{Y}$ - узагальнений критерій рішення; $Y_{Ui}$ , $Y_{qi}$ – приватні показники якості для ідеальної і реальної системи; $a_i$ – ваговий коефіцієнт важливості приватних показників якості, причому

$$\sum_{i=1}^{n} a_i = 1.$$

## 2.4 Поняття і принципи проектування технічних об'єктів

Методологія проектування включає два поняття:

- методологія - вчення про структуру, логічну організацію, методи та засоби діяльності, зокрема про логічної організації проектування як процесу;
- проектування - процес складання опису, необхідного для створення ще неіснуючого об'єкта в заданих умовах.

Розглянемо основні поняття, пов'язані з проектуванням:

1) *Алгоритм проектування* - сукупність приписів, необхідних для виконання проектування;

2) *Мова проектування* - мова, призначена для представлення та перетворення описів при проектуванні (схеми, креслення, діаграми).

3) *Проектна процедура* - сукупність дій, виконання яких завершується проектним рішенням.

4) *Проектне рішення* - проміжне або кінцеве опис об'єкта проектування, необхідне і достатнє для розгляду і переходу до подальшого розгляду або закінчення проектування.

Сучасні методи проектування повинні володіти ергатічностью, тобто розумним поєднанням формалізованих (ПК) і неформалізованих (людина) процедур в процесі проектування.

Між поняттям проектування і конструювання є істотні відмінності. Проектування - термін, визначений стандартом (ГОСТ 22487). Конструювання стандартом не закріплено. Прийнято, що конструювання - складова частина проектування і пов'язана з тими його етапами, де прийняте технічне рішення отримує конкретне конструктивне втілення, наприклад у вигляді об'ємної геометричної моделі, виконаної в CAD-системі.

Сучасні САПР підтримують і дозволяють автоматизувати контроль за інформацією протягом усього життєвого циклу виробу, тобто не тільки в процесі проектування, а й у виробництві, при експлуатації і утилізації.

САПР істотно допомагає при виконанні ряду процедур і операцій, дозволяє автоматизувати наступні функції:

1) Надає засоби об'ємного моделювання
2) Надає засоби кінцево-елементного аналізу.
3) Забезпечує виконання проектувальних та перевірних розрахунків.
4) Надає засоби виконання кінематичного і динамічного аналізу об'єктів, інших інженерних і спеціалізованих розрахунків.
5) Дозволяє оптимізувати параметри при розрахунках.
6) Забезпечує різні види математичного моделювання.
7) Надає засоби інформаційного забезпечення проектування (бази даних і знань), забезпечує інтеграцію з промисловими СУБД і іншими системами рівня управління підприємством (ERP, MES, CRM).
8) Забезпечує управління процесом проектування, обмеження доступу, контроль версій, облік виконаної роботи.
9) Забезпечує управління процесом архівування та зберігання інформації.
10) Забезпечує автоматизацію рутинних процесів:
    - складання специфікацій;
    - генерацію креслень;
    - генерацію документації;
    - копіювання;
    - виведення інформації на різні носії;
    - переклад інформації з паперових носіїв в електронний вигляд.

Автоматизоване проектування проводиться за допомогою систем автоматизованого проектування, які в даний час включають ряд підсистем, що мають різне функціональне призначення (CAD, CAE, CAM, PDM, CRM та інші):

- CAD – системи призначені для створення об'ємних моделей виробів;

– CAE – системи забезпечують виконання інженерних розрахунків (міцність, кінематичні, динамічні, теплові та інші процеси);

– CAM – системи для розробки технологічних процесів обробки деталей на верстатах з ЧПУ, візуалізації процесів обробки;

– спеціалізовані підсистеми для розрахунку процесів лиття, формозміни листових деталей, проектування пружин, зварювання, розводки трубопроводів;

– PDM – системи для вирішення організаційних питань при проектуванні виробів;

– CALS – системи для безперервної інформаційної підтримки всіх стадій ЖЦ продукту.

– CRM – системи для взаємодії зі споживачами продукції.

– MES (Manufacturing Execution Systems) – виробничі виконавчі системи для оперативного планування і управління виробництвом.

– ERP – системи управління ресурсами підприємства.

В даний час у всіх класах перерахованих систем відбувається інтенсивна розробка формалізованої методології їх створення та впровадження у виробництво.

Гірше йде справа зі створенням пошукових САПР, які автоматизують початкові етапи проектування. Пошукові САПР дозволяють:

1) Знаходити аналоги в БД і БЗ, патентної та науково-технічної літератури, фондах технічних рішень, фізичних, хімічних явищ і ефектів (здійснити інформаційну підтримку проектування).
2) Реалізувати алгоритм (послідовність етапів).
3) Здійснити синтез технічних рішень (надати прийоми усунення технічних суперечностей, експертні системи).
4) Здійснити оцінку технічних рішень (надати критерії оцінки, виконати розрахунки).
5) Оптимізувати склад елементів виробу.

В даний час не існує САПР для всіх етапів проектування, бо не зроблена їх формалізація ні в одній області техніки.

В результаті проектування отримують комплект документації, достатній для виготовлення всіх елементів вироби. Згодом форма документації буде змінюватися, проте стадії проектування регламентуються ГОСТом відповідно до ЄСКД. У дужках наведені відповідні (приблизно) стадії розробки програмних продуктів:

1) Технічне завдання (технічне завдання) - забезпечує формування вимог до системи.
2) Технічна пропозиція (концептуальна модель системи).
3) Ескізний проект (логічний проект).
4) Технічний проект (фізичний проект).
5) Робоче проектування (кодування) - забезпечує формування та документування «деталей» проекту).

Дані етапи відповідають процесу розпізнавання об'єкта в середовищі його оточення. Тому перераховані етапи мають місце при проектуванні будь-яких

систем, однак особливості, пов'язані з об'єктом проектування, змінюють зміст етапів і засоби документування проекту.

## 2.5 Процедурна модель проектування

Логічна послідовність діяльності при проектуванні може бути представлена *процедурної моделлю*, яка реалізує системний підхід до цього процесу.

*Процедура* – певна сукупність елементарних операцій по обробці інформації, яка веде до зміни її складу або місця розташування, наприклад, пошуку, розмноження інформації ЄСТПП (ГОСТ 14.104).

Якість проектування залежить від уміння використовувати знання, досягнуті цілим рядом розділів науки, які знайшли відображення в дисциплінах: «Теорія механізмів і машин», «Взаємозамінність, стандартизація та технічні вимірювання», «Надійність машин», «Ергономіка» - теоретичні основи оцінки якості продукції та інші.

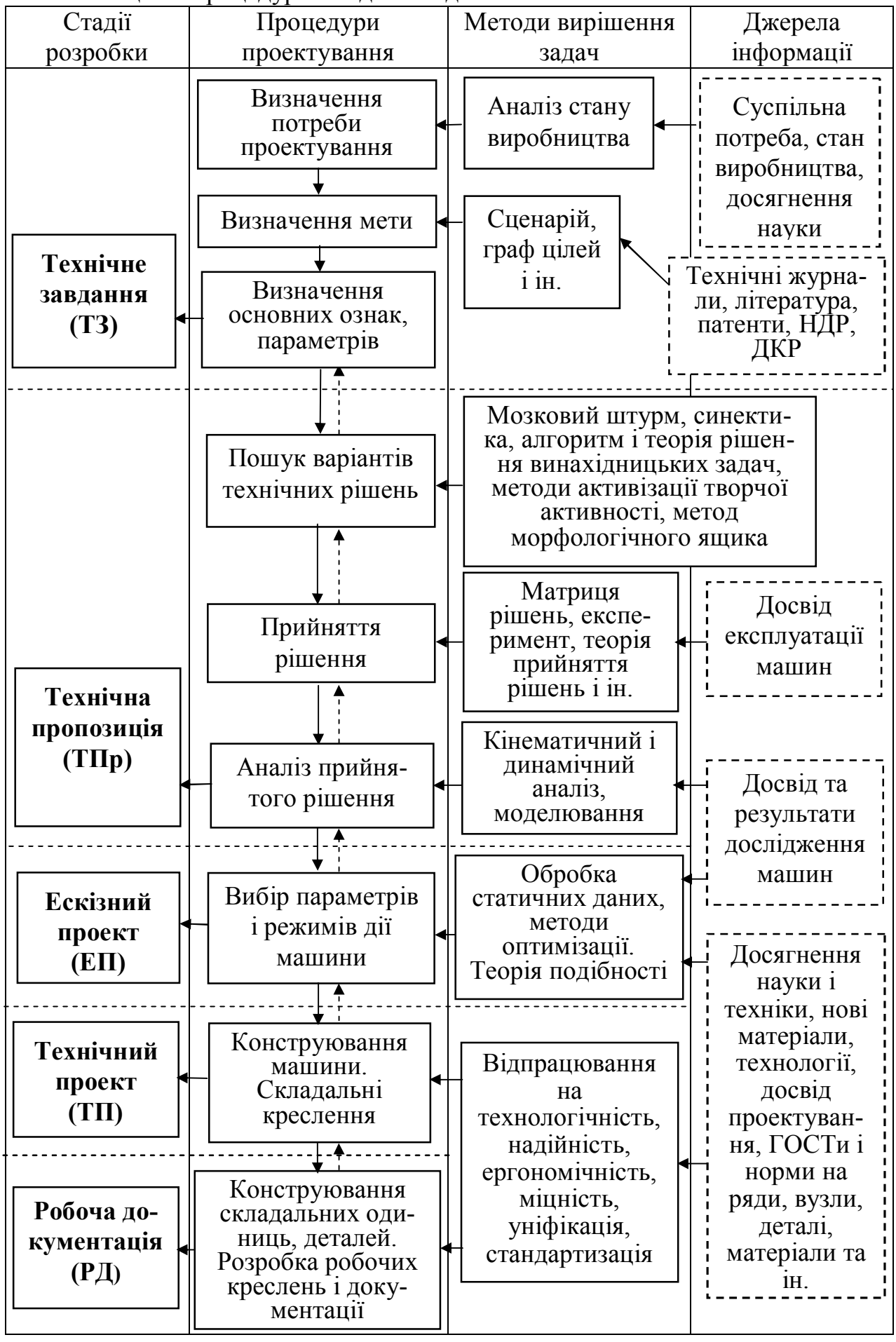
Модель дає уявлення про основні процедурах і операціях проектування, завдання і методи їх вирішення, вказує на джерело інформації. Модель узгоджується зі стадіями ЄСКД.

ЕСКД регламентує стадії розробки конструкторських документів, структурна схема приведена на рис

Для робочої документації настановної серії виробів присвоюється літера «А», в масовому (або серійному) виробництві - літера «Б».


Рисунок 20 - Стадії розробки виробів

Таблиця 1- Процедурна модель моделювання



За результатами проектування складається технічна документація.

Документи поділяються на такі види: *оригінали, подлинники, дублікати, копії*. Види документів при розробці виробів машинобудування та їх зміст наведено у табл. 2.

Таблиця 2 - Номенклатура документів при розробці виробів машинобудування

| Вид документа | Зміст документа |
|---|---|
| 1 | 2 |
| Креслення деталі | Визначає зображення деталі |
| Складальне креслення (СК) | Визначає зображення складальної одиниці |
| Креслення загального вигляду (ВЗ) | Визначає конструкцію виробу, принцип його роботи |
| Теоретичне креслення (ТК) | Визначає геометричну форму, обводи і координати складових частин |
| Габаритне креслення (ГК) | Контурне, спрощене зображення з габаритними і приєднувальними розмірами |
| Монтажне креслення (МК) | Контурне, спрощене зображення з розмірами для установки, може бути фундамент |
| Схема | Показує складові частини виробу і зв'язки між ними |
| Специфікація | Визначає склад складальної одиниці, комплексу або комплекту |
| Відомість специфікацій (ВС) | Перелік специфікацій складових частин |
| Відомість посилальних документів (ВД) | Документи, на які робляться посилання |
| Відомість покупних виробів (ВП) | Перелік покупних виробів |
| Відомість узгодження покупних виробів (ВВ) | Перелік покупних виробів, узгоджених з постачальниками |
| Відомість власників подлинників (ВП) | Перелік підприємств, організацій-розробників, у яких зберігаються документи |
| Відомість технічної пропозиції (ПТ) | Перелік документів, які увійшли в технічну пропозицію |
| Відомість ескізного проекту (ЕП) | Перечень документов, вошедших в эскизный проект |
| Відомість технічного проекту (ТП) | Перелік документів, які увійшли в технічний проект |
| Пояснювальна записка (ПЗ) | Опис, принципи дії, обґрунтування технічних рішень |
| Технічні умови (ТУ) | Вимоги до виробу і його виготовлення, контролю, постачання і т. д. |

Продовження таблиці 2

| 1 | 2 |
|---|---|
| Програма і методика випробувань (ПМ) | Технічні дані, що підлягають перевірці, порядок і методи випробувань |
| Таблиця (ТБ) | Дані, зведені в таблицю |
| Розрахунок (РР) | Розрахунки параметрів, величин на міцність, розмірні ланцюги та ін. |
| Експлуатаційні документи | Наприклад, інструкція з експлуатації та ремонту |
| Ремонтні документи | Дані по ремонту на спеціалізованих підприємствах |
| Патентний формуляр (ПФ) | Дані про патентну чистоту виробу (продукції) |
| Карта технічного рівня і якості продукції (КР) | Технічні та економічні показники якості продукції |

Структура та види виробів

*Виріб* – предмет або сукупність предметів виробництва (рис.21), які потребують подальшого виготовлення на підприємстві (гайка, вал, підшипник, літак).

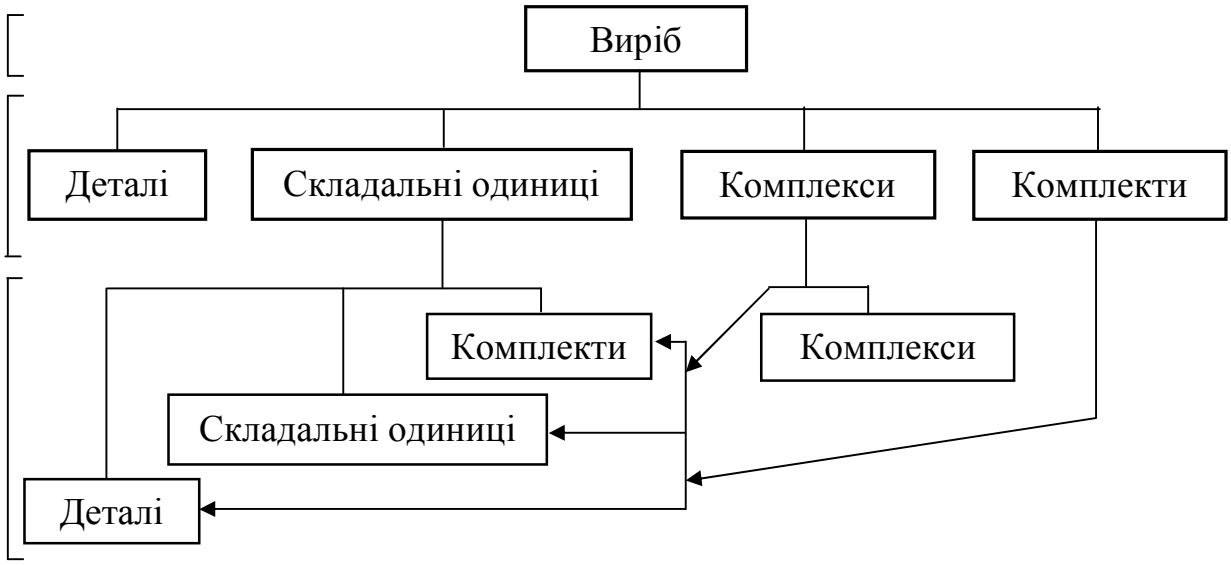



Рисунок 21 - Декомпозиція та види виробів

*Деталь* – це виріб, виготовлений з однорідного матеріалу без застосування складальних операцій (покриття, зварювання, пайка можуть бути використані, але деталь виготовляється з одного шматка матеріалу - трубка, коробка).

*Складальна одиниця* – складові частини підлягають з'єднанню між собою (згвинчуванням, клепкою, зварюванням, обпресуванням, склеюванням, зшиванням і ін.). Можуть бути розібрані на складові частини для транспортування і т.д. Може бути *комплект*, який має загальне функціональне призначення: замок з планкою, готовальня, що містить кілька інструментів в загальному футлярі.

*Комплекс* – два и более изделий, которые функционально связаны, но не соединёны сборочными операциями на заводе - изготовителе, предназначенные для выполнения взаимосвязанных функций (посадочный комплекс для самолётов, ЭВМ как совокупность блоков, цех-автомат, ракета и пусковая установка со средствами управления).

*Комплект* – комплекс експлуатаційного призначення (комплект запчастин, інструментів).

## 2.6 Цільове проектування. Компоненти проектування

*Проектування технічних об'єктів* (ТО) можна розглядати як процес, що полягає в перетворенні вихідного опису ТО в остаточне опис на основі досліджень, виконання розрахунків, конструювання. Проектування в значній мірі пов'язано з вибором наявних (у ряді випадків альтернативних) технічних рішень. При виборі варіанту необхідно оцінити його переваги, відповідають призначенню виробу. Функції системи нерозривно пов'язані з цілями проектування, які також можуть бути представлені у вигляді ієрархічної структури, легко представимо у вигляді графа [6]. Використовуючи граф цілей можна на основі експертних оцінок виділяти пріоритетні цілі, а, значить, і більш важливі функції. При такому підході цілі розглядаються в основному як технічні функції. Розглянемо основні компоненти проектування. Воно представляється як послідовний процес переходу від функцій (цілей) до конструктивних рішень і їх оцінками.

Позначимо компоненти проектування: $A=\{a_1,....,a_m\}$ - безліч цілей проектування; $P=\{p_1,...,p_n\}$ - безліч ознак, що характеризують цілі (чисельні показники досягнення цілей); $X=\{x_1,...,x_k\}$ - безліч технічних рішень, які виконують необхідні функції і характеризуються значеннями ознак; $V=\{\nu_1,....,\nu_l\}$ - безліч оцінок; *m, n, k, l* - потужності множин.

Чотири компонента проектування утворюють граф: вершини - це елементи цих множин, розташовані на чотирьох рівнях; ребра - відносини, зв'язки між ними.

Проектування технічної системи розглядається як відображення на безліч оцінок зрізу твори бінарних відносин двох видів:

- безліч цілей і безліч ознак;
- безліч ознак і безліч конструктивних рішень.

$$F: (\psi * \varphi (A_0)) \rightarrow V,$$

де $\varphi$ - бінарне відношення між множинами А та Р (цілями і ознаками):

$$\varphi \subset (A * P) \text{ при } A_0 \subseteq A ;$$

$\psi$ - бінарне відношення між елементами множин Р та Х (ознаками і рішеннями):

$$\psi \subset (P * X) \text{ при } A_0 \subseteq A.$$

Множини зазвичай ранжирують. Це означає, що функція F встановлює відповідність між дужкою і ставленням.

$$\varphi(A_0) = ((p)(\forall a))\,[a \in A_0 \text{ и } (a, p) \in \varphi\,] .$$
$$\psi(A_0) = ((x)(\forall p))\,[p \in P_0 \text{ и } (p, x) \in \psi] .$$

де $P_0$ - це зріз множини P по підмножині $A_0$.

Добуток бінарних відносин $\varphi$ x $\psi$ дорівнює:

$$\psi \text{ x } \varphi = ((a,x)(\forall p))\,[(a,p) \in \varphi \text{ и } (p,x) \in \psi]$$

є безліч впорядкованих пар (a, x) таких, що для них існує елемент p множини P з яким він вступає в відношення $\psi$ з елементом x. В цьому випадку зріз по підмножині $A_0$ буде:

$$\psi \text{ x } \varphi(A_0) = ((a,x)(\forall p))\,[(a,p) \in \varphi \text{ и } (p,x) \in \psi \text{ и } a \in A_0].$$

Відображення даного зрізу на підмножину оцінок V означає функцію, певну на добутку $\psi$ x $\varphi$ ($A_0$) (тобто область визначення) і приймає значення на множині V. Кожен елемент безлічі V в загальному випадку - багатовимірний вектор, компонентами якого є економічні показники, вартісні характеристики, оцінки корисності та ін. Оскільки параметрів багато, функцію можна оптимізувати в залежності від важливості цих факторів.

$$(F: (\psi \text{ x } \varphi(A_0)) \rightarrow V) \rightarrow \text{opt}.$$

### 2.6.1 Поняття мети проектування. Ієрархія цілей

Цілі як будь-які об'єкти можуть бути розділені за важливістю або за іншими ознаками, тобто мети можна класифікувати. Цілі можуть суперечити одна одній, наприклад: продуктивність і універсальність, якість і вартість. Спроби досягти одних цілей в ряді випадків призводять до погіршення інших показників. Мета може бути представлена у вигляді ієрархії цілей (графів), де цілі і підцілі пов'язані між собою. На 0 рівні абстракції знаходяться інтереси всього людства; на 1-му рівні-державні інтереси; на 2-му рівні - інтереси галузі; на 3-му рівні - інтереси підприємства; на 4-му рівні - інтереси проектувальника; на 5-му рівні - інтереси підрозділу, організації; на 6-му рівні - особисті цілі.

### 2.6.2 Оцінка цілей проектування. Матриця суміжності для орграфа цілей

Для виділення важливіших цілей визначають вагу мети як правило в частках одиниці і на підставі експертних оцінок:

а) ранжируют мети на кожному підрівні;

б) можна виділити частину графа, яка містить найбільш важливі, з точки зору конкретних завдань проектування мети.

i - номер рівня; j - номер цілі на і-му рівні (номер після ранжирування) $i \in \{0,I,II,\ldots.\}$; $j \in \{0,1,2,\ldots.\}$;

$r_{i-j}$ – оцінка ваги мети на і-му рівні без урахування зв'язків (відносна вага), сума ваги на рівні дорівнює 1;

$$\sum_{j=1}^{m} r_{i-j} = 1,\ldots i = const$$

$R_{i-j}$ – абсолютна оцінка ваги мети з урахуванням зв'язків;

υ - місце мети на даному рівні (місця можуть бути однаковими).

Структуру зв'язків представляють у вигляді матриці суміжності або інцидентності. У матриці суміжності рядки - вузли попередники; стовпці - вузли послідовники. Якщо одна вершина пов'язана з іншою, то на перетині відповідного рядка і стовпця ставиться 1, якщо немає - 0.

Таблиця 3 - Матриця інцидентності

| i-j \ k-m | 1-1 | 2-1 | … |
|---|---|---|---|
| 1-1 | 0 | 1 | |
| 2-1 | | 0 | … |
| … | | … | … |

При побудові графа цілей використовують поняття відносного ваги вершини і коефіцієнта зв'язку. Коефіцієнт зв'язку позначається $C_{ij,\,km}$ , і при побудові графа ставиться над стрілкою, що позначає зв'язок (рис. ).
Коефіцієнт зв'язку визначається за формулою:

$$C_{ij,\,km} = r_{ij} * r_{km},$$

де $C_{ij,mk}$ – коефіцієнт зв'язку по призахідним зв'язкам (i, m – номера пов'язаних рівнів вершин; j, k – номера пов'язаних вершин відповідно на рівнях і та m).

## Визначення абсолютної ваги вершини при оцінці цілей проектування

Абсолютна вага дорівнює відносному плюс сума коефіцієнтів зв'язку по призахідним дугам:

$$R_{ij} = r_{ij} + \sum_{1}^{n} C_{ij,km}, \qquad (1)$$

де $R_{ij}$ – абсолютна оцінка ваги мети з урахуванням зв'язків; і – номер рівня абстракції; j – номер цілі на і-му рівні; $r_{ij}$- експертна оцінка ваги j-й цілі на і-му рівні без урахування зв'язків, виражена в частках одиниці (відносна вага);

### 2.6.3 Модель технічної оцінки варіантів рішенньь

Експертні оцінки можливих варіантів технічних рішень застосовують, наприклад, у вигляді матриць, в яких вказуються переваги і недоліки виділених технічних рішень [6]. В якості критеріїв можна використовувати також показники якості, рекомендовані ГОСТ 14.202. Модель технічної оцінки варіанта рішення розглядається у вигляді суми оцінок по кожній цілі для кожного елемента системи [2]:

$$Q = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_j k_{ij} \rightarrow \max , \qquad (2)$$

де Q – оцінка реалізації цілей конкретним варіантом виконання системи; $\alpha_j$ - коефіцієнт значущості j-й цілі, в якості якого можуть бути використані абсолютні ваги цілей; $k_{ij}$ – експертна або інша (економічна) оцінка і-го варіанта елемента системи з точки зору задоволення j-й цілі.

Суттєвим недоліком цільової оцінки є те, що окремо взятий конструктивний елемент виконує не тільки свою основну функцію, але і ряд допоміжних. Так, наприклад, вузол кріплення інструменту виконує не тільки функцію утримання різця в певному просторовому положенні, але також частково реалізує функцію його напрямки, сприйняття навантаження і т.д. (див. рис.  ). В цьому випадку геометричним представленням елемента на дереві буде не вузол дерева, а підграф. Крім того, при об'єднанні різних елементів в системі виникає надсуммарний ефект, виконання деяких функцій дублюється, що призводить до економічно невиправданих витрат при виготовленні.

Інший підхід для вибору технічного рішення заснований на виділенні функцій і використанні вартісних показників для їх реалізації - функціонально-вартісний аналіз і його різновиди [6]. Остаточна оцінка варіанту технічного рішення системи при такому підході може бути дана на основі зіставлення його якісних показників з витратами на виготовлення:

$$k_{еф.} = \sum_{i=1}^{n} \sum_{j=1}^{m} S_{ij} \Big/ \sum_{i=1}^{n} \sum_{j=1}^{m} \beta_j k_{ij} \rightarrow \min , \qquad (3)$$

де $k_{еф.}$ – коефіцієнт ефективності варіанта; n – число основних функцій системи; m – кількість допоміжних функцій, що забезпечують і-ю основну функцію; $S_{i,j}$ – витрати на реалізацію j-й допоміжної функції, що забезпечує і-ю основну; $\beta_j$ – коефіцієнт значущості j-й функції; $k_{ij}$ – оцінка варіанту j-й допоміжної функції, що забезпечує і-ю основну.

Даний підхід спрямований на дослідження найбільш економічних варіантів конструкції, які повністю забезпечують виконання функцій, заданих для певної системи. Але при цьому сам вибір конструктивних варіантів елементів си-

стеми недостатньо чітко формалізований, не розглядається також пріоритетність самих функцій.

Тому можна поєднати можливості цільового вибору рішень і їх економічної оцінки. Контроль повноти системи здійснюється на основі аналізу виконання конструктивними елементами функцій, що забезпечують роботу системи в цілому з мінімізацією дублювання функцій елементами конструкції. Виходячи з вищесказаного, пропонується наступний алгоритм вибору і оцінки конструктивних рішень, який передбачає для підтримки проектного рішення комбіноване використання функціональних, конструкторських, технологічних і економічних даних, який передбачає комбіноване використання функціональних, конструкторських, технологічних і економічних даних для підтримки проектних рішень:

1) Декомпозиція системи, виділення функцій, складання І-АБО-дерева.
2) На основі цільової оцінки вибір на І-АБО-дереві варіантів рішень (вузлів) для всіх функцій (побудова І-дерева).
3) Вибір конструктивних рішень-аналогів для елементів і системи в цілому.
4) Оцінка виконання функцій для наявних аналогів з визначенням ступеня їх дублювання.
5) Вибір оптимального рішення з усуненням дублювання і введенням при необхідності нових конструктивних рішень (доповненням дерева).

Четвертий і п'ятий етапи ітераційно повторюються до тих пір, поки не виходить оптимальна за вартістю конструкція системи, що виконує всі задані цілі і функції. Для реалізації запропонованого підходу необхідна розробка програмного засобу інформаційної та обчислювальної підтримки проектних рішень, яке має забезпечувати знаходження аналогів для системи і її частин при неповних технічних даних, візуалізацію фрагментів рішень, їх оцінку.

Даний алгоритм побудови та оцінки конструктивних рішень може бути використаний для підтримки вибору проектних рішень.

Для автоматизації розрахунків матрицю суміжності перетворять в матрицю вагів, що містить коефіцієнти зв'язку. Алгоритм для розрахунку абсолютних вагів: для кожної вершини підсумовуються коефіцієнти по стовпцях і складаються з відповідними відносними вагами. Таким чином, процедура вибору мети і їх оцінки містить наступні операції:

1) Виділення рівнів абстракції інших навколишніх систем по відношенню до досліджуваної.
2) Складання сценарію розвитку сфер оточення на кожному рівні абстракції: історія розвитку; напрямок розвитку; прогноз; створення прототипу.
3) Виділення цілей і побудова графа цілей.
4) Експертна оцінка відносних ваг.
5) Розрахунок абсолютних вагів.

Граф цілей використовується для вирішення різних завдань, він може бути розширений в зв'язку з удосконаленням розглянутих об'єктів. Граф цілей має власну цінність і є основою для прийняття рішень по конструктивному виконанню проектованої системи.

## 2.7 Технологічний процес розробки автоматизованих систем

Розглянемо технологічний процес структурного аналізу і проектування автоматизованих систем (АС).

*Автоматизовані системи* – це складні вироби, що включають різнорідні за своєю специфікою компоненти: програми, БД, технічні засоби, які в свою чергу також є складними системами. Створення автоматизованої системи (як і інших систем) базується на концепції життєвого циклу (рис. 22).

