МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет інформаційних технологій
Кафедра інформаційних технологій та
комп'ютерної інженерії

В.І. Олевський, Ю.Б. Олевська, Н.О. Соколова

Моделювання інформаційних систем

Конспект лекцій

Дніпро
НТУ «ДП»
2024

Олевський В. І. Конспект лекцій з дисципліни «Моделювання інформаційних систем» для здобувачів ступеня бакалавра спеціальності 126 Інформаційні системи та технології / В. І. Олевський, Ю. Б. Олевська, Н. О. Соколова ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Електрон. дані. – Дніпро : НТУ «ДП», 2024. – 499 с.

Автори:
В.І. Олевський, д-р. техн. наук, проф. каф. ІТКІ
Ю.Б. Олевська, канд. фіз.-мат. наук, доц. каф. ПМ
Н.О. Соколова, канд. техн. наук, доц. каф. ІТКІ

Погоджено науково-методичною комісією спеціальності 126 Інформаційні системи та технології (протокол № 4 від 10.04.2024) за поданням кафедри інформаційних технологій та комп’ютерної інженерії (протокол № 14 від 10.04.2024).

Розглянуто прикладні питання теорії моделювання інформаційних систем, систем масового обслуговування, методи синтезу мереж Петрі, імітаційне моделювання інформаційних систем та дослідження таких моделей. Розкрито теми «Поняття фреймворку. Типи фреймворків», «Методи дослідження систем масового обслуговування», «Метод аналізу ієрархій», «Принципи статистичного імітаційного моделювання», «Оцінка якості інформаційних систем», розглянуто приклади по кожній з тем, надано перелік використаних і рекомендованих джерел.

Відповідальний за випуск професор кафедри ІТКІ В.І. Олевський, д-р. техн. наук, проф.

ЗМІСТ

- **Лекція 1.** Моделювання. Основні поняття. Поняття моделі і моделювання. Формальні методи побудови моделей складних систем і процесів.
- **Лекція 2.** Інформаційні системи (ІС). Види, структура, типи взаємодії ІС. Типи моделей життєвого циклу ІС. Основи гнучкого моделювання. Процеси життєвого циклу ІС.
- **Лекція 3.** Властивості і кваліфікації інформаційних систем. Морфологічна модель інформаційної системи. Матрична форма опису морфологічної моделі системи. Функціональна модель системи. Поняття архітектури інформаційних систем. Типи архітектури.
- **Лекція 4.** Поняття фреймворку. Типи фреймворків. Схема Захмана. Фреймворк TOGAF.
- **Лекція 5.** Типи фреймворків. Фреймворк DoDAF. Фреймворк FEA. Фреймворк GEAF.
- **Лекція 6.** Системи масового обслуговування (СМО). Базові поняття і показники ефективності СМО. Структура СМО та розподіл потоку вимог.
- **Лекція 7.** Методи дослідження СМО. Математична модель СМО з одноканальною чергою і пуасонівським потоком вимог. Дослідження стаціонарного стану СМО.
- **Лекція 8.** Аналітичні моделі найбільш поширених СМО з очікуванням. Формули Літтла. Імітаційне моделювання інформаційних систем. Принципи статистичного імітаційного моделювання.
- **Лекція 9.** Мережі Петрі. Завдання аналізу мереж Петрі.

- **Лекція 10.** Метод аналізу ієрархій. Поняття про ієрархії, побудова ієрархій. Метод аналізу ієрархій. Шкала Сааті. Розрахунок локальних пріоритетів. Метод ранжування факторів інформаційних процесів.
- **Лекція 11.** Оцінка якості інформаційних систем. Стандарти оцінки якості інформаційних систем. Моделі прогностичного оцінювання якості процесу функціонування ІС.

ВСТУП

Мета дисципліни – формування компетентностей щодо сучасних наукових концепцій, методів та технологій розробки і застосування математичних моделей різних систем для проведення дослідження складних об’єктів та систем, зокрема інформаційних систем.

ОЧІКУВАНІ ДИСЦИПЛІНАРНІ РЕЗУЛЬТАТИ НАВЧАННЯ

- ❖Формулювати вимоги до моделей ІС, виконувати вибір методів та засобів аналізу і моделювання ІС.
- ❖Розробляти концептуальні, логічні та фізичні моделі процесів функціонування ІС, використовуючи сучасний інструментарій.
- ❖Знати методи побудови концептуальних, логічних та фізичних моделей даних ІС, принципи та методи об’єктного моделювання ІС.
- ❖Проводити планування та виконувати машинні експерименти.
- ❖Оцінювати результати моделювання ІС.

Лекція 1

МОДЕЛЮВАННЯ. ОСНОВНІ ПОНЯТТЯ

ПОНЯТТЯ МОДЕЛІ І МОДЕЛЮВАННЯ. ФОРМАЛЬНІ МЕТОДИ ПОБУДОВИ МОДЕЛЕЙ СКЛАДНИХ СИСТЕМ І ПРОЦЕСІВ

Мета дисципліни – формування компетентностей щодо сучасних наукових концепцій, методів та технологій розробки і застосування математичних моделей різних систем для проведення дослідження складних об’єктів та систем.

Завданнями курсу є вивчення теорій, сучасних методів моделювання які дають можливість досліджувати складні технічні об’єкти, абстрагуючись від тих властивостей, які не мають суттєвого значення.

Тема 1. Моделювання. Основні поняття. Види моделей, їх класифікація. Вимоги до моделей

“Моделювання – це спосіб дослідження будь-яких явищ, процесів або об'єктів шляхом побудови й аналізу їх моделей.

Моделювання є однією з основних категорій теорії пізнання і мало не єдиним науково обґрунтованим методом наукових досліджень систем і процесів будь-якої природи в багатьох сферах людської діяльності” [1-10] .

Модель

- об'єкт-замінник об'єкта-оригіналу, що забезпечує вивчення деяких властивостей останнього;
- спрощене подання системи для її аналізу і отримання результатів, необхідних для прийняття управлінського рішення

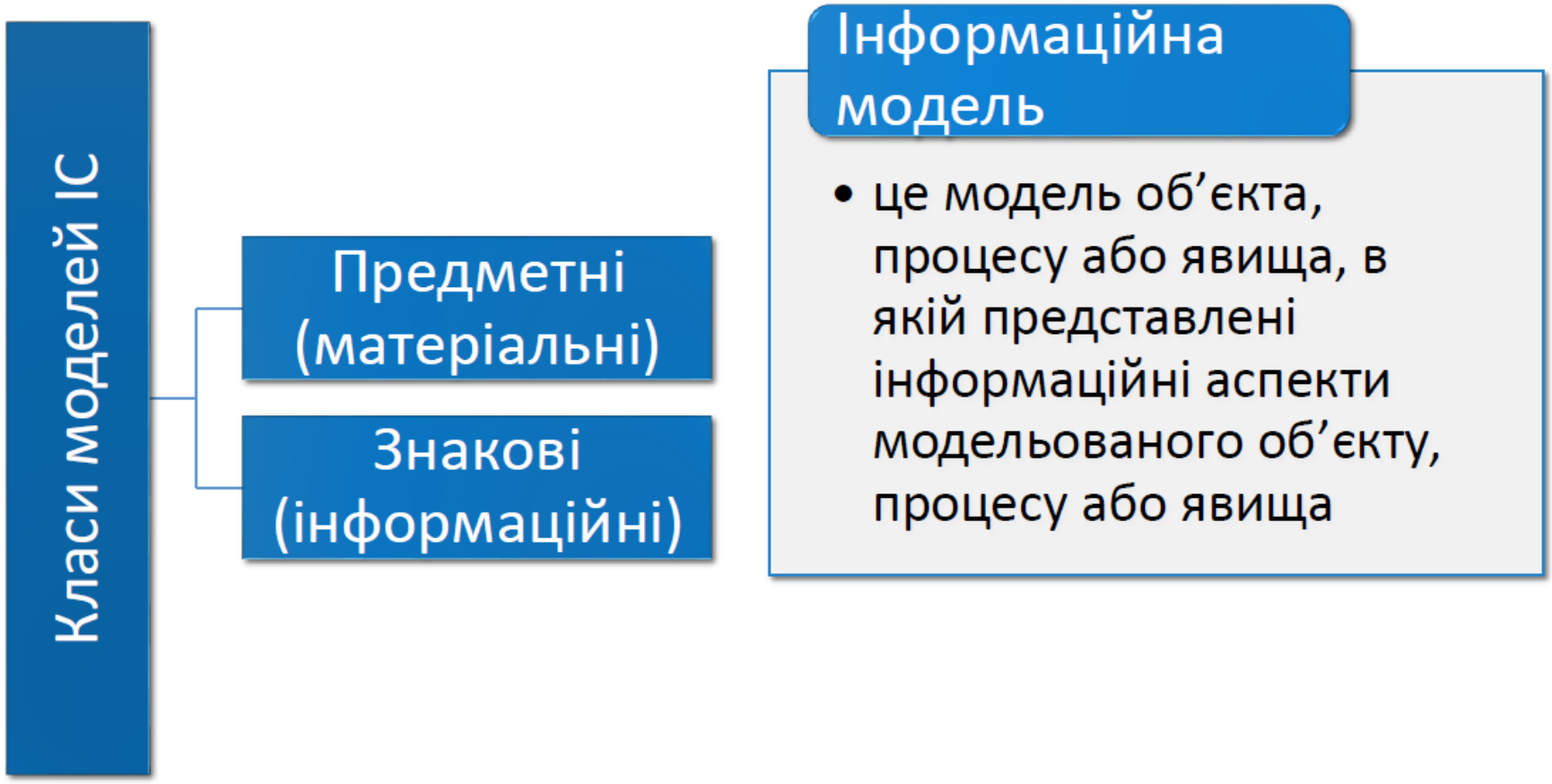
[9]

Моделювання

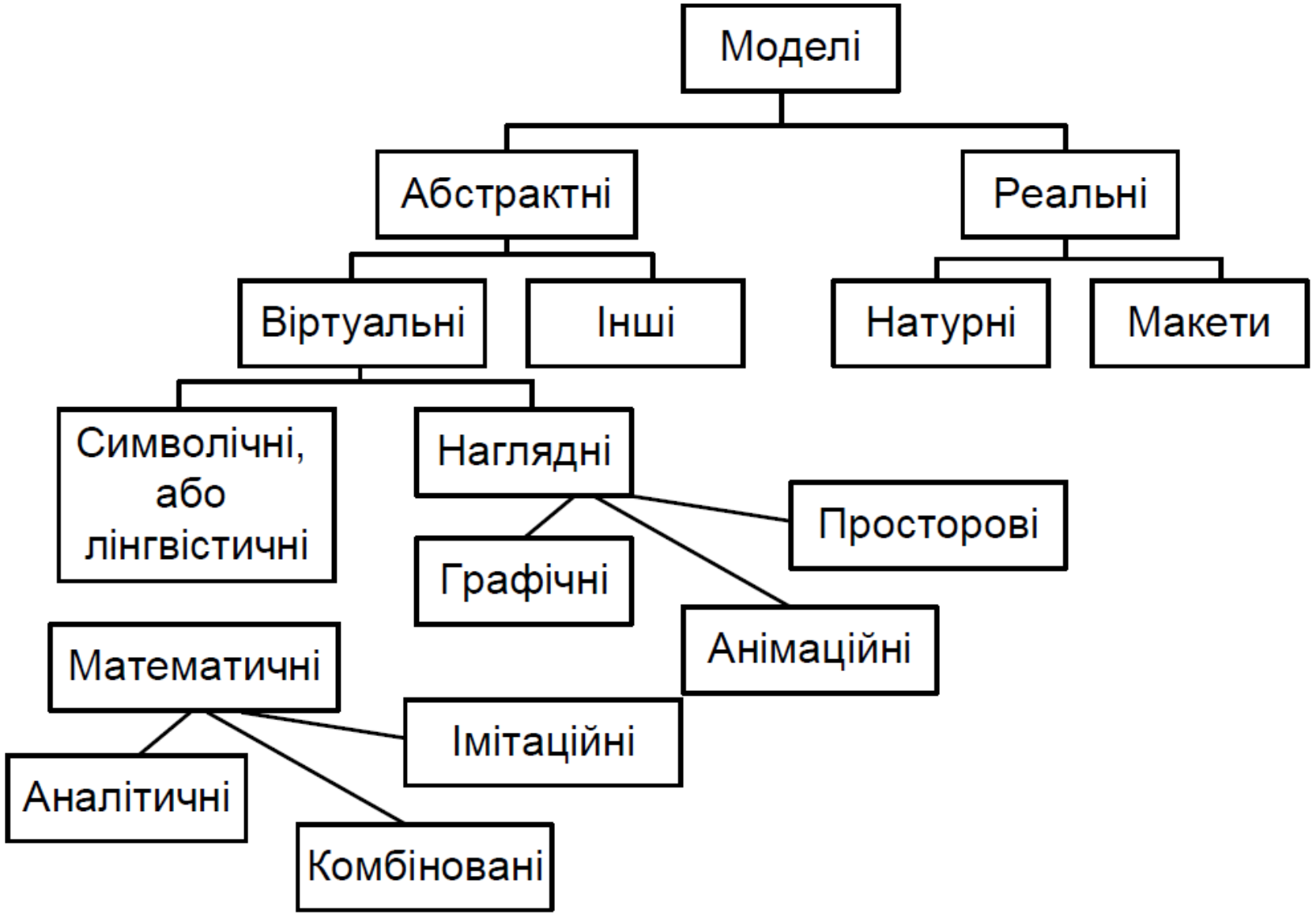
- процес створення точного опису системи;
- метод пізнання, що складається в створенні і дослідженні моделей.

[9]

Класи моделей ІС



[9]


Моделі
Абстрактні
Реальні
Віртуальні
Інші
Натурні
Макети
Символічні, або лінгвістичні
Наглядні
Просторові
Графічні
Анімаційні
Математичні
Імітаційні
Аналітичні
Комбіновані

[9]

ПОНЯТТЯ СИСТЕМИ

Основними поняттями є поняття **"система"** і **"модель"**.

Система – цілісний комплекс взаємозв'язаних “елементів, який має певну структуру і взаємодіє із зовнішнім середовищем.

Структура – це організована сукупність зв'язків між її елементами. Зв'язок - можливість впливу одного елементу системи на інший.

Середовище – це сукупність елементів зовнішнього світу, які не входять до складу системи, але впливають на її властивості.

Система є ***відкритою***, якщо існує зовнішнє середовище, яке впливає на систему, і ***закритою***, якщо зовнішнє середовище відсутнє або не враховується, у зв'язку з поставленими цілями досліджень” [9].

“Система більше, ніж сукупність елементів. Потрібно враховувати оцінку системного (синергетичного) ефекту.

Система – це сукупність взаємопов’язаних об’єктів, що має чотири наступні властивості.

– *цілісність та частковість*. Система – це цілісна сукупність окремих елементів. Зібрані разом елементи можуть створювати систему, а можуть й не створювати. Крім того, система може зруйнуватися без якогось елемента;

– *наявність зв’язків*. Системі притаманна наявність суттєвих стійких зв’язків (відношень) між елементами або властивостями, які перевищують за потужністю (силою) зв’язки цих елементів з елементами, що не входять в дану систему. За фізичним наповненням

зв’язки можна розділити на дійсні, енергетичні, інформаційні, змішані. За напрямком розрізняють зв’язки прямі, зворотні, нейтральні.

– *наявність визначеної організації*. Це формування зв’язків елементів, упорядковане розподілення зв’язків та елементів в часі та просторі. При цьому складається визначена структура системи, а властивості елементів трансформуються в функції (дії, поведінку). Організація проявляється у зменшенні ентропії, у порівнянні з ентропією, **системоформуючих факторів** (*елементів* та *зв’язків*), які визначають можливість створення системи;

– *наявність інтегративних якостей*. Це якості, які притаманні системі в цілому, але не властиві жодному з її елементів окремо.

Визначення поняття системи пов'язані з абстрактною теорією систем, в рамках якої використовуються такі **рівні абстрактного опису**:

– символічний, або лінгвістичний” [10];

– теоретико-множинний;

– абстрактно-алгебраїчний;

– топологічний;

– логіко-математичний;

– теоретико-інформаційний;

– динамічний;

– евристичний.

“Найвищий рівень абстрактного опису систем – ***лінгвістичний***; ґрунтуючись на ньому, можна отримати всі інші рівні.

Система – це окремий випадок теорії, описаний формальною мовою, яка уточнюється до мови об'єктів.

Для визначення деякого поняття використовують певні символи (алфавіт) і встановлюють правила оперування ними. Сукупність символів і правил користування ними утворює абстрактну мову. Поняття, висловлене абстрактною мовою, означає будь-яке речення (формулу), побудоване за граматичними правилами цієї мови.

Якщо існує множина висловлювань *G*, але лише *V* з них істинні, то вважають, що має місце **теорія *L* щодо множини *G***. Таке речення містить змінні, що підбираються, так звані ***конституенти***, які, маючи тільки певні значення, роблять дане висловлювання істинним.

Система – це множина правильних висловлювань.

Усі висловлювання поділяються на два типи: ***терми***, які вказують на предмети (об'єкти), і ***функтори***, які визначають відношення між термами (об'єктами)” [9, 10].

Теоретико-множинне визначенням системи

“Система – це множина об'єктів, між якими існують певні відношення, а також їх атрибути.

Об'єктами - компоненти (елементи) системи. Це, наприклад, підсистеми (тобто може існувати ієрархія підсистем) або окремі об'єкти системи.

Атрибути – це властивості об'єктів.

Відношення задають певний закон, за яким визначається деяке відображення в одній і тій же множині об'єктів.

Згідно з цим визначенням поняття ***множина*** і ***елемент*** є аксіоматичними.

Система S задається парою елементів:

$$S\,(X_S, R_S),$$

де X_S, – множина елементів (об'єктів) системи,

R_S множина відношень між елементами.

У загальному випадку n -відношення R в множинах $X_1, X_2, \ldots, X_n$ є деякою підмножиною декартового добутку $X_1 \times X_2 \times \ldots \times X_n$, який складений з наборів виду $(x_1, x_2, \ldots, x_n)$, де $x_i \in X$, $i=1,2,\ldots,n$.

Якщо відношення R задається функцією, яка визначає зв'язок між елементом $x \in X$ і елементом y підмножини Y , то вважаємо, що f перетворить значення з множини X у значення підмножини Y .

$$f : X \rightarrow Y$$ "[9,10].

Атрибути системи “подібні до функцій, визначених на підмножині об'єктів. Відмінність атрибутів від функцій полягає в тому, що два різних атрибути з точки зору поняття функції можуть бути однаковими. Атрибут А задається парою елементів

$$(i , f),$$

де *i* – ім'я атрибута,

f – функція, визначена на підмножині об'єктів.

У динамічних об'єктів атрибут також може бути функцією від часу *t* .

Система може складатися з **підсистем** або бути одним з елементів **більшої системи**, тобто може існувати **ієрархія систем”** [9, 10].

“Якщо зв'язки між елементами даних множин встановлюються за допомогою деяких однозначних функцій, які відображають елементи множини в саму початкову множину, то має місце ***абстрактно-алгебраїчний*** рівень опису систем. У таких випадках між елементами множини встановлені унарні, бінарні та ін. відношення.

Якщо ж на даних множинах визначені деякі багатозначні функції, то мають місце ***топологічні абстрактні*** моделі, записані мовою загальної топології або її гілок (топологією алгебри, гомологічною топологією і т. п.

Вибір рівня абстрактного опису є завжди найбільш відповідальним кроком у теоретико-системних побудовах і **майже не піддається формалізації**” [9,10].

“ПОНЯТТЯ МОДЕЛІ

Науковою основою моделювання як методу пізнання і дослідження різних об'єктів і процесів є ***теорія схожості***, в якій головним є поняття ***аналогії***, тобто схожість об'єктів за деякими ознаками. Подібні об'єкти називаються аналогами. Аналогія між об'єктами може встановлюватися за якісними і (або) кількісними ознаками.

Основним видом кількісної аналогії є ***математична схожість***, коли об'єкти описуються за допомогою рівнянь і функцій.

Функції і незалежні змінні називаються схожими, якщо вони співпадають з точністю до деяких констант” [9-12].

геометрична схожість, ***часова***, ***фізична схожість***

“У теорії систем визнається **об'єктивність існування систем**.

Якщо реально існують взаємозв'язки між об'єктами, то існують і системи, які їм відповідають.

Дві множини *X, Y* називаються ***ізоморфними***, якщо між елементами цих множин можна встановити взаємно однозначну відповідність.

Постулат функціонально-структурного ізоморфізму об'єктів і явищ природи:

Якщо структура однієї системи і зовнішні функції її елементів ізоморфні структурі іншої системи і зовнішнім функціям її елементів, то зовнішні властивості цих систем не розрізняються в області їх ізоморфізму” [10].

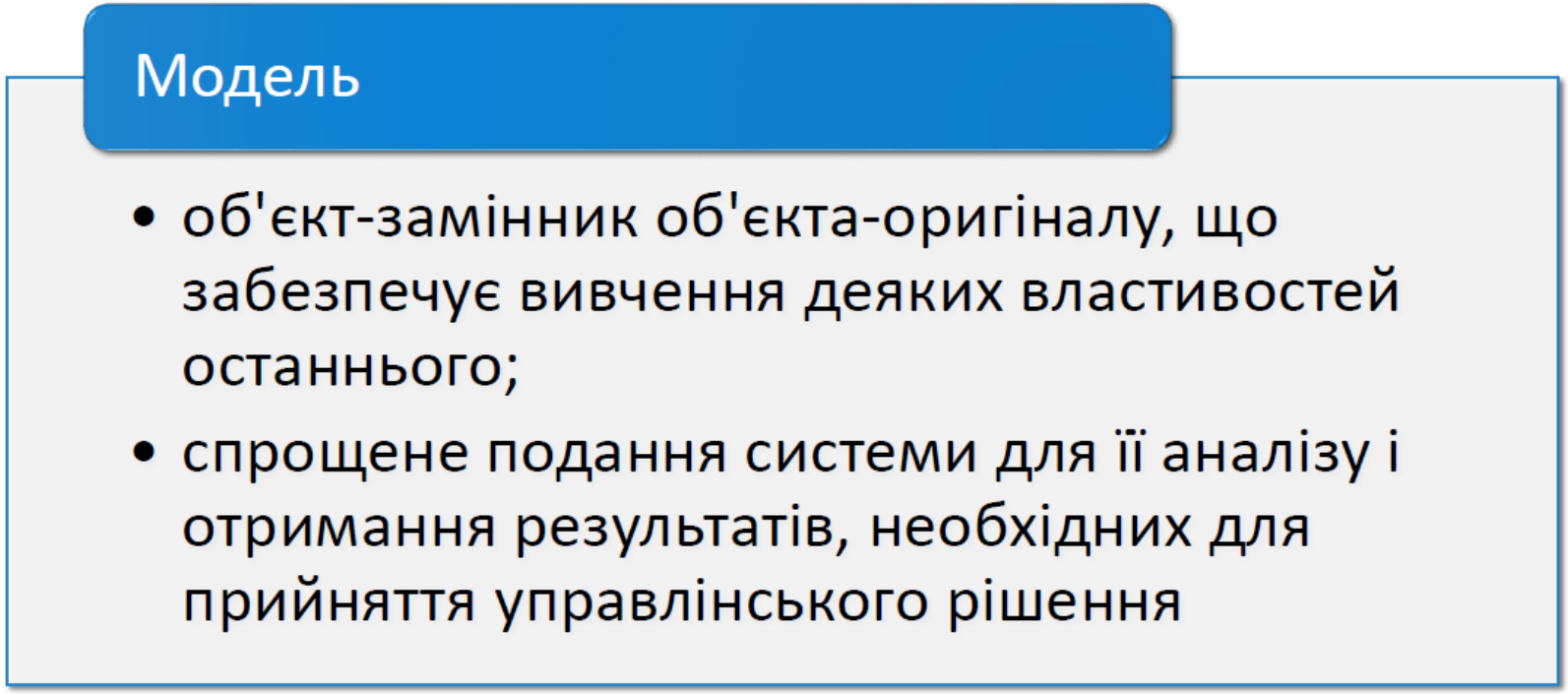
“Одна з таких систем є ***моделлю*** іншої (***оригіналу***) і навпаки.

Таких ізоморфних систем може бути безліч. Виникає проблема вибору або побудови системи, яка може бути моделлю досліджуваної системи.

Якщо система існує реально, то її можна вивчати, досліджуючи, яким чином зв'язані вхідні впливи з виходами системи. На основі результатів досліджень будується деяка ***абстрактна система***.

У ній відношення еквівалентності визначається тільки для тих важливих властивостей і аспектів поведінки, які в початковій та в абстрактній системах повинні бути однаковими. На практиці абстрактна система є ***більш простою***, ніж початкова” [13].

“**Моделлю** можна називати систему, яку використовують для дослідження іншої системи.



Модель – це реально існуюча або абстрактна система, яка, замінюючи і відображаючи в пізнавальних процесах іншу систему – оригінал, перебуває з нею у відношенні схожості” [1-13].

Використовуються ***моделі двох основних типів***: "***феноменологічні***" і "***дедуктивні***".

“Під **феноменологічними** моделями розуміють переважно емпірично відновлені залежності вихідних даних від вхідних, як правило, з невеликою кількістю входів і виходів. Це ***моделі даних***, які не використовують і не відображають яких-небудь гіпотез про фізичні процеси або системи, з яких ці дані отримані. До моделей даних належать всі моделі математичної статистики. Їх існування ув'язується з експериментально-статистичними методами.

Дедуктивне моделювання передбачає з'ясування і опис основних фізичних закономірностей функціонування всіх компонентів досліджуваного процесу і механізмів їх взаємодії. Це ***системні моделі***, які будуються на базі фізичних законів і гіпотез про те, як система структурована і як вона функціонує.

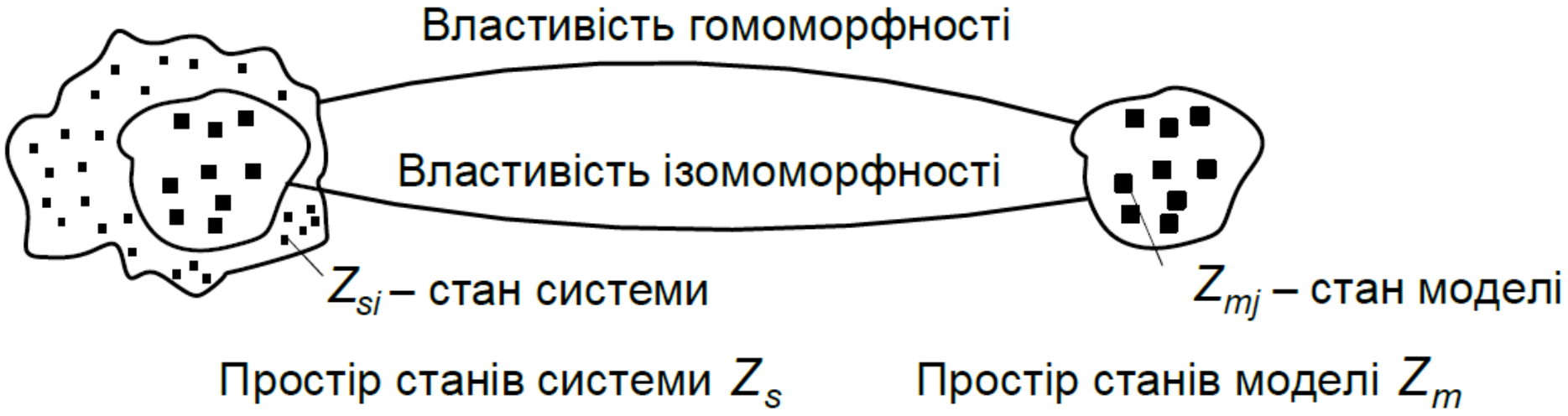
Таким чином, **модель є абстракцією системи і відображає деякі її властивості**. Цілі моделювання формулює дослідник. Тільки завдяки ним можна визначити сукупність властивостей модельованої системи, які повинна мати і модель. Від мети моделювання залежить потрібний ступінь деталізації моделі" [11-14].

Співвідношення між моделлю та системою

“Аналогія, абстракція і спрощення – це основні поняття, які використовуються при моделюванні систем.

Станом динамічної системи (моделі) в деякий момент часу *t* називається множина значень всіх її параметрів (змінних), виміряних одночасно у цей момент. При зміні значення хоча б одного параметра системи в наступний момент часу говорять, що стан системи змінився. Стан системи зручно розглядати як точку в багатовимірному просторі. Множина всіх можливих станів системи називається ***простором станів системи***.

Ізоморфна і гомоморфна залежності між системою і моделлю для просторових станів системи Z_s і моделі Z_m



Система і модель є **ізоморфними**, якщо існує взаємно однозначна відповідність між ними.

Гомоморфні зв'язки визначають однозначну відповідність лише від моделі до системи.

1. **Детерміновані відношення**, коли стан системи однозначно визначає стан моделі і навпаки:

$$P(Z_m = Z_{mj} \mid Z_s = Z_{si}) = P(Z_s = Z_{si} \mid Z_m = Z_{mj}) = 0 \vee 1,$$

де *P* – ймовірність.

2. **Імовірнісні відношення зі скінченою множиною станів:**

$$P(Z_m = Z_{mj} \mid Z_s = Z_{si}) = 0 \vee 1,$$

$$P(Z_s = Z_{si} \mid Z_m = Z_{mj}) \leq 1,$$

Стан системи однозначно визначає стан моделі, але стан моделі визначає стан системи лише з деякою ймовірністю.

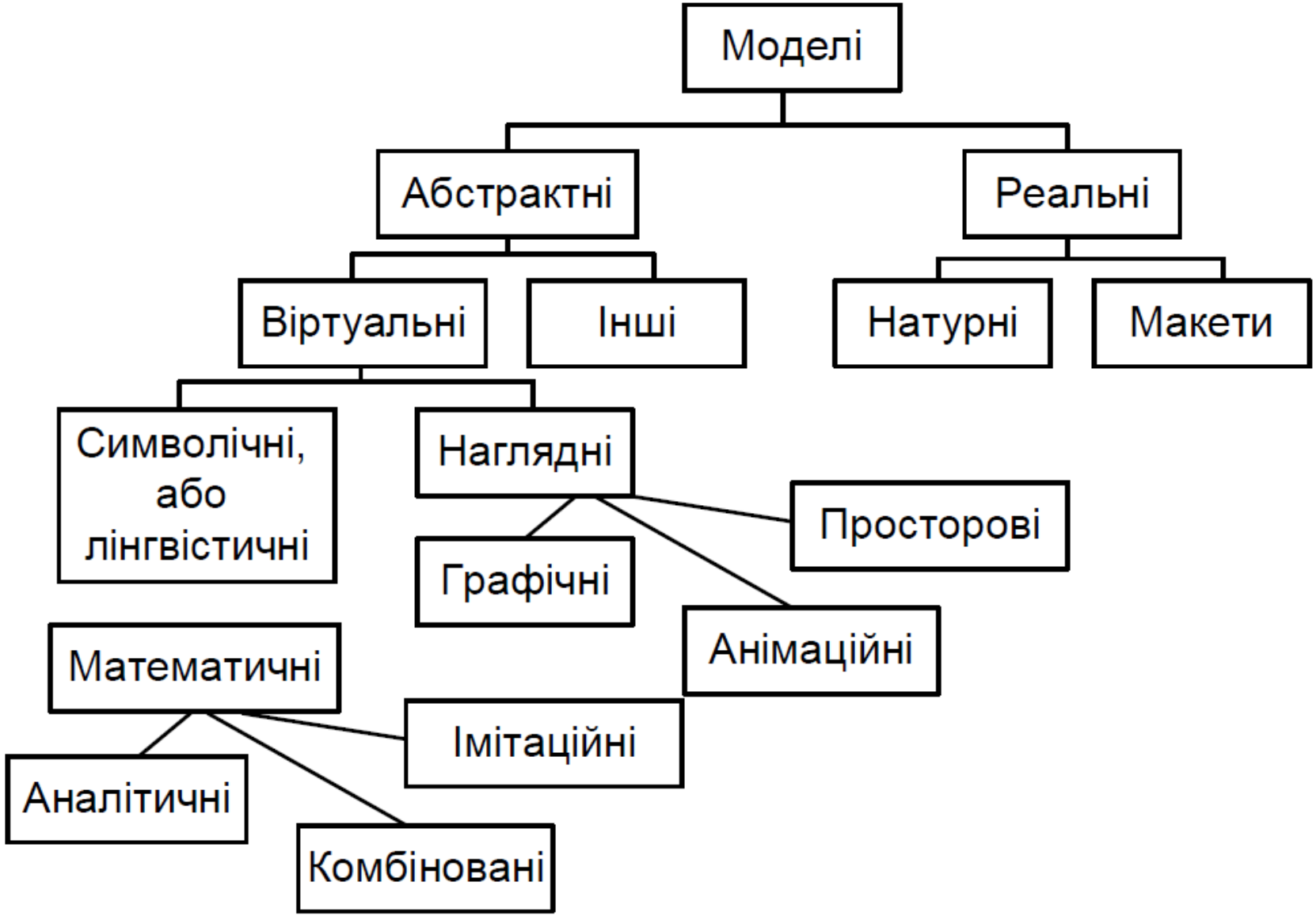
3. **Імовірнісні відношення з нескінченною множиною станів:**

$$P(Z_m = Z_{mj} \mid Z_s = Z_{si}) \le 1,$$

$$P(Z_s = Z_{si} \mid Z_m = Z_{mj}) \le 1.$$

Стани системи і моделі визначають стани один одного лише з деякою ймовірністю” [9-15].

“Види моделей та їх класифікація за різними критеріями



Ознаки класифікації:

Спосіб подання моделі.

За цією ознакою розрізняють ***абстрактні і реальні.***

Астрактні ***конструкції:***

основною є віртуальна (**уявна**) модель:

наглядна ***модель (***за допомогою графічних образів і зображень),

у ***формальному вигляді***, створюють ***символічну***, або ***лінгвістичну***, модель.

Основним видом абстрактної моделі є ***математична модель***.

Вид математичні моделі: ***аналітичні***, ***імітаційні*** і ***комбіновані***.

Для ***аналітичної моделі*** характерно те, що процеси функціонування елементів системи записуються у вигляді деяких функціональних співвідношень (алгебрі, інтегрально-диференціальних, кінцево-різницевих і т. п.) або логічних умов.

В ***імітаційній моделі*** відтворюється процес функціонування системи *S* у часі, причому імітуються елементарні явища, що складають процес.

Використання ***комбінованих*** (аналітико-імітаційних) ***моделей*** при аналізі і синтезі систем дозволяє об'єднати переваги аналітичних й імітаційних моделей.

Залежно від можливості змінювати в часі свої властивості моделі поділяються на ***статичні*** і ***динамічні***.

Залежно від того, яким чином відтворюються в часі стани моделі, розрізняють ***дискретні***, ***неперервні*** і ***дискретно-неперервні*** (комбіновані) моделі.

Відповідно до співвідношень між станами системи і моделі розрізняють ***детерміновані*** і ***стохастичні*** моделі.

Вимоги до моделей

– ***незалежність результатів*** розв'язання задач від конкретної фізичної інтерпретації елементів моделі;

– ***змістовність***, тобто здатність моделі відображати важливі риси і властивості реального процесу, який вивчається і моделюється;

– ***дедуктивність***, тобто можливість конструктивного використання моделі для отримання результату (управління, прогнозування);

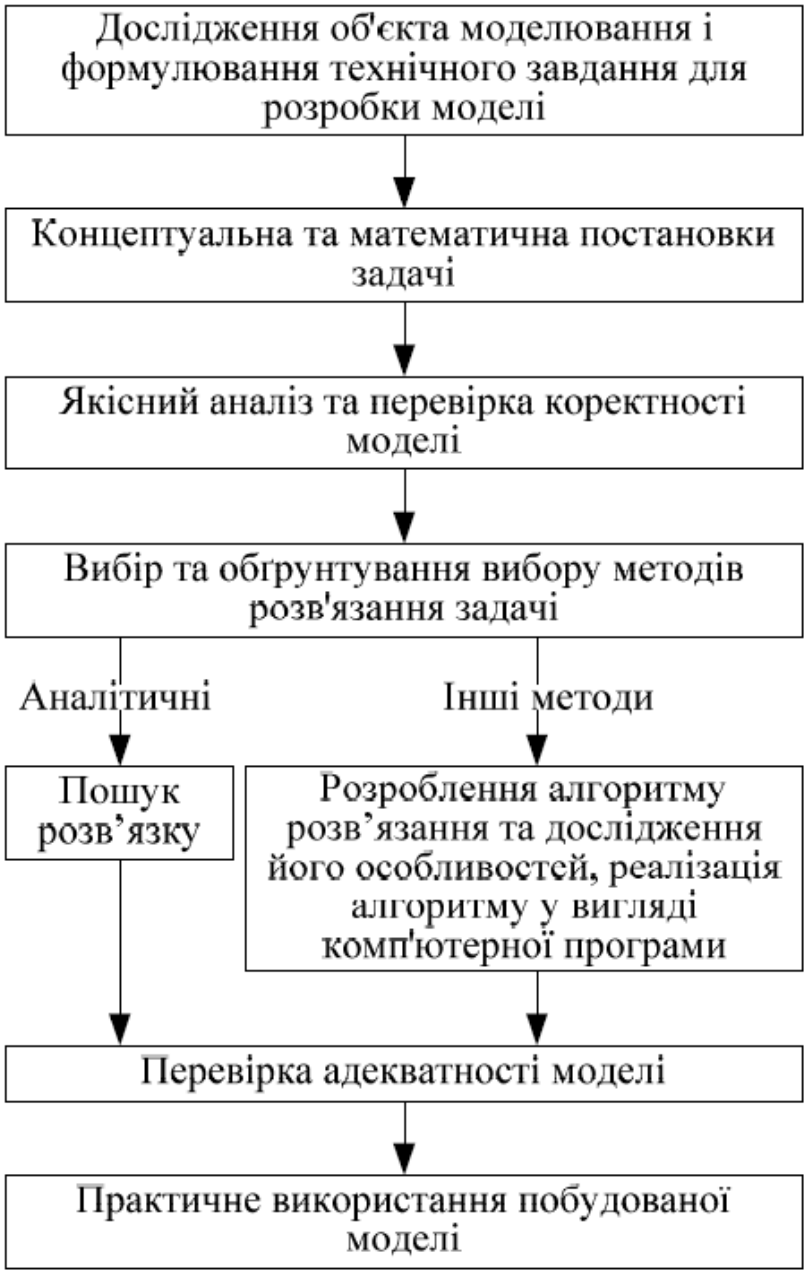
– ***індуктивність*** – вивчення причин і наслідків, від окремого до загального, з метою накопичення необхідних знань.

Висновки

1. Система – це цілісний комплекс взаємопов'язаних елементів, який має певну структуру і взаємодіє із зовнішнім середовищем.

2. Модель – це реально існуюча або уявна система, яка, заміщаючи і відображаючи в пізнавальних процесах іншу систему-оригінал, знаходиться з нею у відношенні подібності.

3. Моделювання – це спосіб дослідження будь-яких явищ, процесів або об'єктів шляхом побудови та аналізу їх моделей” [15, 16].


Дослідження об'єкта моделювання і формулювання технічного завдання для розробки моделі
Концептуальна та математична постановки задачі
Якісний аналіз та перевірка коректності моделі
Вибір та обґрунтування вибору методів розв'язання задачі
Аналітичні
Інші методи
Пошук розв'язку
Розроблення алгоритму розв'язання та дослідження його особливостей, реалізація алгоритму у вигляді комп'ютерної програми
Перевірка адекватності моделі
Практичне використання побудованої моделі

[16]

“На етапі *дослідження об’єкта моделювання* потрібно виконати такі дії:

– аналіз взаємодії об’єкта з зовнішнім середовищем, виділення характеристик вхідних впливів та реакції об’єкта, класифікація їх на вимірні та невимірні, керувальні та перешкоди;

– проведення декомпозиції та дослідження внутрішньої структури об’єкта;

– дослідження порядку функціонування об’єкта, виявлення зв’язку між входом та виходом, формування множини станів об’єкта;

– збір та перевірка існуючих експериментальних даних про об’єкти-аналоги, проведення, за необхідності, додаткових експериментів;

– класифікація об’єкта моделювання на стаціонарний чи нестаціонарний, визначення міри впливу випадкових факторів на об’єкт та порядку нелінійності зв’язків між характеристиками об’єкта;

– аналітичний огляд літературних джерел, аналіз та порівняння побудованих раніше моделей подібних об’єктів;

– аналіз та узагальнення всього накопиченого

На підставі аналізу об’єкта моделювання формується змістовна постановка моделювання, в якій мають бути зазначені:

– мета моделювання;

– тип моделі;

– вимоги до адекватності моделі та якості розв’язку.

Структурні елементи математичних моделей

За структурою математична модель має такі елементи:

1. **Об’єкт моделювання** – це суцільна технічна система або її складові структурні частини (підсистеми).

2. **Постійні параметри** – величини, які в процесі всього моделювання залишаються незмінними.

3. **Змінні параметри** – величини, значення яких потрібно знайти при розв’язуванні задачі за допомогою математичного моделювання.

4. **Критерій оптимальності** – показник міри ефективності досліджуваної технічної системи, величина якого при екстремальному значенні цільової функції (максимальному чи мінімальному) визначає

оптимальний рішення для заданих умов, тобто оптимальне значення змінних параметрів моделі.

5. **Обмеження –** області можливих значень змінних умов за заданих конкретних умов технічної системи, що вивчається і для якої обчислюється найкраще (оптимальне) рішення.

6. **Цільова функція** пов’язує критерій оптимальності зі змінними та постійними параметрами. У процесі створення моделей знаходиться оптимальне рішення та визначаються такі значення змінних параметрів, які надають цільовій функції мінімаксного значення.

Основні види моделювання

математичне (аналітичне), **імітаційне, статистичне**.

Для ***аналітичного (математичного) моделювання*** характерне те, що процеси функціонування елементів системи записуються у вигляді деяких функціональних співвідношень.

Декомпозиція систем та простір станів

Як правило, під час побудови моделі система спрощується, тобто проводиться її декомпозиція (розділення на підсистеми). Якщо систему задати множиною відношень n -го порядку $R[x_1, x_2 \ldots, , x_n]$, то загальний метод декомпозиції можна описати за допомогою операції добутку відношень. Відношення R є добутком відношень R_1 і R_2 , якщо виконується умова

$$(Y\,R\,X) \leftrightarrow [(Y\,R_1\,X) \cap (Y\,R_2\,X)],$$

де X, Y, Z – деякі множини. Завдання дослідника полягає у визначенні відношень R_1 і R_2.