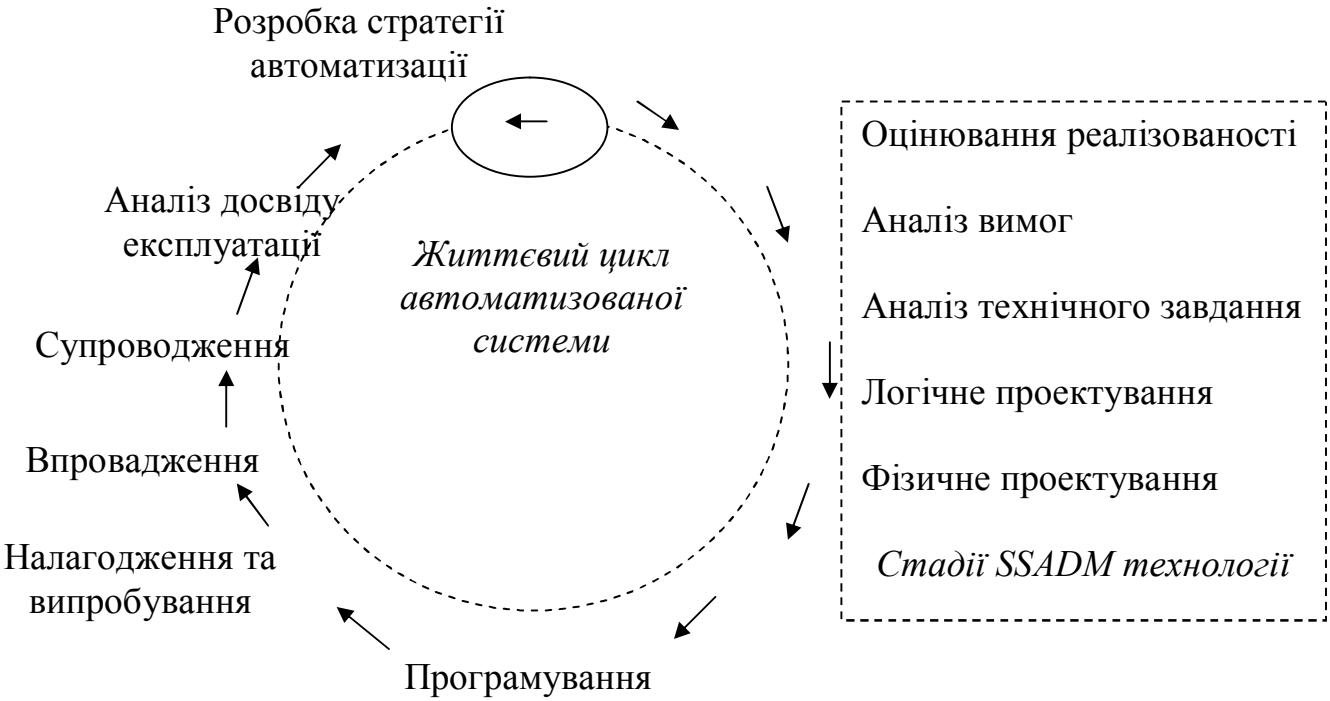



Рисунок 22 - Життєвий цикл автоматизованої системи

Проблеми, що виникають при розробці АС:

1) Забезпечення функціонування в режимі реального часу, тобто необхідним є дотримання вимог щодо швидкодії всієї системи.
2) Обмеженість обчислювальних ресурсів.
3) Складність технології випробування систем.

Існують різні технології розробки АС, які регламентуються ГОСТ 34.601 або іншими стандартами, зокрема технологією SSADM (Structured Systems Analysis and Design Method), яка є державним стандартом Великобританії і отримала широке розповсюдження в Європі [15].

### Принципи технології SSADM і її місце в життєвому циклі АС

При розробці програмного забезпечення застосовують ряд моделей ЖЦ на основі процедурного або ОО підходу до проектування. Розрізняють також два види реалізації технологічного процесу: каскадний і ітераційний. Каскадна модель регламентує сувору послідовність етапів. Спіральна модель передбачає кілька ітерацій до стадії впровадження. SSADM є компромісом між цими класич-

ними підходами. АС розробляється в цілому по каскадної моделі, але з проведенням досліджень, циклами на початкових етапах проектування. Технологія SSADM здійснюється відповідно до таких принципів:

1) Постійне залучення користувачів в процес вироблення, прийняття рішень протягом усього циклу.

2) Чітка структуризація технологічного процесу, взаємна ув'язка всіх стадій, етапів проектних процедур (наприклад, на основі календарного планування).

3) Контроль кожного етапу технологічного процесу з боку керівництва:
   – вбудований контроль якості (тести);
   – виділення формалізованих критеріїв якості.

4) Використання по можливості CASE - засобів.

5) Стикування з існуючими для прийнятих систем проектування, технологіями та управлінням базами даних.

6) Формалізація кошти розробки у вигляді різних методологій, алгоритмів, таблиць, форм.

## Типовий технологічний процес створення АС

Типовий ТП включає сім стадій, кожна стадія складається з етапів, етапи діляться на операції.

Для підтримки проектування технологія SSADM містить близько 50 документів, що включають методики реалізації стадій створення АС.

## Характеристика методики визначення вимог до АС

Методика визначення вимог до АС призначена для виявлення вимог і функцій з боку замовника АС. На даному етапі розробляють каталог вимог, який містить повний опис функцій, термінів даної предметної області, який дозволяє однозначно трактувати базові поняття.

При виконанні стадій 0 - 3 вимоги ітераційно розглядають і уточнюють, після чого зміна вимог заборонена.

На СТ.4 незначно кількість вимог уточнюються, в тому числі вимоги до діалогового взаємодії.

Рішення поставлених задач вирішується на етапі 5

На етапі 6 фіз. Проектування контролюють ступінь виконання зазначених вимог.

У методиці виділяють наступні положення:

1) Функції розробників при формуванні вимог (ролі).
2) Здійснюють збір первинної інформації та вимог до обробки.
3) Постійне ведення каталогу вимог.
4) Выделяются виды требований (группы).
5) Формування характеристик вимог (формування).

Таблиця 4 - Стадії і етапи технологічного процесу створення АС по технології SSADM і відповідно до ГОСТ 34.601

| Стадії і етапи SSADM | ГОСТ 34.601 на створення АС |
|---|---|
| **Стадія 0** Оцінювання реалізованості (необов'язкова).<br>01 Визначити рамки, скласти план розробки.<br>02 Визначити початковий варіант вимог до АС.<br>03 Вибрати варіант оцінювання реалізованості (техніко-економічне обґрунтування).<br>04 Оформити звіт про можливість створення АС.<br>**Стадія 1** Передпроектна обстеження.<br>1.1 Визначити рамки передпроектного обстежування.<br>1.2 Визначити основні вимоги до АС.<br>1.3 Вивчити процеси обробки інформації в існуючій системі.<br>1.4 Вивчити дані, які обробляються.<br>1.5 Розробити логічне опис існуючої системи.<br>1.6 Узагальнити результати обстеження.<br>**Стадія 2** Вибір варіанту автоматизації.<br>**Стадія 3** Розробка ТЗ.<br>3.1 Розробити загальні вимоги до АС.<br>3.2 Розробити необхідну логічну модель (ЛМ) даних.<br>3.3 Уточнити вимоги до функцій і завдань.<br>3.4 Уточнити ЛМ даних.<br>3.5 Розробити демонстраційний прототип.<br>3.6 Розробити вимоги до обробки даних.<br>3.7 Уточнити цілі розробки АС.<br>3.8 Оформити ТЗ на створення АС.<br>**Стадія 4** Вибір варіанту технічної реалізації.<br>4.1 Розробити варіанти технічної реалізації.<br>4.2 Вибрати варіант технічної реалізації.<br>**Стадія 5** Розробка логічного проекту.<br>5.1 Визначити порядок діалогового взаємодій-наслідком.<br>5.2 Розробити постановки задач модифікації БД.<br>5.3 Розробити постановки інформаційних завдань.<br>5.4 Завершити розробку лог-проекту.<br>**Стадія 6** Фізичне проектування.<br>6.1 Розробити план фізичного проектування. Підготувати фізичну організацію БД.<br>6.2 Розробити специфікації вимог до програмних компонентів.<br>6.3 Оптимізувати фізичну структуру БД.<br>6.4 Уточнити специфікації вимог до програмних компонентів.<br>6.5 Узгодити інтерфейс між завданнями і БД.<br>6.6 Оформити фізичний проект. | **Стадія 1** Формування вимог до АС.<br>1.1 Обстеження об'єкта та обгрунтування необхідності створення АС: вивчення об'єкта, видів його діяльності, режимів, оцінка якості функціонування, виявлення проблем для вирішення за допомогою АС.<br>1.2 Формування вимог користувача до АС.<br>1.3 Оформлення звіту і заявки на розробку АС.<br>**Стадія 2** Вироблення концепції АС.<br>2.1 Вивчення об'єкта.<br>2.2 Проведення необхідних НДР.<br>2.3 Розробка варіантів концепції і вибір варіанту, відповідного вимогам користувача (розробка альтернативних варіантів, оцінка їх переваг та недоліків, необхідних ресурсів на реалізацію, вибір оптимального варіанту на підставі зіставлення вимог користувача і можливостей АС).<br>**Стадія 3** Технічне завдання (ТЗ).<br>3.1 Розробка та затвердження ТЗ.<br>**Стадія 4** Ескізний проект (необов'язкова).<br>4.1 Розробка попередніх проектних рішень по АС і підсистем: склад завдань, концепція і структура інформаційної бази, функції СУБД, склад обчислювальної системи, функції та параметри основних програмних засобів.<br>4.2 Розробка документації на АС і її частини.<br>**Стадія 5** Технічний проект.<br>5.1 Розробка проектних рішень по системі і її частинам: по функціонально-алгоритмічної, і оргструктуре АС, технічних засобів, з ведення БД, класифікації та кодування інформації, алгоритмам розв'язання задач, по застосовуваних мов і ПО.<br>5.2 Розробка документації на АС.<br>5.3 Розробка і оформлення документів на поставку або ТЗ на розробку комплектуючих.<br>5.4 Розробка завдань на проектування в суміжних частинах проекту об'єкта автоматизації.<br>**Стадія 6** Робоча документація.<br>6.1 Розробка робочої документації на АС і її частини.<br>6.2 Розробка або адаптація програм.<br>**Стадія 7** Введення в дію.<br>**Стадія 8** Супровід АС. |

# Збір первинних вимог

Методи збору:

- інтерв'ю;
- вивчення документації;
- анкетування користувачів і протоколювання їх діяльності;
- стажування на робочому місці;
- спостереження;
- мозковий штурм;
- прототипування;
- використання власних знань і досвіду.

Незалежно від методу корисні відповіді на наступні питання:

1) Що потрібно від системи?
2) Навіщо вам це потрібно?
3) Кому це потрібно?
4) Наскільки це важливо?
5) Чим можна виміряти ступінь дотримання вимог?

Види вимог:

1) функціональні;
2) нефункціональні.

Функціональні вимоги відповідають на питання «Що повинна робити система?», а нефункціональні «З яким рівнем якості повинна робити система?».

Бажано, щоб вимоги і результати були надані в вимірної формі. Для цього знаходять показники якості, показують їх бажане значення і діапазон допустимих відхилень.

Визначення нефункціональних вимог:

1) Вимоги до якості виконання функцій.
2) Обмеження доступу.
3) Забезпечення вимог до безпеки.
4) Забезпечення моніторингу і контрольованості та ін.

Вимоги до якості:

- години роботи (час доби, дні тижня, особливості для вихідних і святкових днів);
- доступність - частка часу в % виконання функцій;
- оперативність - час функції для систем реального часу або тривалість виконання завдання;
- частота заявок - кількість заявок на вирішення завдань в одиницю часу;
- продуктивність - сумарний обсяг роботи в одиницю часу (кількість зчитувальних даних Мбіт/с);
- тривалість або ранній час початку і пізній час закінчення роботи;
- надійність - середній час напрацювання на відмову;
- середнє і максимальне час на виконання;

Обмеження доступу:

1) Яким даними потрібен захист?

2) Чи слід обмежити доступ для конкретного користувача?

3) Які заходи необхідні для обмеження доступу (пароль на фізичному рівні, організаційні заходи)?

Вимоги до безпеки:

- виготовлення страхувальних копій (архівування) резервування;
- відновлення працездатності при відмовах.

Забезпечення моніторингу та контрольованості: особливе значення має для фінансових систем (ревізія діяльності).

Завдання вирішується шляхом накопичення статистичної інформації для аналізу і оцінки дій. Додаткове питання - частота контролю.

Інші обмеження:

Вимоги до роботи при переході на нову систему.

Вимоги до роботи по сполученню з іншими системами.

Побудова людино-машинного інтерфейсу.

Архівування та знищення даних.

# 3 МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ І ПРОЕКТУВАННЯ ПРОГРАМНИХ СИСТЕМ

## 3.1 Принципи інженерії програмного забезпечення

Підвищення рівня складності завдань, що вирішуються за допомогою ЕОМ, постійно викликає вдосконалення технологій програмування, зокрема зміни організації даних, які обробляє програма. Зокрема, розвиток ЕОМ, мови високого рівня і технологій програмування призвело до модульної організації програми, появи механізму роздільної компіляції модулів, що забезпечують доступ тільки до необхідних даних і процедур. Поступово розвивалися правила побудови межмодульного інтерфейсу. Модулі стали важливим механізмом абстракції, містили ряд даних і логічно пов'язаних процедур, що забезпечують структурування модуля. Процедури в свою чергу забезпечували абстрагування, достатню для опису дій [1].

Наступний етап розвитку технологій розробки програмних продуктів пов'язаний з появою об'єктно-орієнтованого (ООП) підходу до проектування і програмування. Архітектура мов об'єктного і об'єктно-орієнтованого програмування дозволяє описувати абстрактні об'єкти. Теорія типування, втілена в мовах типу Pascal, дає можливість здійснити контроль правильності використання абстрактних типів при програмуванні. У мовах, що підтримують ООП, основним елементом є модуль, що складається з логічно пов'язаних класів і об'єктів, а не процедур [1]. Інкапсуляція даних, розміщених в класах і об'єктах, істотно підвищилася, а доступ до них чітко обмежується інтерфейсом класу. Область глобальних даних зведена до мінімуму або зникла взагалі, що спрощує контроль за роботою програмного продукту, істотно полегшує його модифікацію.

При структурному програмуванні модульна структура ПП має вигляд дерева. Структура програмного продукту при ООП має вигляд графа, елементи якого для складних систем також є графами. ООП представляє більш «природний» підхід до організації інформації про об'єкти, як елементах оточуючого нас світу.

### Структура ЖЦ і її відображення в документації

Розробка ПО здійснюється в рамках галузі знань, яка називається «Програмна інженерія» - це інженерна дисципліна, яка є ПО, а предметом - створення, різні аспекти створення ПО і експлуатація на всіх етапах ЖЦ.

Програмна інженерія поліпшила наступні характеристики програм:

1) читаність і зрозумілість (за рахунок структурування та документування).
2) забезпечила контроль складності при збільшенні розміру за рахунок різних методів декомпозиції і за рахунок виділення точок зору на систему.
3) контроль зв'язку стратегічної структури часу управління.

Принципи використання декомпозиції - структурний або об'єктивно-орієнтований, ієрархічний підхід до подання інформації.

Крім двох базових принципів:

- «розділяй і володарюй» (декомпозиція);
- ієрархічне впорядкування.

Існує ряд принципів не менш важливих:

1) Принцип абстрагування - виділяються тільки суттєві властивості системи у відповідності з аспектом моделювання.

2) Принцип формалізації - визначається і витримується методичний підхід до вирішення проблеми. Наприклад, виконується конструктивна або функціональна декомпозиція.

3) Принцип концептуальної спільності - приймається єдина філософія (підхід) до всіх етапів життєвого циклу виробу. Наприклад, якщо проектування структурне, то і тестування, оновлення і т. д. - теж структурне.

4) Принцип повноти - статичний принцип, що визначає працездатність системи, здійснюється контроль за присутністю зайвих елементів.

5) Принцип несуперечності - обгрунтованість і узгодженість роботи елементів системи.

6) Принцип логічної незалежності - концентрація уваги на логічному проектуванні для забезпечення незалежності від фізичного проектування. Слід розрізняти логічний і фізичний рівні проектування, при цьому логічний рівень повинен бути незалежним від фізичного рівня.

7) Принцип незалежності даних - моделі даних повинні бути проаналізовані і спроектовані незалежно від процесів їх логічної обробки, а також від фізичної структури і розподілу.

8) Принцип структурування даних - полягає в тому, що дані повинні бути структуровані і ієрархічно організовані (дерева, ієрархії класів, бази даних).

9) Принцип доступу кінцевого користувача до бази даних без програмування. Засобом реалізації є графічний інтерфейс, генератори запитів, звітів, візуальні засоби проектування (мови) і т. д.

## 3.2 Життєвий цикл програмного продукту і його етапи

В основі діяльності щодо створення і використання програмного забезпечення (ПЗ) лежить поняття його життєвого циклу (ЖЦ).

ЖЦ є моделлю створення і використання ПЗ, що відображає його різний стан, починаючи з моменту виникнення необхідності в ПЗ і закінчуючи моментом його виходу з ужитку в усіх користувачів автоматизованої системи АС.

ЖЦ утворюється відповідно до принципу *низхідного проектування* і як правило носить *ітераційний характер*. Реалізовані етапи, починаючи з самих ранніх, циклічно повторюються відповідно до уточнення поглядів і вимог замовників, зовнішніх умов, введенням обмежень і т. д.

На кожному етапі ЖЦ створюється певний набір документів і технічних рішень, при цьому для кожного етапу вихідними є документи попереднього етапу. Документи визначають зміст роботи, результати, включають підписи відповідальних за виконання роботи.

Етап завершується *верифікацією* (контролем) розроблених документів і рішень з метою перевірки їх відповідності вихідним вимогам. Всі етапи проходяться ітераційно.

Характеристика етапів

1) *Етап концептуального проектування* - розробляється узагальнена модель системи (організації) для подальшої автоматизації її діяльності. Планується розподіл всіх видів ресурсів.

2) *Етап аналізу* - починається з деталізації концептуальної моделі. Виділяються основні функції системи, процеси і потоки інформації, що протікають між ними (DFD); здійснюється декомпозиція даних; виділяються стани системи і можливості їх зміни (STD) - логічний рівень проектування. Крім побудови діаграм будується їх текстовий опис.

3) *Етап загального проектування* - будують і представляють архітектуру системи у вигляді структурної схеми (ЗС) - це фізичний рівень, який показує взаємодію модему системи. При фізичному проектуванні розробляють більш детальний проект бази даних, інтерфейс користувача й тести.

4) *Детальне проектування* виробляють за методом «білого ящика», розробляють алгоритми на основі функціональних вимог.

5) Проводиться кодування (CASE-засоби).

Резюме.

При реалізації етапів ЖЦ програмна система послідовно представляється на різних рівнях абстракції:

- модель;
- архітектура;
- алгоритми;
- код.

Таблиця 5 - Етапи проектування і моделі програмних систем при структурному підході

| Концептуальне моделювання | Логічне моделювання | Логічне та фізичне моделювання | | Кодування, відлагодження, тестування | Експлуатація та розвиток |
|---|---|---|---|---|---|
| *Стратегічне планування* | *Загальне проектування* | | *Детальне проектування* | *Реалізація* | *Супровід* |
| - концептуальна модель;<br>- план розробки | - модель системи;<br>- словник даних;<br>- розробка специфікацій (міні-специфікації);<br>- послідовна архітектура ПЗ;<br>- план тестування | - остаточна архітектура виробів;<br>- специфікація модулів;<br>- проект архітектурних тестів (тести збірки);<br>- уточнення міні-специфікації | - алгоритми модулів;<br>- проект тестів модулів | - код модулів;<br>- код тестів;<br>- протоколи тестування;<br>- протоколи збірки;<br>- протокол випробувань | - протокол супроводу;<br>- пропозиція щодо розвитку |
| - діаграми, таблиці основних функцій;<br>- діаграми, таблиці інформаційних потреб | - діаграми потоків даних (DFD);<br>- діаграми типу «сутність-зв'язок» (ERD);<br>- діаграми переходів станів (STD) | - структурні схеми;<br>- ієрархічні схеми викликів (SC) | - R – схеми;<br>- діаграми дій;<br>- мови програмування;<br>Flow () | - журнали реєстрації збоїв;<br>- пропозиції клієнтів | |

Існуючі моделі ЖЦ визначає порядок виконання етапів в ході розробки, а також критерії переходу від етапу до етапу. Види моделей ЖЦ відрізняються послідовністю етапів і тривалістю їх виконання. Відповідно до цього набольшее поширення набули 3 моделі ЖЦ.

1) Каскадна модель (70-80 рр.) Передбачає перехід на наступний етап після повного закінчення робіт за попереднім етапу. (Тобто план - закон!). Така модель в значній мірі є ідеалізацією і придатна тоді, коли можна заздалегідь спланувати всі етапи. Наприклад, треба швидко, жорстко що-небудь поставити на ринок. Головним недоліком є те, що замовники недостатньо розуміють що ж вони хочуть від автоматичної системи.

2) Поетапна модель з проміжним контролем (80-85 рр) - итерационная модель розробки ПЗ з циклами зворотного зв'язку між етапами. При необхідності відбувається повернення до попередніх етапів. Перевага в тому, що міжетапні коригування забезпечують меншу трудомісткість в порівнянні з каскадної, однак час життя кожного етапу розтягується на весь період розробки.

3) Спіральна модель (86-90 рр) робить упор на початкові стадії ЖЦ: аналіз вимог, проектування специфікацій, попереднє і детальне проектування. Проводиться послідовне циклічне повторення етапів ЖЦ.

На цих етапах перевіряється і обгрунтовується реалізація технічних рішень шляхом створення прототипів. Кожен виток спіралі відповідає поетапній моделі створення фрагмента або версії ПП, на ньому уточнюються цілі і характеристики проекту, визначається його якість, плануються роботи наступного витка спіралі. Таким чином поглиблюються і послідовно конкретизуються деталі проекту і вибирається обгрунтований варіант для реалізації.

Переваги спіральної моделі:

– накопичення і повторне використання програмних засобів, моделей, прототипів (інтерфейсні модулі, формати файлів...);

– орієнтація на розвиток і модифікацію ПЗ в процесі проектування (версії оновлюються часто 2 рази в рік.- Pro / Enginttr) $;

– аналіз ризику і витрат в процесі проектування;

– завдяки прототіпірованію розробники домагаються уточнення вимог від замовників.

У чистому вигляді спіральна модель придатна в більшій мірі для порівняно невеликих проектів.

Оптимум на думку творців промислової інформаційної технології Великобританії SSADM (Structured Systems Analiysis and Desing Metod) лежать в області каскадної моделі з ітераційним циклами, аналогічними спіральної моделі. (Посередині, як і будь-яка істина).

Застосування певної послідовності етапів дозволяє використовувати такі технології як стандарти (ГОСТ 34.601-90 та ін.). Інформаційна технологія. Комплекс стандартів і керівних документів на автоматизовані системи. (ГОСТ 34.201-89, 34.602-89,РД-50-698-90, ГОСТ 34.003-90, РД-50-34.119-90). –М.: Госстандарт СССР, 1991 – 143с.)

З іншого боку, стандартизація підходів до проектування дозволяє інтенсифікувати створення АС. Виникає база для застосування інструментальних програмних засобів проектування АС. (CASE - Computer-Aided Software Engineering).

В Англії SSADM розвивалася спочатку в державному секторі, потім прийнята як державний галузевий стандарт (1993 г.). Зараз існує Асоціація користувачів (International SSADM Users Group) - більше 3000 індустріальних членів і понад 300 організацій в Західній Європі та Японії.

Паралельно з ПЗ розвиваються і технічні засоби, які сприяють впровадженню нових технологій.

Гордон Мур - один із засновників фірми Intel вивів закон, за яким кількість транзисторів, уміщається на кристалі, подвоюється кожні 18 місяців: кожні 1,5-2 роки з'являється якісно нове покоління ЕОМ. Таким чином час життя ЕОМ становить приблизно 2 роки.

## 3.3 Функції CASE-засобів при автоматизації ЖЦ ПЗ

1) Підтримка всіх етапів ЖЦ.
2) Координація робіт.
3) Підтримка якості ПЗ.
4) Контроль забезпечення функцій до ПЗ.
5) Можливість створення прототипів з подальшим наповненням.
6) Організація роботи великої кількості програмістів.
7) Питання інтеграції, наступності, використання коду.
8) CASE-засоби забезпечують інтеграцію:
   – даних;
   – контролю;
   – уявлення.

Для CASE-засобів відповідно до виконуваних функцій розроблений загальний каркас, що забезпечує наступний набір послуг:

– інтеграція контролю забезпечує комунікацію інструментальних засобів між собою з послугами;

– інтеграція даних забезпечується наявністю депозитарію (службова БД, де зберігаються дані по проекту);

– інтеграція представлення здійснюється стандартним інтерфейсом користувача, загальним для всіх інструментальних засобів і послуг;

– контроль забезпечується наявністю менеджера задач.

Наявність депозитарію дозволяє використовувати попередні напрацювання, навчати персонал.

Стандартизація такого каскаду сприяє переходу на більш сучасні методології проектування.

В цілому використання CASE-засобів дозволяє підвищувати передбачуваність роботи за якістю, терміну, вартості. Наприклад, застосування мов візуального проектування і прототипування дозволяють прискорити процес розробки у 10-20 разів.

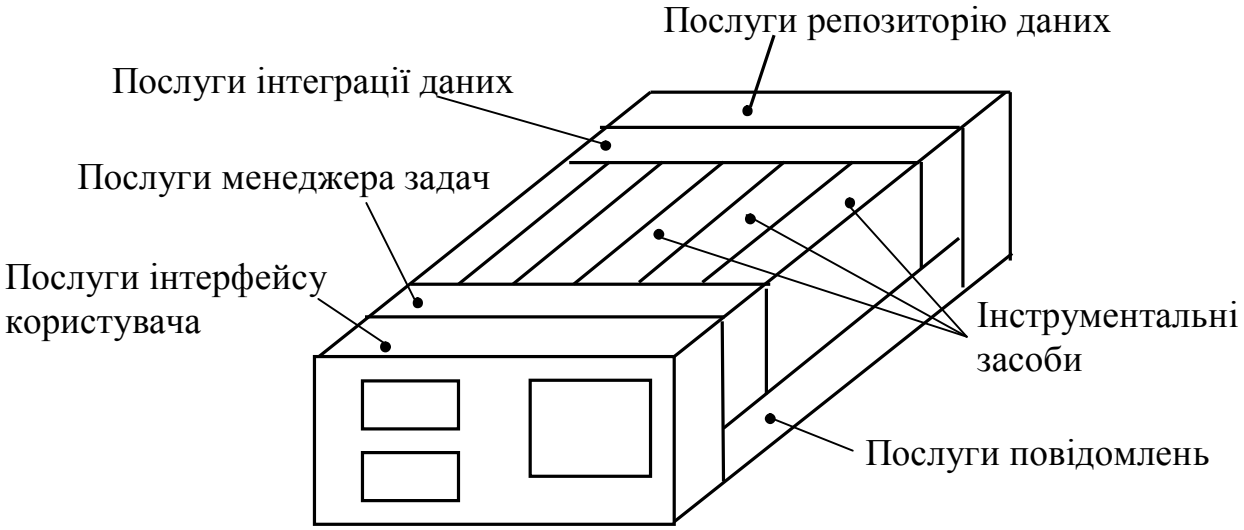



Рисунок 23 – Проблеми впровадження CASE-систем

Складність розробки підвищується при переході від етапів програмування до етапу аналізу і стратегічного планування.

1) Стратегічна інформація слабо структурована, вона нечітка і фрагментарна.

2) CASE-засоби слабо нагадують таку інформацію, потрібна колективна робота експертів.

3) Використання CASE-засобів вимагає знання розробників в області прийнятої методології проектування, тобто потрібна вища культура розробки програмного забезпечення. Для цього потрібно затратити певні зусилля з вивчення CASE-засобу, зусилля по більш точному використанню методології.

4) Простота графічних мов моделювання (діаграми) створюють ілюзію простоти процесу моделювання у відриві мови моделювання.

Порівняння каскадної і итерационной моделей

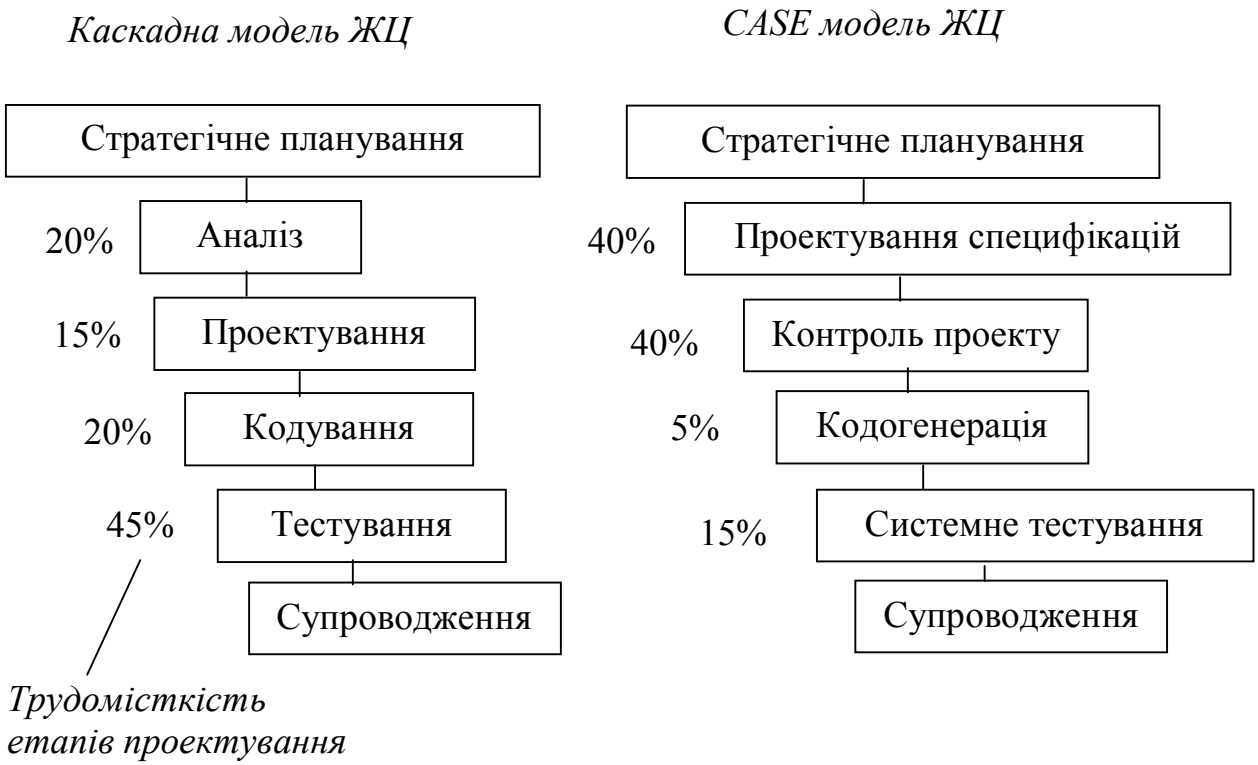



Рисунок 24 - Традиційний лавиноподібний цикл розробки (каскадна модель ЖЦ) і спіральна (CASE) модель створення програмних виробів

Переваги CASE - моделі (спіральна)

1. Підвищення частки інтелектуальної праці при проектуванні.
2. Уточнення вимог за рахунок прототипирования збільшує якість роботи.
3. Спеціалізована БД зберігає накопичений досвід і сприяє навчанню програміста.
4. Спрямованість проекту на розвиток, модифікацію (етап стратегічного планування.).