Якщо ці відношення знайдені, то систему можна розглядати як сукупність двох підсистем

$$R_1[x_1, x_2, \ldots, x_j, Z] \text{ і } R_2[Z, x_{j+1}, x_{j+2}, \ldots, x_n],$$

$$x_j \in X,\ j = 1, 2, \ldots, n.$$

У літературі вказано, що відношення n -го порядку можна розкласти на n -2 тривимірних відношень" [16-19].

Лекція 2

- ІНФОРМАЦІЙНІ СИСТЕМИ (ІС). ВИДИ, СТРУКТУРА, ТИПИ ВЗАЄМОДІЇ ІС
- ТИПИ МОДЕЛЕЙ ЖИТТЄВОГО ЦИКЛУ ІС. ОСНОВИ ГНУЧКОГО МОДЕЛЮВАННЯ
- ПРОЦЕСИ ЖИТТЄВОГО ЦИКЛУ ІС

ІНФОРМАЦІЙНА СИСТЕМА

Інформаці́йна систе́ма (англ. *Information system*) — сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів.

За ДСТУ 2392-94:— комунікаційна система, що забезпечує збирання, пошук, оброблення та пересилання інформації.

Закон України «Про захист інформації в інформаційно-телекомунікаційних системах» визначає інформаційну (автоматизовану) систему як організаційно-технічну систему, в якій реалізується технологія обробки інформації з використанням технічних і програмних засобів.

“До інформаційних систем нового покоління належать системи **підтримки прийняття рішень (СППР)** та інформаційні системи, побудовані на штучному інтелекті (інтелектуальні ІС).

СППР – це інтерактивна комп’ютерна система, яка призначена для підтримки різних видів діяльності при прийнятті рішень із слабо структурованих або неструктурованих проблем.

Штучний інтелект – це штучні системи, створені людиною на базі ЕОМ, що імітують розв’язування людиною складних творчих задач.

Системи підтримки прийняття рішень

- інтерактивна комп’ютерна система, призначена для підтримки різних видів діяльності при прийнятті рішень із слабоструктурованих або неструктурованих проблем

Штучний інтелект

- штучні системи, створені людиною на базі ЕОМ, що імітують розв’язування людиною складних творчих задач

Розрізняють три види **інтелектуальних ІС**:

- інтелектуальні **інформаційно-пошукові** системи (системи типу «запитання – відповідь»), які у процесі діалогу забезпечують взаємодію кінцевих користувачів-непрограмістів з базами даних та знань професійними мовами користувачів, близьких до природних;
- **розрахунково-логічні** системи, які дають змогу кінцевим користувачам розв'язувати в режимі діалогу з ЕОМ свої задачі з використанням складних методів і відповідних прикладних програм;
- **експертні системи**, які дають змогу провадити ефективну комп'ютеризацію областей, в яких знання можуть бути подані в експертній описовій формі, але використання математичних моделей утруднене або неможливе.

За **ступенем централізації обробки інформації** – централізовані ІС, децентралізовані ІС, системи колективного використання.

За **ступенем інтеграції функцій** – багаторівневі ІС з інтеграцією за рівнями управління (підприємство – об’єднання, об’єднання – галузь і т.ін.), багаторівневі ІС з інтеграцією за рівнями планування і т.ін.

За **типом** ІС розподіляються на фактографічні, документальні і документально-фактографічні ІС.

Документальна ІС – це система, в якій об’єктом зберігання і обробки є власне документи.

Фактографічна ІС – це система, в якій, об’єктом або сутністю є дещо, що являє для проблемної сфери багатосторонній інтерес

Типи взаємодії інформаційних систем:

- Довільна взаємодія між двома окремими комп'ютерами, наприклад по модему. Обов'язкова участь оператора на приймаючій і передаючій стороні. Можливий обмін в довільному, але заздалегідь обумовленому форматі;
- Інтерактивна віддалена взаємодія комп'ютера з інформаційною системою, наприклад по протоколу http. Оператор на передавальній стороні. Як правило використовується певна форма HTML документа. Прийняті документи обробляються автоматично;
- Інтерактивна інформаційна система – інформаційно-обчислювальна система, в якій передача та обмін інформацією відбуваються в режимі діалогу. Є два типи інтерактивних ігор: 1.

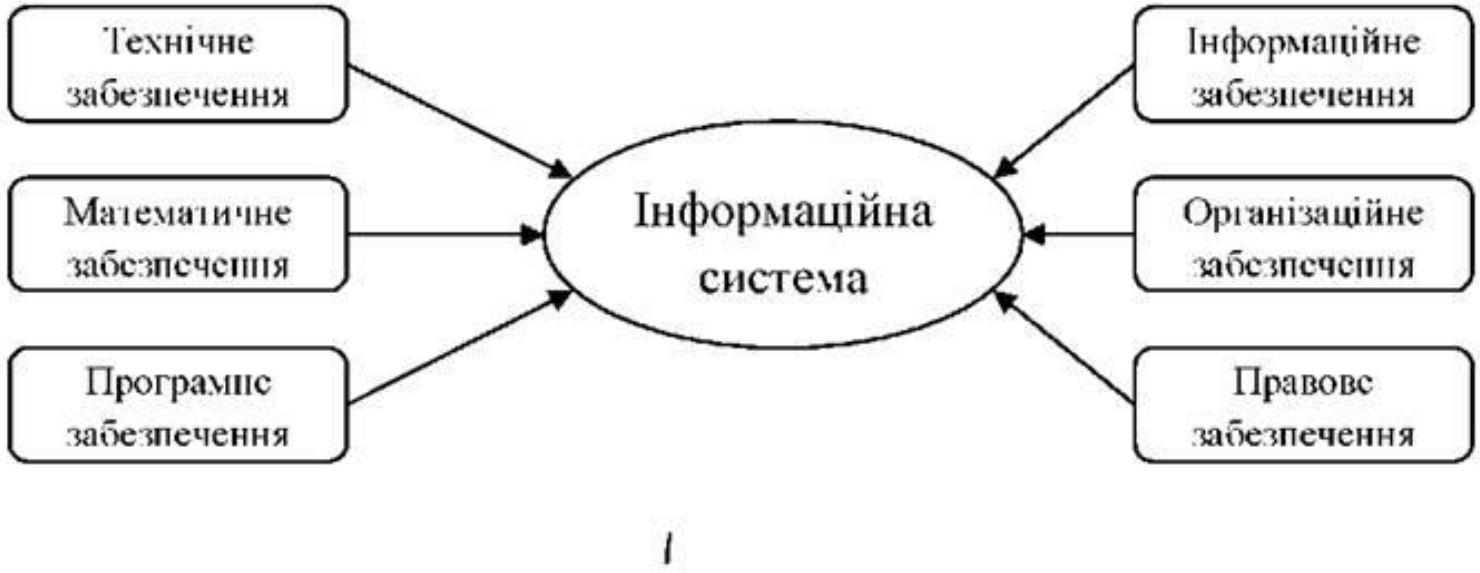
Тип таких ігор призначений для гравців, які активно взаємодіють з ігровою системою, і ця система, у свою чергу, реагує на поведінку гравців. 2. Тип інтерактивних ігор – це ігри, де гравці можуть пасивно спостерігати за сюжетною лінією гри.

- Контрольована потокова обробка, наприклад прийом з e-mail, файл містить HTML форму, запуск якої ініціює процес обробки документа або прийом оператором по e-mail електронних документів в обумовленому форматі і далі запуск програми обробки. Вимагає обов'язкового контролю оператора на приймаючій стороні;

- Повністю автоматизований процес прийому та обробки електронних документів в обумовленому форматі. Участь операторів не потрібно.

Структура інформаційної системи

Структуру інформаційної системи складає сукупність окремих її частин - підсистем. *Підсистема* - це частина системи, яка виділена за певною ознакою. Тому структура будь-якої інформаційної системи може бути представлена як сукупність підсистем, що забезпечують інформаційне, технічне, математичне, програмне, організаційне і правове забезпечення



Інформаційне забезпечення - сукупність єдиної системи класифікації й кодування повідомлень, уніфікованих систем документації, схем інформаційних потоків, що циркулюють в організації, а також методологія побудови баз даних.

Технічне забезпечення - комплекс технічних засобів, призначених для роботи інформаційної системи, а також відповідна документація на ці засоби й технологічні процеси.

Математичне й програмне забезпечення - сукупність математичних методів, моделей, алгоритмів і програм для реалізації цілей і завдань інформаційної системи, а також нормального функціонування комплексу технічних засобів.

Організаційне забезпечення - сукупність методів і засобів, що регламентують взаємодію працівників з технічними засобами й між собою в процесі розробки й експлуатації інформаційної системи.

Правове забезпечення - сукупність правових норм, що визначають створення, юридичний статус і функціонування інформаційних систем, що регламентують порядок одержання, перетворення й використання відомостей.

Життєвий цикл ІС та його структура

Життєвий цикл інформаційної системи - період часу, який починається з моменту прийняття рішення про необхідність створення інформаційної системи і закінчується в момент її повного вилучення з експлуатації.

Методологія проектування інформаційних систем описує процес створення і супроводу систем у вигляді життєвого циклу (ЖЦ) ІС, як **послідовність стадій і процесів**.

Для кожного етапу визначаються склад і послідовність виконуваних робіт, отримані результати, методи і засоби, необхідні для виконання робіт, ролі та відповідальність учасників і т.д.

Такий формальне опис ЖЦ ІС **дозволяє спланувати та організувати процес колективної розробки і забезпечити управління цим процесом.**

Повний життєвий цикл інформаційної системи включає в себе

- стратегічне планування,
- аналіз,
- проектування,
- реалізацію,
-впровадження,
-експлуатацію.

У загальному випадку життєвий цикл можна в свою чергу розбити на ряд **стадій**. У принципі, цей поділ на стадії досить довільно. Ми розглянемо один з варіантів такого поділу, пропонований корпорацією Rational Software - однієї з провідних фірм на ринку програмного забезпечення засобів розробки інформаційних систем (універсальний CASE-засіб Rational Rose).

Стадії життєвого циклу ІС

Стадія - частина процесу створення ІС, обмежена певними часовими рамками і закінчується випуском конкретного продукту (моделей, програмних компонентів, документації), що визначається заданими для даної стадії вимогами.

Співвідношення між процесами і стадіями також визначається використовуваної моделлю життєвого циклу ІС.

Згідно з методологією, пропонованої Rational Software, життєвий цикл інформаційної системи поділяється на чотири стадії.

Межі кожної стадії визначено деякими моментами часу, в які необхідно приймати певні критичні рішення і, отже, досягати певних ключових цілей.

1) Початкова стадія

На початковій стадії встановлюється область застосування системи і визначаються граничні умови. Для цього необхідно ідентифікувати всі зовнішні об'єкти, з якими повинна взаємодіяти розробляється система, і визначити характер цієї взаємодії на високому рівні. На

початковій стадії ідентифікуються всі функціональні можливості системи і проводиться опис найбільш істотних з них.

2) Стадія уточнення

На стадії уточнення проводиться аналіз прикладної області, розробляється архітектурна основа інформаційної системи.

При прийнятті будь-яких рішень, що стосуються архітектури системи, необхідно брати до уваги систему, що розробляється в цілому. Це означає, що необхідно описати більшість функціональних можливостей системи та врахувати взаємозв'язки між окремими її складовими.

У кінці стадії уточнення проводиться аналіз архітектурних рішень і способів усунення головних факторів ризику у проекті.

3) Стадія конструювання

На стадії конструювання розробляється закінчений виріб, готовий до передачі користувачеві. По закінченні цієї стадії визначається працездатність розробленого програмного забезпечення.

4) Стадія передачі в експлуатацію

На стадії передачі в експлуатацію розроблене програмне забезпечення передається користувачам. При експлуатації розробленої системи в реальних умовах часто виникають різного роду проблеми, які вимагають додаткових робіт по внесенню коректив у розроблений продукт. Це, як правило, пов'язано з виявленням помилок і недоробок. У кінці стадії передачі в експлуатацію необхідно визначити, досягнуті цілі розробки чи ні.

Стандарти життєвого циклу ІС

Сучасні мережі розробляються на основі стандартів, що дозволяє забезпечити, по-перше, їх високу ефективність і, по-друге, можливість їх взаємодії між собою.

Серед найбільш відомих стандартів можна виділити наступні:

ГОСТ 34.601-90 - поширюється на автоматизовані системи і встановлює стадії та етапи їх створення. Крім того, в стандарті міститься опис змісту робіт на кожному етапі. Стадії й етапи роботи, закріплені в стандарті, більшою мірою відповідають каскадної моделі життєвого циклу.

ISO / IEC 12207 (International Organization of Standardization / International Electrotechnical Commission) 1995 - стандарт на процеси

і організацію життєвого циклу. Поширюється на всі види замовленого ПЗ. Стандарт не містить опису фаз, стадій та етапів.

Rational Unified Process (RUP) пропонує ітеративну модель розробки, що включає чотири фази: початок, дослідження, побудова та впровадження. Кожна фаза може бути розбита на етапи (ітерації), в результаті яких випускається версія для внутрішнього або зовнішнього використання. Проходження через чотири основні фази називається циклом розробки, кожен цикл завершується генерацією версії системи. Якщо після цього робота над проектом не припиняється, то отриманий продукт продовжує розвиватися і знову мине ті ж фази. Суть роботи в рамках RUP - це створення і супровід моделей на базі UML.

Microsoft Solution Framework (MSF) подібна з RUP, так само включає чотири фази: аналіз, проектування, розробка, стабілізація, є ітераційної, припускає використання об'єктно-орієнтованого моделювання. MSF у порівнянні з RUP більшою мірою орієнтована на розробку бізнес-додатків.

Extreme Programming (XP). Екстремальне програмування (сама нова серед розглянутих методологій) сформувалося в 1996 році. В основі методології командна робота, ефективна комунікація між замовником і виконавцем протягом усього проекту з розробки ІС, а розробка ведеться з використанням послідовно допрацьовуємо прототипів.

Моделі життєвого циклу

Модель життєвого циклу ІС - структура, що визначає послідовність виконання та взаємозв'язку процесів, дій і завдань протягом життєвого циклу. Модель життєвого циклу залежить від специфіки, масштабу і складності проекту та специфіки умов, в яких система створюється і функціонує.

Модель ЖЦ ІС включає в себе:

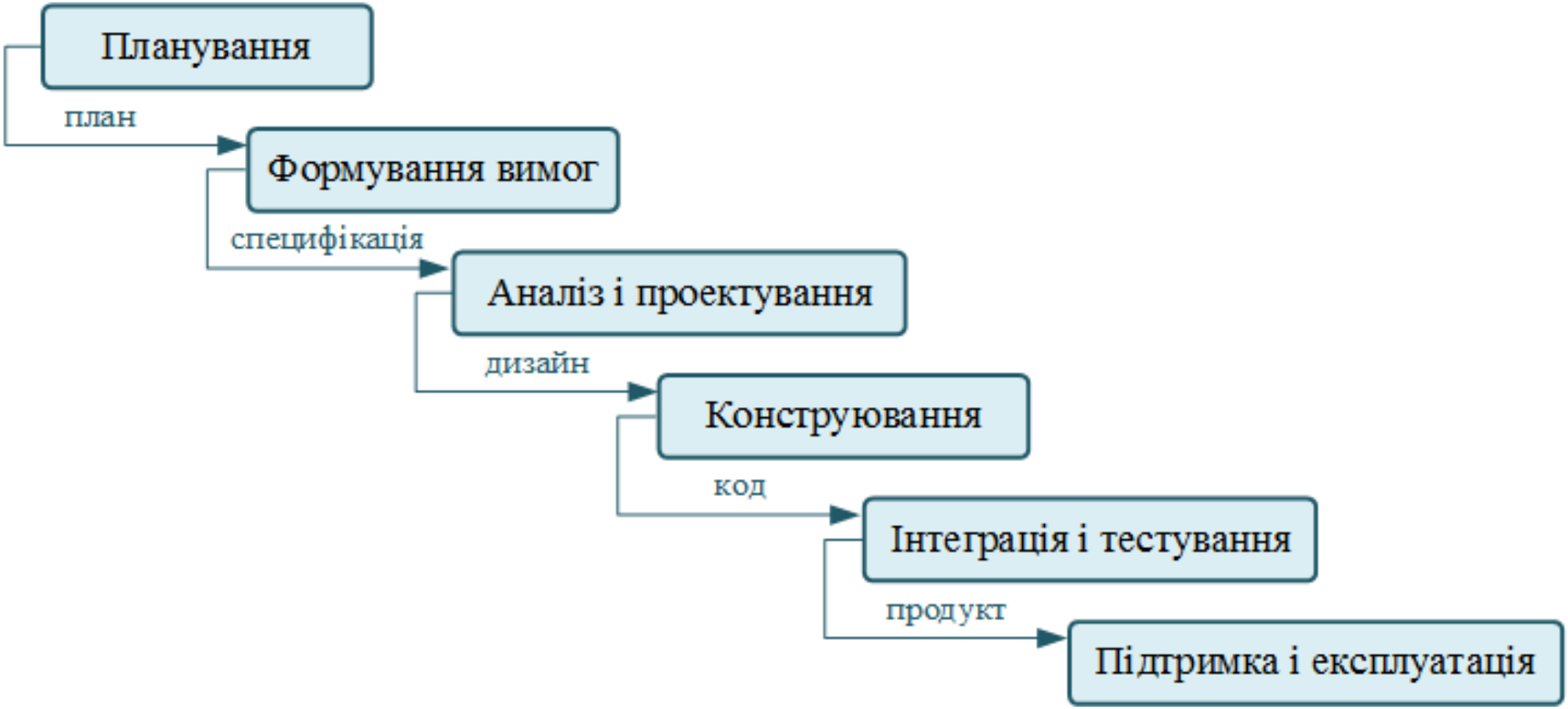
- стадії;
- результати виконання робіт на кожній стадії;
- ключові події - точки завершення робіт і прийняття рішень.

Типи моделей життєвого циклу ІС

В даний час відомі і використовуються наступні моделі життєвого циклу:

Каскадна модель передбачає послідовне виконання всіх етапів проекту в чітко фіксованому порядку. Перехід на наступний етап означає повне завершення робіт на попередньому етапі.

Поетапна модель з проміжним контролем. Розробка ІС ведеться ітераціями з циклами зворотного зв'язку між етапами. Міжетапні коригування дозволяють враховувати реально існуюче взаємовплив результатів розробки на різних етапах; час життя кожного з етапів розтягується на весь період розробки.