Таблиця 6 – Порівняння традиційної розробки і CASE-технології

| | Традиційна розробка | CASE - технологія |
|---|---|---|
| 1 | Основні зусилля на кодування і тестування | Основні зусилля на аналіз і проектування |
| 2 | «Паперові» специфікації | Швидке итеративное прототипування |
| 3 | Ручне кодування | Автоматична кодогенерація |
| 4 | Ручне документування | Автоматична генерація документації |
| 5 | Тестування кодів | Автоматичний контроль проектів |
| 6 | Супроводження кодів | Супровід специфікацій проектування |

CASE - модель життєвого циклу ПЗ

CASE засоби і технології пропонують новий, заснований на автоматизації підхід до концепції життєвого циклу ПЗ.

При використанні CASE змінюються всі фази життя ЖЦ, при цьому найбільші зміни стосуються фаз аналізу і проектування.

В CASE - моделі фаза прототипування замінює традиційну фазу системного аналізу.

Найбільш автоматизованими є фази контролю проекту і кодогенерації (хоча всі інші також підтримуються CASE засобами)

## 3.4 Життєвий цикл розробки програмного забезпечення при ООП проектуванні

Розробка ПП досі залишається трудомістким процесом і засноване на ручних методах навіть на заключних етапах розробки.

Тому будь-які теоретичні розробки не виключають практичних питань, прийомів управління процесами розробки ПЗ. При розробці ПЗ незалежно від технології реалізуються такі принципи:

- розподіл ресурсів;
- встановлення проміжних етапів;
- управління конфігурацією;
- перевірка версій.

Всі накопичені прийоми, підходи структурного проектування використовуються і в ООП проектуванні.

Кожна модель (лавиноподібна) традиційна, має такі недоліки:

- непридатність для розробки складних програм систем, що складаються з великої кількості автономних модулів, а також для організації процесу внесення в систему наступних змін;
- обов'язкове послідовне виконання всіх етапів розробки;
- несовмемтімость з еволюційним підходом, який впроваджується завдяки прототіпіруванію;
- несумісність з можливостями автоматичного програмування, трансформації програм, CASE технологій, допоміжних засобів, заснованих на базі знань;
- ООП являє собою послідовний ітеративний процес, тобто підхід еволюційний, природний. Для ООП більш підходить спіральна модель створення ПП.

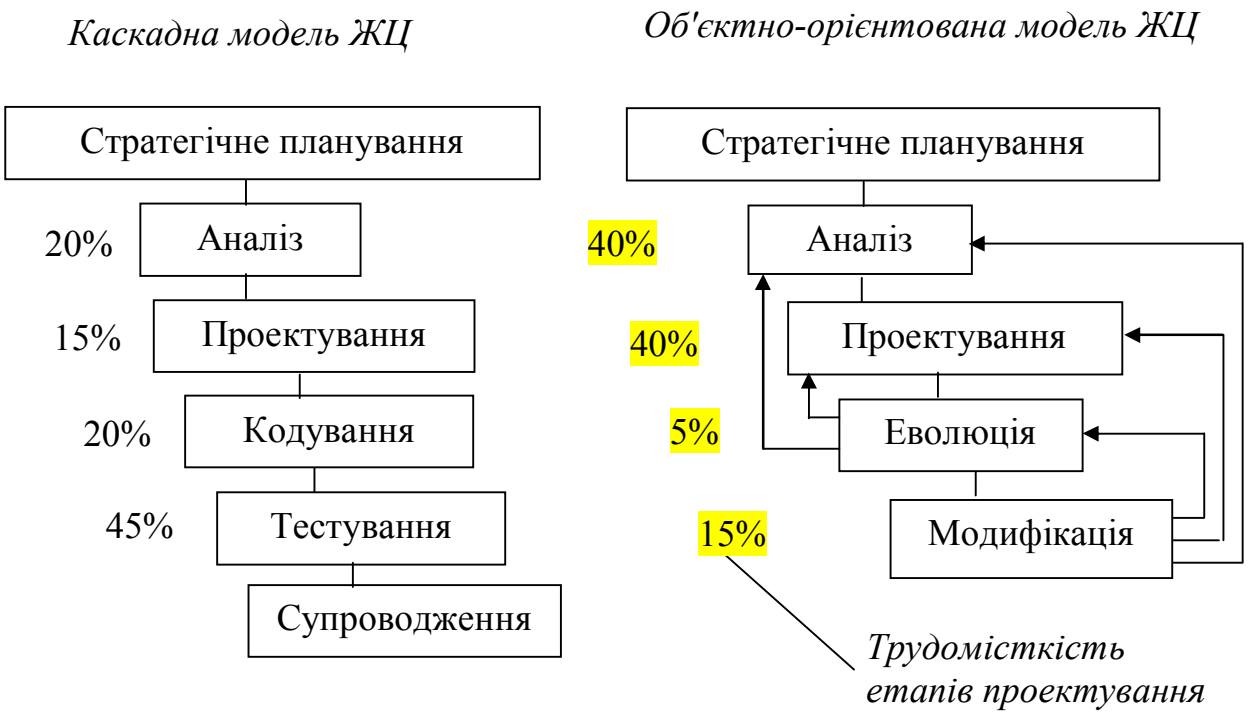



Рисунок 25- Традиційний лавиноподібний цикл розробки (каскадна модель ЖЦ) і спіральна (об'єктно-орієнтована) модель створення програмних виробів

Порівняння циклів розробки показує, що є ряд спільних рис, і ряд принципових відмінностей.
ООП проектування не є монолітним, а являє собою один із кроків на шляху послідовної итеративной розробки системи.

Характеристика етапів ЖЦ при ОО підході

Метою аналізу є відомості про завдання та вимог до системи.

У контексті аналізу вимог до системи *функція* - це абсолютний, спостережуваний і контрольований процес поведінки (фрагмент поведінки). Наприклад, функцією системи організації інформації є забезпечення різних видів пошуку в базі даних. Наприклад, на етапі виконується аналіз ключових понять (абстракцій) предметної області.

*Методи аналізу* - при ООП можна використовувати будь-який метод для отримання вихідної інформації, в тому числі і структурні (наприклад DFD), хоча це і **не рекомендується**. Вивчається видиме ззовні поведінку системи. Аналізуються прецеденти використання системи, проводиться ОО декомпозиція.

*Проектування* - полягає в поділі завдання на підзадачі, розробці абстракцій, але без зайвої деталізації. Підзадачі повинні виконувати фахівці, які зроблять це краще універсалів. Проводиться побудова певних схем, діаграм, що описують архітектуру системи. Архітектор будує будинок, вказує, що він повинен опалюватися, мати водо- і електропостачання, може вказати місця введення трубопроводів. Але проектувати окремі системи повинні фахівці.

*Эволюція системи* – поєднує традиційні етапи: складання програм, їх тестування та інтеграцію (комплексування): відбувається послідовна розробка ряду прототипів.

Складання прогресивних прототипів стимулює розробку і оцінку альтернативних рішень, що дозволяє забезпечити розумний компроміс яке обмеження дійсно важливо.

В процесі еволюції з'ясовується, яке обмеження дійсно важливо: розмір і вага ЕОМ, пам'яті (космічний корабель), розмір програми, який пов'язаний з ОЗУ.

*Стратегія еволюції* повинна бути наступною:

– проектування, спрямоване на виконання заданих функцій (все поліпшити);

– потім перевірка виконання характеристик обмежень (що при цьому погіршується);

– примирення цілей - термін, що означає знаходження компромісного рішення після визначення «вузьких місць», суперечностей в системі (див. рис ).

Види змін при ОО розробці ПП;

1) Додавання нового класу - розширює функціональність або гнучкість.
2) Зміна реалізації класу (внутрішні зміни - підтримуються инкапсуляцией і визначення функцій-членів класу).
3) Зміна уявлення класу - зміна його інтерфейсу користувача.
4) Реорганізація структури класу - (внутрішні зміни - підтримуються инкапсуляцией і внутрішньою структурою зберігання даних).
5) Зміна інтерфейсу класу для клієнтів.

*Модифікації.* Модифікація пов'язана зі зміною ПЗ, збільшує складність системи, іноді збільшується зв'язність і порушується інкапсуляція. При ООП модифікація є природною операцією, відбувається розвиток «свідомості» класу [Т. Сван].

При вдосконаленні ПЗ діють закони:

1) Закон збільшення складності при зміні ПЗ. При зміні ПЗ, воно стає більш складним (загальна тенденція), якщо при цьому не додаються додаткові зусилля для зменшення складності.
2) Закон безперервного зміни ПЗ. Програма, яка реально використовується, повинна змінюватися, інакше вона знижує свої споживчі властивості.

В цілому розробка моделей ЖЦ CASE - засобів і технологій створення ПЗ перетворила проектування в інженерну дисципліну, що забезпечує документування і контроль якості розробки.

Існують дві базові моделі: структурна і ОС.

Крім логічного і фізичного рівнів з'явилися методології концептуального моделювання.

Розвинені методи підтримки загальних процесів проектування:

- управління проектом;
- управління якістю;

– репозиторії.

В цілому структура ЖЦ є стандартом ISO/IEC 12207. ЖЦ базується на трьох групах процесів:

1) Основні процеси ЖЦ ПЗ - це придбання, постачання, розробка, експлуатація, супровід.

2) Допоміжні процеси забезпечують основне: документування, управління конфігурацією, забезпечення якості, верифікації, атестація, оцінка, аудит, рішення різних проблем.

3) Організаційні процеси: управління проектами, створення інфраструктури проекту, визначення етапів, оцінка та поліпшення самого ЖЦ, навчання.

## Алгоритмічна і ОО декомпозиція

Алгоритмічна декомпозиція використовуються при структурному проектуванні за методом зверху - вниз.

Процес відбувається каскадно, проводиться поділ алгоритмів, виділення функцій.

В результаті створюється дерево функцій (рис. 26).

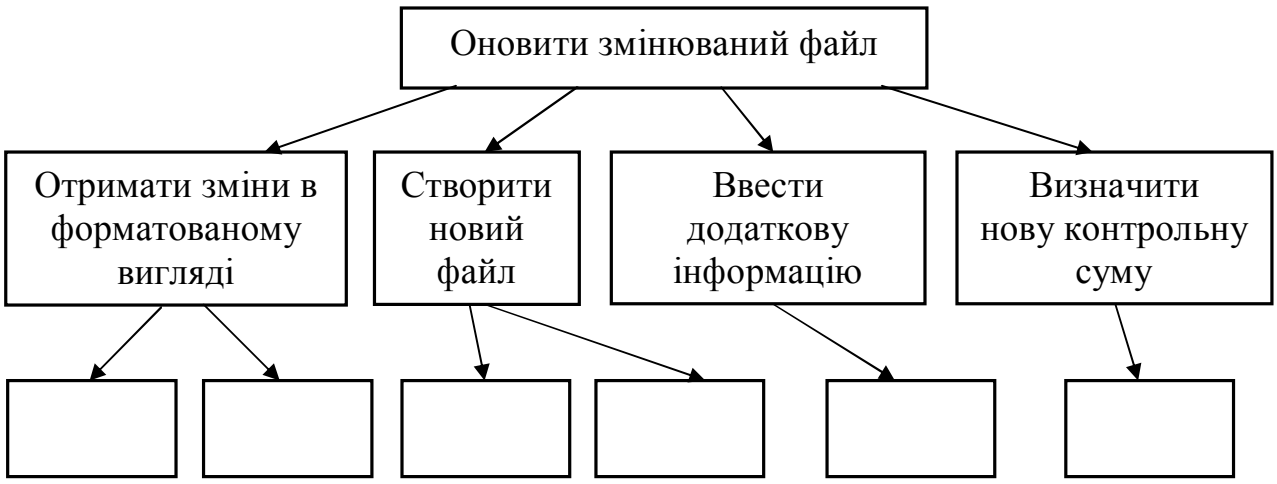



Рисунок 26 – Дерево функцій

При функціональної декомпозиції завдання розбивається на кроки, при цьому увага акцентується на порядку, послідовності виконання функцій.

Алгоритм декомпозиції документують, за допомогою діаграми потоків даних DFD (логічна модель).

## ОО декомпозиція

Це альтернативний спосіб поділу системи. Базується на виділенні понять (істотних) даного ПЗ. Поняття зберігається в словнику даних.

Наприклад, «основний файл (керуючий)», файл змін «контрольна сума».

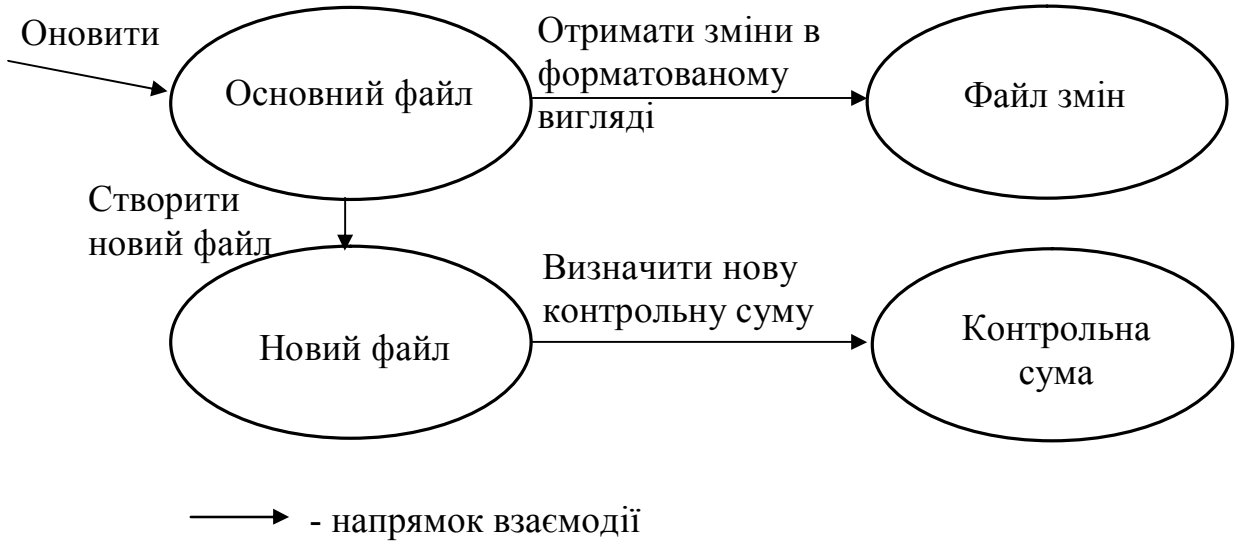



Рисунок 27 – Декомпозиція

Розробка ПЗ двома шляхами відразу неприпустима. ОО програмування (підхід) відразу застосовувати краще, а потім можна перейти і до функціонального. Кожен з них заснований на застосуванні ряду засобів аналізу систем. Для структурного аналізу використовують діаграми потоків даних (DFD), діаграми сутність-зв'язок (ERD) і діаграми переходів станів (STD) [3].

В ООП кожен об'єкт характеризується станом, поведінкою і індивідуальністю [4]. Для логічного представлення системи застосовують діаграми класів і об'єктів, однак не виключають і перераховані кошти структурного аналізу. Наприклад, кошти структурного аналізу використовують при розгляді структури даних в класі (ERD), при вивченні динамічних аспектів функціонування системи застосовують STD, при розробці архітектури процесів - DFD. В цьому випадку діаграми адаптовані до концепції ООП.

Види діаграм при ООП

1) Діаграма прецедентів (варіантів використання - Use case).
2) Діаграма класів - показує статичне відношення між класами.
3) Модель поведінки - найчастіше показують за допомогою діаграми станів (автомат), діаграма діяльності.
4) Модель динамічної взаємодії між об'єктами. Показується за допомогою діаграми послідовностей і діаграми кооперації об'єктів.
5) Модель реалізації системи - діаграма розподілу (вузлів системи).
6) Діаграма компонентів, як фізичних програмних одиниць (елементів) системи.

Між об'єктами і структурним підходом є відмінності і спільне.

## Структурний алгоритмічний підхід

1. Каскадна модель ЖЦ зверху вниз.
2. Виділення (декомпозиція функцій, процесів, алгоритмів).
3. Увага на порядок подій, що відбуваються.
4. Структура даних передається між процесами.

## ОО – підхід

1. ОО спіральна модель ЖЦ (ітерації, спрямованість на модифікацію, еволюцію).
2. Виділення понять (словник предметної області, декомпозиція на об'єкти).
3. Увага на взаємодію об'єктів.
4. Функції - операції над об'єктами.

## Переваги ООП

1. При ОО підході для великих проектів обсяг року менше. Таким чином частково вирішується проблема надмірності даних за рахунок використання загальних механізмів і класів.
2. ОО системи більш відкриті (за рахунок відкритих класів), здатні більш до модернізації (кооперація спадкування).
3. ОО системи базуються на стійких проміжних формах. Розробка діаграм класів дозволяє розвивати, модернізувати систему досить тривалий час.
4. ОО підхід передбачає еволюційний шлях розвитку, при цьому зменшується ризик створення надскладних програмних систем.
5. Об'єктна декомпозиція на етапі аналізу ПЗ дозволяє відразу оцінити складність системи.
6. Структурний підхід не дозволяє виділити абстракції у вигляді класів в ООП, для виділення абстракцій служать класи.
7. Структурний підхід не забезпечить захист даних (инкапсуляцию).
8. Структурний підхід являє менше коштів для паралельної розробки.
9. При ОО підході спрощується модернізація за рахунок класів.

## Базові поняття в ООП

### Поняття об'єкта

Об'єкт - це відчутний (видимий) предмет, так як він має фізичну природу.

Об'єкт - це дещо, що сприймається мисленням або на що спрямована діяльність.

Об'єкт - це щось, що має чітко визначені межі по відношенню до інших об'єктів.

Об'єкт має статки.

Об'єкт проявляє поведінку.

Об'єкт має індивідуальність.

Сукупність станів і поведінки визначає загальний для об'єктів клас, під визначення якого підходять ці об'єкти.

Індивідуальність - контрольний набір значень атрибутів, що визначає стан об'єкта.

Атрибути - властивості, характеристики об'єктів.

Стан об'єкта характеризується переліком всіх можливих властивостей об'єкта.

Перелік властивостей:

1. Статична складова.
2. Динамічна складова значення кожної складової.

Значення властивостей змінюється набагато швидше, ніж якість властивостей.

Значення параметрів можуть бути кількісними, можуть бути об'єктами, над якими можна здійснювати будь-які дії.

Індивідуальність визначається ім'ям об'єкта, місцем в пам'яті, значенням атрибутів.

Індивідуальність визначається ім'ям об'єкта, місцем в пам'яті, значенням атрибутів.

Поведінка об'єкта характеризує як об'єкт впливає на інших і/або як піддається впливу інших з точки зору зміни стану і передачі повідомлень. Поведінка об'єкта визначається сукупністю операцій.

Операція - певний вплив одного об'єкта на інший.

Поняття «Операція» та «Повідомлення» в більшості випадків збігаються.

Операції реалізуються методами класів.

Види операцій:

- конструктор;
- деструктор;
- функція типу Get (селектор) - повертає;
- функція типу Set (модифікатор) - змінює стан об'єкта;
- функція итератор - функція, яка забезпечує доступ до елементів об'єкта по частинах, в певній послідовності.

Загальнодоступні функції, розміщені поза класів, для роботи з ними називаються утилітою.

Методи і утиліти визначають оболонку поведінки об'єкта, так як охоплюють і внутрішній стан зовнішню поведінку об'єкта. Їх сукупність називають протоколом об'єкта.

Залежно від реалізованого поведінки об'єкти можуть бути активними (автономні - володіють власним автоматом), пасивними (реактивними) можуть бути акторами (виконують функції активного і пасивного об'єкта).

Для поведінки активних об'єктів використовується теорія кінцевих автоматів.

## 3.5 Принципи організації інформації про систему для ефективної обробки на ЕОМ

### Аналіз сучасних засобів структурного аналізу систем і їх застосування

Засоби структурного аналізу систем застосовуються при побудові логічної моделі системи, що є обов'язковою частиною дипломного проекту студентів спеціальності «Інформаційні технології проектування». В процесі виконання дипломних проектів студенти виконують ряд етапів розробки програмних продуктів, пов'язаних з аналізом різних систем і формалізацією інформації про їх будову і функціонування. Метою аналізу є повне всебічне опис системи та побудова функціональної логічної моделі системи (специфікації системи). Потім розглядається фізична модель системи.

Зокрема при об'єктно-орієнтованому проектуванні (ОПП) моделі проектування можна представити в наступному вигляді (рисунок 28) [32].

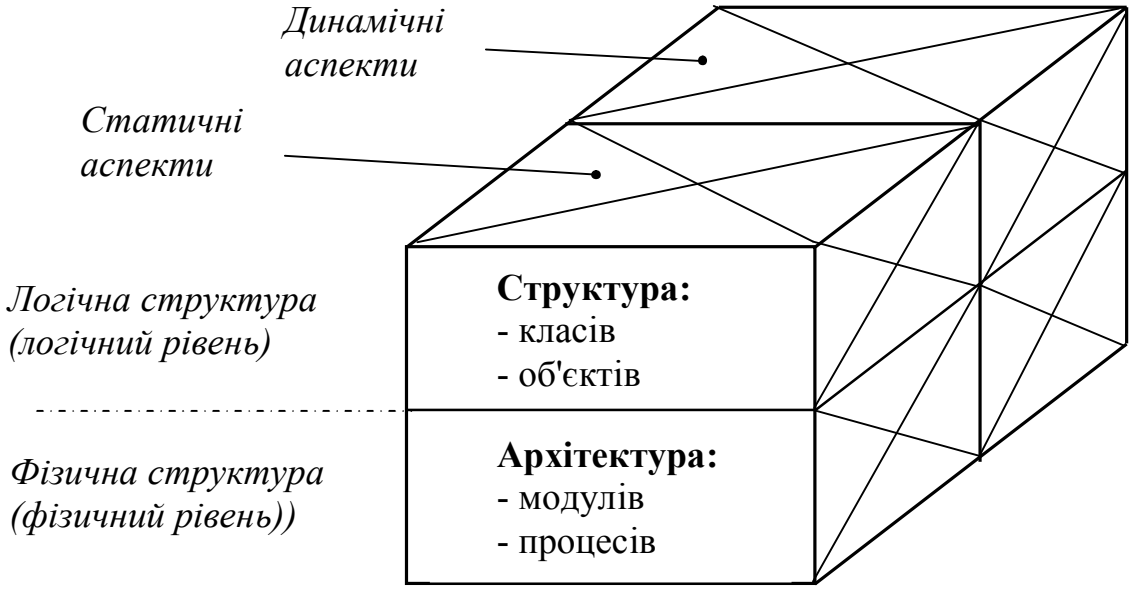



Рисунок 28 - Види моделей при ОО аналізі і проектуванні систем

При ОО - підході також будується логічна модель системи. При розробці логічної структури системи виділяються:

– статичні аспекти (показують структурні відносини - вони більш стійкі). Поведінка системи описується при розгляді;

– динамічні аспекти системи, зокрема, базовими логічними структурами є структура чи діаграма класів і діаграма об'єктів (кооперація). Діаграма об'єктів менш статична і може істотно відрізнятися від діаграми класів.

Діаграма класів показує, які класи існують в програмі і як вони взаємопов'язані між собою, як забезпечується видимість об'єкта класів (включення, посилання і об'єкт).

Діаграма об'єктів - це діаграма часу виконання. Показує механізми (іноді послідовність) взаємодії об'єктів.

Обидві діаграми структурні.

Існують і діаграми поведінки.

На фізичному рівні кошти реалізації не розглядають.

При розробці фізичної структури розробляють модульну структуру, де визначають в якому місці оголосити класи об'єкти.

Такими фізичними елементами фізичної структури є компоненти у вигляді DLL.

Будують також діаграми процесів. Вони показують, яким процесору в системі приписати конкретний процес і як управляти процесами.

Перші дві діаграми служать для опису ключових абстракцій ПО, інші дві - описують конкретні програмні реалізації, а також апаратні реалізації.

Проаналізуємо існуючі методики розробки та подання логічної структури системи і її фізичної структури.

На етапі розробки логічних і фізичних моделей (статичні аспекти) вирішуються питання, які при розгляді представляються у вигляді:

*логічне уявлення системи:*

- діаграма класів, які класи і як вони пов'язані між собою;
- діаграма об'єктів, які механізми забезпечують взаємодію об'єктів;

*фізична структура системи:*

- діаграма модулів, в якому місці оголошувати класи і об'єкти;
- діаграма процесів, якому процесу приписати конкретний процес, як управляти процесами.

Перші дві діаграми служать для опису ключових абстракцій проекту. Решта дві діаграми описують конкретні програмні і апаратні компоненти реалізації проекту. Застосовуються і засоби структурного аналізу систем.

При структурному підході для цілей моделювання систем взагалі і структурного аналізу зокрема використовують три групи інструментальних засобів, що ілюструють:

- функції, які система повинна виконувати;
- відносини між даними;
- залежне від часу поведінка системи (аспекти реального часу).

Подання інформації вимагає використання наочних діаграмних методик. Існує безліч різних засобів візуалізації інформації, які застосовуються на практиці. Вибір цих коштів залежить від розв'язуваної задачі, тому розглянемо ряд найбільш часто використовуваних для зазначених завдань діаграм і методик [4]:

**- DFD** (Data Flow Diagrams) - діаграми потоків даних; спільно зі словниками даних і специфікаціями процесів (мініспеціфікація) ілюструють функції (процеси), які система повинна виконувати;

**- ERD** (Entity Relationship Diagrams) - діаграми «сутність-зв'язок» показують відносини між даними;

**- STD** (State Transition Diagrams) - діаграми переходу станів показують залежне від часу поведінка системи (аспекти реального часу).

Перераховані кошти дають повний опис системи незалежно від її новизни (рисунок 2.2). Проводиться побудова функціональної логічної специфікації -

докладний опис того, що повинна робити система, без розгляду шляхів реалізації (чітке уявлення про кінцеві результати). Логічна DFD показує зовнішнє по відношенню до системи джерела і стоки (адресати) даних, ідентифікує логічні функції (процеси) і групи елементів даних, що пов'язують одну функцію з іншого (потоки), ідентифікує сховища (накопичувачі) даних.

Структури потоків даних зберігаються і аналізуються в словниках даних. Кожна логічна функція може бути деталізована за допомогою DFD нижнього рівня. У разі наявності реального часу використовують STD - діаграми.