Каскадна модель ЖЦ ІС

Переваги та недоліки каскадної моделі

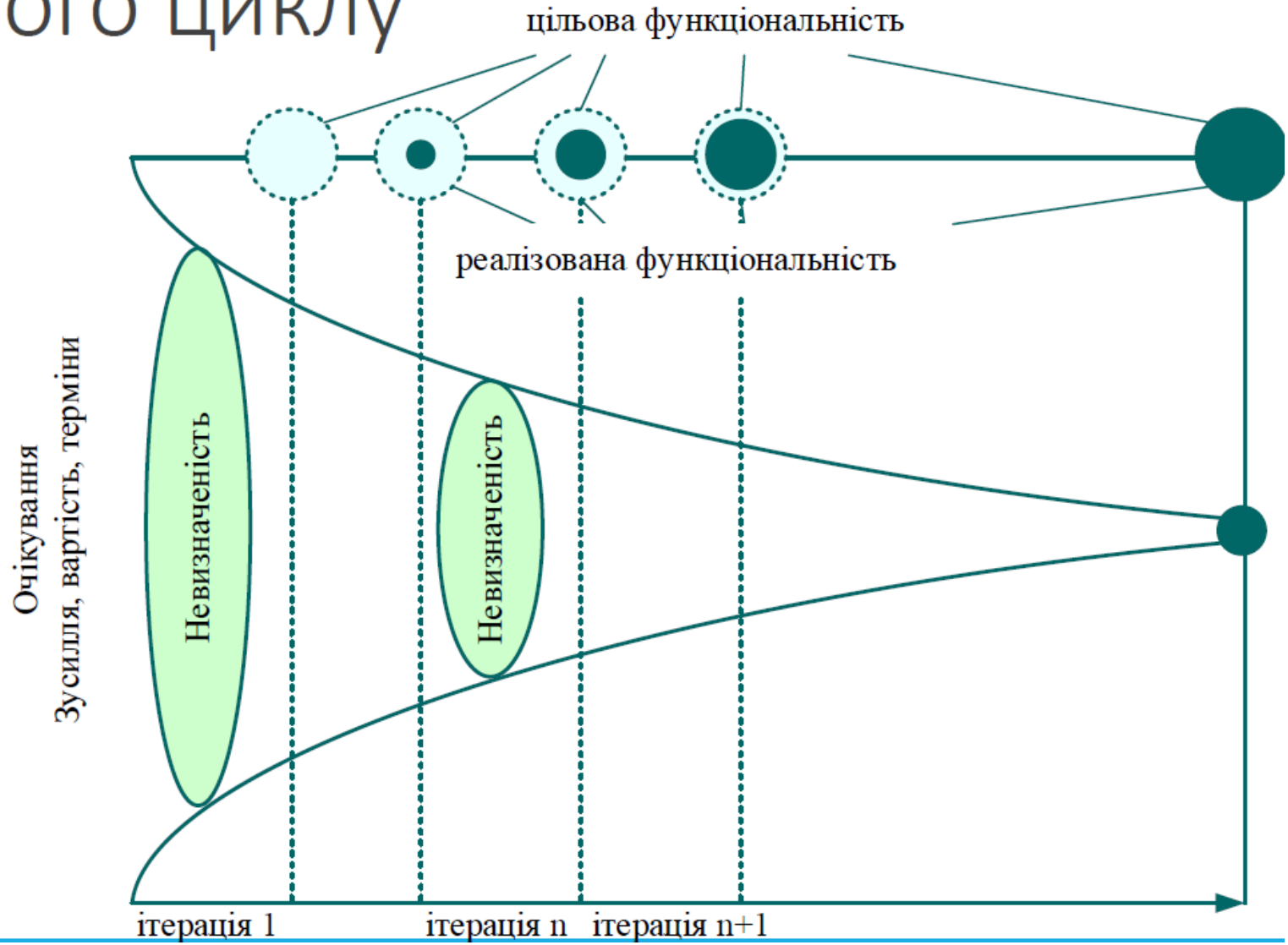
+ стабільність вимог;
+ послідовне усунення складнощів;
+ визначеність кроків моделі;
+ спрощення можливості здійснення планування, контролю та управління проектом;
+ ефективність для проектів з чіткими і зрозумілими, високими вимогами

− повна коректність результату попередньої фази;
− складність формулювання вимог на початку ЖЦ;
− статичність вимог;
− лінійна структура;
− непридатність проміжного продукту для використання

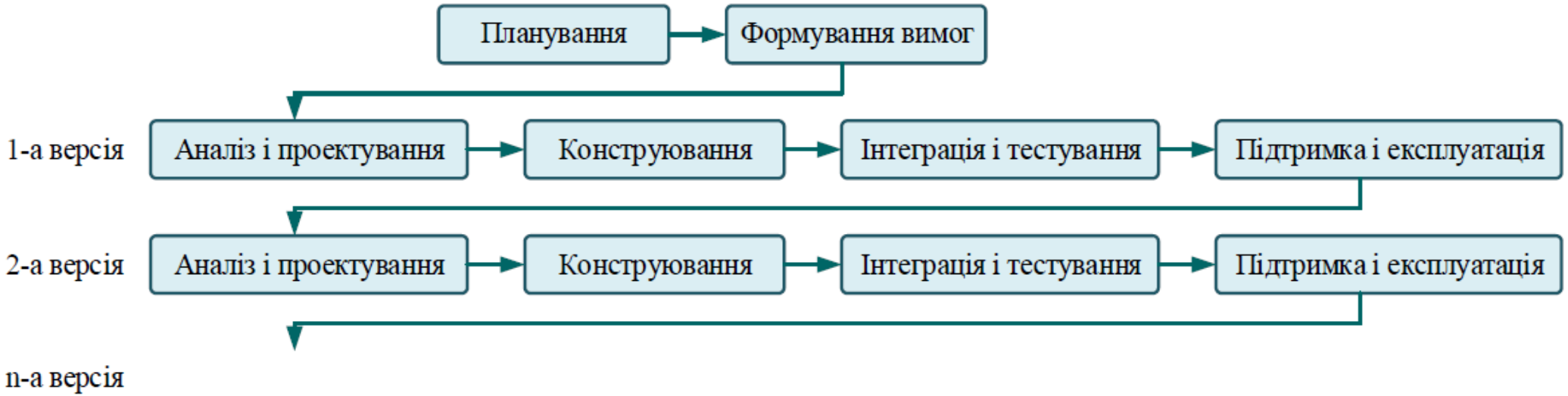
Область застосування каскадної моделі

- ✓ проекти з чіткими, незмінними протягом ЖЦ вимогами;
- ✓ проекти, орієнтовані на побудову системи з прототипами;
- ✓ проекти, пов’язані з модернізацією існуючого продукту або системи;
- ✓ проекти, пов’язані з перенесенням існуючого продукту на нову платформу;
- ✓ при виконанні великих проектів, в яких задіяно декілька великих команд розробників.

Ітеративна й інкрементальна модель життєвого циклу



Ітеративна й інкрементальна модель життєвого циклу



Переваги та недоліки ітеративної й інкрементальної моделі

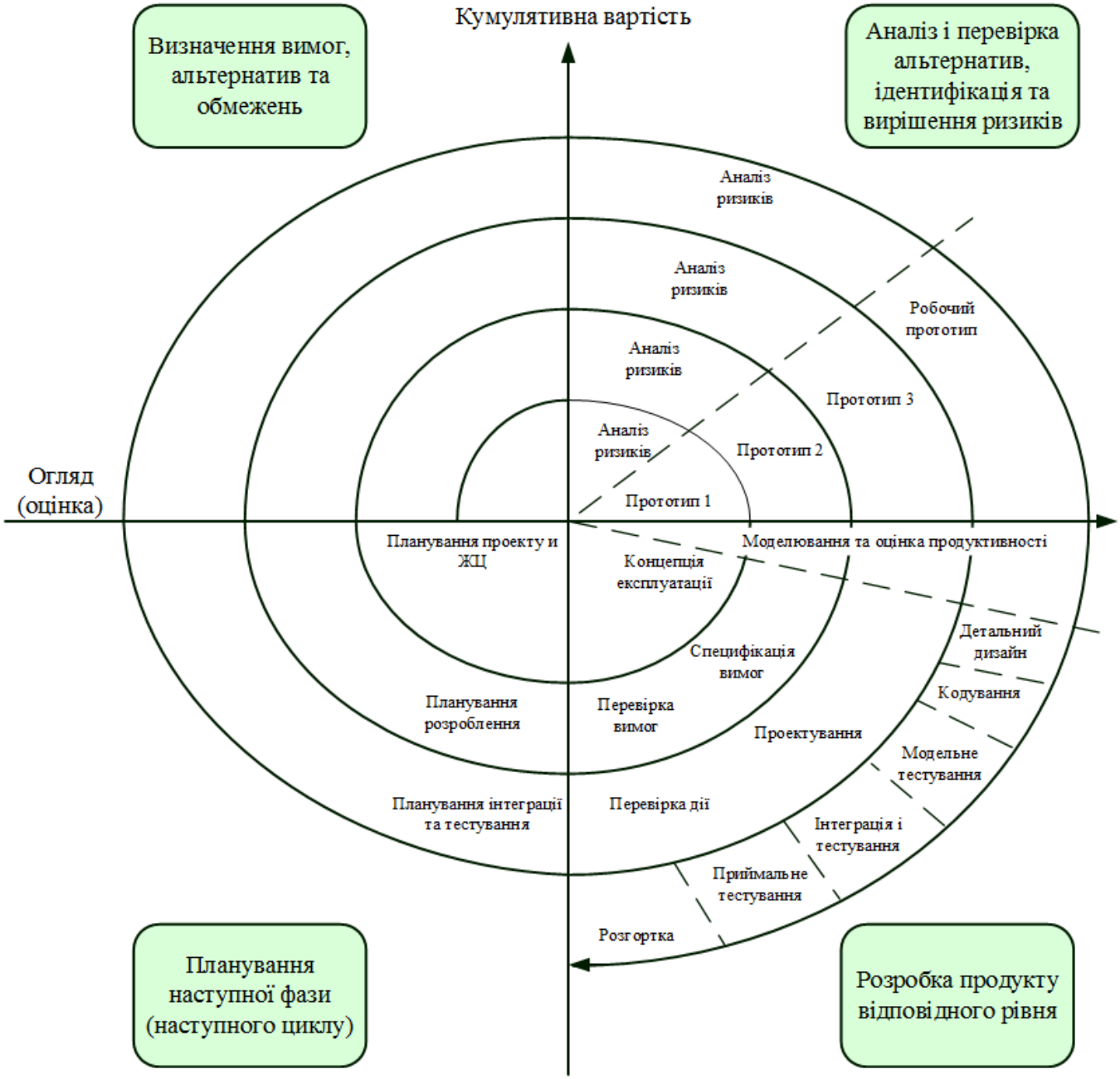
+ зниження ризиків ;
+ організація ефективного зворотного зв’язку ;
+ швидкий випуск мінімально цінного продукту

– проблеми з архітектурою і накладні витрати;
– немає фіксованого бюджету і термінів;
– висока вірогідність зміни начальних вимог

Спіральна модель. На кожному витку спіралі виконується створення чергової версії продукту, уточнюються вимоги проекту, визначається його якість, і плануються роботи наступного витка. Особливу увагу приділяється початковим етапам розробки.

Властивості моделі

- спеціальна увага ризикам, що впливає на організацію життєвого циклу
- пропонує спектр можливостей адаптації вдалих аспектів існуючих моделей процесів життєвого циклу


Кумулятивна вартість
Визначення вимог, альтернатив та обмежень
Аналіз і перевірка альтернатив, ідентифікація та вирішення ризиків
Планування інтеграції та тестування
Розробка продукту відповідного рівня
Огляд (оцінка)
Аналіз ризиків
Аналіз ризиків
Аналіз ризиків
Аналіз ризиків
Робочий прототип
Прототип 3
Прототип 2
Прототип 1
Планування проекту и ЖЦ
Концепція експлуатації
Моделювання та оцінка продуктивності
Специфікація вимог
Планування розроблення
Перевірка вимог
Проектування
Детальний дизайн
Кодування
Модельне тестування
Перевірка дії
Інтеграція і тестування
Приймальне тестування
Розгортка

Переваги та недоліки спіральної моделі

+ швидко готовий працездатний продукт;
+ зміна вимог;
+ гнучкість в управлінні проектом;
+ надійна і стійка система;
+ удосконалення процесу розробки;
+ зменшення ризиків замовника

– невизначеність у розробника в перспективах розвитку проекту;
– ускладнення операції тимчасового і ресурсного планування всього проекту в цілому

Сучасні моделі ЖЦ. Об’єктно-орієнтована модель

Властивості моделі

- конструювання програмного рішення з готових об’єктів, для яких визначаються правила їх взаємодії, що переводять об’єкти з одного стану в інший
- повна відповідність процесу розробки положенням об’єктно-орієнтованої методології

- ✓ великі проекти
- ✓ жорсткий контроль дотримання розробниками встановлених правил

Методики проектування ІС

- Rational Unified Process (RUP);
- Enterprise Unified Process (EUP);
- Microsoft Solutions Framework (MSF);
- Agile-практики (eXtreme Programming (XP), Feature Driven Development (FDD), Dynamic Systems Development Method (DSDM), SCRUM та ін.).

Основи гнучкого моделювання

Для чого потрібна методологія гнучкої розробки?

По-перше, це прискорення виведення продукту на ринок. Якщо необхідно розробити систему швидше, потрібно робити це відповідно до Agile. Дуже простий приклад. Є дві компанії, у них приблизно однаковий бізнес. Одна пише ТЗ, потім проектує систему і малює дизайн – це водоспадна модель, на розробку якої може піти кілька місяців. У другій компанії, що працює по Agile, до цього часу може бути вже запущений сайт, випущено ПЗ, вона почне заробляти гроші і захоплювати ринок, що найголовніше.

По-друге, управління змінами в пріоритетах. Якщо розроблюється проект тривалістю кілька місяців, то у вас обов’язково зміняться вимоги. Якщо говорити про комерційну розробку, то проблема в тому, що ми, програмісти, аналітики та дизайнери, ніколи не знаємо, що потрібно не тільки замовнику, який нам сплачує, але і користувачам. Зазвичай всі підходять до питання так: поки користувач не спробує функціонал сайту або програми, ви не знаєте, чи потрібен він чи ні.

По-третє, поліпшення взаємодії між IT і бізнесом. Це головний біль, особливо для великих компаній, адже у бізнесу періодично змінюються вимоги, кожен говорить своєю мовою. В результаті сторони один одного не розуміють.

Маніфест гнучкої розробки

Цінності.

Маніфест складається з декількох частин. Перша частина називається «Цінності» (Values). Це чотири «зважування»:

- Якщо ви хочете побудувати гнучкий процес, вам потрібно взаємодіяти і спілкуватися між собою. У чому це виражається розглянемо нижче на прикладі Scrum. При цьому ви можете (і обов'язково будете) використовувати якісь інструменти і процеси, наприклад, трекери – JIRA, Redmine і т.д. Але ваша робота повинна спиратися на різні мітинги, зустрічі і взаємодію, а не на налаштування трекерів.

- Працюючий продукт, який ми робимо, набагато важливіше, ніж документація по ньому. Вище було наведено приклад з двома компаніями: у однієї є готовий продукт, який можна дати користувачам, замовнику, захопивши ринок; а друга пише ТЗ, малює макети і т. д. Вся ця документація, яку користувач не може застосувати по причини не готовності продукту, не приносить цінності цьому користувачеві. Якщо ми навчимося працювати, мінімізуючи ці кроки, або роблячи їх невеликими шматочками, то у нас вийде більш гнучкий процес.
- Співпраця та взаємодія з замовником важливіше жорстких контрактних обмежень. Зазвичай підписується договір, в якому зазначено, що до конкретної дати за певну суму розробник

зобов’язується виконати обумовлений обсяг робіт. Природно, до договору прикладається ТЗ. Тобто фіксується час, обсяг робіт і терміни. Це називається Fixed Price. Такий підхід не дуже хороший, якщо ви хочете працювати на довгострокову перспективу і бути гнучкими. У цьому випадку правильніше вибудовувати партнерські відносини з замовником. Якщо говорити про контрактні оформлення, то зазвичай це виливається в контракти за схемою «час – матеріали», коли розробнику просто оплачується витрачений час. Найголовніше, що тут починається пошук партнерства і ситуації Win-Win, коли перемагає і замовник, і його підрядник.

- Готовність до змін в зважуванні зі проходженням первинним планом.

Таким чином, в Agile існує чотири цінності:

- люди і взаємодія між ними;
- робочий продукт;
- співпраця і вибудовування партнерських відносин із замовником;
- готовність до змін.

Принципи Agile.

Цінності зумовлюють 12 принципів Agile:

1. *Найвища цінність* – це задоволення потреб замовника завдяки регулярній і максимально ранній поставці цінного для нього ПЗ. Якщо замовник хоче отримати від нас великого слона, але ми можемо дати йому частину цього слона не через рік, а через три місяці, потім ще через три місяці ще одну частину, а після щомісяця видавати шматочки, то чим частіше ми це будемо робити і чим раніше, тим краще.
2. Ми завжди *готові змінювати вимоги*, навіть на пізніх стадіях проекту, якщо дізнаємося щось нове. Таким чином, ми

створюємо бізнесу або зовнішньому замовнику конкурентну перевагу. Припустимо, працюють дві компанії: один написала ТЗ і за рік зробила продукт, а ми зробили концепцію продукту (неважливо, в якому вигляді) і поступово його викочуємо і розгортаємо. Тоді наш продукт буде більше відповідати вимогам замовників, користувачів і ринку в цілому.

3. При використанні Agile *працюючий продукт випускають максимально часто*. У маніфесті прописані терміни – від кількох тижнів до кількох місяців. Насправді це тиждень / місяць, якщо ви використовуєте Scrum. А якщо роблять якийсь веб-проект, то зазвичай використовують одну з варіацій Kanban, значить, релізи можна робити щодня. У HeadHunter зазвичай щодня виходить

кілька релізів, що створює великі проблеми для наших конкурентів. Це можуть бути правки багів, але ми вміємо додавати щось цінне для наших користувачів.

4. *Бізнес обов'язково повинен працювати разом з програмістами*, допомагати їм зрозуміти специфіку даного ринку. Найбільш частою проблемою, з мого досвіду, є недоступність бізнесу – коли розробник не може отримати у співробітника потрібну інформацію. При використанні Agile важливо уникати виникнення подібних ситуацій.

5. *Команда – один з наріжних каменів* Agile. Найкращих результатів досягає команда замотивованих професіоналів. У Agile керівник передусім має створювати умови для команди і

забезпечувати всебічну підтримку, проводити коучинг, стежити за атмосферою в колективі.

6. Є багато досліджень, які показують, що *краще спілкування – лицем до лиця*. Причому бажано, щоб було якесь засіб візуалізації, на якому можна писати: аркуш паперу, дошка зі стікерами і т. д. Найпростіший і ефективний спосіб дізнатися вимоги клієнта, замовника або користувача – поговорити з ними.

7. *Працюючий продукт*. Ступінь готовності проекту повинна вимірюватися не словами, про те, що ТЗ вже написано і 50% макетів намальовано, а кількістю функціоналу, випущеного в production.

8. У Agile важливий *ритм, постійні поліпшення*. Бізнес і програмісти завжди повинні мати можливість робити процес стійким, постійно його покращувати.

9. У вас повинна бути хороша *гнучка архітектура*, в яку можна додавати різні елементи і при необхідності легко їх змінювати. І якщо команда не приділятиме максимум уваги технічною якістю (писати хороший код, використовувати інженерні практики, автоматизувати процеси), то ніякого Agile у вас не буде.

10. *Простота*. Вона проявляється в технічній складовій, в дизайні. Це один з принципів екстремального програмування. Простота дуже важлива також з точки зору випуску продукту: коли

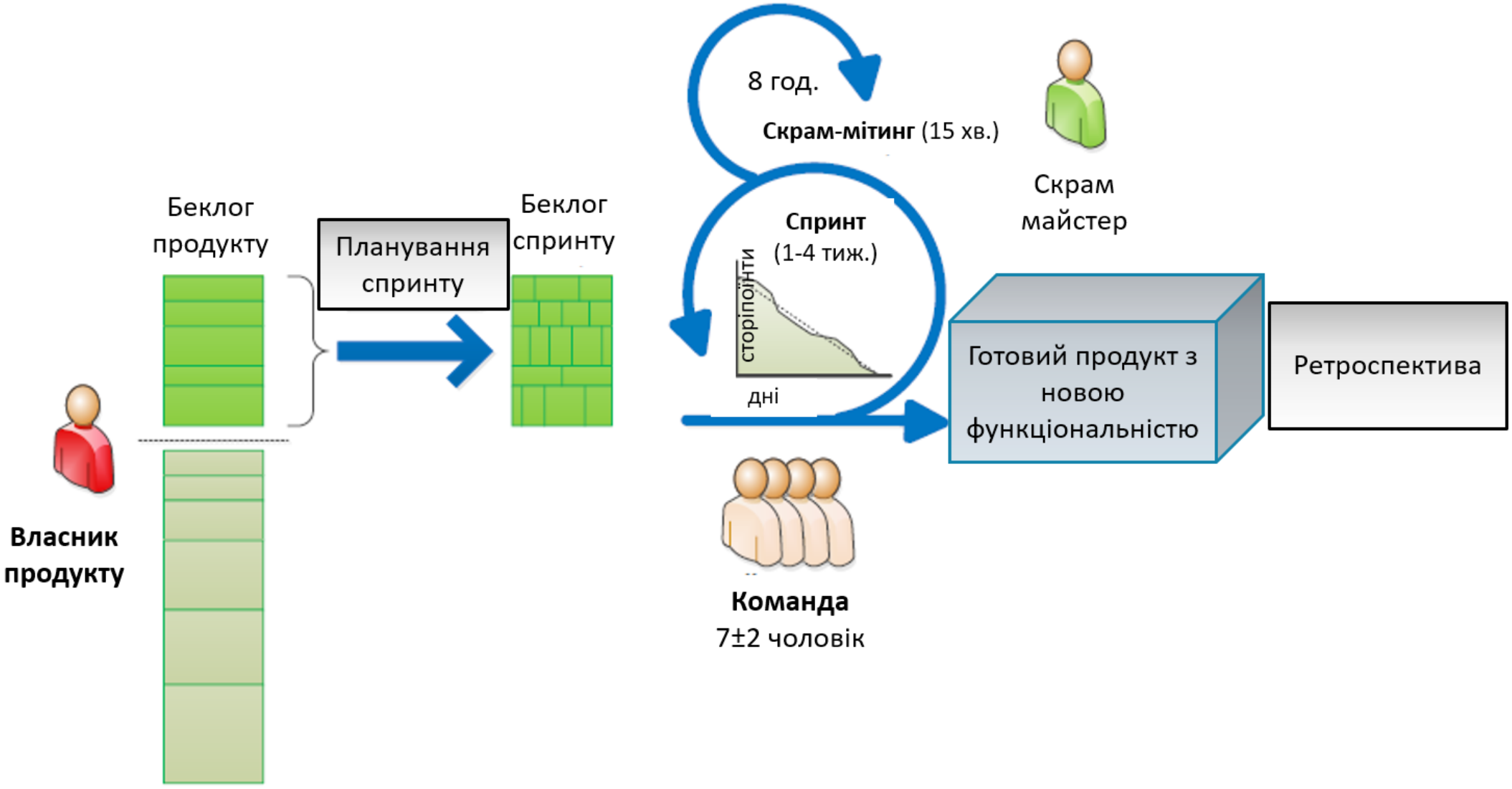
ви хочете «нарізати» того «слона», краще почати з простої частини.

11. *Менеджер* (керівник, Scrum-коуч, Agile-коуч) в команді змінює свою роль: він не стільки займається організацією процесу, скільки *вчить команду*, тому *команда повинна бути самоорганізованою*. Команда повинна *постійно аналізувати свою роботу*, процеси: що вийшло, як вони цього добилися, і постійно покращувати організацію робіт.

Отже, у нас виходить піраміда, що складається з чотирьох цінностей, на яких збудовано 12 принципів. Тепер з’являються конкретні практики.

Основи Scrum

Найбільш розвинені представники гнучкої методології – Scrum, екстремальне програмування (XP), DSDM, Crystal, FDD, Kanban. Більше половини компаній, що застосовують Agile, використовують Scrum. На другому місці комбінація Scrum і XP, коли беруться управлінські практики з Scrum і додаються інженерні. Комбінація Scrum і Kanban використовується приблизно в 8% компаній. Зараз це один із трендів, мода. Назва Scrum прийшла з регбі і перекладається як «сутичка». До розробки продукції цей термін вперше застосували в 1980-х роках два японці – Хіротака Такеуті і Ікудзіро Нонака. Вони проаналізували, як різні компанії створюють свої продукти.


8 год.
Скрам-мітинг (15 хв.)
Скрам майстер
Беклог продукту
Планування спринту
Беклог спринту
Спринт (1-4 тиж.)
сторіпоінти
дні
Готовий продукт з новою функціональністю
Ретроспектива
Власник продукту
Команда
7±2 чоловік

Середньостатистична команда складається з семи осіб (плюс-мінус дві особи). Якщо набагато менше, то, швидше за все, в команді не вистачає якихось фахівців, тому що Scrum-команда може самостійно зробити з беклога готовий продукт з новою функціональністю.

Ролі	Артефакти	Процеси
• Власник продукту • Скрам-майстер • Команда розробки • Команда	• Беклог продукту • Беклог спринту • Інкремент продукту	• Планування спринту • Огляд спринту • Ретроспектива • Скрам-мітинг • Спринт

Власник продукту, product manager – людина, відповідальна за максимізацію цінності продукту, він відповідає за продукт.

Команда розробки повинна складатися з мотивованих професіоналів, які можуть протягом кожного спринту робити поставку, тобто на підставі вимог створювати готовий продукт.

Процеси. Щоденний скрам – це зустріч, на якій члени команди розповідають, що зроблено, з якими проблемами зіткнулися, що планується зробити найближчим часом, щоб кожен розумів, хто і що робить.

Спринт – ітерація з фіксованим терміном, тобто в Scrum повинен бути ритм.

Планування спринту. Перед початком спринту всі збираються і визначають, що потрібно зробити протягом спринту і в якому вигляді.

Огляд спринту або демонстрація: всі збираються і демонструють, що нового в інкременті продукту, дивляться, фіксують зауваження, може бути, міняють найближчі плани.

Ретроспектива: всі збираються і думають, що можна поліпшити в наступному спринті. Наприклад, було багато багів, користувачі незадоволені. Давайте спробуємо в наступному спринті використовувати модульне тестування. Пробуємо, на наступній ретроспективі дивимося, допомогло чи ні, вигадуємо щось ще.

Беклог спринту – верхня частина беклогу. Кількість елементів зазвичай визначається швидкістю команди на заході, який називається «планування спринту», і проводиться перед початком самого етапу спринту. Спринт триває 1-4 тижні (найчастіше 2 тижні). Чим швидше і дешевше можна реліз, тим менше тривалість даного етапу.

Щодня команда проводить 15-хвилинний стендап, *Scrum-мітинг*, на якому всі члени команди синхронізують свої дії. Найчастіше ці зустрічі називають просто «скрам».

Після завершення спринту проводиться демонстрація – «*Огляд*», сенс якої полягає в тому, що команда показує зроблену роботу

власнику продукту або комусь ще. Потім проводиться ретроспектива, на якій команда обговорює, що вийшло, а що ні, відбувається розбір робочих процесів, атмосфери в колективі, робляться спроби щось поліпшити. Після цього запускається наступний спринт. У підсумку роботу Scrum-команди можна представити як ланцюжок безлічі спринтів.

Артефакти. Це можуть бути картки на дошці з коротким описом, що собою представляє конкретний функціонал. При цьому зміст картки може являти собою обговорення з власником продукту. Зазвичай це виливається в тікети в трекері, який ви використовуйте.

Беклог продукту – це все тікети, картки або інші вимоги у вигляді елементів беклогу, які є у вашому продукті.

Беклог спринту – найцінніші і пріоритетні вимоги.

Переваги і недоліки моделей життєвого циклу ІС

У ранніх проектах досить простих ІС кожен додаток являло собою єдиний, функціонально та інформаційно незалежний блок. Для розробки такого типу додатків ефективним виявився каскадний спосіб. Кожен етап завершувався після повного виконання та документального оформлення всіх передбачених робіт.

Можна виділити наступні позитивні сторони застосування каскадного підходу:

на кожному етапі формується закінчений набір проектної документації, який відповідає критеріям повноти та узгодженості;

виконувані в логічній послідовності етапи робіт дозволяють планувати терміни завершення всіх робіт і відповідні витрати.

Каскадний підхід добре зарекомендував себе при побудові відносно простих ІС, коли на самому початку розробки можна досить точно і повно сформулювати всі вимоги до системи. Основним недоліком цього підходу є те, що реальний процес створення системи ніколи повністю не вкладається в таку жорстку схему, постійно виникає потреба в поверненні до попередніх етапах і уточнення або перегляд раніше прийнятих рішень. У результаті реальний процес створення ІС виявляється відповідним поетапної моделі з проміжним контролем.

Спіральна модель ЖЦ була запропонована для подолання перелічених проблем. На етапах аналізу і проектування реалізація технічних рішень і ступінь задоволення потреб замовника

перевіряється шляхом створення прототипів. Кожен виток спіралі відповідає створенню працездатного фрагмента або версії системи. Це дозволяє уточнити вимоги, цілі і характеристики проекту, визначити якість розробки, спланувати роботи наступного витка спіралі. Таким чином поглиблюються і послідовно конкретизуються деталі проекту і в результаті вибирається обгрунтований варіант, який задовольняє дійсним вимогам замовника та доводиться до реалізації.

Основна проблема спірального циклу - визначення моменту переходу на наступний етап. Для її рішення вводяться тимчасові обмеження на кожен з етапів життєвого циклу, і перехід здійснюється відповідно до плану, навіть якщо не вся запланована робота

закінчена. Планування виробляється на основі статистичних даних, отриманих у попередніх проектах, та особистого досвіду розробників.

Незважаючи на наполегливі рекомендації експертів у галузі проектування і розробки ІС, багато компаній продовжують використовувати каскадну модель замість будь-якого варіанту ітераційної моделі. Основні причини, по яких каскадна модель зберігає свою популярність, наступні:

-Звичка - багато ІТ-фахівці здобували освіту в той час, коли вивчалася тільки каскадна модель, тому вона використовується ними і в наші дні.

-Ілюзія зниження ризиків учасників проекту (замовника і виконавця). Каскадна модель передбачає розробку закінчених продуктів на кожному етапі: технічного завдання, технічного проекту, програмного продукту і для користувача документації. Розроблена документація дозволяє не тільки визначити вимоги до продукту наступного етапу, а й визначити обов'язки сторін, обсяг робіт і терміни, при цьому остаточна оцінка термінів і вартості проекту проводиться на початкових етапах, після завершення обстеження. Очевидно, що якщо вимоги до інформаційної системи змінюються в ході реалізації проекту, а якість документів виявляється невисоким (вимоги неповні і / або суперечливі), то насправді використання каскадної моделі

створює лише ілюзію визначеності і на ділі збільшує ризики, зменшуючи лише відповідальність учасників проекту.

Проблеми впровадження при використанні ітераційної моделі. У деяких областях спіральна модель не може застосовуватися, оскільки неможливо використання / тестування продукту, що володіє неповною функціональністю (наприклад, військові розробки, атомна енергетика тощо). Поетапне ітераційне впровадження інформаційної системи для бізнесу можливо, але пов'язане з організаційними труднощами (перенесення даних, інтеграція систем, зміна бізнес-процесів, облікової політики, навчання користувачів). Трудовитрати при поетапному ітераційному впровадженні виявляються значно вище.

Процеси життєвого циклу ІС

Процес визначається як сукупність взаємопов'язаних дій, що перетворюють вхідні дані у вихідні. Опис кожного процесу включає в себе перелік вирішуваних завдань, вихідних даних і результатів.

Відповідно до базовим міжнародним стандартом ISO / IEC 12207 всі процеси ЖЦ ПЗ діляться на три групи:

Основні процеси життєвого циклу

Придбання (дії і завдання замовника, що здобуває ІС)

Постачання (дії і завдання постачальника, який постачає замовника програмним продуктом або послугою)

Розробка (дії і завдання, що виконуються розробником: створення ПЗ, оформлення проектної та експлуатаційної документації, підготовка тестових та навчальних матеріалів і т. д.)

Експлуатація (дії і завдання оператора - організації, що експлуатує систему)

Супровід (дії і завдання, що виконуються супроводжує організацією, тобто службою супроводу). Супровід - внесень змін в ПЗ з метою виправлення помилок, підвищення продуктивності або адаптації до нових умов роботи або вимогам.

Найважливіші: **розробка, експлуатація і супровід**. характеризується завданнями і методами їх вирішення, вихідними даними, отриманими на попередньому етапі, і результатами.

Розробка

Розробка інформаційної системи включає в себе всі роботи по створенню інформаційного програмного забезпечення і його компонентів відповідно до заданих вимог. Розробка інформаційного програмного забезпечення також включає:

оформлення проектної та експлуатаційної документації;

підготовку матеріалів, необхідних для тестування розроблених програмних продуктів;

розробку матеріалів, необхідних для навчання персоналу.

Розробка є одним з найважливіших процесів життєвого циклу інформаційної системи і, як правило, включає в себе стратегічне планування, аналіз, проектування і реалізацію (програмування).

Експлуатація

Експлуатаційні роботи можна підрозділити на підготовчі та основні.

До підготовчих відносяться:

конфігурування бази даних і робочих місць користувачів;

забезпечення користувачів експлуатаційної документації;

навчання персоналу.

Основні експлуатаційні роботи включають:

безпосередньо експлуатацію;

локалізацію проблем і усунення причин їх виникнення;

модифікацію програмного забезпечення;

підготовку пропозицій щодо вдосконалення системи;

розвиток і модернізацію системи.

Супровід

Служби технічної підтримки грають вельми помітну роль в житті будь-якої корпоративної інформаційної системи. Наявність кваліфікованого технічного обслуговування на етапі експлуатації інформаційної системи є необхідною умовою вирішення поставлених перед нею завдань, причому помилки обслуговуючого персоналу можуть призводити до явним або прихованим фінансових втрат, порівнянними з вартістю самої інформаційної системи.

Основними попередніми діями при підготовці до організації технічного обслуговування інформаційної системи є:

виділення найбільш відповідальних вузлів системи та визначення для них критичності простою (це дозволить виділити найбільш

критичні складові інформаційної системи і оптимізувати розподіл ресурсів для технічного обслуговування);

визначення завдань технічного обслуговування та їх поділ на внутрішні, які вирішуються силами обслуговуючого підрозділу, і зовнішні, можуть бути вирішені спеціалізованими сервісними організаціями (таким чином проводиться чітке визначення кола виконуваних функцій і поділ відповідальності);

проведення аналізу наявних внутрішніх і зовнішніх ресурсів, необхідних для організації технічного обслуговування в рамках описаних завдань і поділу компетенції (основні критерії для аналізу: наявність гарантії на обладнання, стан ремонтного фонду, кваліфікація персоналу);

підготовка плану організації технічного обслуговування, в якому необхідно визначити етапи виконуваних дій, терміни їх виконання, витрати на етапах, відповідальність виконавців.

Допоміжні процеси життєвого циклу

Документування (формалізоване опис інформації, створеної протягом ЖЦ ІС)

Управління конфігурацією (застосування адміністративних і технічних процедур на всьому протязі ЖЦ ІС для визначення стану компонентів ІС, управління її модифікаціями).

Забезпечення якості (забезпечення гарантій того, що ІС і процеси її ЖЦ відповідають заданим вимогам і затверджених планів)

Верифікація (визначення того, що програмні продукти, які є результатами певної дії, повністю задовольняють вимогам або умовам, обумовленим попередніми діями)

Атестація (визначення повноти відповідності заданих вимог і створеної системи їх конкретному функціональному призначенню)

Спільна оцінка (оцінка стану робіт за проектом: контроль планування та управління ресурсами, персоналом, апаратурою, інструментальними засобами)

Аудит (визначення відповідності вимогам, планам і умов договору)

Дозвіл проблем (аналіз і рішення проблем, незалежно від їх походження чи джерела, які виявлені в ході розробки, експлуатації, супроводу або інших процесів)

Організаційні процеси

Управління (дії і завдання, які можуть виконуватися будь-якою стороною, що управляє своїми процесами)

Створення інфраструктури (вибір і супровід технології, стандартів та інструментальних засобів, вибір і встановлення апаратних і програмних засобів, що використовуються для розробки, експлуатації або супроводу ПЗ)

Удосконалення (оцінка, вимірювання, контроль та вдосконалення процесів ЖЦ)

Навчання (початкове навчання і подальше постійне підвищення кваліфікації персоналу)

Управління проектом пов'язано з питаннями планування та організації робіт, створення колективів розробників і контролю за термінами і якістю виконуваних робіт. Технічне і організаційне забезпечення проекту включає:

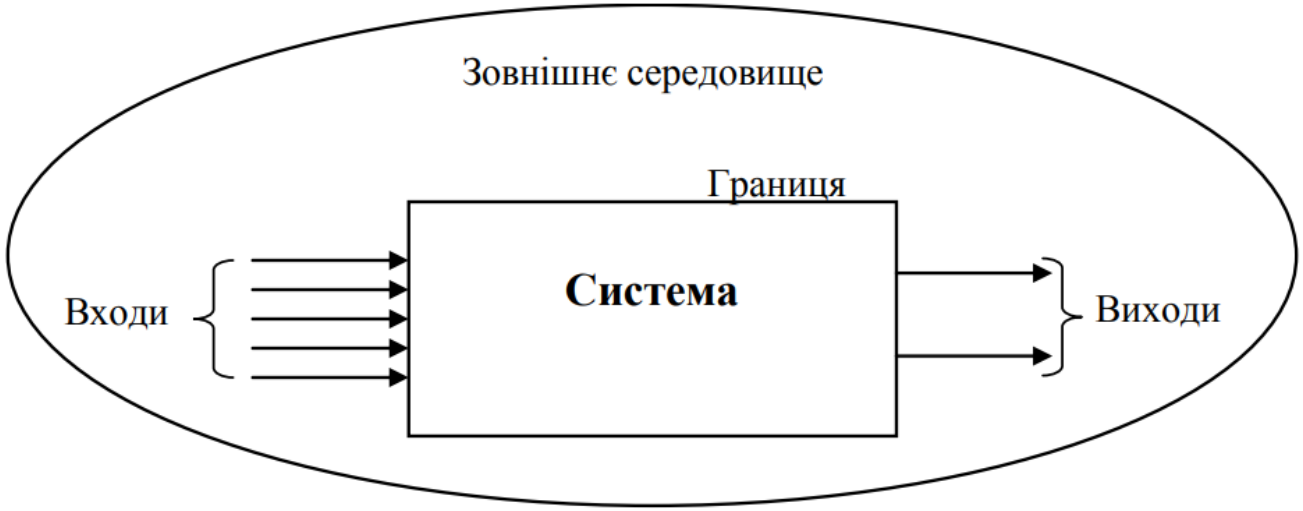
- вибір методів та інструментальних засобів для реалізації проекту;
- визначення методів опису проміжних станів розробки;
- розробку методів і засобів випробувань створеного програмного забезпечення;
- навчання персоналу" [9, 10, 16].

Лекція 3

- ВЛАСТИВОСТІ І КВАЛІФІКАЦІЇ ІНФОРМАЦІЙНИХ СИСТЕМ. МОРФОЛОГІЧНА МОДЕЛЬ ІС
- МАТРИЧНА ФОРМА ОПИСУ МОРФОЛОГІЧНОЇ МОДЕЛІ СИСТЕМИ. ФУНКЦІОНАЛЬНА МОДЕЛЬ СИСТЕМИ
- ПОНЯТТЯ АРХІТЕКТУРИ ІНФОРМАЦІЙНИХ СИСТЕМ. ТИПИ АРХІТЕКТУРИ

Морфологічна модель інформаційної системи. Матрична форма опису морфологічної моделі системи.

“Морфологічна модель системи являє собою сукупність моделей "границі системи"; "зовнішнє середовище системи"; "входи системи"; "виходи системи"; "склад системи"; "структура системи". Сукупність моделей границя, зовнішнє середовище, входи і виходи системи називають моделлю "чорна скриня"



Матрична форма опису структури системи базується на методології графоаналітичного представлення складної системи. Основою цього представлення є гіперкомплексна матриця. Під гіперкоплексністю розуміється властивість системи, яка полягає в тім, що її субстратний склад представляє набір різнорідних і різнофункціональних елементів, що мають ієрархічну структуру. При описі системи за допомогою гіперкомплексної матриці виконують наступну по-слідовність процедур.

1. Визначається гіперкомплексність досліджуваної системи, тобто її ієрархічна структура: число ієрархічних рівнів і число елементів на кожному рівні.

2. Встановлюється наявність і напрямок взаємозв'язків між рівнями й елементами системи на кожному рівні.

3. Формується вихідна матриця, умовна довжина сторін якої визначається гіперкомплексністю досліджуваної системи.

4. Сторона квадрата (матриці) розбивається на частини, кількість яких залежить від кількості елементів на вищому ієрархічному рівні системи.

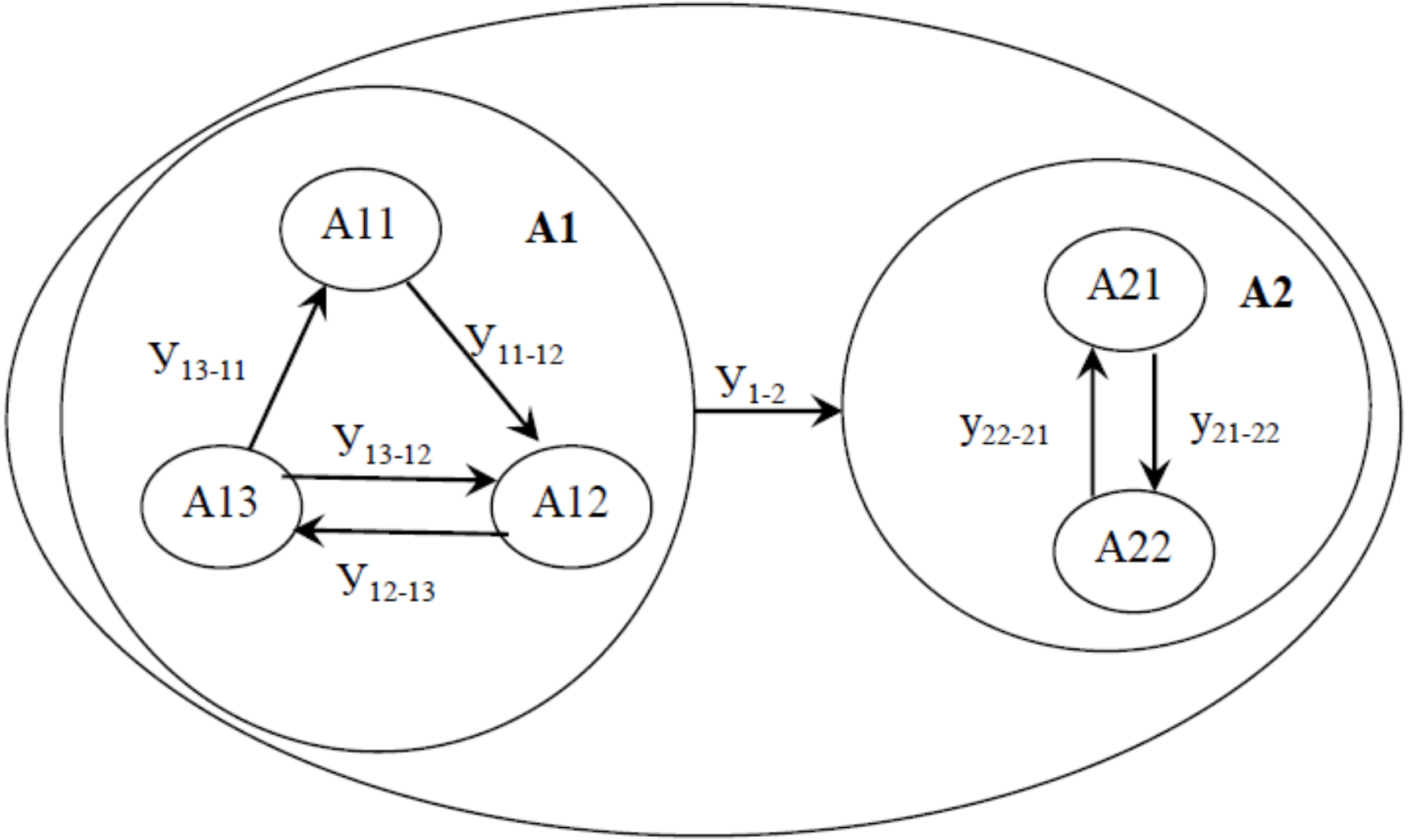
5. Отримані квадрати на головній діагоналі вихідної матриці також розбиваються на частини, що відповідають кількості елементів на наступному рівні. Процес ієрархічної розбивки продовжується доти, поки всі рівні будуть враховані. У такий спосіб на головній діагоналі матриці представлена ієрархічна модель складу розглянутої системи.