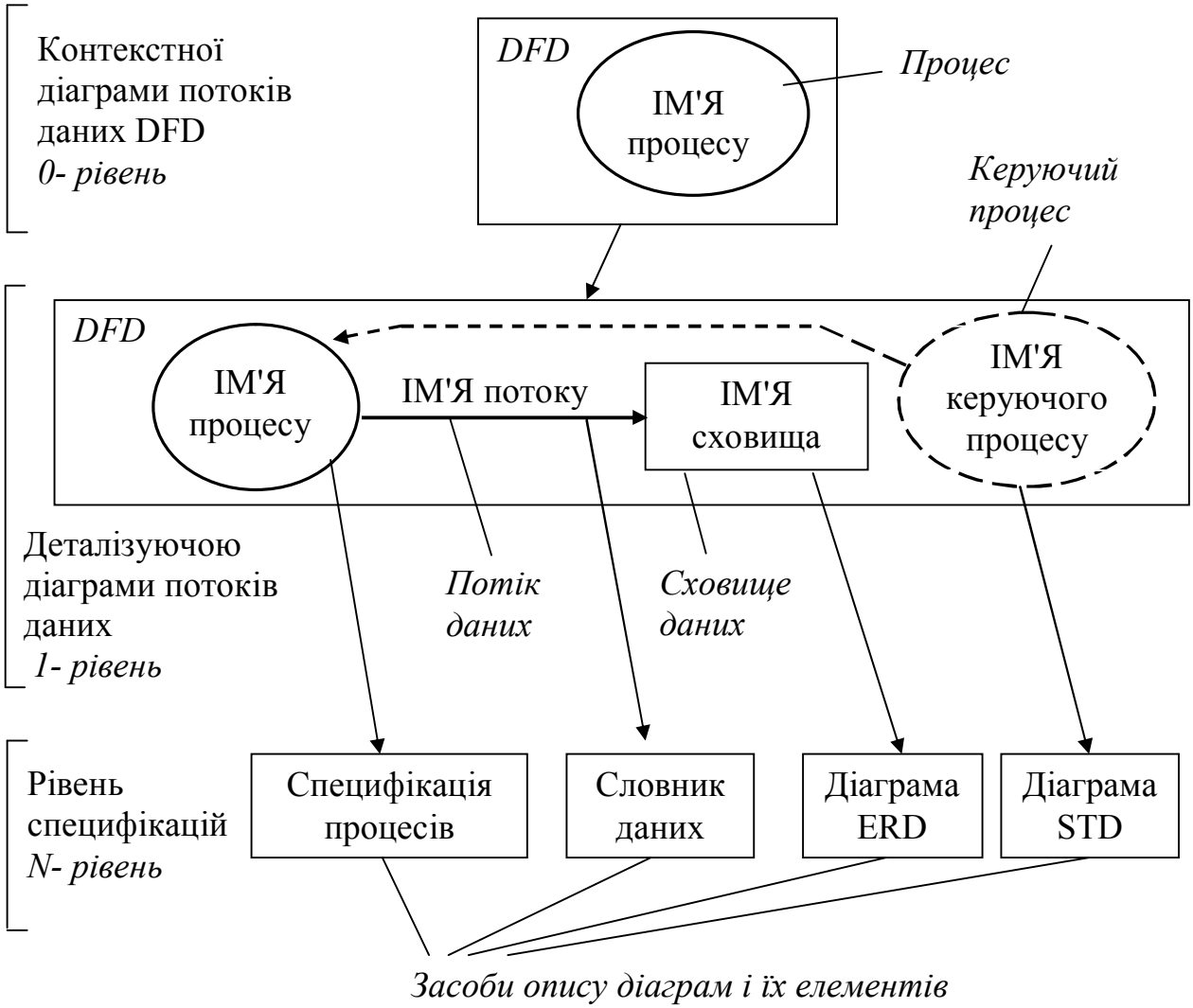



Рисунок 29 - Компоненти логічної моделі системи і види діаграм

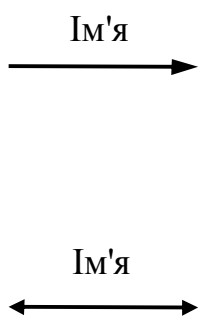
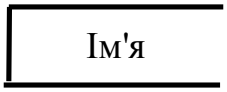
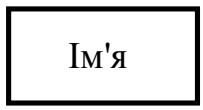
### 3.5.1 Діаграми потоків даних

Діаграми потоків даних DFD (Data Flow Diagrams) - є основним засобом моделювання функціональних вимог до проектованої системі. З їх допомогою ці вимоги розбиваються на функціональні компоненти (*процеси*), пов'язані *потоками* даних і подаються у вигляді графа.

Головна мета таких засобів - продемонструвати, як кожен процес перетворює свої *вхідні дані* у *вихідні*, а також виявити відносини між цими процесами.

Для зображення DFD традиційно використовують наступну нотацію, представлену у таблиці 7.

Таблиця 7 - Позначення на діаграмі потоків даних

| Найменування і призначення елемента | Позначення |
|---|---|
| *Потік даних* – механізм, що використовується для моделювання передачі інформації (або навіть фізичних компонентів) з однієї частини системи в іншу. *Ім'я потоку* – ідентифікує його зміст і є *іменником*.<br>Потік зображуються *іменованими стрілками*, орієнтація - напрямок передачі (руху) інформації. Рух можливий в одному напрямку - *односпрямований потік*, або в двох: обробка, повернення; моделюються двома потоками або одним *двонаправленим* |  |
| *Процес* – продукує вихідні потоки з вхідних відповідно до дії, що задається ім'ям процесу.<br>*Ім'я процесу* – містить *дієслово в невизначеною формі* з подальшим доповненням (наприклад, «Обчислити максимальну висоту»).<br>*Номер* – унікальний номер процесу для посилань на нього всередині DFD. Може використовуватися спільно з номером діаграми для отримання унікального індексу моделі. |  |
| *Сховище* (накопичувач даних) – дозволяє на певних ділянках визначати дані, які будуть зберігатися в пам'яті між процесами. Фактично це «зрізи» потоків даних у часі.<br>Інформація, яку вони містять може використовуватися в будь-який час після її визначення. Дані можуть вибиратися в будь-якому порядку.<br>Ім'я *сховища* – ідентифікує його зміст і є *іменником*.<br>Якщо потік даних входить або виходить зі сховища і містить (відповідає) структурі сховища, він повинен мати те ж ім'я (не потрібно відображати на діаграмі). |  |
| *Зовнішня сутність* (термінатор) – представляє сутність поза контекстом системи, що є джерелом або приймачем системних даних.<br>*Ім'я* – повинно містити іменник, (наприклад, «Склад товарів»). Передбачається, що об'єкти, що представлені такими вузлами, не беруть участі ні в якій обробці. |  |

### 3.5.2 Контекстна діаграма і деталізація процесів

Важливу роль в моделі грає спеціальний вид DFD - *контекстна діаграма*, - моделює систему найбільш загальним чином (на найвищому рівні). *Контекстна діаграма* моделює (відображає) інтерфейс зв'язку системи із зовнішнім світом, а саме, інформаційні потоки між системами і зовнішніми сутностями, з якими вона повинна бути пов'язана. зовнішні по відношенню до системи витоки даних, основний процес (Ф) обробки Вона ідентифікує *зовнішні сутності*, а також як правило, *єдиний процес*, що відображає *головну мету* або природу системи. Кожен проект має тільки одну контекстну діаграму (О - рівня).

Контекстна діаграма відображає інтерфейс зв'язку системи із зовнішнім світом, ідентифікує зовнішні сутності, ідентифікує головну мету системи, в якості яких можуть бути підрозділи організації, користувачі, пристрої для зберігання і передачі інформації, користувачі з різних розширенням, ін. програмні системи.

Функції контекстної діаграми потоків даних:

1) Відображає інтерфейс зв'язку системи із зовнішнім світом.
2) Ідентифікує зовнішні сутності.
3) Ідентифікує головну мету системи.

Декомпозиція контекстної діаграми здійснюється за рівнями: 1-й і 2-й.

Вкладеність декомпозиції визначається складністю розглянутих процесів.

Кожен рівень додає індекс.

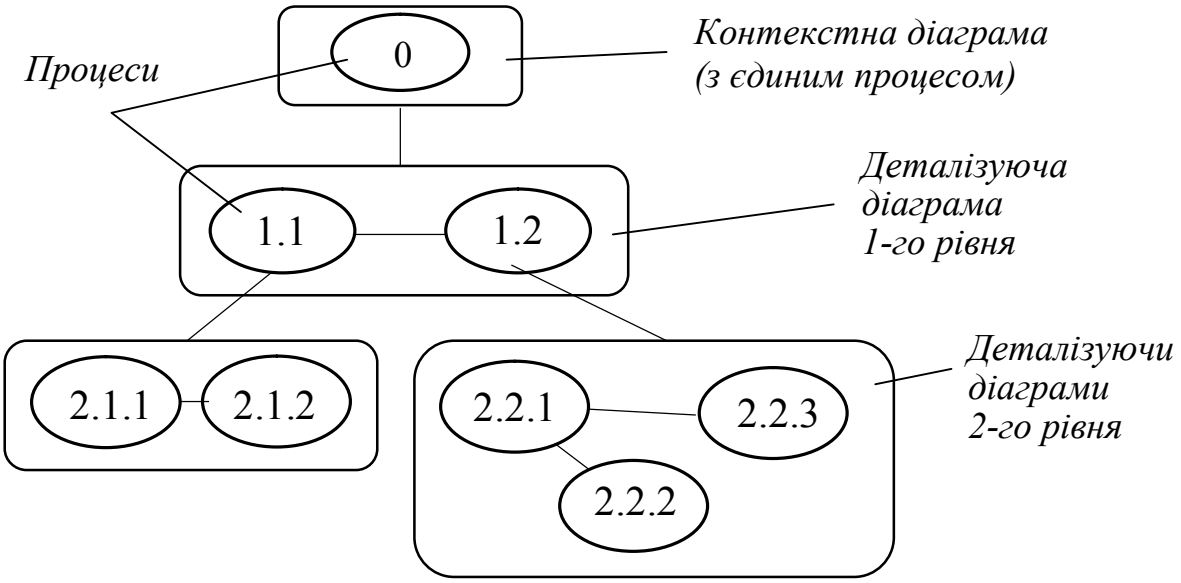



Рисунок 30 - Схема нумерації вузлів при декомпозиції діаграми

Зокрема, DFD першого рівня будується як декомпозиція процесу контекстної діаграми. Діаграма першого рівня має безліч процесів з номерами, які використовуються при подальшій декомпозиції. Процес першого рівня декомпозиції містить процеси 1.1, 1.2, 1.3, ... . При переході на наступний рівень додається індекс - 2.1.1, 2.1.2, ... і т. д. (см. рисунок 30).

Розглянемо приклад для завдання обслуговування клієнта банківською си-

стемою віддалених платежів за допомогою кредитних карт (рисунок 31 ).

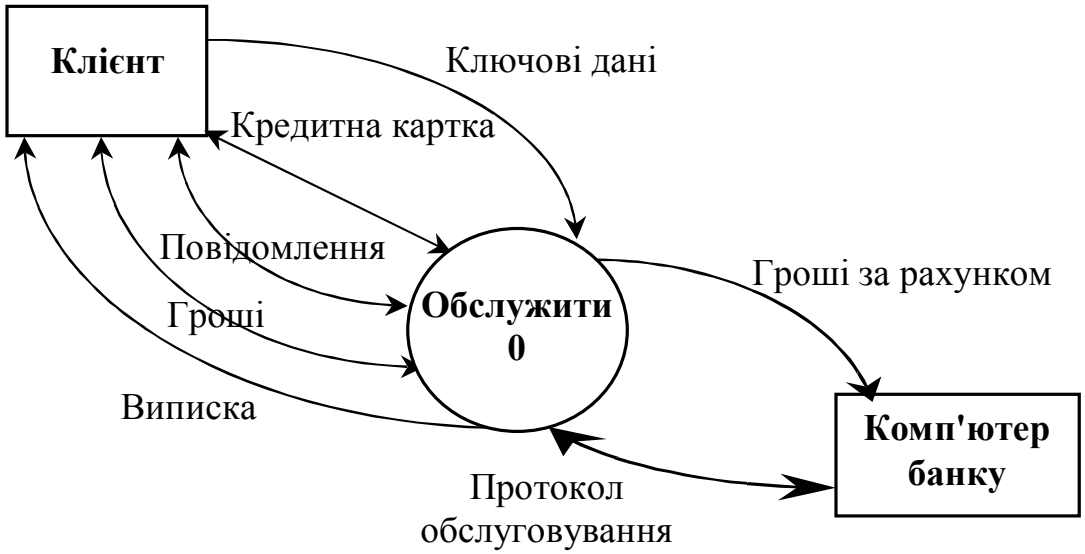



Рисунок 31 - Контекстна діаграма банківської задачі

Сценарій розвитку ПЗ - використання банкомату.

Зовнішні сутності в даній системі: **Клієнт**, **Комп'ютер банку**. Клієнт звертається до системи, а комп'ютер зберігає інформацію про рахунки клієнта. Клієнт і комп'ютер банку не входять в систему і не обробляють інформацію.

Основний процес в системі - **Обслужити**, оперує інформацією про рахунки всіх клієнтів.

Потоки даних, якими обмінюється проектована система з зовнішніми об'єктами, наприклад, **Клієнт** представляє **Кредитну карту** для автоматичного зчитування з неї інформації (Пароль, Ліміт грошей, Деталі клієнта), а також повідомляє свої **Ключові дані** (Пароль, Запит на обслуговування, тобто необхідну послугу, наприклад, зняти готівку з рахунку).

Банківська система обслуговування повинна забезпечити такі функції для клієнта:

- видавати **Повідомлення** - запрошення клієнту ввести **Ключові дані**;
- видати **Виписку про гроші**, **Виписку по бланку**, **Виписку по операції**, проведеної банком.

**Процес** і **Комп'ютер банку** повинні обмінюватися такою інформацією:

- **Дані по рахунку** клієнта в банку;
- **Протокол обслуговування,** що включає інформацію про оброблену документацію, що вилучається **Грошовою суми** і **Дані з історії запиту**.

Клієнт підходить до банкомату, бачить «запрошення до роботи». Клієнт вводить кредитну карту, банкомат приймає її і встановлює зв'язок з комп. Банку для контролю. Просить пароль. Після ідентифікації, якщо все нормально, то запитуються ключові дані по операції (сума, потрібна виписка і ін.). Після закінчення діалогу, якщо він успішний, видаються дані і виписка, повертається карта.

Клієнт і комп'ютер банку - зовнішні сутності.

Формалізований опис

Зовнішні сутності отримують і надають інформацію.

Основний процес містить і обробляє інформацію про рахунок клієнта.

Банковое обслуговування повинно забезпечувати наступне:

1) Видавати повідомлення як запрошення ввести кредитну карту, а потім ключові дані.
2) Видавати дані, виписку за рахунком (баланс), виписку з операції.

Процес і комп'ютер банку обмінюються такою інформацією:

1) Дані за рахунком клієнта в банку, пароль що зберігається.
2) Протокол обслуговування, який включає:
   - інформацію про запит;
   - грошову суму, яка знімається з рахунку;
   - дані по ідентифікації;
   - дані по історії запиту (час і т. д.).

Деталізація процесу Обслужити з використанням DFD першого рівня

Деталізація DFD здійснюється на основі декомпозиції процесів: кожен процес розкривається за допомогою DFD нижнього рівня або специфікації процесу, якщо досягнуть необхідний рівень деталізації (спрощення).

Деталізація процесу **Обслужити** приведена на рисунку 32. Основний процес поділений на ряд підпроцесів зі своїми функціями.

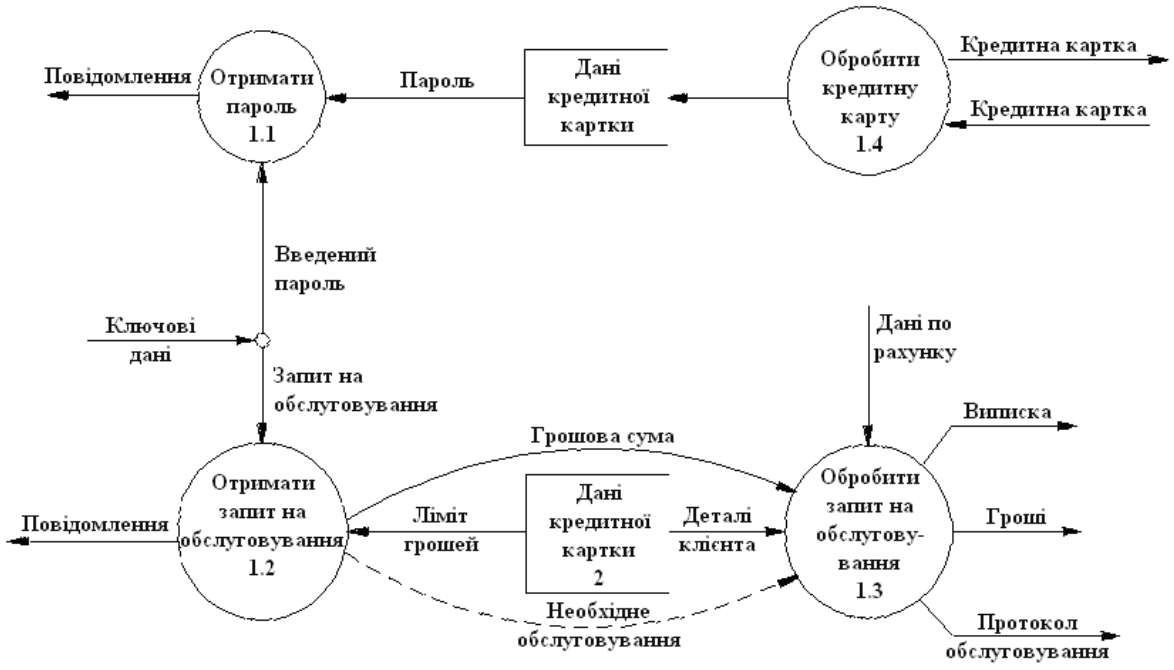



Рисунок 32 - Деталізація процесу **Обслужити** за допомогою DFD першого рівня

Розглянемо функції процесів, представлених на діаграмі:

**Процес 1.1** Здійснює прийом і перевірку пароля клієнта і має на вході і виході потоки:

– зовнішній вихідний потік **Повідомлення** про готовність прийняти пароль;

– вхідний потік **Введений пароль** як елемент зовнішнього потоку **Ключові дані**;

– вхідний потік **Пароль зі сховища**, **Дані кредитної картки** для перевірки введеного клієнтом пароля.

**Процес 1.2** Здійснює прийом і перевірку запиту клієнта на проведення необхідної банківської операції і має на вході/виході такі потоки:

– зовнішній вихідний потік **Повідомлення** для інформування клієнта про готовність прийняти запит на обслуговування;
– вхідний потік **Запит** на обслуговування як елемент зовнішнього потоку **Ключові дані**;
– вхідний потік **Ліміт грошей** зі сховища **Дані кредитної картки** для контролю наявності грошей на рахунку клієнта.

**Процес 1.3** Має зовнішній вхідний потік **Дані по рахунку** (з зовнішньої сутності **Комп'ютер банку**), вхідний потік **Деталі клієнта** (зі сховища), зовнішні вихідні потоки: **Виписка**, **Гроші**, **Протокол обслуговування.**

**Процес 1.4** Здійснює зчитування інформації з кредитної картки і має на вході зовнішній потік **Кредитна карта**, а на виході потік **Дані кредитної картки**.

**Процеси 1.1, 1.2, 1.4** є елементарними (DFD - далі не застосовуються) - деталізація проводиться за допомогою **Специфікації процесів**.

**Процес 1.3** може бути деталізований діаграмою DFD 2-го рівня.

**Рекомендації** для побудови ієрархії DFD:

– доцільно на кожній діаграмі розглядати від 3 до 6-7 процесів;

– не захаращувати діаграму несуттєвими на даному рівні деталями;

– декомпозицію потоків виробляти паралельно декомпозиції процесів (одночасно);

– вибирати ясні, що відображають суть справи імена процесів і потоків (без скорочень);

– однозначно визначати функціонально ідентичні процеси на са-мом верхньому рівні, де він необхідний, і посилатися на нього на нижніх рівнях;

– користуватися найпростішої диаграммной технікою;

– відокремлювати керуючі структури від обробних структур (тобто процесів), локалізувати керуючі структури. Необхідно відокремлювати і виділяти керуючі процеси від обробних, прагнути до локалізації елементів, процесів, потоків, даних, що відповідають за логіку.


Рисунок 33 – Створення графа моделі

### 3.5.3 Декомпозиція даних і розширення позначень потоків даних для DFD

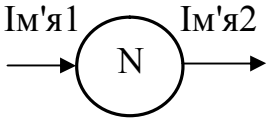
Індивідуальні дані часто незалежні, проте іноді необхідно групувати дані. Наприклад, є потоки *Яблука*, *Апельсини*, *Груші*, які можуть бути згруповані в потік *Фрукти,* який складається з декількох елементів - нащадків. Потік *Фрукти* може міститися в потоці *Їжа* разом з потоками *М'ясо*, *Овочі*. Такі потоки називаються груповими.

Застосовується і зворотна операція – розщеплення потоку на підпотоки, здійснюється за допомогою групового вузла, що дозволяє розщепити потік на кілька підпотоків (таблиця 2.2).

Аналогічно здійснюється і декомпозиція потоків через кордони діаграм для спрощення деталізуючий DFD. Вхідний потік в деталізуємий процес *Фрукти* можна не показувати, а привести потоки *Яблука*, *Апельсини*.

Застосування операцій над даними забезпечують структуризацію, наочність і читабельність діаграм.

Таблиця 8 - Таблиця розширення позначень на діаграмі потоків даних

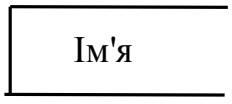
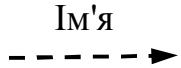
| Найменування і призначення елемента | Позначення |
|---|---|
| *Груповий вузол* - призначений для розщеплення і об'єднання потоків (може вироджуватися в точку) |  |
| *Вузол - предок* - дозволяє пов'язати вхідні та вихідні потоки між деталізуючим процесом і деталізуючею DFD |  |
| *Невикористаний вузол* - застосовується, коли декомпозиція проводиться в груповому вузлі, але потрібні не всі елементи входящого до вузла потоку |  |
| *Вузол зміни імені* - дозволяє неоднозначно іменувати потоки, наприклад, при стикуванні різних DFD, де розрізняються назви. Один потік на вході, інший - на виході, імена різні, а вузол забезпечує еквівалентність потоків даних |  |
| *Текст* - вільне застосування в будь якому місці програми | Текст |

### 3.5.4 Розширення позначень реального часу потоків даних для DFD (керуючі процеси)

Розширення позначень реального часу використовується для доповнення моделі функціонування системи (ієрархії DFD) засобами опису керуючих аспектів в системах реального часу (таблиця 2.3).

Керуючий процес - командний пункт, що реагує на зміни зовнішніх умов, які передаються керуючими потоками і продукує, виконувані процесами команди.

Таблиця 9 - Застосування символів діаграми потоків даних (DFD) для керуючих процесів

| Найменування і призначення елемента | Позначення |
|---|---|
| *Керуючий процес* – інтерфейс між DFD і специфікаціями управління - перетворювач керуючих вхідних потоків в керуючі вихідні потоки.<br>*Ім'я* – тип керуючої діяльності. Опис в специфікації. |  |
| *Керуючий сховище* – «сріз» керуючого потоку в часі, сенс той же, що і для звичайного сховища, але містить тільки керуючі потоки. |  |
| *Керуючий потік* – «трубопровід», через який проходить керуюча інформація.<br>Зазвичай має дискретне значення. |  |

*Керуючий процес* - командний пункт, що реагує на зміни зовнішніх умов, які передаються керуючими потоками і продукує, виконувані процесами команди.

Є такі типи керуючих потоків:

- *Т - потік* (triqqer flow) - викликає виконання процесу однією короткою операцією. Аналогічно - вимикач світла: лампа загоряється (або гасне) від одного натискання вимикача (процес «запускається»);

- *А - потік* (activator flow) - забезпечує безперервність виконання процесу поки потік «включений» (тобто тече безперервно). З «вимиканням» потоку виконання процесу завершується. Аналогічно - перемикач лампи, лампа може бути включена і вимкнена;

- *E/D - потік* (enable/ disable flow) - може перемикати виконання окремого процесу. Перебіг по Е-лінії викликає виконання процесу до тих пір, поки не збуджується хід по D-лінії. Аналогічно виключення з двома кнопками: одна для включення, інша для виключення. Можна використовувати 3 типи таких потоків: *E, D, E/D - потоки*.

Один фрагмент даних можна представити потоками різних типів. Наприклад, проводиться контроль досягнення критичного значення і формування керуючого потоку. використовують позначення: Р - *вузол зміни типу потоків.*

### 3.5.5 Словник даних і специфікація процесів

На DFD не вистачає опису деталей компонентів системи:

- яка інформація перетворюється процесами;
- як інформація перетворюється процесами.

Текстові засоби моделювання, призначені для опису структури преутвореної інформації - називаються *словниками даних*.

*Словник* - організований список всіх елементів даних з їх визначеннями - забезпечує загальне для всіх розуміння всіх потоків і сховищ.

Визначення елементів виконання за допомогою описів наступних видів:

– опис значень потоків і сховищ (зображених на DFD);

– опис композицій агрегатів даних, що рухаються уздовж потоків, їх декомпозиції (наприклад, Адреса покупця містить символи (поля), Поштовий індекс, Місто, Вулиця, і т.д.) - тобто структури даних;

– опис композиції (структури) групових даних в сховищі;

– специфікація значень і областей дії елементарних фрагментів інформації в потоках даних і сховищах.

– опис деталей, відносин між сховищами.

Вміст словника даних

Для кожного потоку даних в *словнику* зберігається: *ім'я* потоку, *тип* і *атрибути.*

Інформація про потоки наводиться у вигляді опису або таблиці.

*Тип потоку*:

– простий або груповий (елементарний або комплексний);

– внутрішній (тільки всередині системи) або зовнішній (пов'язує з іншими системами);

– потік даних або потік управління;

– безперервний або дискретний потік.

*Атрибути* потоку даних включають (рис. 34):

– імена - синоніми потоку даних для вузлів зміни імені;

– визначення для групових потоків (об'єднання);

– одиниці виміру потоку;

– діапазон значень для безперервного потоку, типове значення (за замовчуванням) інформація з обробки екстремальних значень;

– список значень для дискретного потоку і їх зміст (може бути посилання на таблицю);

– список номерів діаграм, в яких потік зустрічається;

– список потоків, в які цей потік входить як зовнішній груповий і розщеплюваний (може не бути на діаграмі);

– коментар, що включає додаткову інформацію (мета введення даного потоку).

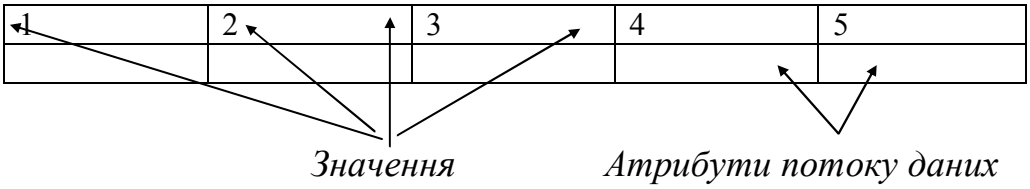



Рисунок 34 - Атрибути потоку даних і їх значення

Наприклад, дано *груповий вузол* з *вхідним потоком* X і *вихідними підпотоками* Y та Z.

У словнику даних може бути визначено, не тільки як X=Y+Z, але може бути: X=A+B; Y=A+B; Z=B+C; - розщеплення потоків.

Процеси, наведені на діаграмі описуються за допомогою специфікації процесу (СП), яка використовується якщо немає необхідності в DFD для нього, тобто якщо процес простий для розуміння. *Специфікація процесу* - це укрупнений алгоритм, який перетворює вхідні дані у вихідні.

СП містить списки вхідних і вихідних даних, ім'я процесу і специфікацію (зображення) алгоритму - по ГОСТ. Методи завдань специфікацій процесів можуть бути різні, а мета - автогенерація коду.

Підвищення складності проектування вимагає ускладнення опису:

- текстовий опис;
- структурована природна мова;
- таблиця рішень;
- дерево рішень;
- візуальна мова;
- мова програмування.

Підвищення складності проектування

Специфікація управління. Діаграми переходів станів STD

*Специфікації управління* призначені для моделювання і документування аспектів систем, що залежать від часу або реакції на подію. Вони дозволяють здійснювати декомпозицію керуючих процесів і описують відносини між вхідними керуючими потоками на керуючому процесі - предка.

SDT дозволяють документувати і моделювати аспекти системи, які залежать від часу і реакції на події.

Ці два фактори можуть змінювати стан системи.

SDT діаграми дозволяють виконувати декомпозицію процесів і описують відношення між вхідними та вихідними керуючими потоками. В цілому SDT діаграма моделює подальше функціонування на основі поточного та попереднього функціонування.

В процесі функціонування система знаходиться в певному (одному з безлічі) стані і в залежності від ситуації або часу (повідомлень, подій) може переходити в інший стан. Такі переходи повинні бути строго визначені.

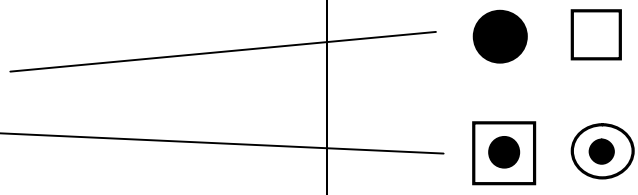
STD моделює подальше функціонування системи на основі її попереднього і поточного функціонування. Модельована система в будь-який момент часу знаходиться в одному з кінцевого безлічі станів. В часі вона змінює стан, причому всі переходи повинні бути точно визначені.

*Стан* - умова стійкості для системи.

*Ім'я* - має відображати реальну ситуацію, в якій знаходиться система: НАГРІВАННЯ, ОХОЛОДЖУВАННЯ і т.д.

Перебуваючи в певному стані, ми маємо достатньо інформації, щоб по історії системи і поточним вхідним подіям визначити майбутній (черговий) стан системи.

Таблиця 10 - Таблиця розширення позначок на діаграмі переходів станів (STD)

| Найменування, визначення | Позначення |
|---|---|
| *Стан* – це умова стійкості для системи.<br>*Ім'я* - має відображати стан або реальну ситуацію, в якій знаходиться система. Наприклад, нагрівання, охолодження, світіння, відбиття, гальмування, обертання. | Ім'я |
| Перехід - визначає переміщення системи з одного стану в інший.<br>Перехід ідентифікує подію, яке є його причиною і управляє переходом. Ім'я переходу в явному вигляді не вказують. | [Умова]<br>[Дія] |
| Варіанти станів:<br>- вихідне (початкове);<br>- кінцеве. |  |

*Початковий стан* - вузол STD, який є стартовою для початкового системного переходу.

Всі поняття структурного підходу, пов'язані з SDT, застосовуються в ОПП.

Перебуваючи в певному стані і знаючи історію системи, можна по біжучим вхідним впливам (подій) визначити майбутнє (чергове) стан системи.

STD має тільки одне *початковий стан*, Відповідне, наприклад, станом системи інсталяції, але перед початком реальної обробки, а також будь-яке (кінцеве) число *завершальних станів*. (таблиця 10).

Події, що викликають перехід, складаються зазвичай з керуючого потоку (сигналу), який виникає із зовнішнього світу (дії оператора), всередині системи або тимчасова подія.

Подія виникає при виконанні певного умови. Наприклад, «кнопка натиснута», «необхідну кількість деталей отримано», «цикл обробки завершений».

Таким чином можна вважати, що умова - це подія, що викликає перехід.

Обмеження:

1) Не всі події викликають перехід.
2) Подія не завжди викликає перехід.
3) Подія не завжди викликає перехід в той же самий стан. Наприклад - стек.

З переходом можуть зв'язуватися дії - це операція, яка виконується при здійсненні переходу.

Дія може бути прив'язана до стану як пост - або передумови.

Умова ідентифікується ім'ям, яке наводиться лапками.

Приклади застосування SDT-діаграми

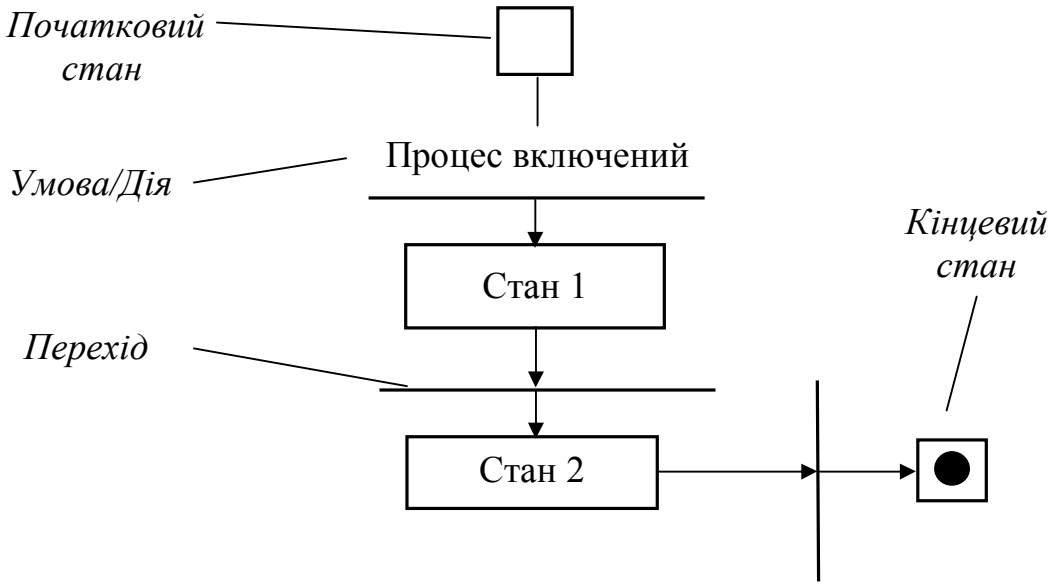



Рисунок 35 - Приклад застосування STD-діаграми

За допомогою STD виконаємо декомпозицію керуючого процесу «Управління обслуговування кредитної карти»


Рисунок 36 - SDT-діаграма

SDT-діаграму можна представити у вигляді таблиці (табл. 11).

Таблиця 11 - Приклад таблиці переходів станів для системи

| Поточний стан | Умова | Дія | Наступний стан |
|---|---|---|---|
| Початковий стан | Активізується кожен раз | — | Обробка |
| Очікування | Введена кредитна картка | Отримати пароль | Обробка |
| Обробка | Некоректний пароль | Видалити кредитну картку | Очікування |
| Обробка | Коректний пароль | Забезпечити вимоги сервісу<br>Видалити кредитну карту | Очікування |

*Перехід* визначає переміщення модельованої системи з одного стану в інший.

*Ім'я переходу* - ідентифікує подію, яка є причиною переходу і керує ним. *Подія* зазвичай складається з *керуючого потоку* (сигналу), що виникає із зовнішнього світу або всередині системи при виконанні деякої *умови* (ЛІЧИЛЬНИК = 99 КНОПКА НАТИСНУТА). Необхідно мати на увазі, що:

- в повному обсязі події викликають переходи;
- події не завжди викликають переходи;
- подія не завжди викликає перехід в той же самий стан.

*Умова* - подія, що викликає перехід і дозволяє ідентифікувати вас ім'ям переходу. З переходом може зв'язуватися дію або ряд дій, що виконуються при переході.

*Дія* - операція, яка може мати місце при виконанні переходу.

*Стани* - вузли, а *переходи* - дуги.

*Умова* - ідентифікуються ім'ям переходу (у ″ ″).

*Дії* - відгуки на події. Прив'язуються до переходів і записуються під умовою. При побудові STD рекомендується слідувати правилам:

- будувати STD на якомога більш високому рівні;
- будувати якомога простіші STD;
- по можливості деталізувати STD (ієрархія);
- використовувати ті ж принципи іменування станів, подій, дій, що і при іменуванні процесів і потоків.

Існує два способи побудови STD:

1) Ідентифікація всіх можливих станів і дослідження не безглуздих зв'язків (переходів) між ними.

2) Побудова зверху вниз від початкового стану (аналогічно побудові дерева, графа).

Контроль STD - діаграми: чи стану визначені і мають Ім'я; чи стану досяжні; чи стану мають вихід; для кожного стану визначається: чи реагує система відповідним чином на всі можливі умови (особливо на ненормальні); чи всі вхідні (вихідні) потоки керуючого процесу відображені в умовах (діях) на STD.

Матриця зображення переходів станів наведена в таблиці 2.6.

Таблиця 12 - Матриця зображення переходів станів

| Список станів, з яких відбувається перехід | Список станів, в які відбувається перехід | Ім'я стану 1 | Ім'я стану 2 |
|---|---|---|---|
| Ім'я стану 1 | Ім'я стану 2,... | | |
| Ім'я стану 2 | Ім'я стану 3,... | Умови/дії при переході → | |

Рекомендації з побудови SDT

1) Будувати SDT на високому рівні без зайвої деталізації.
2) Намагатися управляти SDT.
3) Виконати декомпозицію SDT на наступному рівні.
4) Використовувати ті ж принципи іменування елементів, що і в DFD, забезпечити збіг імен.

Способи побудови і контроль SDT діаграм

1) Ідентифікуються всі можливі стани.
2) Ідентифікуються всі можливі зв'язки (побудова знизу - вгору).
3) Зверху вниз шляхом послідовної декомпозиції.

Контроль діаграм:

1) Чи всі стани визначені і мають ім'я.
2) Чи всі стани досяжні.
3) Чи всі стани мають вихід.

Для кожного стану визначається:

1) Чи реагує система належним чином на вхідні події.
2) Чи всі вихідні і вхідні потоки відображені в умовах або діях.

## 3.6 SADT - технологія аналізу і проектування

SADT (Structured Analysis and Design Technique -1973г.) - використовується в рішеннях широкого спектру завдань: телефонні мережі, системна підтримка і діагностика, довгострокове і стратегічне планування, автоматизоване виробництво і проектування конфігурація комп'ютерних систем, навчання, вбудоване ПО для оборонних систем, управління фінансами та матеріально-технічним постачанням та ін. SADT технологію доцільно використовувати на

ранніх етапах життєвого циклу системи для розуміння сутності функцій і взаємозв'язків в системі.

У даній технології використовуються поняття:

– предмети системи (як сукупність даних);

– активності системи (функції системи).

Повна модель містить два види діаграм:

– наводяться активності (блоки), які висловлюють свої відносини через предмети системи (зв'язку);

– наводяться моделі даних (предмети), пов'язані активностями (функціями).

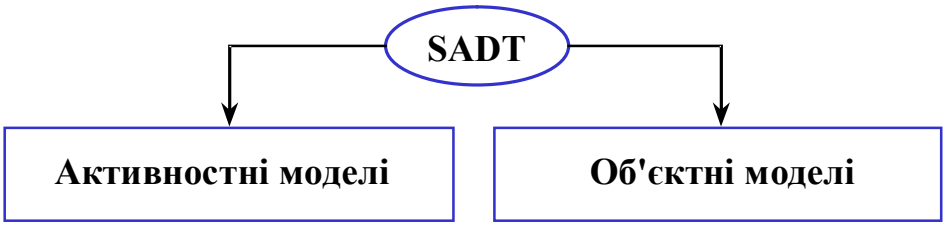



Рисунок 37 - Повна модель SADT технології (види діаграм)

Активносна (функціональна) модель проектованої системи

Розглянемо активносну модель, яка знайшла найбільш широке застосування. Модель SADT об'єднує діаграми в ієрархічні деревоподібні структури.

У діаграмі одного рівня 3-6 блоків, замість однієї громіздкою моделі використовуються кілька взаємопов'язаних моделей, (декомпозиція), що забезпечує структуризацію проблеми (системи).

*Блоки* - означають активності і супроводжуються текстами природною мовою, що описує ці активності. Блоки нумеруються.

Кожна сторона блоку має своє призначення і показує принципи функціонування блоку (рисунок 5.2):

– *входи* - перетворюються у *виходи*;

– *управління* - обмежує і наказує умови виконання діяльності;

– *виконавці* - описують, за рахунок чого (ким) виконуються перетворення.

*Дуги* позначають набори предметів і супроводжуються текстом на природній мові. Предмети складаються в 4-х можливих відносинах з активностями (Вхід, Вихід, Управління, Виконавці).

Таким чином сторони блоку графічно сортують предмети, зображувані дугами.

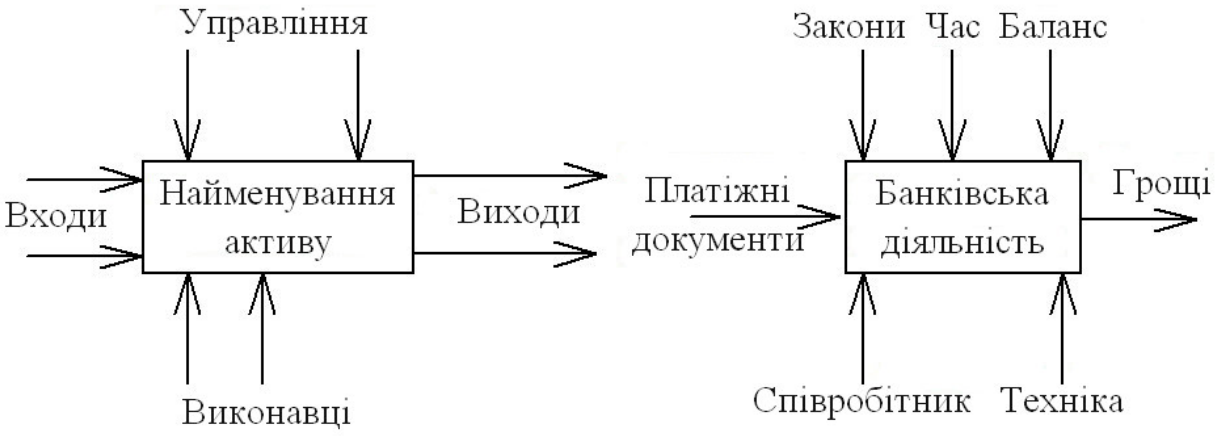



Рисунок 38 - Призначення сторін блоку і приклад SADT - блоку

Розміщення блоків на діаграмі проводиться за ступеневою схемою відповідно до їх *домінуванням* - впливом одного блоку на інший. Номери проставляються, наприклад, відповідно до домінування. Взаємовплив виявляється в пересиланні Виходу однієї активності до іншої - для подальшого перетворення, або у виробленні керуючої інформації, яка наказує, що повинна робити інша активність.

**Висновок**. SADT діаграми - наказують правила перетворень предметів і описують перетворення між Входом і Виходом (перетворення предметів, інформації, даних). Нове, в порівнянні з іншими структурними технологіями (діаграмами):

– виділення, закріплення рядків блоку за конкретними типами предметів (зв'язків);

– «ступенева» (ієрархічне) розташування блоків - домінування.

Відносини між активностями системи

Використовується п'ять типів взаємозв'язків (відносин) між активностями (рисунок 39).

Зворотні зв'язки (в, г) відображають ітерацію або рекурсію - виходи з однієї функції А впливають на майбутнє виконання інших функцій, що згодом впливає на вихідну А (активність).

Зв'язок Вихід - Виконавець - виникає, коли Вихід однієї функції (активності) А стає засобом досягнення мети інший А.

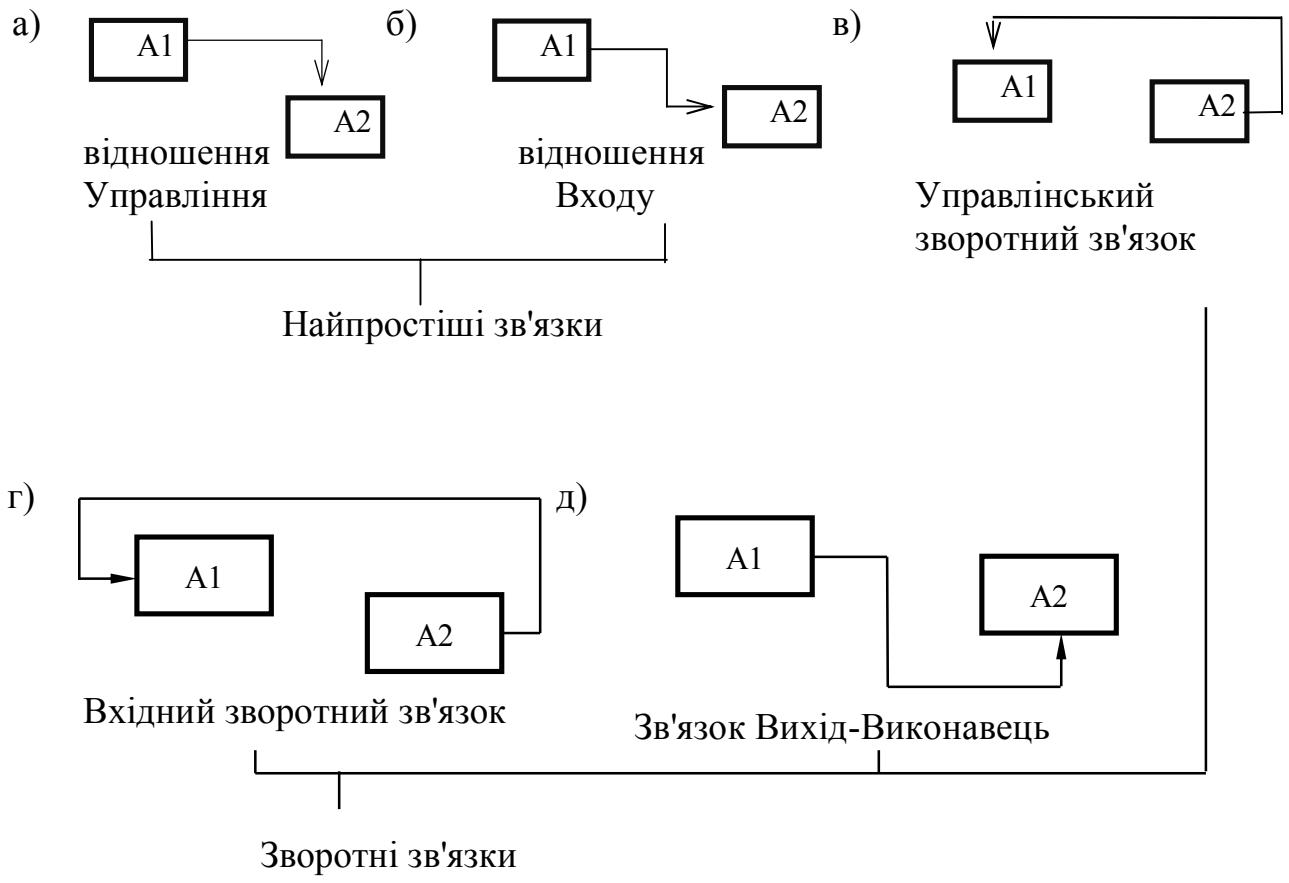
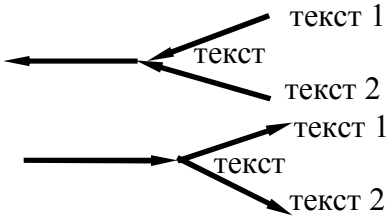



Рисунок 39 - Типи взаємозв'язків активностей системи

Дуги можуть з'єднуватися і розгалужуватися.

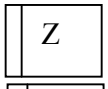


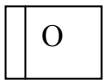

Тексти відображають набори предметів, при чому може бути: текст 1 ≤ текст; текст 2 ≤ текст;

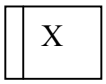
У разі рівного розподілу текстів текст 1 не призводять

Рисунок 40 - Типи розгалужень дуг між активностями системи

У стандарті IDEF3 для деталізації вставлення використовують вузли:

1. логічні «і»; 


2. логічні включаючи «або»; 


3. логічні які виключають «або». 

У IDEF3 використовуються довідкові об'єкти  для посилань на інші діаграми, для вказівки циклічних переходів.

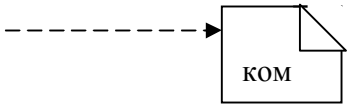



Рисунок 41 - Пов'язує блок з коментарями

При використанні SADT технології варіанти діаграм розробляються кілька разів, щоб вибрати кращий для практичної реалізації системи. Технологію SADT доцільно використовувати на ранніх етапах життєвого циклу для розуміння сутності системи.

Переваги SADT - поєднання графа (діаграми) ("активність - предмет"), відображення управління, зворотного зв'язку, виконавців і одночасний показ домінування.

Приклади застосування SADT технології

SADT – диаграмма деятельности компании, осуществляющей распределения товаров по заказам, приведена на рисунке 5.6. Пример применения SADT технологи для разработки функциональной диаграммы работы гидравлического пресса, приведен на рисунке 5.7.

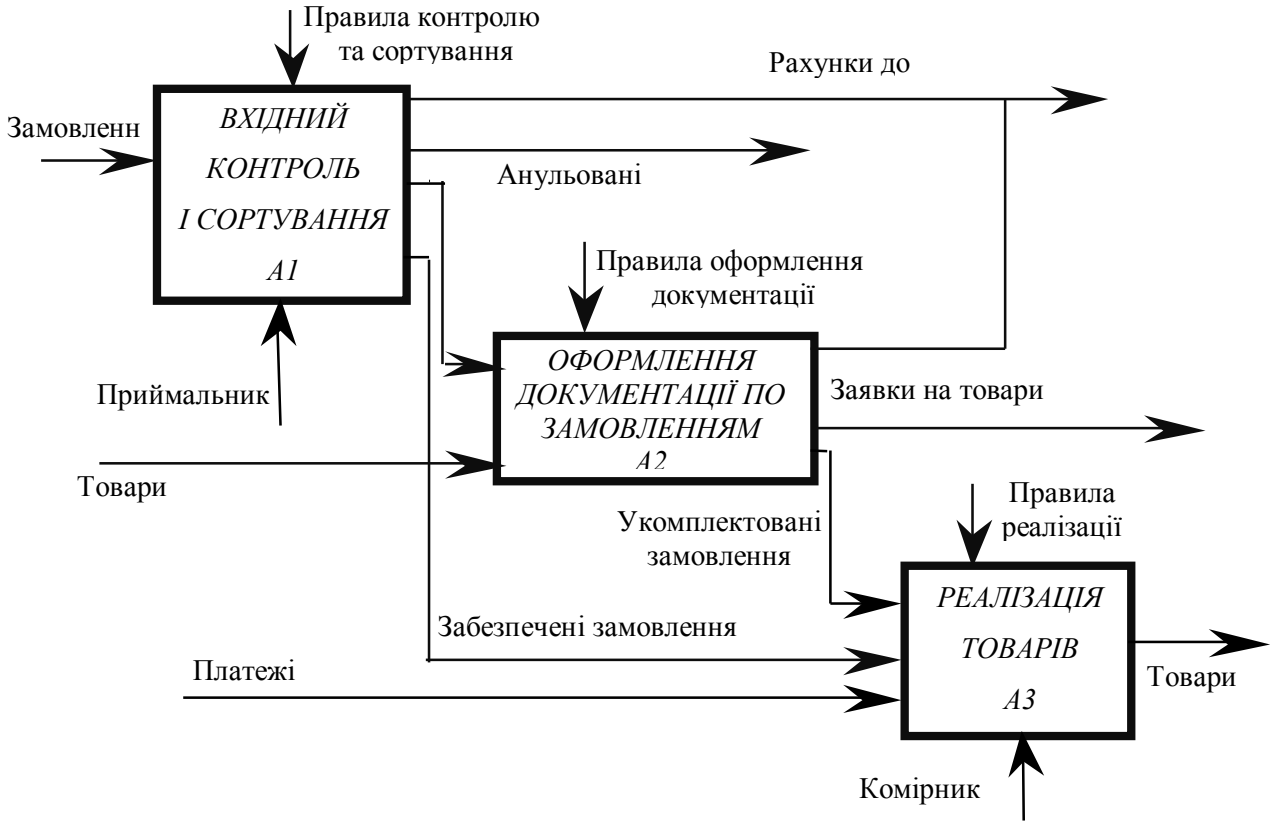



Рисунок 42 – Діяльність компанії розподілу товарів на замовлення (SADT - діаграма)

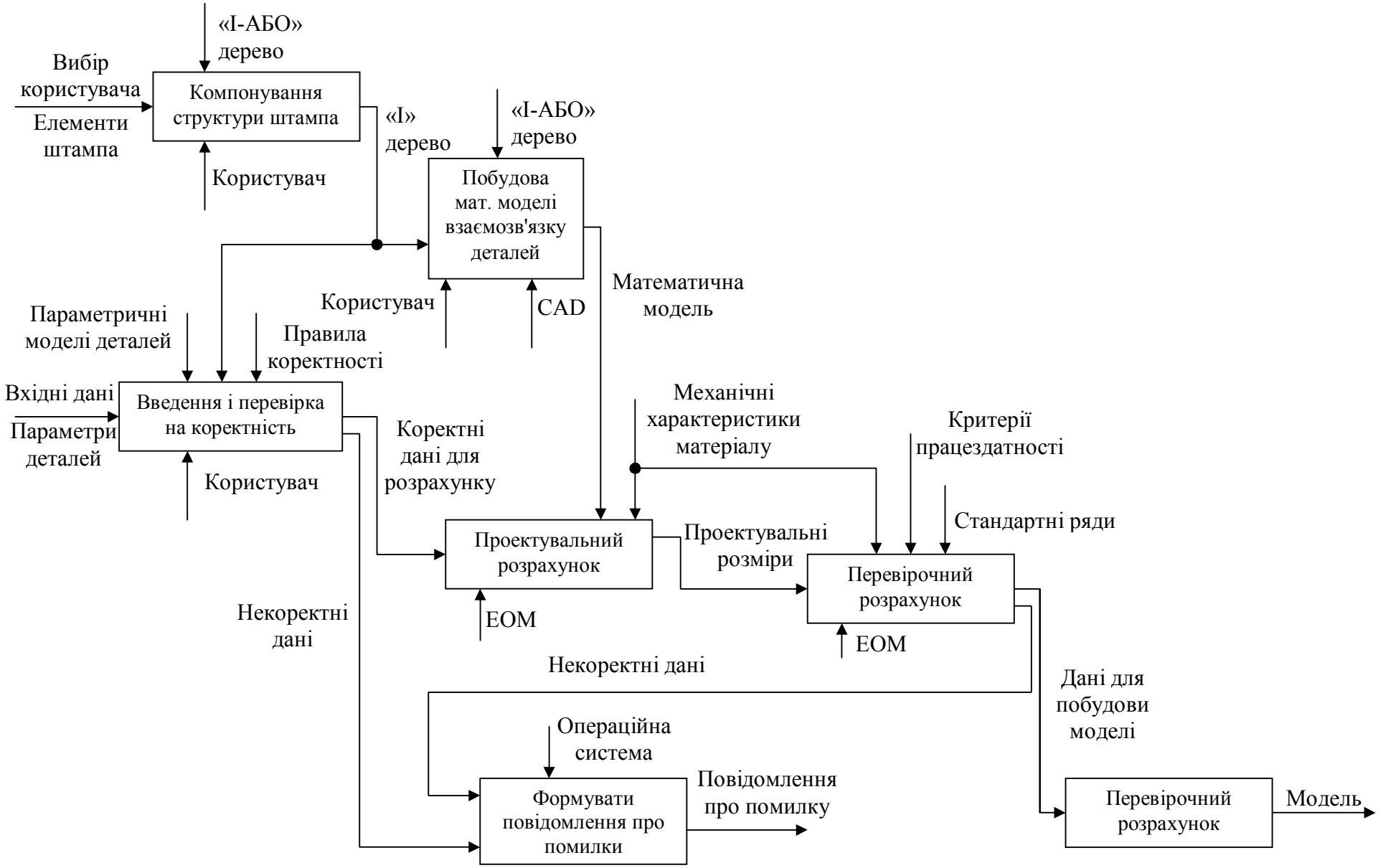



Рисунок 43 – Функциональная SADT диаграмма работы гидравлического пресса

## Засоби структурного проектування

Ми розглянемо засоби структурного системного аналізу, які дозволяють побудувати модель вимог (логічну модель) системи.

## Засоби проектування фізичної моделі системи

Розглянемо засоби структурного системного аналізу, які дозволяють побудувати модель вимог (логічну модель) системи при її проектуванні. Проектування - фаза ЖЦ, на якій виробляється реалізація вимог користувача. Жорсткого кордону між етапами проектування немає. Для візуалізації розбиття програми на модулі і їх взаємозв'язку зазвичай використовують *структурні карти*:

- Константайна - опис відносин між модулями програми ;
- Джексона - опис внутрішньої структури модулів.

## Структурні карти Константайна

Структурні карти Константайна показують відносини ієрархії між модулями (рисунок 7.1).

Розрізняють такі основні види зв'язків між блоками:

а) послідовний зв'язок - тільки послідовно;
б) паралельний зв'язок - одночасний виклик;
в) виклик співпрограми;
г) ітераційний зв'язок - виконання блоку в циклі;
д) умовний зв'язок - вибір альтернатив;
е) выполнение происходит только 1 раз;
ж) звернення до модуля;
з) звернення до деякого елементу модуля.

Структурні карти Константайна показують відносини ієрархії між модулями.

| | | | |
|---|---|---|---|
| Позначення стандарту | *Вузли* | - | модулі або області даних, |
| IBM, ISO, ANSI | *Потоки* | - | міжмодульні виклики зв'язку: |

| | | | | |
|---|---|---|---|---|
| потік-виклик модуля | → | - | за даними | ●→ |
| | | - | з управління | ○→ |

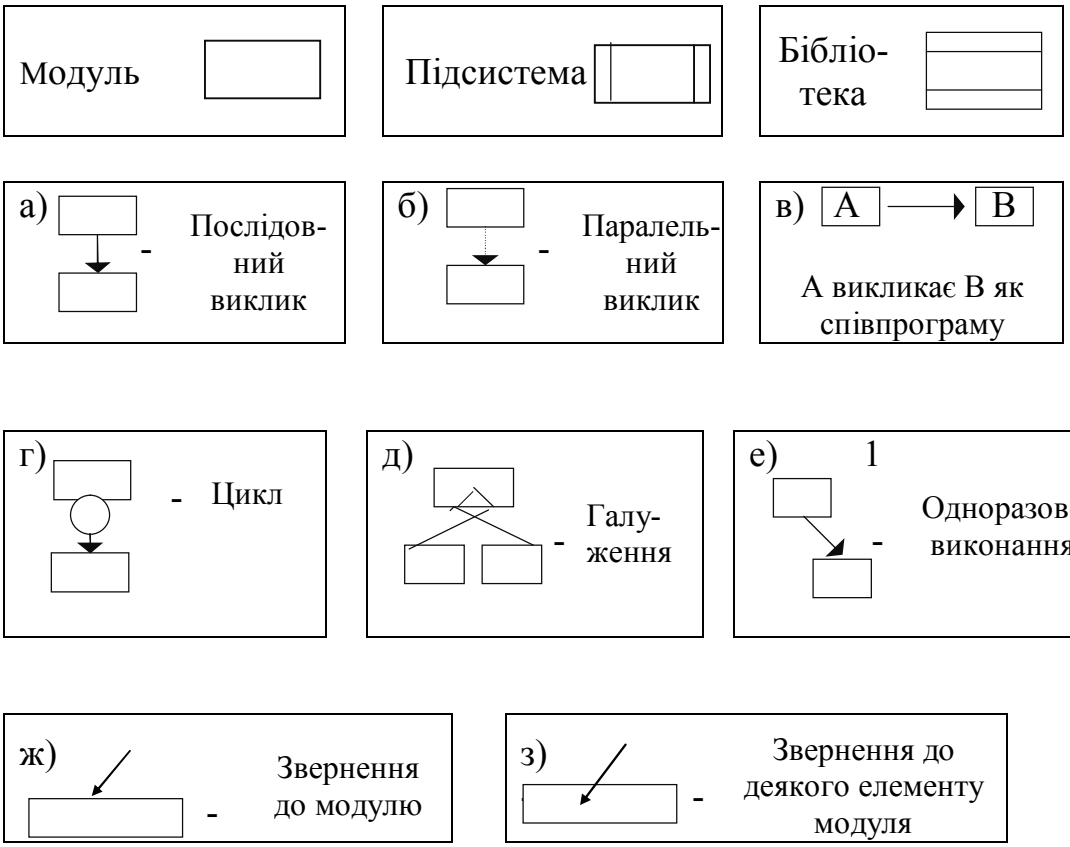



Рисунок 44 – Види умовних позначень на структурних картах

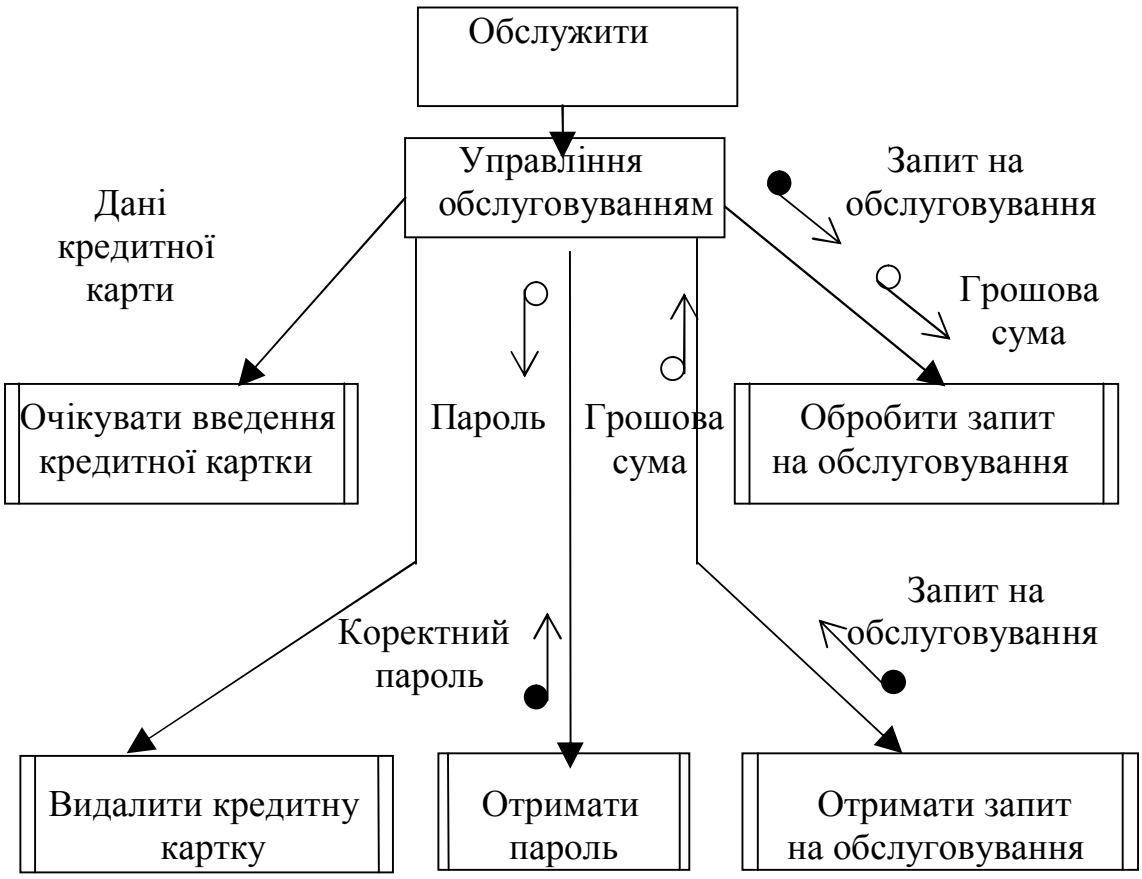



Рисунок 45 – Приклад структурної карти Константайна для фрагмента банківської системи

Структурні карти Джексона

Застосування структурних карт Джексона засноване на методології структурного програмування. Вони використовуються для ілюстрації внутрьомодульних (іноді міжмодульних) зв'язків і документування проекту архітектури системи ПО. Придатне для нижнього рівня проектування (близькі до блок-схем).