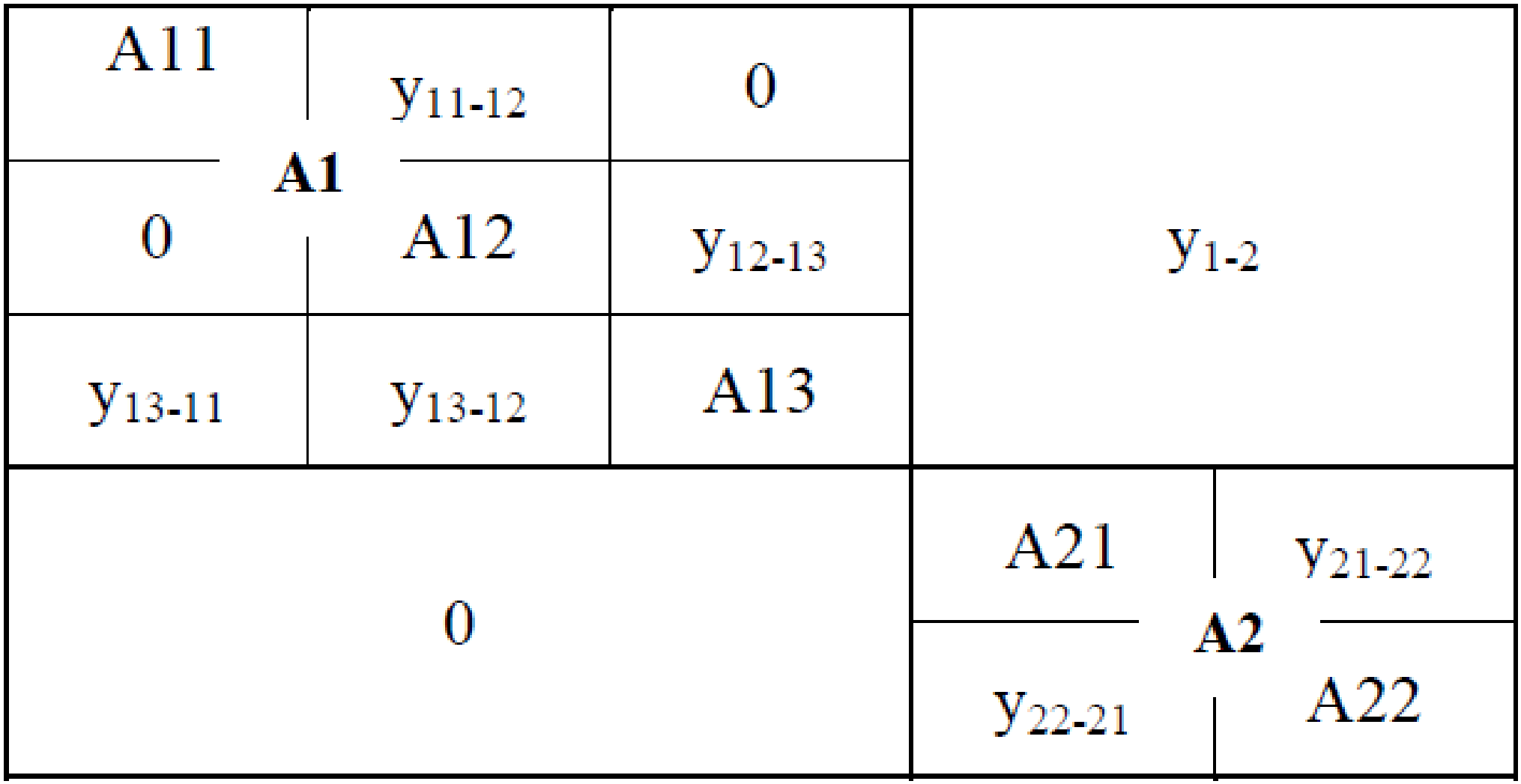
6. Наступною процедурою є формування моделі структури системи шляхом установлення формалізованих відносин між елементами одного рівня і відносинами між рівнями. Ця модель відображається зв'язками між елементами і рівнями за схемою моделі "чорного ящика", де для кожного елемента системи відображаються входи і виходи. Входи і виходи системи - це фізичні або інформаційні потоки, які система і її підсистеми перетворюють у відповідності зі своїм призначенням.

Методика. Зв'язки показуються тільки між елементами одного рівня. Верхня відносно головної діагоналі частина матриці відображає зв'язки між і-м елементом даного рівня і (і+1)-м елементом того ж рівня

(i = 1, 2,..., n) за правилом: вихід з і-го елемента є входом у (i+1)-й елемент. Приклад: зв'язок між елементами 1-го рівня А1 і А2 описується моделлю y_{1-2}, а 2-го рівня А11 і А12 - моделлю y_{11-12}.

Нижня щодо головної діагоналі частина матриці показує зв'язки між (i+1)-м елементом розглянутого рівня й і-м елементом того ж рівня за правилом: вихід з (i+1)-го елемента є входом в і-й елемент. Приклад: структура відносин між елементами 2-го рівня А11 і А13 описується моделлю y_{13-11}. Якщо в визначеному напрямку відношення між елементами відсутнє, то ставиться 0.


A11
A1
A21
A2
y_{13-11}
y_{11-12}
y_{1-2}
y_{13-12}
y_{22-21}
y_{21-22}
A13
A12
A22
y_{12-13}


A11
y11-12
0
A1
0
A12
y12-13
y1-2
y13-11
y13-12
A13
0
A21
y21-22
A2
y22-21
A22

Моделювання системи за допомогою гіперкомплексної матриці має бага-то переваг у порівнянні з традиційним моделюванням системи. Головні з них – це наочність, висока інформаційна насиченість і можливість машинної реаліза-ції моделі без втрати сутності об'єкта, що підлягає моделюванню.

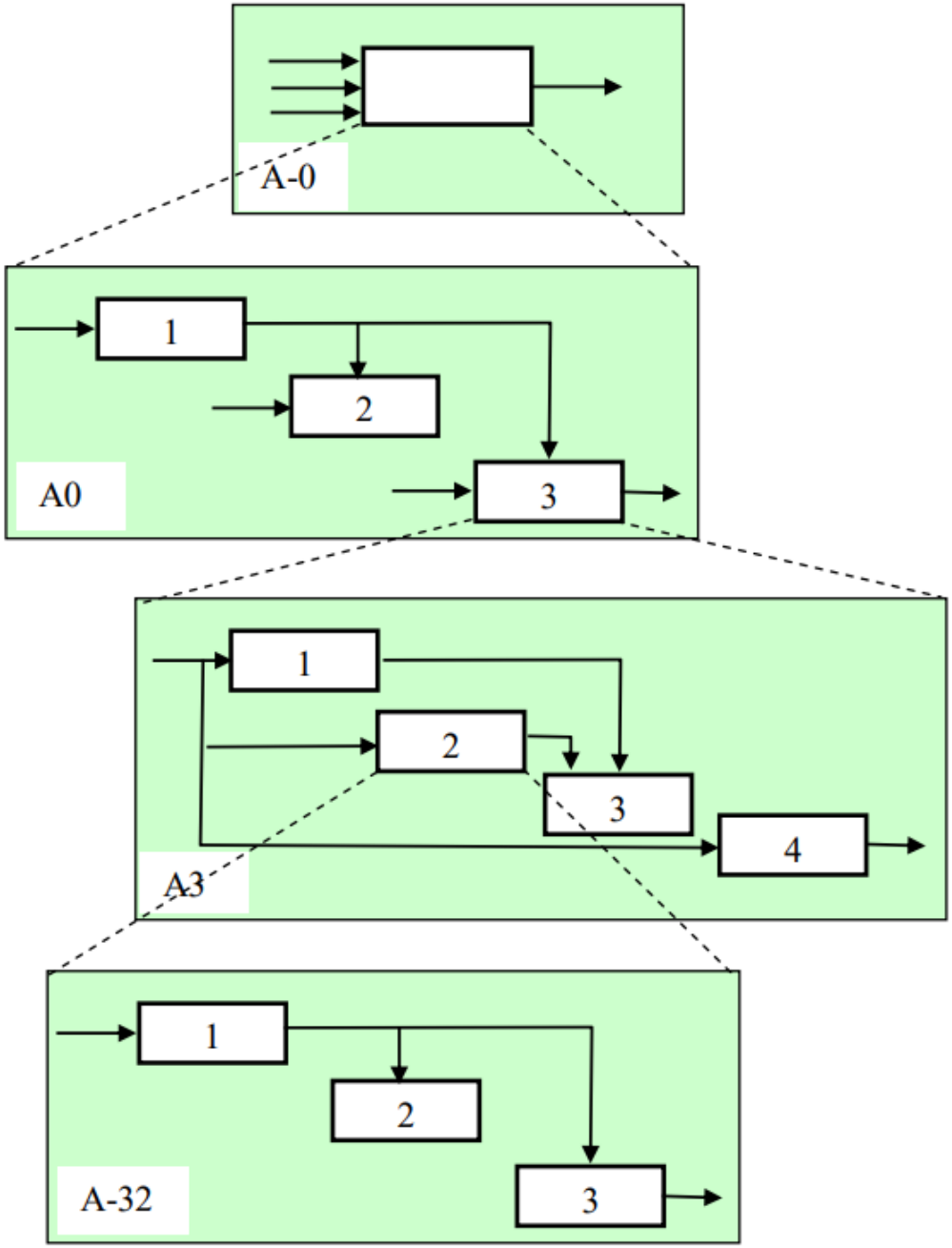
ФУНКЦІОНАЛЬНА МОДЕЛЬ СИСТЕМИ

Функціональна модель – це структурне зображенням функцій системи, а також інформації й об'єктів, які пов'язують ці функції. При побудові функціональних моделей систем широко застосовується IDEF - методологія (IDEF – Icam Definition). З позиції IDEF –методології модель може акцентувати на функціях системи або на об’єктах системи. IDEF –модель орієнтована на функції системи називається функціональною моделлю, а модель яка орієнтована на об’єкти системи називається моделлю даних або інформаційною моделлю. IDEF - методологія вимагає точного вирішення двох важливих питань: перше - визначення границь системи, друге – мети створення (дослідження) системи. Інакше кажучи, IDEF вимагає, щоб модель

розглядалася увесь час з однієї і тієї ж позиції - мети створення системи. Ця позиція називається "точкою зору" даної моделі. Після визначення мети і точки зору моделі починається перша ітерація процесу побудови моделі по методології IDEF. Системний аналітик визначає, що включати в модель, а що ні. Точка зору диктує авторові моделі вибір потрібної інформації і спосіб її подачі. Мета стає критерієм закінчення моделювання. Кінцевим результатом цього процесу є набір ретельно взаємодіючих описів, починаючи з опису самого верхнього рівня всієї системи і кінчаючи докладним описом деталей або операцій системи. Кожний з таких ретельно погоджених описів називається діаграмою. IDEF - модель поєднує й організує

діаграми в ієрархічні структури, в яких діаграми верхнього рівня менш деталізовані, ніж діаграми нижнього рівня.

Кожна IDEF - діаграма містить блоки і дуги. Блоки зображують функції системи. Дуги зв'язують блоки разом і відображають взаємодію і взаємозв'язки між ними. Кожна діаграма має назву, що розташовується в нижній частині блоку. Функціональні блоки на діаграмах зображуються прямокутниками. Блок представляє функцію або активну частину системи, тому назвами блоків служать дієслова або дієслівні обороти. Наприклад: "розрахувати собівартість", "оцінити якість", "визначити кількість працівників" тощо.


A-0
1
2
3
A0
1
2
3
4
A3
1
2
3
A-32

В IDEF-методології кожна сторона блоку має особливе, цілком визначене призначення. Ліва сторона блоків призначена для входів, верхня - для керування, права - для виходів і нижня - для механізмів. Таке позначення відбиває визначені системні принципи: входи перетворюються даною функцією у виходи; керування обмежує або пропонує умову виконання перетворення; механізми показують, хто, що і як виконує функції. Блоки IDEF розміщаються на діаграмі по ступені важливості (у розумінні автора моделі). Такий порядок розташування блоків в IDEF називають домінуванням. Домінування розуміється як вплив одного блоку на інші блоки діаграми. Найбільш домінуючий блок розташовується у верхньому лівому куті діаграми, а найменш домінуючий - у правому нижньому куті. У результаті виходить

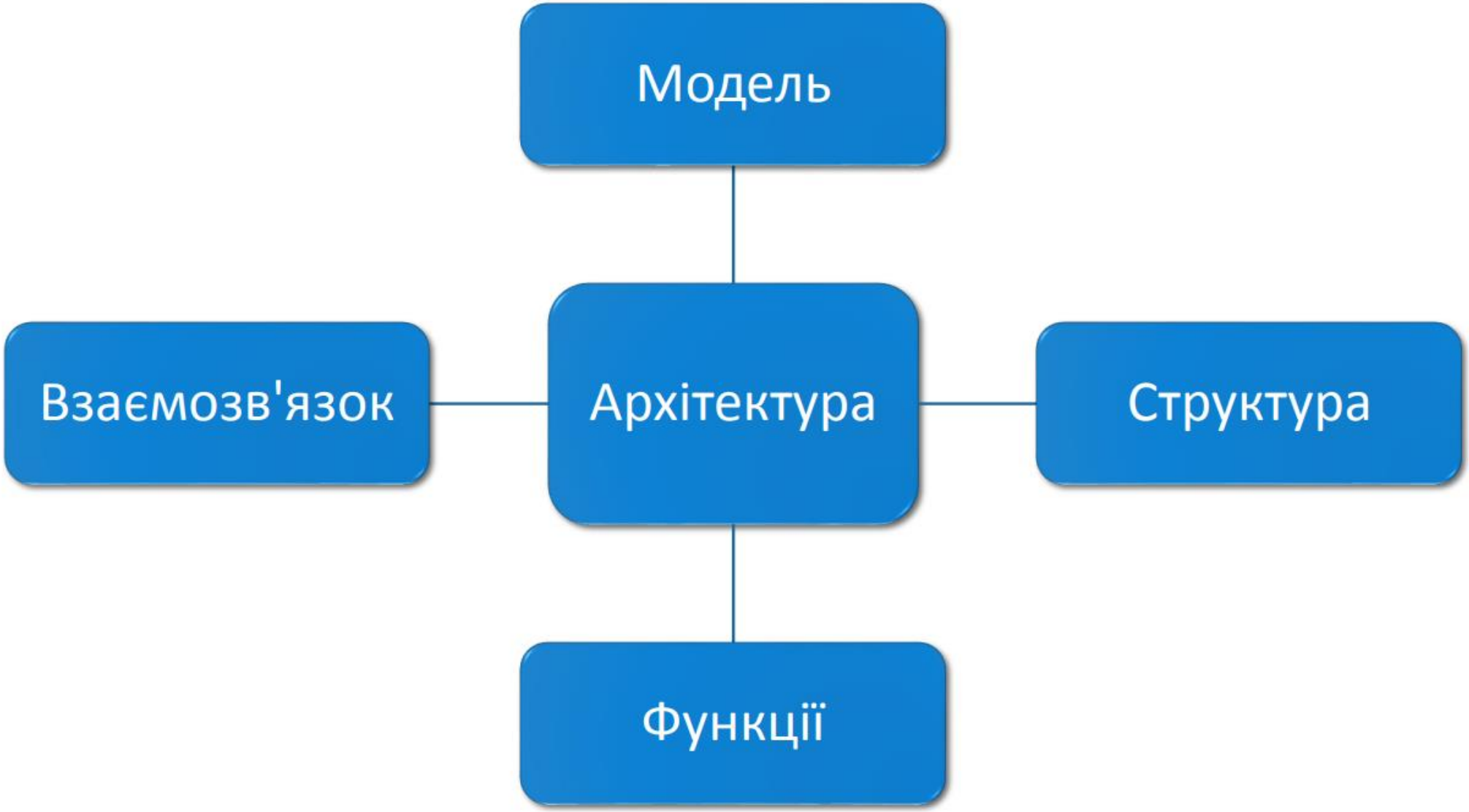
східчаста схема, показана на трьох нижніх діаграмах моделі IDEF (рис. 5.1). Порядок домінування позначається цифрою, розташовуваною в правому нижньому куті блоку. Блок з найбільшим домінуванням позначається цифрою 1, зі зменшенням домінування цифра зростає (2, 3, 4, 5, 6). Номера блоків служать однозначним ідентифікатором для системних функцій і автоматично організують ці функції в ієрархію моделі. Використовуючи номера блоків і оцінюючи їхній вплив, дослідник може організувати модель за принципом функціонального домінування. Це дозволяє погодити ієрархічний порядок функцій у моделі з рівнем впливу кожної функції на іншу частину системи. Тому IDEF рекомендує нумерувати блоки відповідно до порядку їхнього домінування. Дуги в IDEF-діаграмах показують, як функції системи

взаємозалежні, як вони обмінюються даними і здійснюють керування один одним. Для функціональних IDEF-моделей дуги представляють множину об'єктів, під якими розуміються, наприклад, такі, як "плани", "деталі", "інформація".

Так як дуги позначають об'єкти, то вони позначаються іменниками або іменниками з визначеннями, що розташовуються в достатній близькості до лінії дуги. Наприклад: "наступний крок завдання", "інструменти і креслення", "програма" тощо. Можна сказати, що дуги на IDEF-діаграмах показують зображують інтерфейси між функціями системи, а також між системою і її навколишнім середовищем.

Поняття архітектури інформаційних систем

Рівень розвитку сучасних технологій настільки високий, що дозволяє побудувати інформаційну систему будь-якого масштабу, складності й функціональності. Однак, з огляду на вимоги бізнесу, засновані на показниках різних бізнес-оцінок, виникають додаткові складнощі, вирішення яких зводиться до забезпечення раціонального підходу до процесу проектування, реалізації й подальшій експлуатації інформаційних систем. Виходячи із цього, можна однозначно вважати обрану архітектуру одним з основних показників ефективності створюваної інформаційної системи, а, отже, і успішності бізнесу.


Модель
Взаємозв'язок
Архітектура
Структура
Функції

Визначити поняття "архітектура інформаційної системи" можна безліччю способів. Це зв’язано:

1. З відсутністю загальноприйнятого визначення самої інформаційної системи. З огляду на складність структури, достатнім способом описати її можливо тільки при консолідації декількох точок зору, що в кожному конкретному випадку може приводити до різних результатів.
2. З різноманіттям трактувань самого терміну "архітектура".

Архітектуру інформаційної системи можна описати як концепцію, що визначає модель, структуру, виконувані функції й взаємозв’язок компонентів інформаційної системи.

Процедура вибору архітектури для проектованої інформаційної системи, у ринкових умовах, зводиться **до визначення вартості** володіння нею. Вартість володіння інформаційною системою складається із **планових витрат і вартості ризиків**.

Планові витрати містять у собі вартість технічного обслуговування, модернізації, зарплату обслуговуючого персоналу й т.д.

Вартість ризиків визначається з вартості всіх типів ризиків, їхніх імовірностей і матрицею відповідності між ними. Матриця відповідності визначається обраною архітектурою інформаційної системи.

Можна виділити **найбільш важливі типи ризиків**:

- проектні ризики (ризики при створенні системи);
- ризики розробки (помилки, недостатня оптимізація);
- технічні ризики (простої, відмови, втрата даних);
- бізнес-ризики (виникають через технічні ризики й пов’язані з експлуатацією системи);
- невизначеності (пов’язані з варіативністю бізнесів-процесів і складаються з необхідності внесення змін у систему й неоптимальну процедуру функціонування);
- операційні (мають на увазі невиконання набору операцій, можуть виникати через технічні ризики й бути ініціаторами бізнесів-ризиків).

Концепція архітектури інформаційної системи повинна формуватися ще на етапі техніко- економічного обґрунтування.

Необхідно відповісти на ряд питань:

1. Що робить система?
2. На які складові частина вона розділена?
3. Яким чином відбувається взаємодія цих частин?
4. Як і де ці частини розміщені?

Таким чином, можна вважати архітектуру інформаційної системи моделлю, що визначає вартість володіння через наявну в даній системі інфраструктуру.

Типи архітектур

Розглядаючи архітектуру великих організацій, прийнято використовувати поняття «корпоративна архітектура». Її можна представити у вигляді сукупності декількох типів архітектур:

- бізнес архітектура (Business architecture);
- IT-архітектура (Information Technology architecture);
- архітектура даних (Data architecture);
- програмна архітектура (Software architecture);
- технічна архітектура (Hardware architecture).


Бізнес-стратегії та бізнес-процеси
5
Бізнес-архітектура
Середовище для реалізації та інтеграції бізнес-додатків
4
IT-архітектура
Сховища даних та СУБД
3
Архітектура даних
Програмні додатки
2
Програмна архітектура
Апаратні засоби
1
Технічна архітектура

Технічна архітектура є першим рівнем архітектури інформаційної системи. Вона описує всі апаратні засоби, що використовуються при виконанні заявленого набору функцій, а також включає засобу забезпечення мережної взаємодії й надійності. У технічній архітектурі вказуються периферійні пристрої, мережні комутатори й маршрутизатори, жорсткі диски, оперативна пам'ять, процесори, сполучні кабелі, джерела безперебійного живлення й т.п.