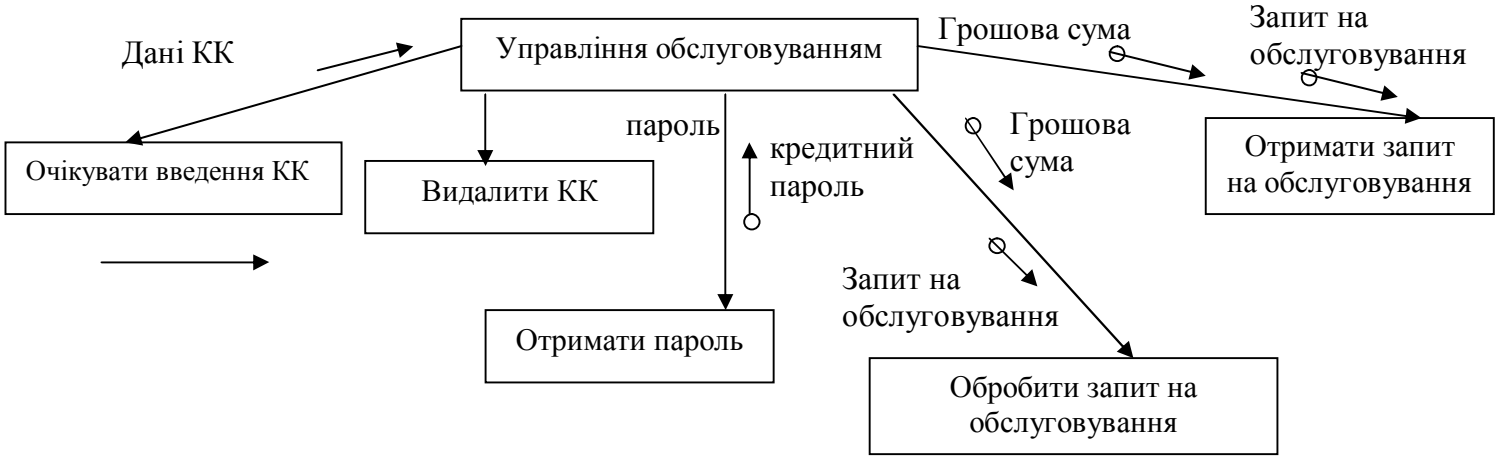



Рисунок 46 – Приклад структурної карти Джексона для фрагмента банківської системи

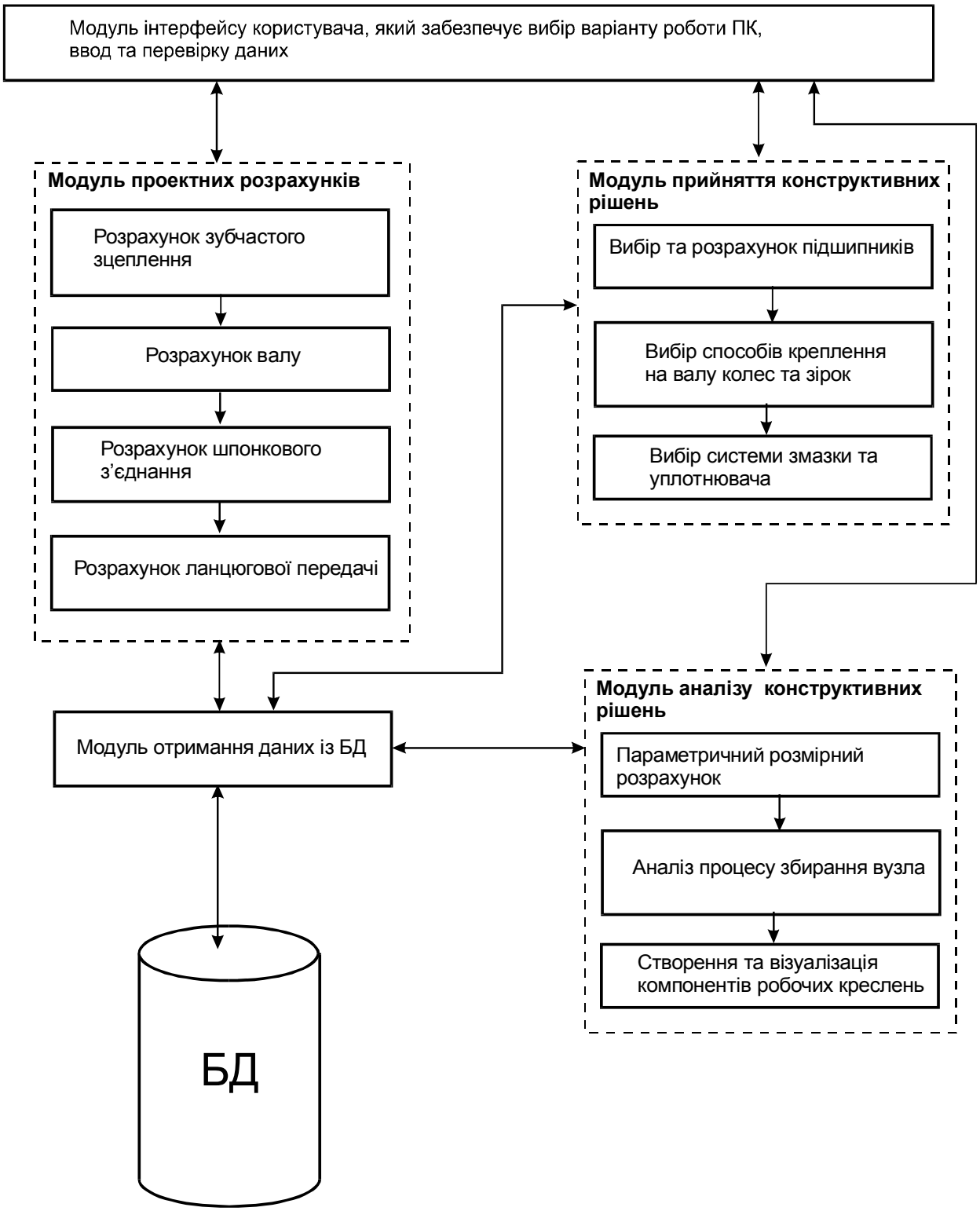



Рисунок 47 – Приклад структурної карти Джексона для роботи з базою даних

Операційні процеси програмног о комплексу (ПК)

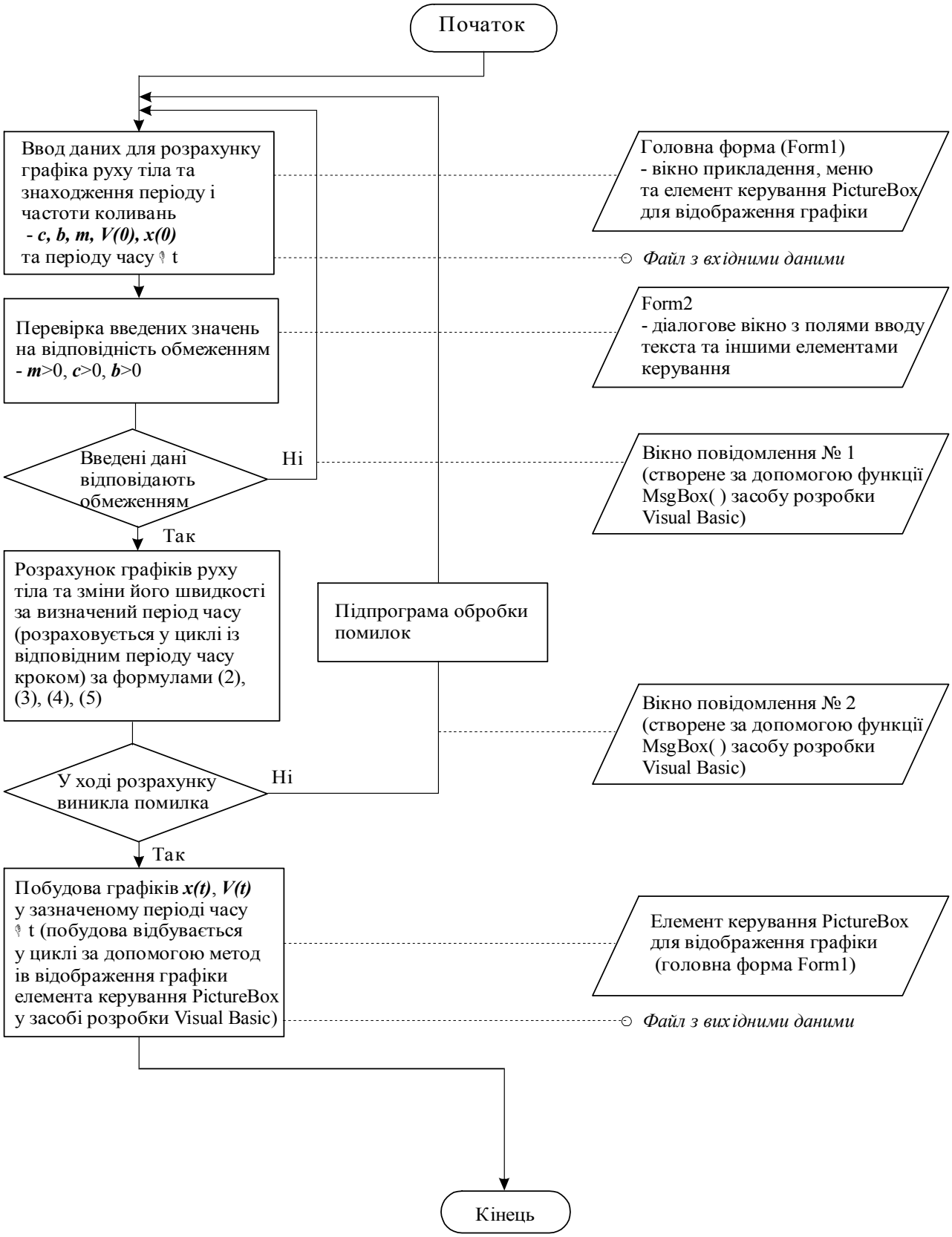



Рисунок 48 – Ілюстрація внутрьомодульних зв'язків


Рисунок 49 - Приклад структурної карти Джексона

Візуальна мова програмування специфікацій

ВМПС дозволяє структурувати опис ходу, робити його більш наочним і автоматично генерувати код і текстові документи.

Flow – форма

послідовне виконання блоків

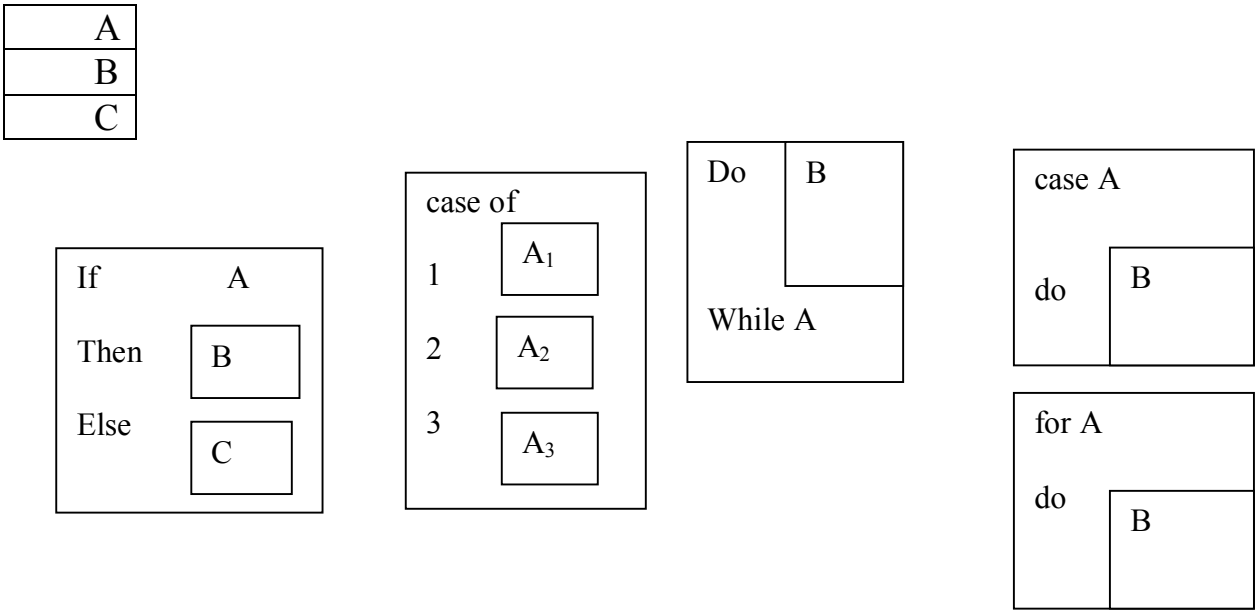



Рисунок 50 – Візуальна мова програмування специфікацій

Коментарі містяться у верхній частині блоку у вигляді звичайного тексту.
Приклад.

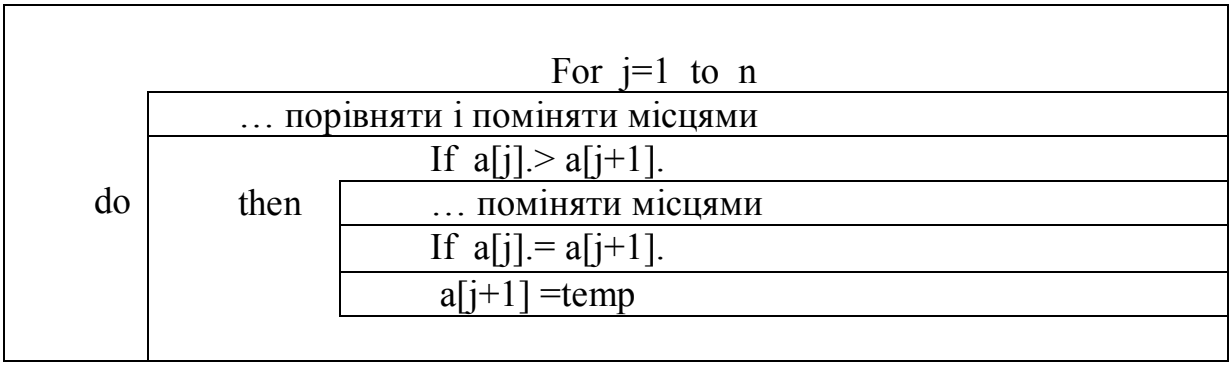



Рисунок 51 – Приклад порівнянь

Кожен блок всередині іншого показує свій цикл обробки, розгалуження, тобто демонструє вкладеність.

*Тестування* - це етап життєвого циклу програмного виробу, спрямований на поліпшення якісних характеристик цього виробу.

При каскадній моделі на тестування йде до 40% часу.

Сучасні технології проектування припускають паралельну або попереджувальну розробку тестів.

## Складнощі тестування

1. Відсутність еталона, якому повинна відповідати програма.
2. Складність програмного забезпечення та відсутність єдиної методики тестування.
3. Велика кількість варіантів станів системи, тобто відсутність можливості повного тестування.

Тестування - це процес багаторазового повторення програми, наприклад, з різними даними і логічними умовами, з метою виявлення помилок.

Таким чином тестування - деструктивний процес.

## Принципи тестування

1. Тестування більш ефективною не розробником програми, так як оцінюється за кількістю помилок. Тестування пов'язано з налагодженням, тобто виправленням помилок.
2. Необхідно робити опис можливих значень результату. Тобто тестові набори повинні включати вихідні дані і результат.
3. Результати кожного тесту ретельно вивчають. Як правило більшість помилок знаходяться на початку тестування.
4. Тести повинні передбачатися для виправлених і непередбачених даних. Причому тести з даними поза ОДЕ володіють великою здатністю до виявлення.

5. Слід перевіряти не тільки те, що повинна робити система, але і те, що вона робить, але не повинна робити.

6. Імовірність наявності невиявлених помилок в частині програми пропорційно кількості вже виявлених помилок в цій частині.

## Методи тестування програм

Тестування є частиною процесу налагодження.

Мета налагодження - локалізація помилок і виправлення їх.

Налагодження - процес отримання ПІ, яке функціонує з заданими характерами надійності в заданій області зміни вхідних даних.

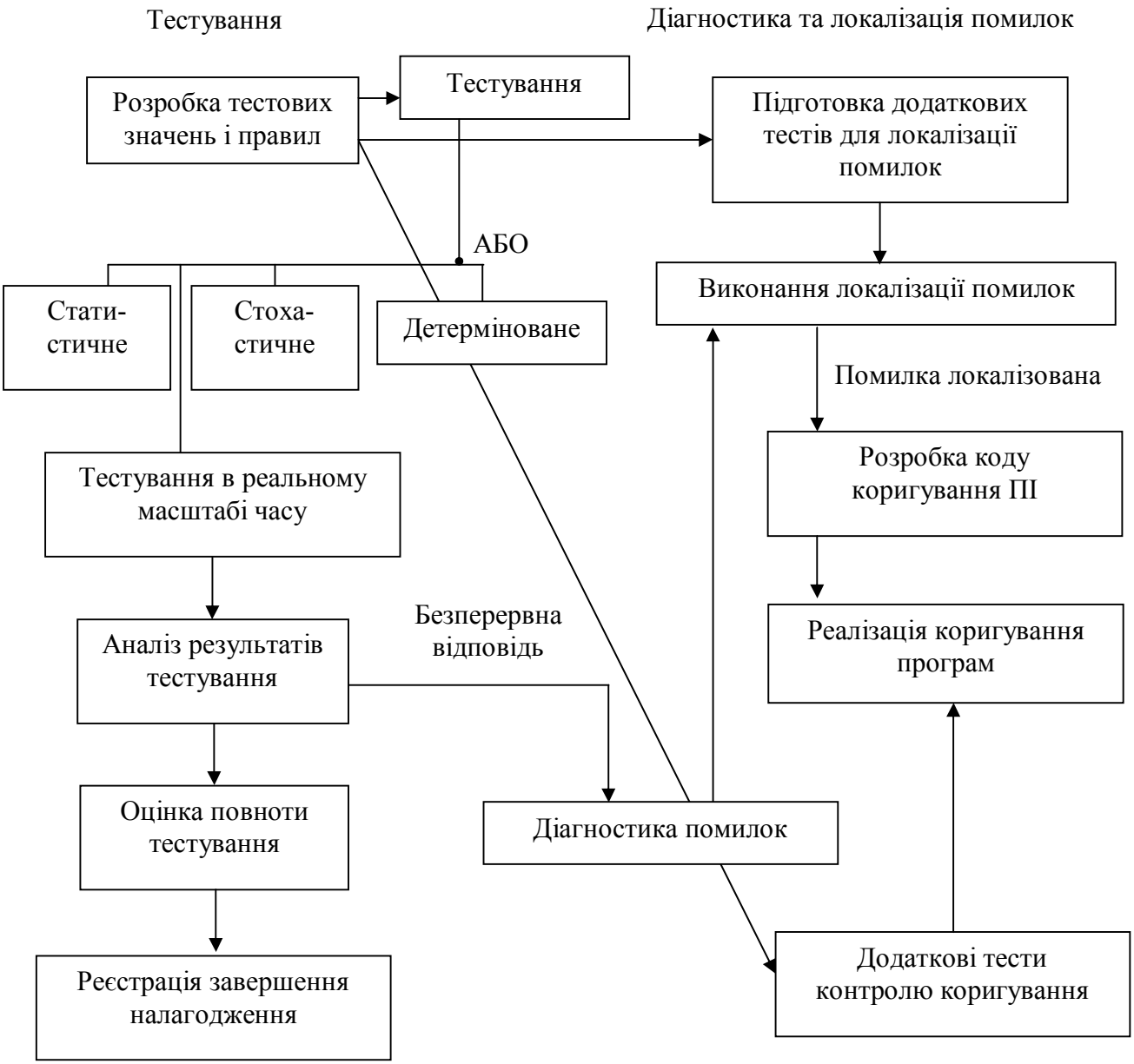



Рисунок 52 – Схема налагодження вхідних даних

Розробка тестів для тестування програмних систем здійснюється постійно.

*Статичне тестування* - це формальний аналіз тексту програми на мові програмування. Це символічне тестування.

*Детерміноване тестування* – вимагає багаторазового виконання програми на ЕОМ з використанням спеціального набору даних. Контролюється кожна комбінація даних, результати розрахунків для перевірки кожного затвердження з специфікації програми.

Детерміноване тестування трудомістке, тому застосовується для окремих модулів, блоків програми.

*Статистичне тестування (ймовірнісне).* Перебрати всі можливі сукупності вхідних величин іноді неможливо. Тому використовується безліч спеціальних величин з відповідними законами розподілу. Розглядаються закони розподілу вхідних величин.

## Метод Монте-Карло

Після виявлення помилки, як правило застосовують детерміноване тестування. Цей вид найбільш зручний для автоматизації.

## Тестування в реальному масштабі часу

Для систем реального часу перевіряються результати обробки вихідних даних з урахуванням часу їх надходження, динаміки використання пам'яті, тривалості та пріоритетності в обробці.

При виявленні помилки - перехід до детермінованого тестування.

Статистичне тестування проводять з використанням ручних методів. Виявляється 3-40% помилок. Результати розглядають на нараді. Мета - виявлення помилок без виправлення.

Процедури статистичного тестування:

1. Інспекція вихідного тексту (4 чол). Текст вивчається заздалегідь, а також специфікація.
2. Виступає автор. Йому ставлять запитання. Наскрізні перегляди. Проглядається текст програми поспіль.
3. Динамічні перегляди.
4. Текст вивчається відповідно до переходів.

## Методи проектування тестових наборів даних

Найбільш ефективним є детерміноване тестування, при якому відомо і контролюється кожна комбінація вихідних даних і відповідні їй результати виконання програми.

Для чисельних методів розв'язання, як правило, вибираються тестові завдання, які мають аналітичне рішення і порівнюються результати чисельного та аналітичного рішення.

Наприклад, для МКЕ в якості тестової використовують завдання напруги в циліндрі під дією внутрішніх сил.

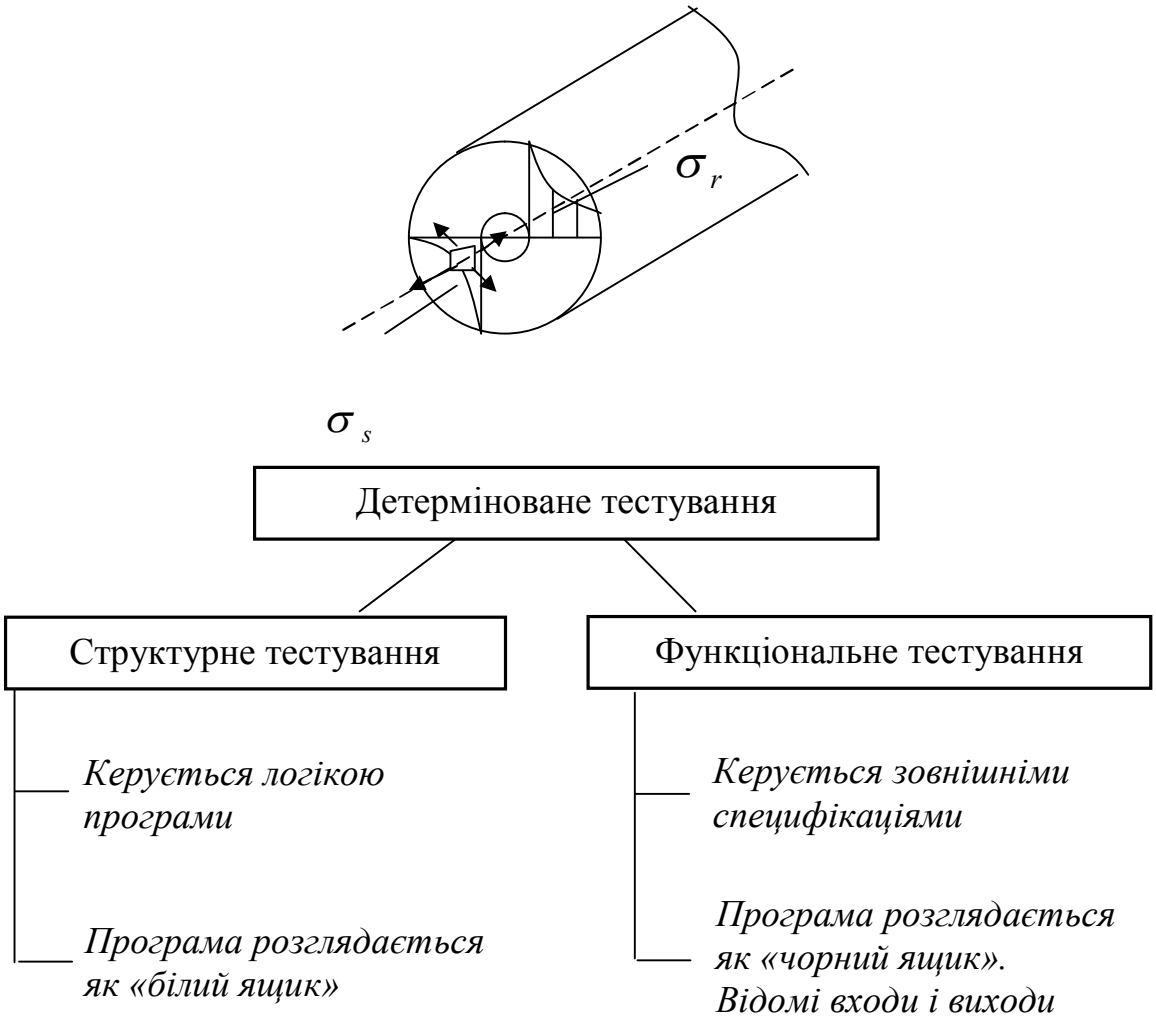


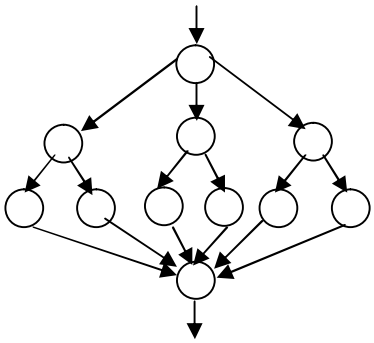

Рисунок 53 – Детерміноване тестування

Структурне тестування здійснюється на основі вивчення прогонки програми, тобто багаторазового повторення програми з різними наборами даних, які забезпечують максимальне число маршрутів її виконання.

Функціональне тестування програми здійснюється за типом вхід-вихід: перевіряються специфікації програми.

Ефективний тестовий набір даних

Поняття введено, оскільки повний перебір неможливий. Дана ділянка доводиться в циклі: $n \leq 20$.



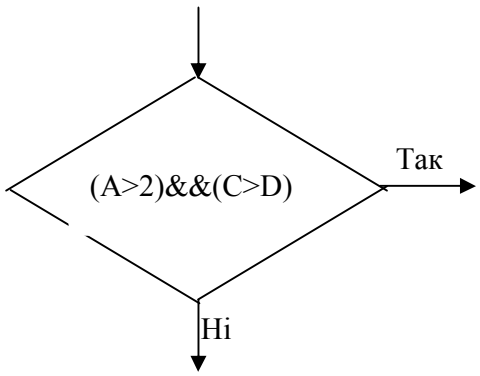
$N_5$=20*6 =120 – загальна кількість прогонів

Рисунок 54 – Ефективний тестовий набір даних

Повне тестування є надмірним. Ефективний набір - підмножина всіх можливих тестів, які мають максимальну ймовірність виявлення помилок.