Програмна архітектура являє собою сукупність комп'ютерних програм, призначених для рішення конкретних завдань. Даний тип архітектури призначений для опису додатків, що входять до складу інформаційної системи. На даному рівні описують програмні інтерфейси, компоненти й поведінку.

Архітектура даних поєднує в собі як фізичні сховища даних, так і засоби керування даними. Крім того, до неї входять логічні сховища даних, а при орієнтованості розглянутої компанії на роботу зі знаннями, може бути виділений окремий рівень – архітектура знань (Knowledge architecture). На цьому рівні описуються логічні й фізичні моделі даних, визначаються правила цілісності, складаються обмеження для даних.

Слід особливо виділити рівень **IT-архітектури**, оскільки він є сполучним. На ньому формується базовий набір сервісів, які використовуються як на рівні програмної архітектури, так і на рівні архітектури даних. Якщо будь-яка особливість функціонування для цих двох рівнів не була передбачена, то сильно зростає

ймовірність збоїв у роботі, а, отже, втрат для бізнесу. У деяких випадках неможливо розділити IT-архітектуру й архітектуру окремого додатка. Таке можливо при високому ступені інтеграції додатків. Прикладом IT-архітектури може служити SharePoint від компанії Microsoft. Цей продукт надає сервіси для спільної роботи й зберігання інформації, що є дуже важливим аспектом функціонування будь-якої компанії. Його базові системні модулі відносяться до IT-архітектури, а користувальницькі – до програмного. Основною функцією IT-архітектури є забезпечення функціонування важливих бізнесів-додатків для досягнення позначених бізнесів-цілей. Якщо деяка функція потрібна відразу в декількох додатках, то її доцільно перенести на рівень IT-

архітектури, тим самим підвищивши інтеграцію системи й знизити складність архітектури додатків.

Останнім в ієрархії є рівень бізнес-архітектури або архітектури бізнесів-процесів. На цьому рівні визначаються стратегії ведення бізнесу, способи керування, принципи організації й ключові процеси, що представляють для бізнесу величезну важливість.

Мікроархітектура й макроархітектура

Терміни мікроархітектура й макроархітектура більшою мірою застосовуються для опису програмних систем.

будова всієї ІС, як сукупності підсистем

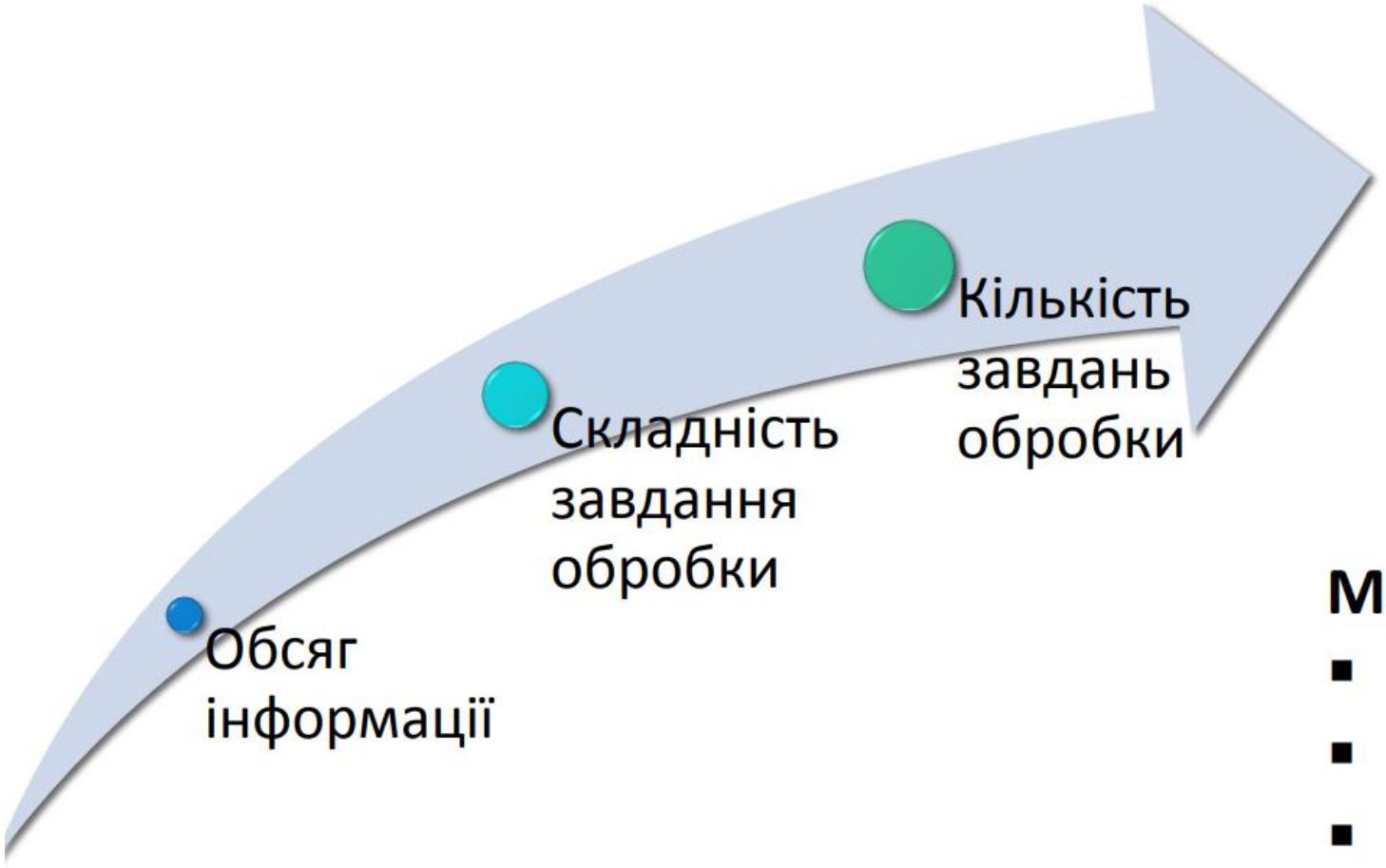
внутрішня будова конкретної підсистеми

У відповідність із розглянутою моделлю рівнів архітектур корпоративних інформаційних систем, мікроархітектуру можна віднести до рівнів програмної архітектури й архітектури даних, а макроархітектуру – до рівня IT-архітектури.

Мікроархітектура описує внутрішній будову конкретного компонента або підсистеми, а **макроархітектура** описує будову всієї ІС, як сукупності її компонентів або підсистем.

Без застосування якого-небудь архітектурного підходу при побудові складних систем, їхнє створення, обслуговування й модифікація, зрештою, стануть нерентабельними для бізнесу.

Для рішення даної проблеми можна використовувати методи абстракції, декомпозиції, інкапсуляції.



Методи:

- абстракції
- декомпозиції
- інкапсуляції

Існують **два принципи**, що дозволяють оцінити взаємний вплив компонентів системи один на одного:

- **Low Coupling (слабка зв'язаність);**
- **High Cohesion (сильне зчеплення).**

low coupling

- мале число залежностей між підсистемами;
- слабка залежність однієї підсистеми від змін в іншій;
- високий ступінь повторного використання підсистем

Принцип Low Coupling сприяє розподілу функцій між компонентами системи таким чином, щоб ступінь зв'язаності між ними залишалася низкою. Ступінь зв'язаності (coupling) – це міра взаємозалежності підсистем. Даний принцип пов'язаний з одним з основних принципів системного підходу, що вимагає мінімізації інформаційних потоків між підсистемами.

У свою чергу, принцип High Cohesion задає властивість сильного зчеплення всередині підсистеми. У результаті підсистеми виходять сфальцьованими, керованими й зрозумілими.

high cohesion

- сильне зчеплення всередині підсистеми
- тісний зв’язок функції між собою
- невеликі обсяги роботи

Зчеплення (функціональне зчеплення) – це міра зв’язаності й сфокусованості функцій підсистеми. Підсистема має високий ступінь зчеплення, якщо її функції тісно зв’язані між собою, і вона не виконує великих обсягів роботи.

Підсистема з низьким ступенем зчеплення виконує безліч різних функцій ніяк не зв'язаних між собою. Такі підсистеми створювати небажано, оскільки вони приводять до виникнення таких проблем:

- труднощі розуміння.
- складність при повторному використанні.
- складність підтримки.
- ненадійність, постійна схильність змінам.

Підсистеми з низьким ступенем зчеплення не мають чіткого функціонального призначення й виконують занадто різнопланові функції, які можна легко розподілити між іншими підсистемами.

Слід відмітити, що зв’язаність є характеристикою системи цілком, а зчеплення характеризує окремо взяту підсистему.

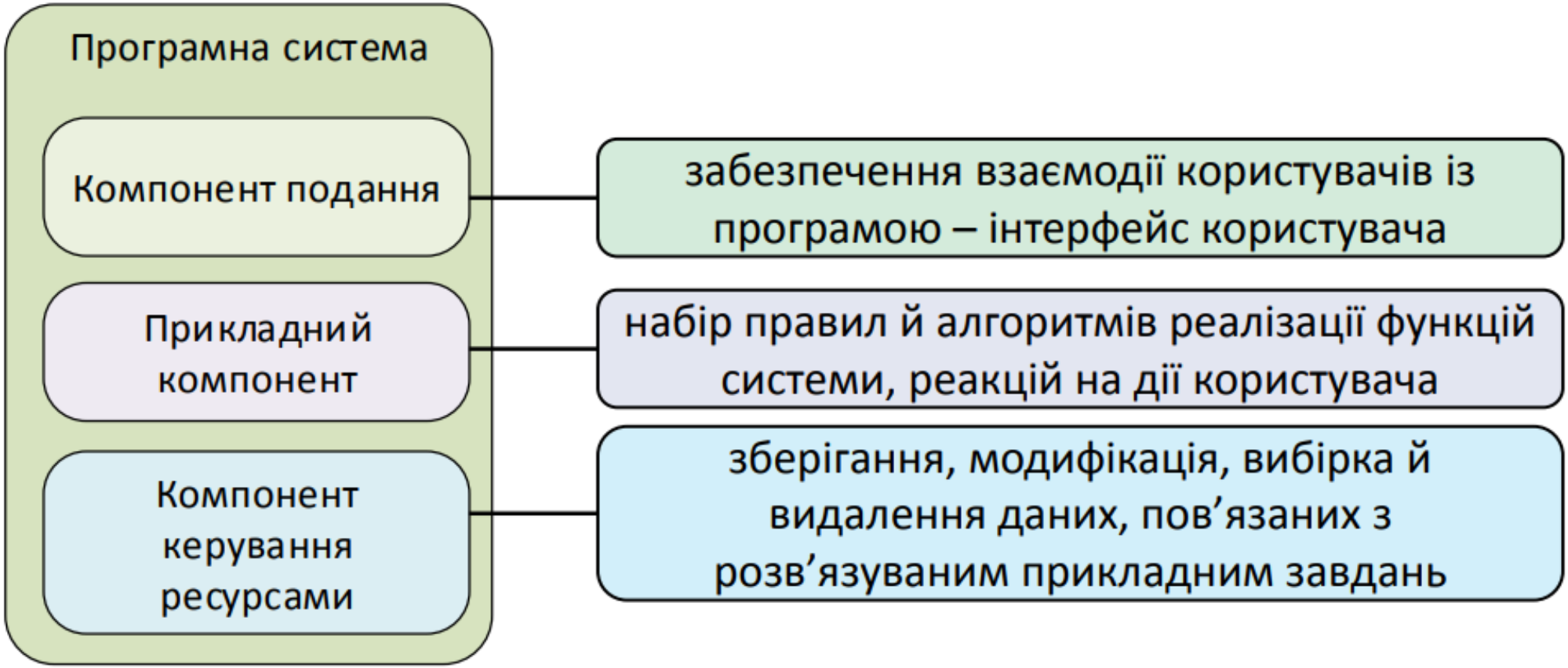
Зв’язаність (coupling) і зчеплення (cohesion) є загальносистемними характеристиками й можуть застосовуватися при проектуванні будь-яких систем.

Функціональні компоненти інформаційної системи

З огляду на принцип декомпозиції, прийнято проектувати інформаційні системи з поділом функціонального призначення їхніх компонентів, тобто створювати багаторівневе подання.

Можна виділити три основні функціональні групи, призначені для рішення різних за змістом завдань:

1. Взаємодія з користувачами.
2. Бізнес-логіка.
3. Керування ресурсами.


Програмна система
Компонент подання
Прикладний компонент
Компонент керування ресурсами
забезпечення взаємодії користувачів із програмою – інтерфейс користувача
набір правил й алгоритмів реалізації функцій системи, реакцій на дії користувача
зберігання, модифікація, вибірка й видалення даних, пов’язаних з розв’язуваним прикладним завдань

Архітектурний підхід

Суть

- створення фреймворка – каркаса з легкою адаптацією під потреби конкретної системи

Задачі

- розробка багаторазово використовуваного каркаса
- створення системи на основі каркаса

Переваги

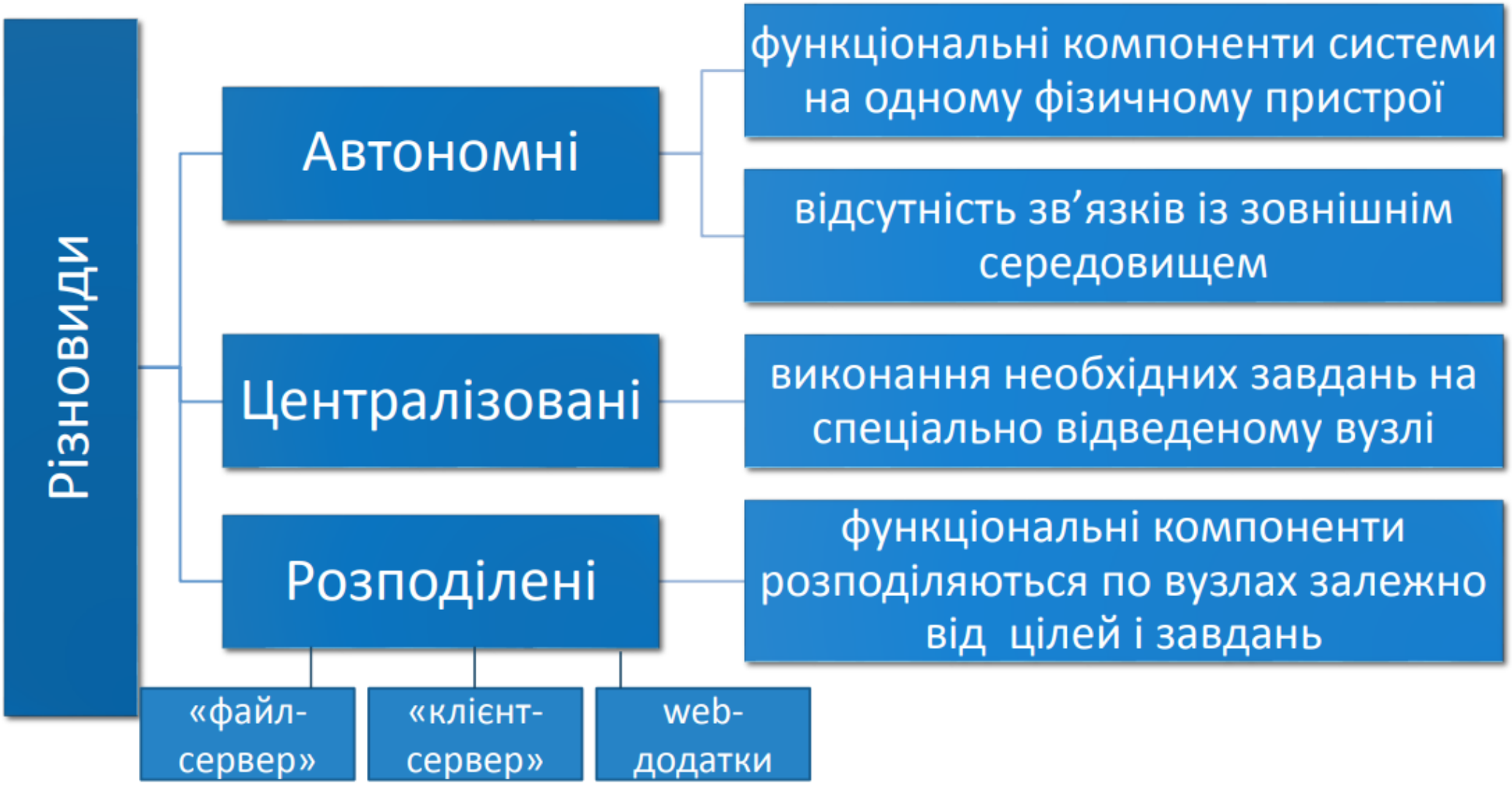
- можливість досить швидко змінювати функціональність системи за рахунок ітеративності процесу проектування

Платформні архітектури інформаційних систем

Можна виділити **три напрямки** розвитку платформних архітектур:

1. Автономні.
2. Централізовані.
3. Розподілені.

Автономна архітектура має на увазі наявність всіх функціональних компонентів системи на одному фізичному пристрої, наприклад, комп'ютері, й не повинна мати зв'язків із зовнішнім середовищем. Прикладом таких систем можуть служити системні утиліти, текстові редактори й досить прості корпоративні програми.


Різновиди
Автономні
функціональні компоненти системи на одному фізичному пристрої
відсутність зв’язків із зовнішнім середовищем
Централізовані
виконання необхідних завдань на спеціально відведеному вузлі
Розподілені
функціональні компоненти розподіляються по вузлах залежно від цілей і завдань
«файл-сервер»
«клієнт-сервер»
web-додатки

Централізована архітектура має на увазі виконання всіх необхідних завдань на спеціально відведеному вузлі, потужності якого досить, щоб задовольнити потреби всіх користувачів. Компоненти системи в цьому випадку розподіляються між обчислювальним вузлом, що називається мейнфрейм (mainframe), і термінальною станцією, за якої працює користувач. Термінал містить компонент подання, а мейнфрейм – прикладний компонент і компонент керування ресурсами. Слід зазначити, що термінал виступає винятково у вигляді пристрою виводу й не має інших функціональних можливостей.

Переваги:

- відсутність необхідності адміністрування робочих місць;
- легкість обслуговування й експлуатації системи, оскільки всі ресурси зосереджені в одному місці.

Недоліками подібної архітектури є:

- функціонування всієї системи повністю залежить від головного вузла (мейнфрейма);
- всі ресурси й програмні засоби є колективними й не можуть бути змінені під потрібні конкретних користувачів.

Щоб позбутися від останнього недоліку, у сучасних інформаційних системах застосовуються технології віртуалізації, завдяки чому стає

можливим виділити кожному користувачеві необхідну кількість ресурсів й установити необхідне програмне забезпечення.

Слід відмітити, що за допомогою технологій віртуалізації можна створити практично будь-яку архітектуру, використовуючи при цьому тільки ресурси мейнфрейма.

Поява й розвиток розподілених архітектур пов’язані з інтенсивним розвитком технічних і програмних засобів. У даному типі архітектури функціональні компоненти інформаційної системи розподіляються по наявних вузлах залежно від поставлених цілей і завдань.

Можна виділити шість основних характеристик архітектури розподілених систем:

- спільне використання ресурсів (як апаратних, так і програмних);
- відкритість – можливість збільшення типів і кількості ресурсів;
- паралельність – можливість виконання декількох процесів на різних вузлах системи (при цьому вони можуть взаємодіяти);
- масштабованість – можливість додавати нові властивості й методи;
- відмовостійкість – здатність системи підтримувати часткову функціональність за рахунок можливості дублювання інформації, апаратної і програмної складових.

Розподілена архітектура ІС. «файл-сервер»

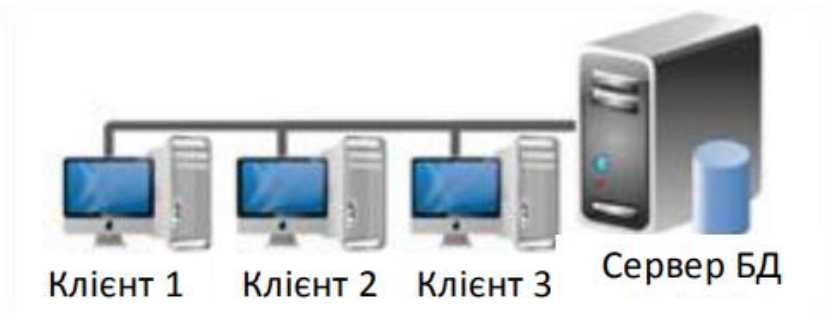
+ багатокористувальницький режим роботи;
+ централізоване керування правами доступу до загальних даних;
+ низька вартість розробки;
+ висока швидкість розробки

– послідовний доступ до даних
– відсутність гарантії їхньої цілісності;
– продуктивність

До недоліків розподілених систем варто віднести:

- структурна складність;
- складно забезпечити достатній рівень безпеки;
- на керування системою потрібне велика кількість зусиль;
- непередбачена реакція на зміни.