Види структурного тестування

1. Покриття операторів. Тестовий набір гарантує виконання всіх операторів програми не менше 1 разу. Критерій слабкий.
2. Покриття вузлів розгалуження. Це тестові набори, що забезпечують перехід по гілках «істина» і «брехня» хоча б 1 раз. У кожному вузлі розгалуження. При цьому зазвичай виконується п.1
3. Покриття умов. Якщо вузол розгалуження містить кілька умов (> 1) то кожна умова має бути виконано хоча б раз.
4. Комбінаторне покриття умов. всі комбінації.




Так: A>2 C=D | A=3, C=0, D=0
Ні: A≤2 C≠D | A=2, C=0, D=1

Рисунок 55 – Види структурного тестування

Методи функціонального тестування

1. *Метод еквівалентного роздроблення.* Проводять в два етапи:
   - виділення класів еквівалентності;
   - побудова тестів.

*Клас еквівалентності* (КЕ) – Множина вхідних значень, що володіють однаковою ймовірністю виявлення помилок конкретного типу.

Класи еквівалентності виділяються шляхом аналізу. Всі вхідні значення ділять на дві або більше групи. Одна група з них правильна.

Таблиця 13 - Вхідні умови і класи еквівалентності

| | Вхідні умови | Правильні КЕ | Неправильні КЕ |
|---|---|---|---|
| 1 | Ідентифікатор (I) містить не більше 8 символів | 1 символу ≤ I ≤ 8 символів | а - I містить 0 символів<br>б - I > 8 символів<br>в - I - не буква<br>с - I починається з цифри |
| 2 | Спосіб передачі інформації (набір значень)<br>Пошта - 1<br>Телеграф – 2<br>e-mail - 3 | Пошта - 1<br>Телеграф – 2<br>e- mail - 3 | а - SMS – 5<br>б - Авіапошта - 1 |
| 3 | Перший символ - буква:<br>A, A1, 1A, 2A | Перший символ - буква:<br>A, B, a, в | а - перший символ цифра<br>б - перший символ підкреслення<br>в - перший символ не буква<br>&, L,\, \|, / |

1. Якщо вхідні умови описують ОДЗ від і до, тобто інтервал, визначається 1, правильний КЕ всередині ОДЗ і два неправильних КЕ (більше і менше ОДЗ).
2. Крім того можуть контролюватися границі ОДЗ.
3. Якщо описується ситуація типу «має бути», то виділяється один правильний КЕ і неправильні КЕ (≥ 1).

Крім того виконується аналіз граничних умов. Наприклад, якщо ОДЗ: від -1 до -1.

Розглядаються варіанти:

- 1,0  1,0…

-1,001  1,001

*Метод функціональних діаграм* (причинно-наслідкові зв'язки)

Вхідна специфікація програми перетвориться на основі розбору булевских відносин проводиться побудова таблиці рішень на основі якої будуються набори тестових даних.

## 3.7 Загальні принципи подання інформації про системи

Загальні принципи подання інформації про системи відомі: блочність, ієрархічність, структурність, виділення аспектів розгляду, стадій проектування і т. д. [1]. Розробка нової системи, а також засобів автоматизованого проектування засновані на ряді загальних підходів до створення *метасістеми*, яка описує (представляє) її склад і функціонування. Таким чином, для створення і вивчення систем потрібно надання інформації про цілі проектування (граф цілей), функціях системи і послідовності їх виконання (функціональний граф-схема), типові варіанти структури систем з конструктивними варіантами виконання її елементів і системи в цілому (дерево «І-АБО»). Структурні елементи (вузли, підвузли) пов'язані з реалізацією функцій, є більш загальними поняттями в порівнянні з конструктивними і не містять інформації про фізичні принципи роботи. Конструктивні елементи: містять фізичні та геометричні характеристики об'єктів, що взаємодіють між собою і виконують функції системи.

У загальному випадку система представляється графом G, що моделює її структуру. Граф - сукупність безлічі (непорожньої, кінцевого) елементів - вершин V (G) і сімейство (кінцеве) невпорядкованих пар елементів, які називаються ребрами. Математичний апарат - теорія графів дозволяє оцінити і досліджувати розроблену систему. В якості загальних функцій, які можна застосувати до подання системи у вигляді графа можна виділити: побудова матриць суміжності і інцидентності, укладання графів (виняток перетинів ребер на площині), прив'язку вузлів до сітки, порівняння графів, виділення подграфов і дерев на графі, контроль зв'язності і т . д. [2]. Велике значення має окремий випадок графів - дерева, які широко використовуються для різних завдань подання інформації в ЕОМ, наприклад, у випадках класифікацій об'єктів в різних предметних областях. Прикладом уявлення і дослідження систем є математичне моделювання, засноване на методі еквівалентних електричних схем.

Для достатнього узагальнення завдань при побудові програмного продукту необхідно виділити загальні функції, необхідні для роботи незалежно від прикладного завдання. Це дозволить розробити структуру та ієрархію необхідних класів для програмної реалізації. Наприклад, зображення елементів, зазвичай використовуваних для створення діаграм, мають вигляд кола, прямокутника, ромба і ін.. У загальному випадку з кожним вузлом або ребром зв'язується піктограма, отримана за допомогою відомих редакторів ресурсів. Для кожного блоку можна закріпити за сторонами інформацію різного типу, як це робиться при побудові SADT - діаграм [3]. Існуючі програмні засоби в певній мірі реалізують наведені функції в конкретних прикладних областях. До таких систем можна віднести ER-WIN (створення ER-діаграм), BP-WIN (створення DFD-діаграм потоків даних), засоби управління проектом в CAD системах. З програмних продуктів, розроблених в Росії, можна відзначити Stratum 2000 (моделювання систем). Перераховані вище функції є в ряді випадків допоміжними, що забезпечують сервіс робіт з основною системою. Для ЕОМ такий сервіс для роботи з файлами забезпечують, наприклад, FAR, Провідник Windows.

# Приклад розробки програмної системи для моделювання динамічних режимів роботи гідропресового устаткування

А.Ф. Тарасов, М.А. Вінніков (ДДМА, м Краматорськ)

## Сценарій розвитку предметної області

Проблема вивчення механічних коливань актуальна для багатьох галузей сучасної техніки. В одних випадках коливальні явища небезпечні і здатні завдати значної шкоди, в інших - цілеспрямовано застосовуються в техніці. Традиційний підхід до вивчення коливань заснований на аналізі рішення системи диференціальних рівнянь, що описують поведінку досліджуваного об'єкта. Такий підхід досить трудомісткий і застосовується лише до простих систем з невеликим числом елементів [1]. Існуюче програмне забезпечення (ПЗ) для універсального моделювання динамічних систем вельми «важкувате» і дороге. Тому існує потреба в розробці інструментального засобу, що дозволяє швидко створювати моделі коливальних систем різного ступеня складності і проводити з ними чисельні експерименти. Зокрема, такий інструментальний засіб необхідний для моделювання динамічних режимів роботи гідропресового устаткування, що виникають при швидкій зміні тиску в гідросистемі, наприклад, в процесі виконання розділових операцій (вирубка, пробивка).

В даний час необхідна ефективність вирішення завдань моделювання технічних об'єктів може бути забезпечена тільки системним підходом. Структура предметної області може бути представлена у вигляді ключових понять на трьох рівнях абстракції: *метасистема* (гідропресового устаткування), *системи* (механічна і гідравлічна), *елементи* (станина, гідроциліндр і ін.) [2].

На початковому етапі для розробки способу представлення системи, стосовно гідропресового устаткування, розглянемо узагальнену структуру преса, для якого складемо наступну функціональну діаграму (рисунок 1). Виконавчий механізм перетворює накопичену в приводі енергію в технологічне зусилля на заготовку (А1). При ході рухливих частин (А2) до моменту торкання заготівлі і в процесі деформування (А3) в приводі відбуваються різні види перетворення енергії. При цьому сили замикаються в рамі (А4). Застосуємо для опису бізнес-процесу роботи гідравлічного преса SADT технологію.

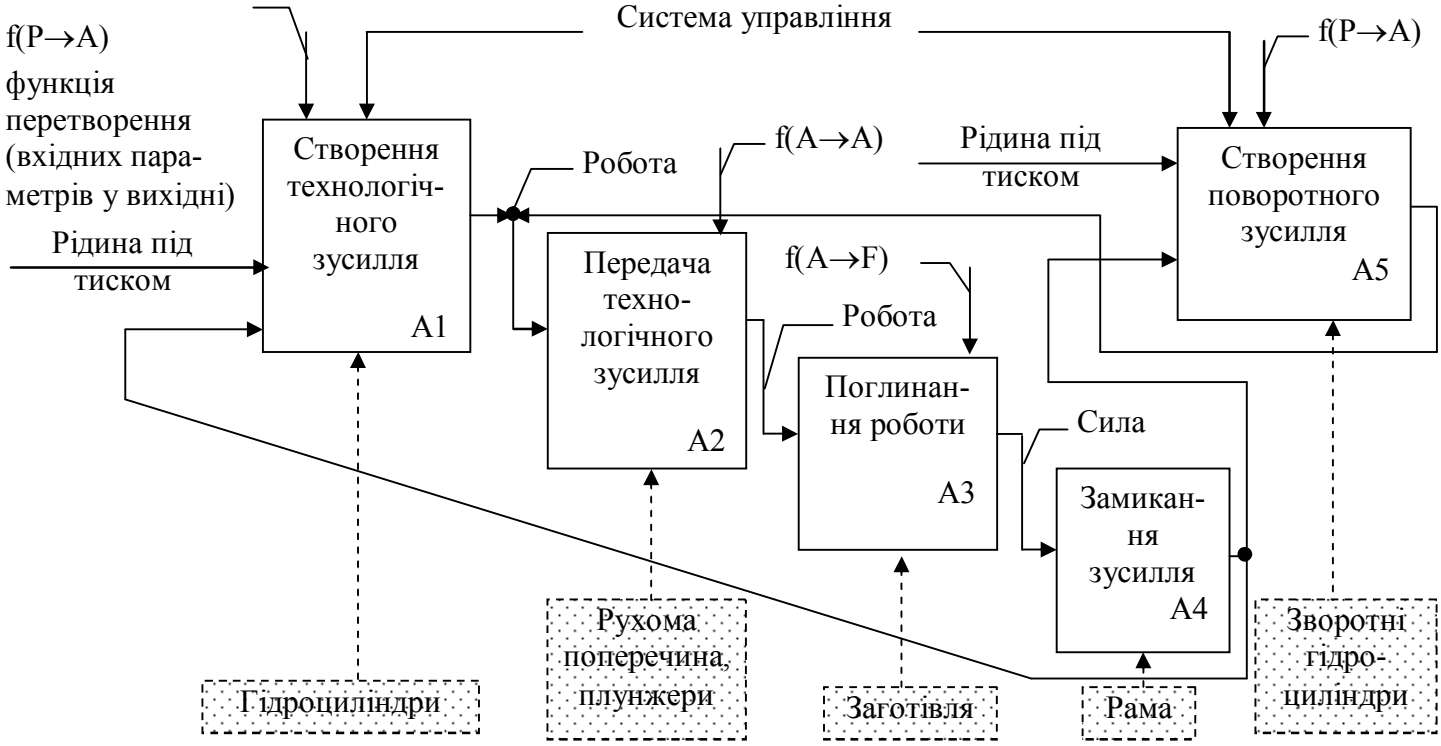



Рисунок 56 – Функціональна схема (опис бізнес-процесу) роботи гідравлічного преса

Використання функціональних діаграм дозволяє відобразити зв'язку структурних складових, управління, показати виконавців (конструктивні рішення) відповідно до послідовності дій. Функціональний підхід при поданні системи уможливлює її аналіз абстрагуючись від конкретного конструктивного виконання [3].

Виділимо елементи об'єкта моделювання для подальшої розробки класів програмної системи.

До складу гідропресового устаткування входить обмежена кількість обов'язкових конструктивних складових. Вони можуть бути представлені у вигляді елементів, які моделюють деталі гидромеханической системи преса і досить точно описують її поведінку при роботі. Функціональний аналіз показав, що незалежно від конструктивного виконання будь-яку коливальну систему можна представити як сукупність кінематичних (клас «маси») і топологічних елементів (клас «зв'язку»). Основна функція елементів першого класу - акумулювати кінетичну енергію, другого - накопичувати потенційну енергію. Основними атрибутами класу «маса» є: інерційність, жорсткість (абсолютна), геометричні розміри; класу «зв'язок» - пружність, невагомість і геометричні параметри.

Змістом наступного етапу була розробка класифікації елементів класів «маса» і «зв'язок». В основу класифікаційних ознак елементів типу «маса» були покладені: тип розподілу маси за обсягом і здатність переміщатися (рисунок 57).

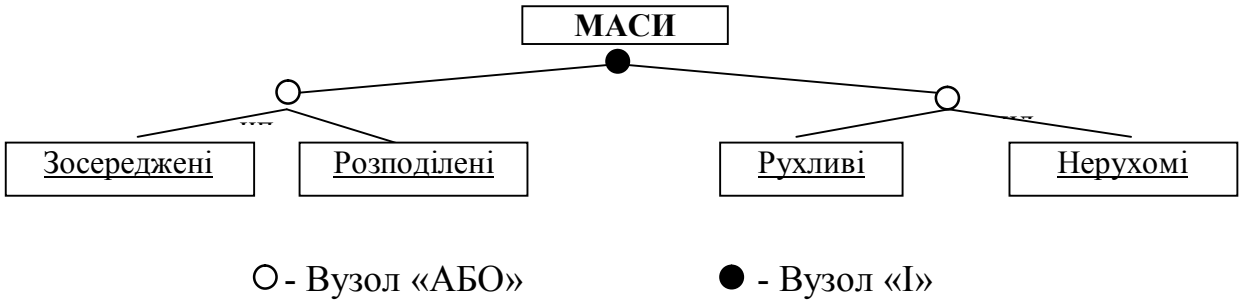



Рисунок 57 – «І-АБО» дерево елементів типу «маса»

При класифікації елементів типу «зв'язок» розглядали вид (лінійність) залежно переданої сили і вплив зв'язку на енергію системи (рисунок 58).

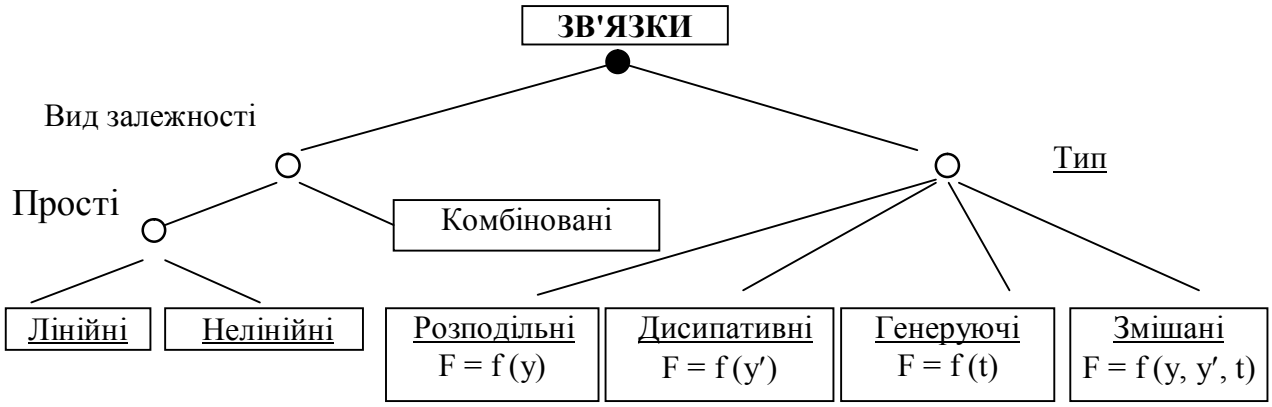



Рисунок 58 –« І-АБО »дерево елементів типу «зв'язок»

Прикладом комбінованих зв'язків можуть служити елементи з зазором і нарізні сполучення, що мають різну жорсткість на розтягування і стиснення, а змішаних - послідовне з'єднання пружного елемента і в'язкого тертя (упруговязкіх з'єднання). В'язкопружну з'єднання (паралельне з'єднання пружного елемента і в'язкого тертя) по даній класифікації являє собою два зв'язку - що розподіляє (без втрат енергії) і дисипативний. Поведінка кінематичних елементів об'єкта моделювання описується класичною системою диференціальних рівнянь виду:

$$m_i \ddot{y}_i = \sum_{j=0}^{N_i} F_j , \qquad (1)$$

де вид $F_j$ див. рисунок 3, причому для кожного дискретного моменту часу права частина рівняння визначається тільки діючими на масу зв'язками.

Сучасний рівень розвитку комп'ютерної техніки і технології дозволяє комплексно вирішити проблему моделювання динамічних систем на основі об'єктно-орієнтованого методу проектування (ООП) програмних продуктів [4]. Із застосуванням ООП можлива розробка програмного забезпечення для систем практично будь-якого ступеня складності. Найбільш важливою є проблема коректної постановки задачі на розробку ПЗ і визначення вимог, що пред'явля-

ються до об'єктно-орієнтованої бібліотеці (ООБ) як інструментального засобу моделювання. Виділимо наступні загальні вимоги до ООБ та ПЗ в цілому:

– адекватність - має на увазі необхідну і достатню повноту опису досліджуваного об'єкта, достовірність даних про нього;

– продуктивність - задовольняє користувача швидкість виконання операцій;

– гнучкість - можливість як горизонтального, так і вертикального розвитку структури бібліотеки класів, її модифікованості.

На ці вимоги накладаються обмеження по часу і вартості ПЗ, ресурсоспоживання, а також обмеження, що залежать від інструментальних засобів розробки.

Основними завданнями при розробці ООБ для моделювання гідропресового устаткування як складної коливальної системи є наступні [5]:

– об'єктно-орієнтована декомпозиція предметної області і виділення класів;

– побудова ієрархії класів;

– розробка внутрішньої будови класів.

Ці завдання в даний час не мають закінченого формального рішення - до них застосовні тільки евристичні методи, що враховують специфіку кожного конкретного випадку і засновані на досвіді, інтуїції розробників [6].

На підставі цих принципів була розроблена ООБ для моделювання одновимірних коливальних систем. В якості базових були виділені два класи: абстрактна маса і абстрактний зв'язок, які не мають певного конструктивного уявлення, мають тільки атрибутивними ознаками і найбільш загальними методами обробки даних. Подальша ієрархія класів будувалася на їх основі шляхом успадкування. Ієрархічна діаграма розроблених до теперішнього часу компонент ООБ приведена на малюнку 4. Клас «Зосереджена маса» вже володіє «поведінкою» - чисельно вирішуючи заданим способом рівняння (1), він визначає власні кінематичні параметри (переміщення і швидкість) для даного моменту часу, на підставі яких приєднані зв'язку визначають свою «поведінку». Для уявлення одержуваної в ході моделювання інформації кожен клас реалізує можливість візуального представлення процесу, що відбувається коливань, будує графіки зміни своїх властивостей в часі (переміщення, швидкості, сили, що діє і т. д.).

Внутрішня будівля класів визначалася компромісом вимог видимості і захисту інформації. Проблема продуктивності функціонування ООБ вирішувалася шляхом підвищення ефективності виконання найбільш критичних з точки зору витрат часу операцій по ініціалізації - деініціалізацію об'єктів, передачі повідомлень безпосередньому оброблювачу.

Розроблений набір елементів дозволяє моделювати багато досить складних одномірних коливальних систем. Прикладом може служити модель пресножиць для різання прокату, скрапу і т. д. (рисунок 59), в яких джерелом виникнення динамічних коливань є різке падіння технологічного навантаження в момент поділу.

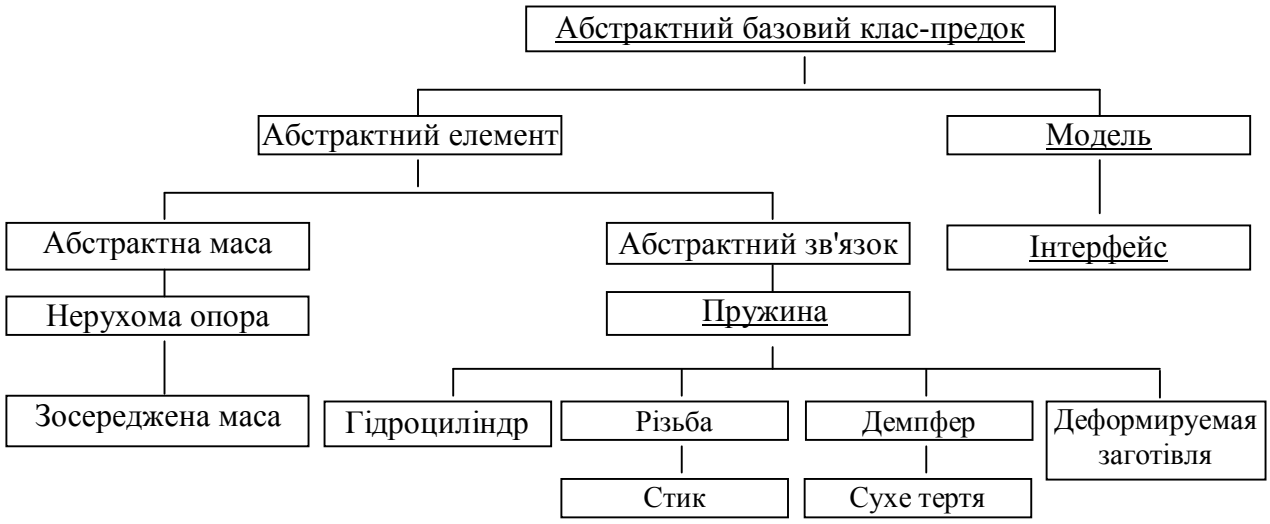



Рисунок 59 – Ієрархія класів ООБ для моделювання гідропресового устаткування

Даний підхід, не пов'язаний в реалізації з конкретною прикладною задачею, дозволяє (розробивши алгоритми обробки даних про систему, представлених у вигляді графів) застосувати проектовану програмну систему до множини важливих практичних застосувань. Такими додатками можуть бути завдання теорії розкладів (транспорт, комунікації, розподіл робіт, товарів), вивчення процесів, що описуються ланцюгами Маркова і т. д.

В даному ПК можна виділити наступні бібліотеки класів, що забезпечують створення, ефективний супровід і розвиток проектованих систем: класи графічного представлення інформації (на базі стандартної бібліотеки коштів розробки MFC, VCL), класи створення, зберігання і роботи з динамічними структурами даних, класи імпорту та експорту даних (бази даних, CAD системи, OLE між додатками); класи, що забезпечують задані функції обробки інформації про створені структури даних (отримують об'єкт класу Graf, List, Tree), а також зовнішні функції, що забезпечують алгоритми обробки даних відповідно до завдань предметної області та ін.

Узагальнення підходів до подання інформації вимагає використання сучасних засобів реалізації програмних продуктів на ЕОМ:

– об'єктно-орієнтованого підходу до проектування;

– використання динамічних структур (даних, об'єктів);

– реалізації можливості маніпулювання різними об'єктами (поліморфізм);

– можливість розвитку, конкретизації інформації про елементах системи на основі успадкування.

Ці можливості реалізовані в сучасних засобах розробки програмного забезпечення (Visual C++, Visual Basic, Delphi та ін.).

Можна виділити ряд загальних вимог до створюваних об'єктів: вони повинні відображати ключові (основні, базові) абстракції предметної області, які дозволяють характеризувати процеси, що відбуваються в системі; вони повинні

бути інтелектуальними, тобто містити логічні умови і принципи роботи, умови зміни станів і т. д. Вони повинні містити або бути пов'язаними з інформаційними базами знань (зокрема містити інформацію «за замовчуванням»); вони повинні мати якісний інтерфейс, достатній для робіт з об'єктом.

Важливим питанням є розподіл інформації між елементом і системою, в яку вона входить. Введемо поняття «оточення» деталі, вузла і т. д. - інформація про зовнішнє середовище, що не відноситься безпосередньо до деталі, але пов'язана з нею, що необхідно передавати від деталі (елемента) до вузла (системі) при її компонуванні. Для побудови системи не обов'язково знати особливості реалізації компонента, необхідний тільки інтерфейс, що забезпечує їх об'єднання. Тому для правильного функціонування деталі у вузлі вона повинна сформувати і передати інформацію про необхідні для її роботи умовах, наприклад, простір для зворотно-поступального руху, граничних навантаженнях, температурі і т. д.

В даному випадку система розглядається на двох рівнях абстракції: «деталь» - атомарний елемент і «збірка» - сукупність елементів «деталей», пов'язаних між собою. Ієрархія класів програмної системи приведена на рисунку 60.

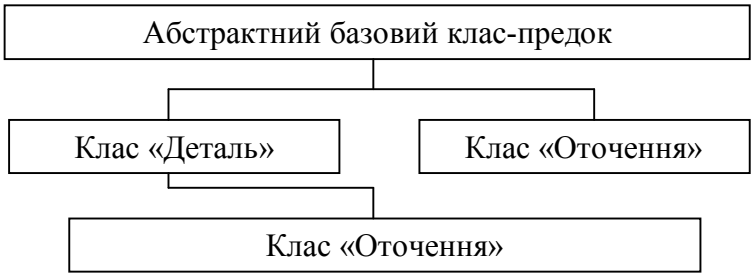



Рисунок 60 - Ієрархія класів системи

Абстрактний базовий клас-предок володіє найбільшим ступенем спільності. Реально він може, наприклад, реалізовувати механізм роботи з динамічною структурою параметрів на основі використання TList (додавання, видалення, зміна, збереження-відновлення та ін.). Причому, в якості параметрів може виступати практично будь-яка інформація - числові змінні, графічні зображення, логічні правила, посилання на бази даних і т.д. Шляхом наслідування отримуємо класи «Деталь», «Оточення» і «Збірка» (рисунки 2.7-2.8). Виділення класу «Оточення» дозволяє удосконалювати логіку застосування деталей, розглядати різні варіанти окремо від уявлення «Деталей». Логіка складання базується на інформації про деталі, яку вони надають про себе, тобто про «Оточення» деталей.

Структуру класу, що описує абстрактну деталь, можна представити таким чином.

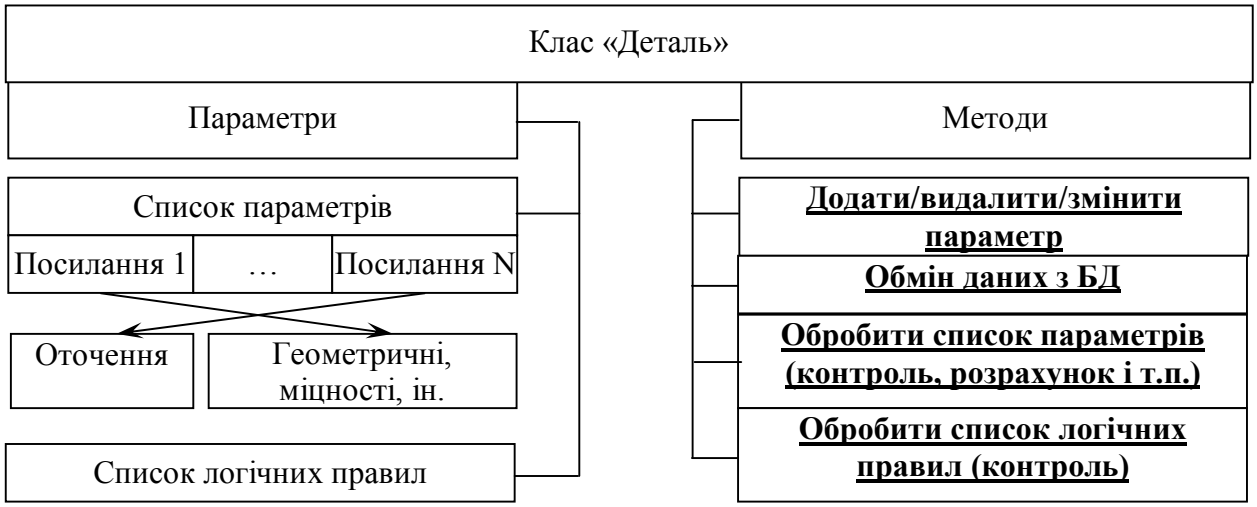



Рисунок 61 - Узагальнена структурна схема класу «Деталь»

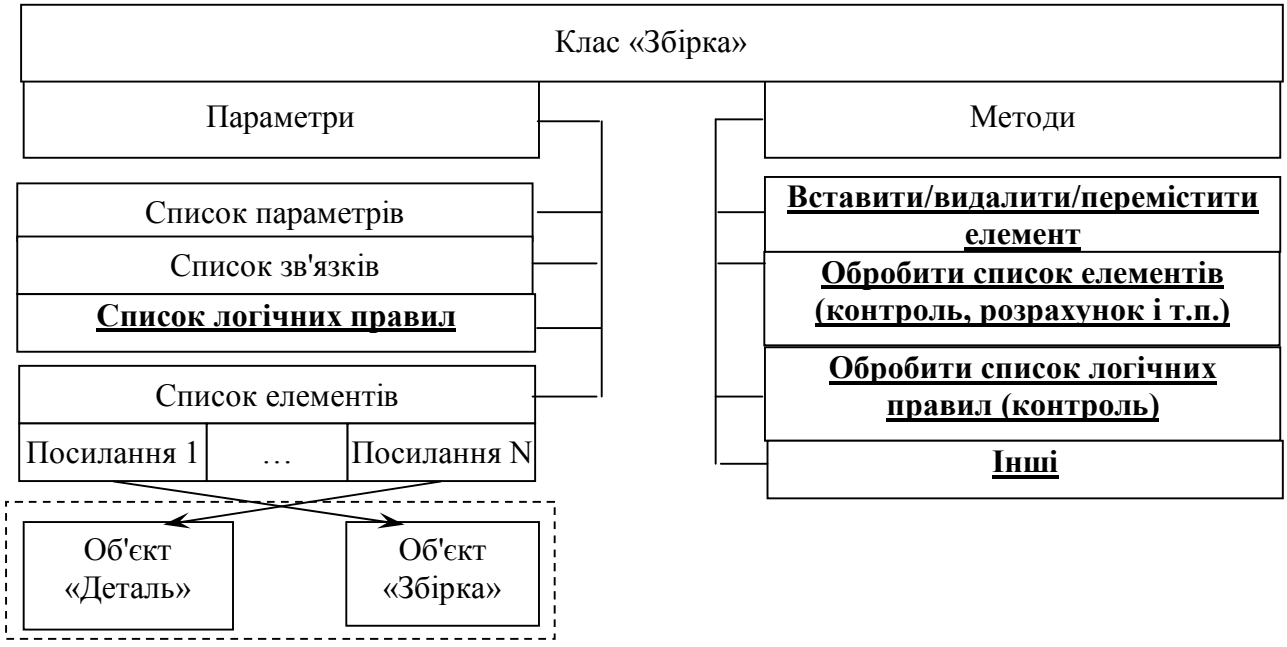



Рисунок 62 - Узагальнена структурна схема класу «Збірка»

Так як елементом складання може бути як деталь, так і збірка (підзбірка), то можна створювати ієрархічні структури необхідної глибини.

Розглянемо можливі варіанти (програмної) реалізації такої логічної структури. Аналіз показав, що в цьому випадку одне з головних вимог до структури системи - динамічність, тобто ситуативне її фо-рмірованіе. У найбільш поширеному в даний час сімействі операційних систем Windows реалізований механізм динамічних структур даних на основі ООП. Так, наприклад, клас TList містить динамічний список посилань на об'єкти і основні методи роботи з цим списком (додати посилання, видалити посилання, впорядкувати список, знайти об'єкт по посиланню і ін.). Використовуючи даний клас як базового, шляхом успадкування та/або включення можна створювати досить універсальні динамічні структури, що описують об'єкт, що проектується.

Створюваний програмний продукт (ПП), що дозволяє представити систему у вигляді графа, включає графічну оболонку, що виконує стандартні функції [ ], зокрема ПП містить редактор для побудови графа, що відображає структуру об'єкта, його функції, або подання іншої інформації про систему. Оболонка містить також ряд функцій, які обслуговують роботу програми, кошти для створення та оперування динамічними структурами даних, які включають текстову, числову, графічну й іншу інформацію, пов'язану з вузлом графа або зв'язком між вузлами.

Функції-члени класу можна розділити на обслуговуючі, що забезпечують інтерфейс для внутрішніх і зовнішніх зв'язків (наприклад, внутрішні ітератори і т.д.), інтеграції з іншими програмними продуктами, а також функції-члени, що забезпечує реалізацію ключових абстракцій предметної області (рисунок 63). Таким чином можна виділити системну частину і прикладну, що забезпечує роботу прикладних алгоритмів обробки інформації.

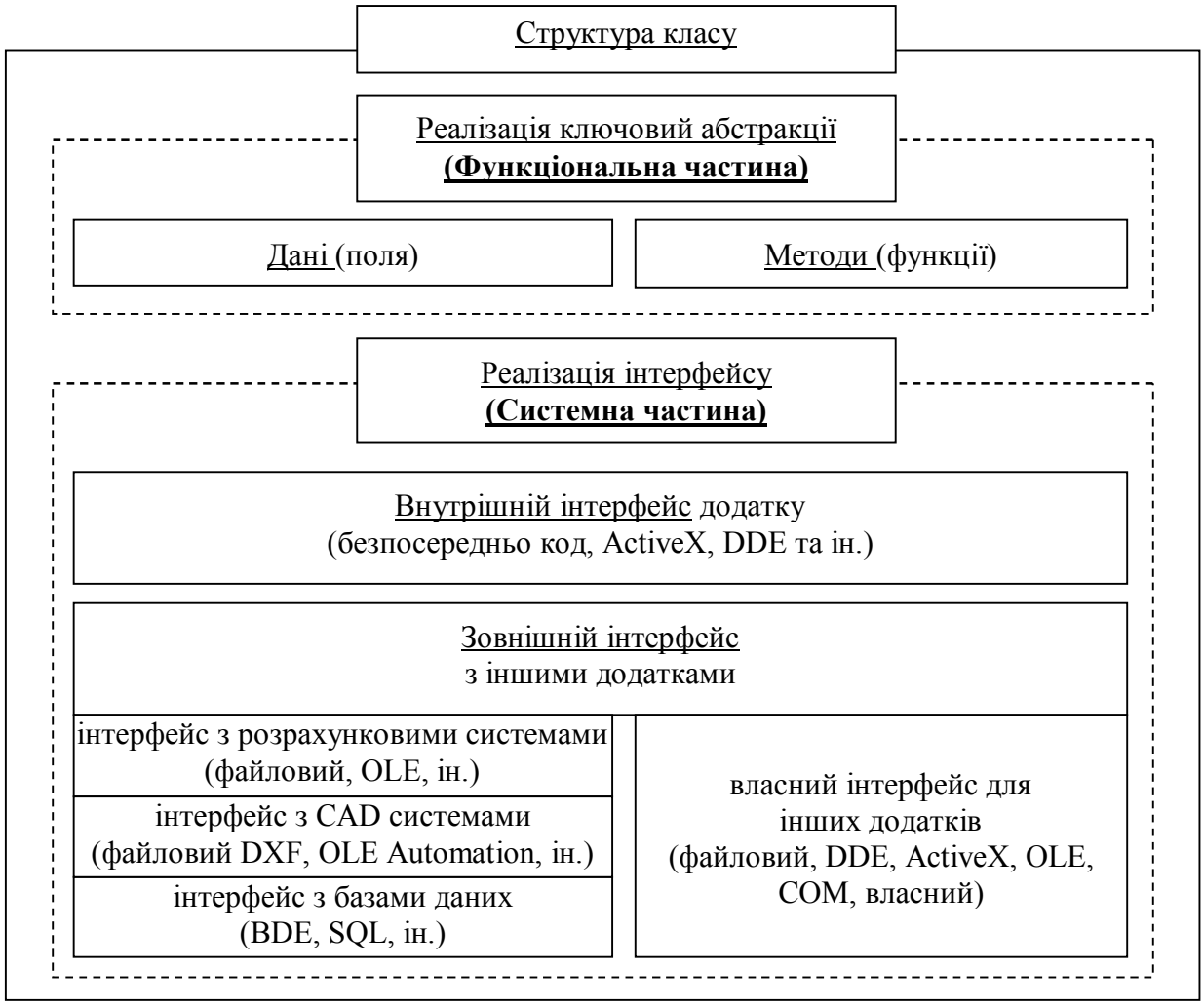



Рисунок 63 - Узагальнена схема структури класу

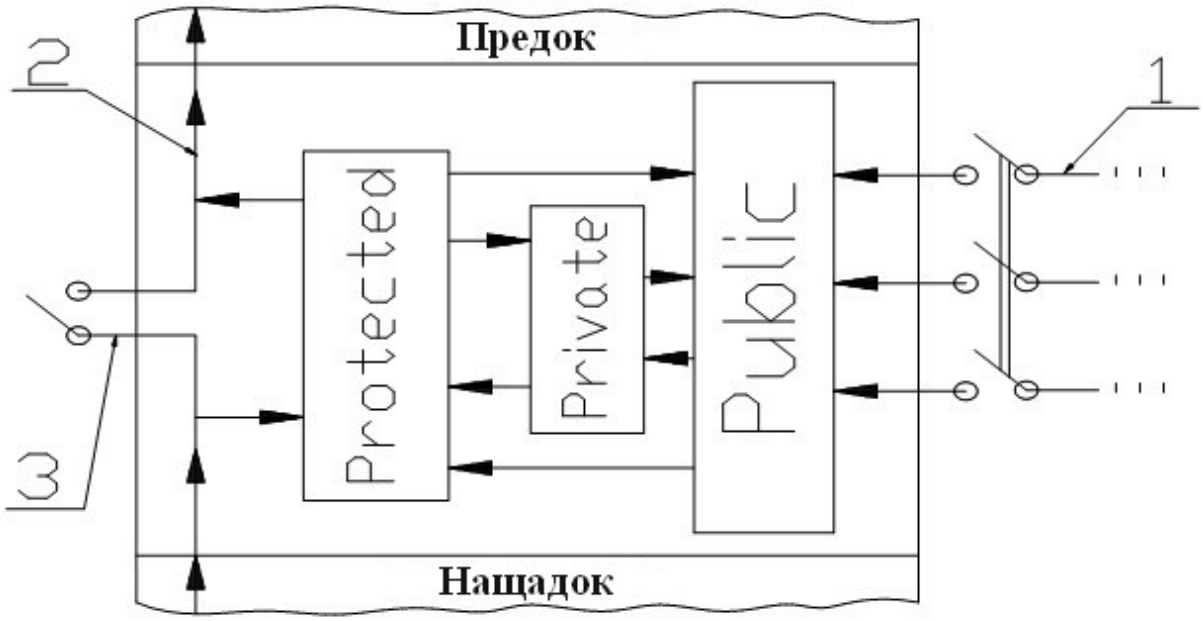
Побудова (організація) системної частини ПП, її спільність і гнучкість визначає можливості застосування в різних предметних областях. На такому під-

ході заснована генерація шаблонів додатків в генераторах (Майстрів) додатків і класів.

Істотне узагальнення завдання різко збільшує складність і вартість ПП, тому більш ефективно звузити можливості системної частини і звузити ряд алгоритмів предметної області. Оскільки не існує формалізованих методів оцінки функції «вартість - спільність постановки», то вибір застосовуваних рішень здійснюється інтуїтивно проектувальниками і програмістами, або на основі досвіду, накопиченого в організації.

### 3.8 Інформаційна взаємодія класів при різних видах спадкування

До теперішнього часу розроблено досить багато об'єктно-орієнтованих реалізацій мов програмування. Однією з останніх мов, що містить ряд нових можливостей, є Borland C++. Зокрема, при створенні ієрархії класів можливо спадкування не тільки типу Public, як в більшості мов, а й Protected, Private. Це дозволяє більш гнучко вирішувати завдання побудови ієрархії класів. У той же час виникає ряд труднощів, пов'язаних з ускладненням інформаційної взаємодії членів класів нащадків і предків. Дана проблема може бути вирішена за допомогою візуального представлення обмежень доступу до членів класів предків у вигляді такої структурної схеми (рисунок 64) [32].




1 - зовнішній інформаційний канал (відкритий інтерфейс), використовуваний для звернення до даних і методів класу ззовні; доступний тільки при спадкуванні типу Public;

2 - внутрішній інформаційний канал, який використовується для доступу до членів класів-предків;

3 - перемикач внутрішнього інформаційного каналу замкнутий при спадкуванні типу Public і Protected (при спадкуванні типу Private розімкнути)

Рисунок 14 - Структурна схема візуалізації доступу до членів класів предків

# 4 ВИДИ І ХАРАКТЕРИСТИКА СИСТЕМ АВТОМАТИЗАЦІЇ НА ВИРОБНИЦТВІ

## 4.1 Технологічна зрілість виробництва

Забезпечення можливості використання сучасних засобів автоматизації.

Вперше поняття технологічної зрілості виникло приблизно у 80-і роки при розробці програмного забезпечення для автоматизації систем дорогих проектів (космічні дослідження, системи військового призначення).

CASE-технології - всі ідеї реалізовані в цій технології.

Рівні технологічної зрілості досягли певного стану виробництва, дозволяють контролювати етапи і забезпечувати якість виконуваних проектів.

### Рівні управління промисловим підприємством

Виробничі виконавчі системи (MES) - необхідний елемент ефективного управління підприємством

(Виробничі виконавчі системи (MES) - шлях до ефективного підприємству В.Н Леньшин., В.В.Куминов, (ЗАО "РТСофт") (URL: http://www.asutp.ru/go/?id=600359&url=www.rtsoft.ru)) CALS 1

У більшості реалізованих проектів, пов'язаних зі створенням інтегрованих систем управління промисловим підприємством (у всякому разі, в Росії) існує цілий пласт функцій, що не покриваються ні класом ERP, ні класом АСУТП. На умовної моделі підприємства (рис.2), можна показати, що ERP-системи не забезпечують оперативного управління виробництвом, обмежуючись стратегічним плануванням, що зумовлює існування значного функціонального розриву між рівнем ERP і рівнем АСУТП. А саме в цьому "неохоплених" інформаційними технологіями шарі оперативного управління виробництвом знаходиться цілий клас життєво важливих для підприємства виробничих процесів, що створюють додаткову вартість продукції, і роблять значущий вплив на ефективність підприємства в цілому.

Цей клас завдань не новий і добре відомий. Засоби автоматизації цих процесів розроблялися, в тому числі і в нашій країні, більше 20 років тому і носили назву АСУ виробничих процесів (АСУПП).

В даний час ці системи позиціоновані в класі виробничих виконавчих систем (MES - Manufacturing Execution Systems), орієнтованих на інформатизацію завдань оперативного планування і управління виробництвом, оптимізації виробничих процесів і виробничих ресурсів, контролю та диспетчеризації виконання планів виробництва з мінімізацією витрат. Як і для ERP-систем, в даний час в класі MES-систем відбувається етап інтенсивної розробки формалізованої методології створення та впровадження даного класу виробничих систем (див. Статтю О.Ю Нестерової "MES-Системи управління виробництвом. Скористайтеся очевидними перевагами. Огляд" на Стор.).

На Заході використання MES систем вважається очевидним, і при вирішенні завдань комплексної автоматизації підприємства одночасно шукаються рішення для трьох взаємопов'язаних рівнів управління: АСУТП, MES і ERP. У Росії ж подібні системи практично невідомі і ігнорування їх необхідної ролі, на наш погляд, є причиною істотних проблем при створенні комплексних систем автоматизації промислових підприємств.


Рисунок 65 – Рівні технологічної зрілості

Функції, що реалізуються в MES-системах, аналогічні методам управління в ERP-системах, але тільки в інших часових масштабах і з іншими об'єктами контролю і управління. MES - це автоматизована виконавча система виробничого рівня, що надає ряд можливостей, які доповнюють і розширюють функції ERP-систем. Використовуючи фактичні технологічні дані, MES-системи підтримують всю виробничу діяльність підприємства в режимі реального часу. Швидкий результативний відгук на умови, що змінюються, в комбінації з орієнтацією MES на зниження витрат, допомагають ефективно управляти виробничими операціями і процесами. Крім того, MES-системи формують дані про поточні виробничі показники, необхідні для функціонування ERP-систем. Таким чином, MES-система - це сполучна ланка між орієнтованими на фінансово-господарські операції ERP-системами і оперативної виробничою діяльністю підприємства на рівні цеху, ділянки або виробничої лінії.

Звідси випливає, що інтегровану автоматизовану систему управління промисловим підприємством можна представити у вигляді трьох взаємопов'язаних рівнів управління.

При цьому кожен рівень виконує свою основну управлінську функцію:

верхній рівень управління підприємством (адміністративно-господарський) вирішує стратегічні завдання, а відповідна ERP-система забезпечує управління ресурсами в масштабі підприємства в цілому, включаючи частину функцій підтримки виробництва (довгострокове планування і стратегічне управління в масштабі: річне, квартальне, місячне);

середній рівень управління (виробничий) вирішує завдання оперативного управління процесом виробництва, а відповідна автоматизована система забезпечує ефективне використання ресурсів (сировини, енергоносіїв, виробничих засобів, персоналу), а також оптимальне виконання планових завдань (змінне, добове, декадні, місячне) на рівні ділянки , цеху, підприємства;

нижчі рівні технологічного управління вирішують класичні задачі управління технологічними процесами.

Треба відзначити, що при передачі частини функцій управління від систем ERP в MES-системи на виробничий рівень (керівництву виробництва, цеху, виробничої дільниці, технологу, начальнику служби експлуатації, і т.д.), відбувається раціональна сегментація контурів управління підприємством в цілому.

При цьому кожен контур управління характеризується своїм рівнем інтенсивності циркулюючої в ньому інформації, своїм масштабом часу і своїм набором функцій:

контур управління рівня АСУТП (технологічний) є самим інтенсивним за обсягом інформації і найжорсткішим за часом реакції, яке може становити секунди і навіть мілісекунди. У верхньому рівні шару АСУТП - в SCADA-системах відбувається накопичення і обробка великого числа технологічних параметрів і створюється інформаційна база вихідних даних для MES-рівня.

контур управління рівня MES (оперативно-виробничий) спирається на відфільтровану і оброблену інформацію, що надходить як від АСУТП, так і від інших служб виробництва (постачання, технічної підтримки, технологічних, планово-виробничих і т. д.). Інтенсивність інформаційних потоків тут істотно нижче і пов'язана з завданнями оптимізації заданих виробничих показників (якість продукції, продуктивність, енергозбереження, собівартість і т.д.). Типові часи циклів управління складають хвилини, години, зміни, добу. Оперативне управління виробництвом в цьому контурі управління здійснюється фахівцями, які більш детально, ніж вищий менеджмент, володіють виробничої ситуацією (керівники виробничих цехів, дільниць, головні технологи, енергетики, механіки та ін.). У зв'язку з цим має підвищуватися якість і ефективність прийнятих рішень в межах делегованих зверху повноважень.

## 4.2 Інтегровані і гнучкі виробничі системи (ІВС і ГВС, ІСУВ)

Ми розглядали поняття "складної системи" - С, які включають безліч різноманітних компонентів, що входять в БС, функцій, зв'язків між ними і зовнішніх умов, що впливають на роботу С.

Компоненти представляють певну цілісність, яка визначає єдине цільове призначення С. Так, сучасне велике промислове підприємство включає в себе цілий ряд виробництв, склади, транспорт, органи постачання, контролю та управління, що утворюють в сукупності велику систему.

В основі управління великими С лежить автоматизація, яка використовує досягнення кібернетики і ЕОМ.

Основна роль в автоматизації промислових підприємств відводиться автоматизованим системам управління (АСУ). За допомогою засобів автоматизації і ВТ автоматизуються всі види виробничої діяльності. Основні напрямки автоматизації:

- АСУ технологічними процесами (АСУТП);
- Автоматизація оперативного управління;
- Створення ГАП;
- Автоматизація адміністративно-організаторської діяльності (АСТП);
- Автоматизація систем масового обслуговування;
- Автоматизація проектно-конструкторських робіт (САПР);
- Автоматизація наукових досліджень (АСНІ);

Об'єднання компонентів перерахованих систем зі створенням інтегрованих систем управління виробництвом (ІСУВ) або інтегрованих виробничих систем (ІВС).

Крім ЕОМ різних типів значна увага приділяється техніці зв'язку і УПУ. На базі використання ЕОМ, техніки зв'язку і УПУ процеси управління, передачі і перетворення інформації на всіх етапах діяльності підприємства об'єднані в єдину систему - від планування і розробки вироби до контролю готової продукції.

Використання ІСУВ забезпечує значне розширення потужностей підприємств. Завдяки своєчасній і правильній передачі інформації на всі ділянки виробництва ліквідуються втрати, що виникають через дублювання, неповноти, помилок і затримок передачі інформації в звичайному виробництві. При цьому підвищуються гнучкість, скорочуються виробничі запаси і відповідні витрати, а також виробничий цикл.

У структурі ІСУВ зазвичай виділяють кілька рівнів:

– етапи підготовки виробництва відносяться до верхнього (першого) рівня;

– етапи безпосереднього виконання (другий рівень), пов'язані в єдину систему єдиним інформаційним потоком.

Планування різних рівнів, маркетинг і збут, адміністративне управління і т. д. Автоматизована розробка кон-трукцій (САПР).

Технологічні розробки, маршрут і операційні карти, документ з виготовлення та збирання, розробка управління програм для ЧПУ, плановане використання верстатів і матеріалів, управління якістю.

ЕОМ управляє ГВС, ділянками, архів програм, верстати з ЧПУ, транспортні, завантаження-розвантаження пристрої, автоматичні склади, вимірювальні пристрої.

Ступінь інтеграції компонентів може бути різною, комплектність і підпорядкованість визначається призначенням системи. Поділ ліній зв'язку на два рівня дозволяє краще забезпечити різні вимоги по швидкодії і помехозащищенности ліній. Крім того, можна реалізувати різні ступені двох основних показників:

- гнучкість АСУ і ІВС, ГВС;
- продуктивність ІВС, які суперечать один одному.

*Приклад.* П'ятирівнева ієрархічна інформаційно-керуюча мережа фірми Proctor & Gamble.

Таблиця 14 – П'ятирівнева ієрархічна інформаційно-керуюча мережа фірми Proctor & Gamble

| Рівні | Призначення | Час реакції | Засоби обчислювальної та іншого обладнання | Основні характеристики |
|---|---|---|---|---|
| 5 | Вище адміністративне управління | Місяці / роки | Великі універсальні ЕОМ і ПК з високою продуктивністю СУБД | Великий обсяг даних, секретна інформація |
| 4 | Адміністративне управління виробництвом | Тижні / місяці | Інтерактивні обчислювальні системи і ПК з СУБД | Гнучкість, невисокі вимоги до часу відповіді |
| 3 | Оперативне управління виробництвом | Години / дні | | |
| 2 | Контроль за виконанням технологічних операцій | Хвилини / години | Обчислювальні системи реального часу, міні-мікро ЕОМ | Високі вимоги до часу відповіді. Min t простою |
| 1 | Управління за технологічними операціями | Мілі-секунди / хвилини | Розподілені системи управління, контролери, системи ЧПУ, роботи, системи управління допоміжним обладнанням, мікропроцесорні системи | Спеціалізоване компактне обладнання |

## Передумови створення ІСУВ

1. Розробка з початку 70-х окремих систем автоматизованого проектування і управлінням виробництвом;

2. Використання систем ЧПУ, інтерактивної машинної графіки і технічних нововведень «острівці автоматизації»;

3. Здешевлення засобів обчислювальної техніки - розширення застосування в різних галузях техніки;

4. Розробка методів системного аналізу стосовно діяльності промислового підприємства;

5. Методи СА дозволяють уявити виробничий процес у вигляді ієрархічного ряду узгоджених, але досить незалежних функцій, що дає можливість забезпечити: розробку по одній загальній схемі, поетапне впровадження елементів.

## Вимоги до АСУВ

1. Відкритість архітектури;
2. Можливість інтеграції окремих підсистем в єдину систему;
3. Сумісність з існуючими підсистемами і методами організації виробництва;
4. Забезпечення динамічного зв'язку окремих процесів, реалізованих в інтерактивному режимі;
5. Незалежність інформації, з якою працюють користувачі, від способу її організації, зберігання і обробки;
6. Забезпечення різних сполучень підсистем;
7. Гнучкість системи, що забезпечується її програмними і апаратними засобами;
8. Можливість зміни конфігурації підсистем ІСУВ в залежності від змін в роботі системи;
9. Сумісність зі стандартами;
10. Продуктивність системи, стійкість до відмов.

Принцип відкритої архітектури ІСУВ дозволяє гнучко реагувати на нестандартні ситуації в роботі, також моделювати процес обробки виробів при «гнучких» обмеженнях.

Основним фактором, що визначає ефективність ГПС, є не ступінь автоматизації, а гнучкість: широкий діапазон серійності оброблюваних деталей, можливість зміни їх конструкції, а також маршрутів обробки деталей. Окремі станції ГПС зазвичай пов'язані з комп'ютерним управлінням єдиної транспортної системою. Всі операції в ГПС здійснюються під контролем центральної ЕОМ.

Найважливішим засобом підвищення ефективності ГПС є комп. Моделювання, що дозволяє аналізувати роботу ГПС в динаміці.

## Приклад ГПС

Призначення: обробка корпусних виливків.

- Необроблені заготівки;
- Токарні верстати;
- Конвеєрна транспортна система;

– Зона завантаження-розвантаження і фіксації на пристосування Х-супутниках;
– Машина для миття;
– Роботи;
– Контрольно-вимірювальна станція;
– Робокари;
– Горизонтально-фрезерні верстати.

Заготовки завантажуються на палети по 16 шт. на 1 і подаються до токарного верстата, обробляються. Потім мийна машина, робокар (разом із супутником) перевозить до верстатів: 10 однакових ГФС (5 верстатів - 1 операція, 2 верстати - 2 операції, 3 верстати - 3 операції). Після другої операції деталі розгортаються на 180° (із супутником). Після обробки - контроль.

Верстати можуть виконувати 1 операцію, але можуть і кілька, різні, якщо забезпечені магазинами з інструментом. Розрізняють 3 типи гнучкості: щодо номенклатури продукції, обсягу і термінів випуску виробів.

Для вибору оптимального варіанту складу і планування роботи ДПС необхідно знати параметри ГПС, що дозволяють оцінити її якість: (за даними Японії)
– Тип гнучкості системи;
– Простота управління виробництвом, технологічні новинки.

Стандартні параметри оцінки ГПС:

Продуктивність, номенклатура виробів, термін ефективної експлуатації, загальний обсяг капітальних вкладень, термін окупності, ефективність, відсоток браку, середня кількість операторів на 1 обладнання, кількість вивільненого персоналу в порівнянні зі звичайним виробництвом, зростання виробництва, у розрахунку на одного оператора, на одиницю площі , період роботи ГПС без поповнення сировини і матеріалів, ємність накопичувачів для окремих виробничих ділянок.

В Японії період окупності, як правило, 2-3 роки іноді 3-5 років (Німеччина – 3-4 роки).

Підвищення ефективності ГПС можливо за рахунок вдосконалення управління і зв'язку. Будь-яка ГПС включає три основних компоненти:
– ЕОМ і виконавчими механізмами);
– Система управління (з апаратної і програмної частин);
– Система зв'язку (з програмного забезпечення і пристроїв зв'язку у вигляді інтерфейсів між пристроями.

Виконавчі механізми (верстати, роботи і т. д.) постачання системами ЧПУ.

Вартість комп'ютерної системи управління зазвичай від 15 до 40% від повної вартості ГПС, причому основні витрати пов'язані зі створенням ПЗ.

Системи зв'язку в ГПС схожі один на одного незалежно від виду ГПС, оскільки у них збігаються основні функції:
– Забезпечення обміну інформацією всередині ГПС між її окремими елементами, програмами і т. п .;
– Передачі керуючих команд виконавчими пристроями (запуск, зупинка і т. д.);

– Передача інформації про стан пристроїв (зайняті, вільні і т. д.).

Одна з основних проблем, що виникають при створенні ГПС - об'єднання інтерфейсів всіх елементів в єдину систему. Для вирішення цієї проблеми використовується автоматичне протоколювання процесу виробництва (МАР - стандарт Manufacturing Automation Protocol), що регламентує побудова інформаційної структури ІСУВ. МАР заснований на застосуванні семи типів описів ієрархії зв'язків Мехді елементами системи (зверху-вниз):

Запуск: виконання операцій безпосереднього прикладними програмами;

Відтворення: перетворення і подання даних в узгодженому форматі, і їх зворотне перетворення;

Узгодження: синхронізація і управління даними;

Передача: передача необхідних даних від одного елемента системи до іншого;

Створення мережі: передача необхідних даних між що не примикають один до одного елементами системи;

Перевірка даних: пошук помилок в повідомленнях між сусідніми елементами системи;

Реальна дія: перекодування команд в код реальних пристроїв і їх передача цих пристроїв.

Однією з найбільш важливих завдань при створенні ГАП і АСУВ в цілому є вироблення загальної концепції при визначенні структури бази даних. Розвиток систем управління супроводжується, з одного боку, розширенням функцій і підвищенням складності і потужності автономних систем управління верстатами та іншими пристроями, а з іншого боку, збільшенням централізації управління, координації і контролю робочих процесів обробки і складання виробів.

Роботи по вдосконаленню ГПС ведуться за такими основними напрямками:

– Забезпечення нових системних функцій, організація роботи ГПС в автономному режимі у вечірню та нічну зміни, розробка системи управління інструментом, впровадження мереж, МАР і т. д.

– Розробка програмного забезпечення для управління виробничим графіком, моделювання роботи ГПС та інтеграції програмних засобів підсистем (АСУ, САПР, АСУТВ і т. д.)

– Удосконалення окремих елементів ГПС: обладнання для установки заготовок, видалення стружки; обробка центрів; системні транспортування матеріалів і заготовок, діагностичного обладнання, обладнання контролю точність обробки. Датчиків визначення аварійних ситуацій.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Биков В.П. Методичне забезпечення САПР в машинобудуванні -Л: Машино-будування, Ленингр. отд-е, 1989. - 255 с.

2 Половинкин А.І. Основи інженерної творчості. Вид. 2-е перераб. і доп. - М: Машино-будування, 1988. - 362 с.

3 Калянов Г.Н. CASE - структурний системний аналіз. - М: ЛОРИ, 1996. - 242 с.

4 Граді Буч. Об'єктно-орієнтоване проектування. - К .: Діалектика. ІВК (Москва)

1992-519с.

5 Автоматізація схемотехнічного проектування в машинобудуванні. //А.І. Петренко, В. В. Ладогубец, В. В. Чкалов. К .: УМК ВО, 1988. - 180 с.

6 Одрін В.М., Кратавов С.С. Морфологічний аналіз систем.-К: Наукова думка, 1977.- 183 с.

**Автоматизоване проектування складних об'єктів і систем**

**Конспект лекцій**

Укладач: А. В. Люта, к.т.н., доц.

Редактор:

Підписано до друку Формат 60х84/16
Різографіч. печатка Ум. друк л.
Тираж Заказ № .
ДДМА. 84313, м. Краматорськ, вул. Академічна, 72