Розподілена архітектура ІС. «клієнт-сервер»

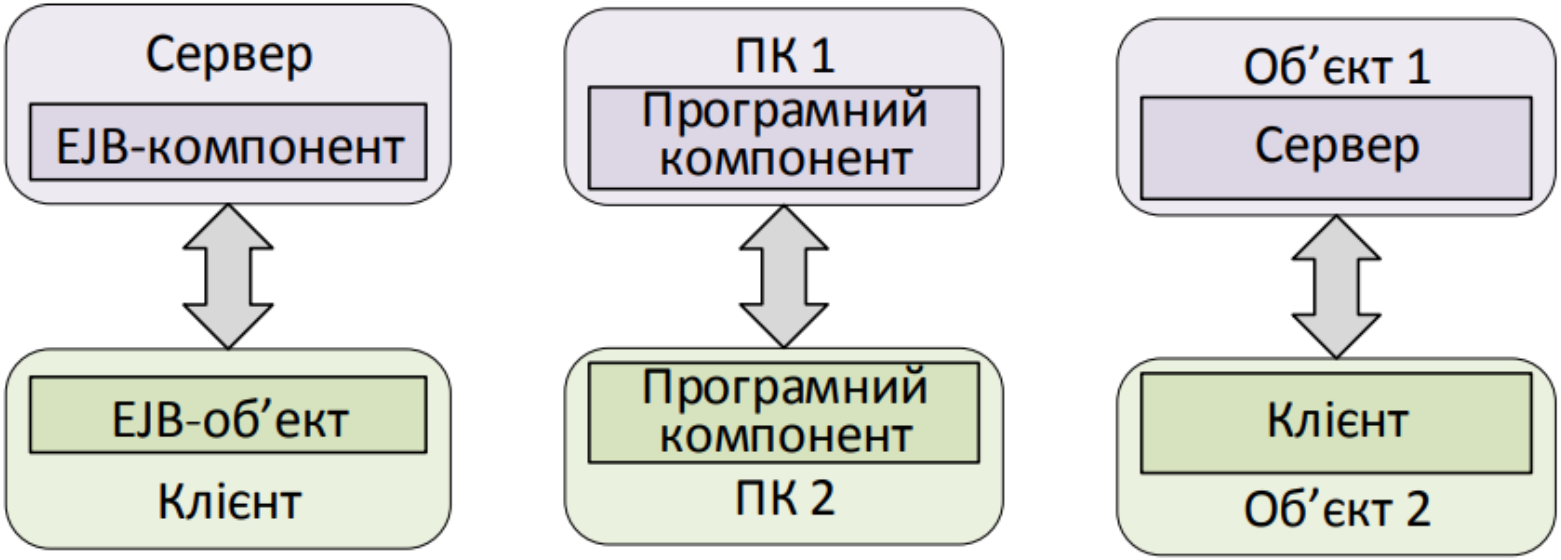


+ багатокористувальницький режим роботи;
+ гарантія цілісності даних;
+ наявність механізмів керування правами доступу до ресурсів сервера;
+ розподілу функцій між вузлами мережі

– високий рівень технічного персоналу;
– вихід з ладу сервера може спричинити непрацездатність всієї системи;
– висока вартість устаткування

Розподілена архітектура ІС. web-сервіс

EJB (Enterprise Java Beans).	DCOM (Distributed Component Object Model).	CORBA (The Common Object Request Broker Architecture)



Можна виділити три технології, які можливо використати для побудови розподіленої архітектури web-сервісу:

- EJB (Enterprise JavaBeans).
- DCOM (Distributed Component Object Model).
- CORBA (The Common Object Request Broker Architecture).

Ідеєю EJB була розробка інфраструктуру для легкого додавання й видалення компонентів зі зміною функціональності сервера. EJB дозволяє створювати власні додатки із створених модулів. При цьому можлива їхня зміна, що робить процес розробки гнучким і набагато більше швидким. Дана технологія сумісна з CORBA й Java API.

Взаємодія між клієнтів і сервером у цьому випадку предствляється як взаємодія EJB-об’єкта, що генерується спеціальним генератором, і

EJB-компонента, написаного розробником. При необхідності EJB-компонента, що перебуває на сервері, викликається однойменний метод EJB-об’єкта, розташованого на стороні клієнта, що зв’язується з необхідним компонентом і викликає необхідний метод.

Переваги EJB:

- просте й швидке створення;
- Java-оптимізація;
- кросплатформність;
- вбудована безпека.

Недоліки EJB:

- складність інтегрування з додатками;
- погана масштабованість;

- низька продуктивність;
- відсутність міжнародної стандартизації.

DCOM являє собою розподілену програмну архітектуру від компанії Microsoft. З її допомогою програмний компонент одного комп'ютера може передавати повідомлення програмному компоненту іншого комп'ютера, причому з'єднання встановлюється автоматично. Для надійної роботи потрібно забезпечити захищене з'єднання між зв'язаними компонентами і створити систему перерозподіл трафіку.

Переваги DCOM:

- незалежність від мови;
- динамічне знаходження об'єктів;
- масштабованість;

- відкритий стандарт.

Недоліки DCOM:

- складність реалізації;
- залежність від платформи;
- пошук через службу Active Directory;
- відсутність іменування сервісів через URL.

Технологія CORBA розглядає всі додатки в розподіленій системі як набір об’єктів. Об’єкти можуть одночасно виступати в ролі клієнта й сервера, викликаючи методи інших об’єктів і відповідаючи на їхні виклики. Застосування даної технології дозволяє будувати системи, що перевершують по складності й гнучкості системи з архітектурою клієнт-сервер (як дворівневої, так і трирівневої).

Переваги CORBA:

- незалежність від платформи;
- незалежність від мови;
- динамічні виклики;
- динамічне виявлення об’єктів;
- масштабованість;
- індустріальна підтримка.

Недоліки:

- відсутність іменування по URL;
- практично повна відсутність реалізації CORBA-сервісів” [9,10, 16-19].

Поняття архітектурного стилю

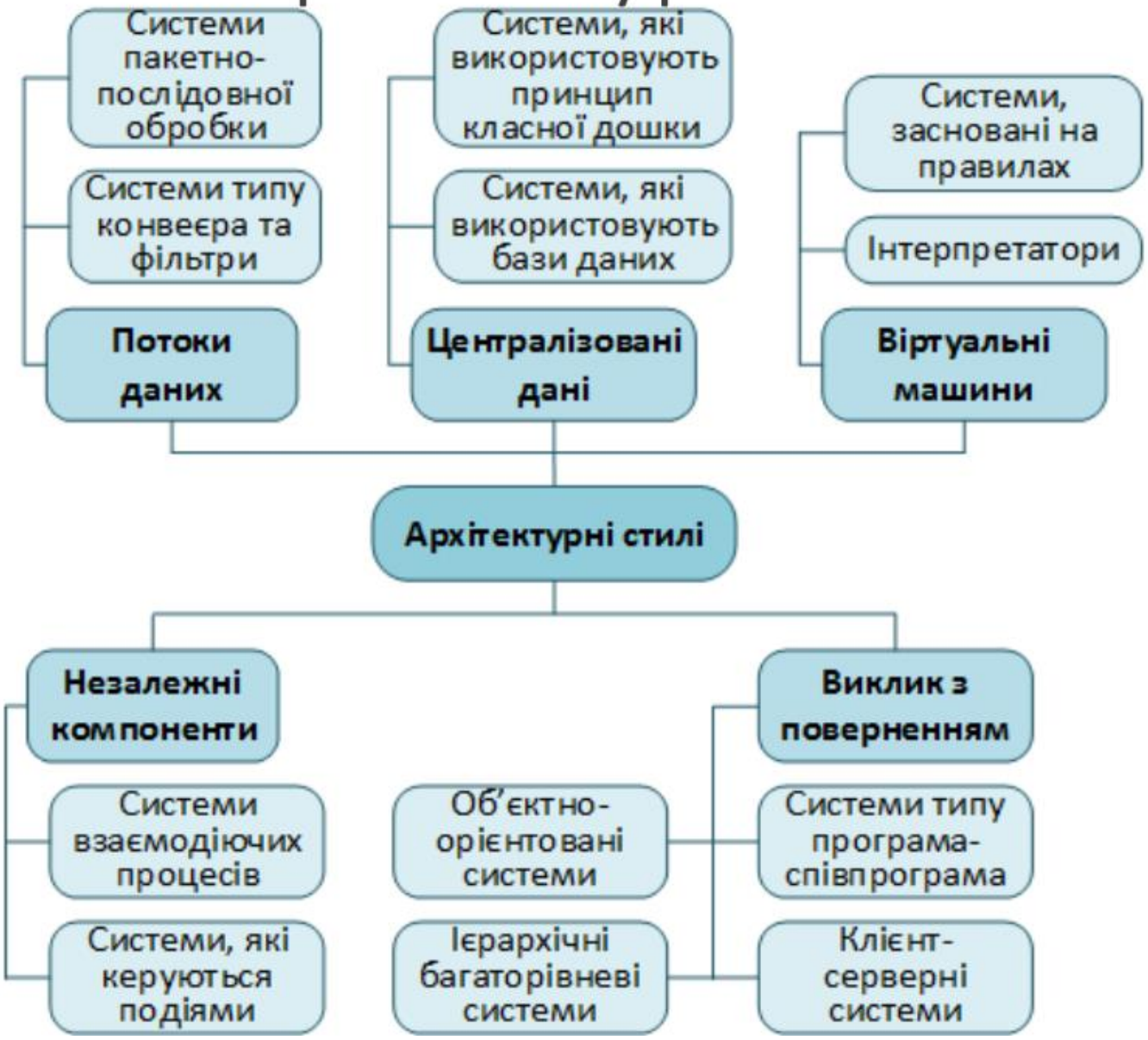
Архітектурний стиль

- подібність у підходах до реалізації поставлених завдань, обумовлене досвідом

- ❑ перелік компонентів системи
- ❑ способи взаємодії компонентів
- ❑ умови взаємодії компонентів

[9]

Типи архітектурних стилів



[9]

Лекція 4

ПОНЯТТЯ ФРЕЙМВОРКУ. ТИПИ ФРЕЙМВОРКІВ. СХЕМА ЗАХМАНА

ФРЕЙМВОРК TOGAF

Поняття фреймворку

Фреймворк

- загальноприйняті архітектурно-структурні рішення й підходи до проектування

- загальне рішення складного завдання

[9]

Фреймворки (каркаси)

Архітектурний фреймворк - сукупність угод, принципів і практик, які використовуються для опису архітектур і прийнятих відповідно деякому предметному домену й (або) у співтоваристві фахівців (зацікавлених осіб) (*Стандарт* ISO/IEC/IEEE *42010*:2011)

Architecture (system) - fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution

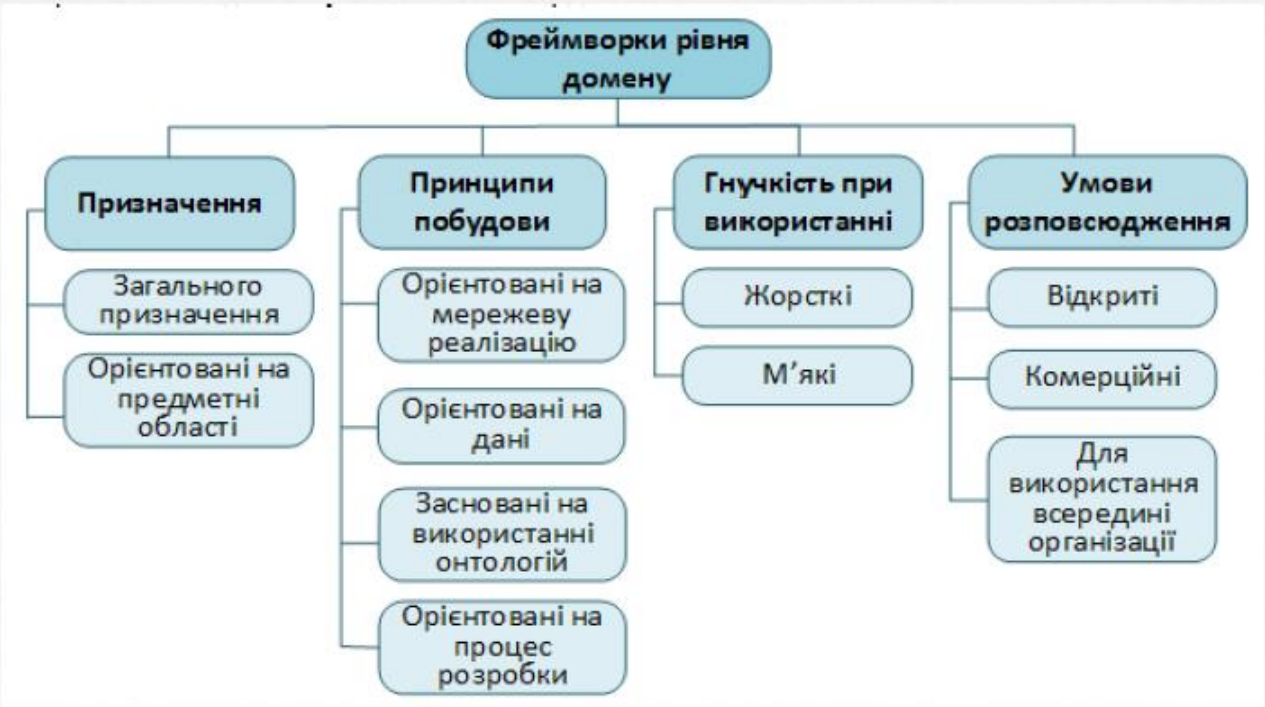
Класифікація фреймворків

[9]

“Інфраструктурні фреймворки (System Infrastructure Frameworks) спрощують процес розробки інфраструктурних елементів, застосовуються усередині організації

Фреймворки рівня проміжного програмного забезпечення (Middleware Frameworks) Проектування інформаційних систем застосовуються для вбудовування додатків і компонентів” [9,10].

Фреймворки рівня додатків (application frameworks) надають функціонал по реалізації типових додатків (GUI, бази даних і т.д).
Фреймворки рівня домену (Domain Frameworks) застосовуються для створення додатків у певній предметній області.
Допоміжні фреймворки (Support Frameworks) застосовуються для рішення приватних завдань



Класифікація фреймворків рівня домену

[9]

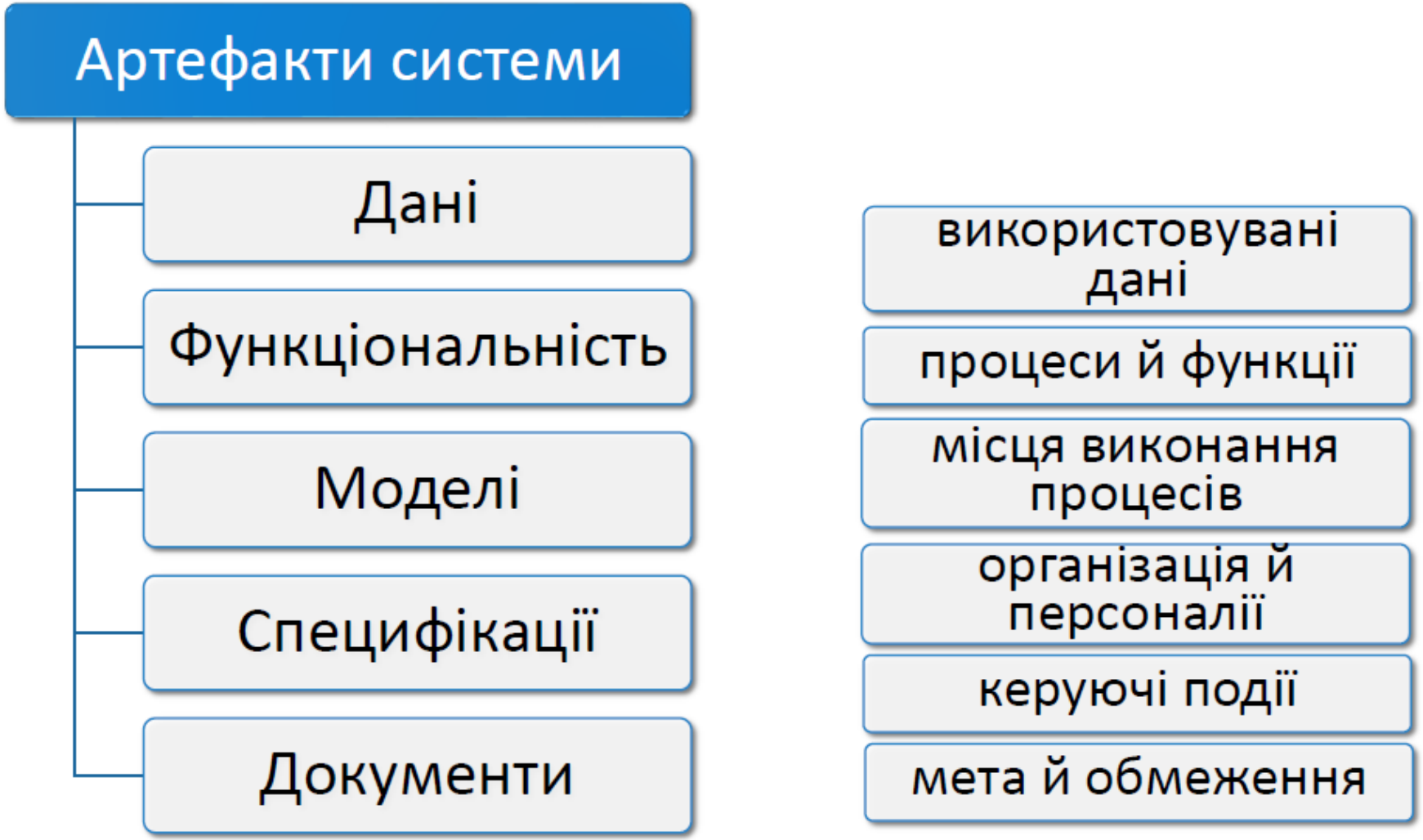
“Фреймворки, використовувані за принципом білого ящика (Architecture-driveng framework), застосовують методи спадкування й динамічного зв’язування для формування основних елементів додатка. Такі фреймворки визначаються через інтерфейси об’єктів, що додають у систему. Для роботи з ними необхідна докладна інформація про класи, розширення яких необхідно.

Фреймворки, що функціонують за принципом чорного ящика, також називають фреймворками, керованими даними. Основними механізмами формування додатків, у цьому випадку, виступають композиція й параметризація, при цьому функціональність забезпечується додаванням додаткових компонентів. Слід зазначити, що процес використання фреймворків, що працюють за принципом

чорного ящика простіше, ніж працюючих за принципом білого ящика, однак їхня розробка складніше.

На практиці застосовують підхід сірого ящика (grey box), що є комбінацією обох підходів” [9-16].

Фреймворк Захмана



[9]

СХЕМА ЗАХМАНА (ZACHMAN FRAMEWORK)

Розробник: авторська методологія (приватної компанії). Рік формування: 1987. Форма: приватна методика. Тип: таксономія.

Ключові особливості: матриця Захмана є таксономию, що дозволяє зв'язати різні моделі, що надаються зацікавленим сторонам.

Короткий опис. До найбільш значущих на сьогоднішній день з ранніх моделей, присвячених опису і розробки архітектури підприємства, відноситься схема Захмана. Історично була створена для проектування і впровадження IT-систем, потім підхід був узагальнений для підприємства в цілому.

Приклади фреймворків. Фреймворк Захмана. Опис системи

	Використовувані дані (що?)	Процеси й функції (як?)	Місця виконання процесів (де?)	Організації й персоналії (хто?)	Керуючі події (коли?)	Мета й обмеження	
Контекст	Список основних сутностей	Основні бізнес-процеси	Територіальне розміщення організації	Важливі зовнішні організації	Список подій	Бізнес-стратегія	Аналітик
Бізнес-модель	Відносини між сутностями	Докладний опис бізнес- процесів	Система логістики	Модель потоків даних	Базовий графік робіт	Дерево цілей, Бізнес-план	Топ-менеджер
Системна модель	Концептуальні моделі даних	Архітектура додатків	Архітектура розподіленої системи	Інтерфейси користувача	Модель роботи з подіями	Бізнес-правила	Архітектор
Технологічна модель	Фізична модель даних	Програмно-апаратна архітектура	Технологічна архітектура	Архітектура подання	Алгоритми обробки подій	Правила обробки подій	Розробник
Детальний опис	Специфікації форматів даних	Виконуваний код	Архітектура мережі	Ролі й права користувачів	Обробка подій за допомогою переривань	Алгоритми роботи системи	Адміністратор
Функціонуюча організація	Дані	Реалізована функціональність	Функціонуюча мережна інфраструктура	Організаційна структура організації	Історія функціонування системи	Реалізовані стратегії	Користувач

Основна ідея схеми Захмана “полягає в тому, щоб забезпечити можливість послідовного опису кожного окремого аспекту підприємства в координації з іншими. Метод переслідує дві основні мети: з одного боку, логічно розбити всі опис архітектури на окремі розділи, з іншого - забезпечити можливість розгляду цілісної архітектури на декількох рівнях абстракції” [9,10]. Для цього застосовується матриця 6 х 6, в якій кожна клітинка задає свій тип опису (моделей) властивостей підприємства. Вся сукупність осередків розділена на шість стовпців матриці - шість аспектів діяльності підприємства:

- «ЩО робиться», або об'єкти / дані;
- «ЯК робиться», або функції / процеси;

- «ДЕ робиться», - розміщення або інфраструктура;
- «ХТО робить» - люди, організаційні одиниці;
- «КОЛИ робиться» - графіки подій і робіт;
- «НАВІЩО робиться» - стимули, мотиви та стратегії діяльності.

Ці аспекти пропонується описувати в шести різних, пов'язаних між собою рівнях, згрупованих в рядки матриці: від рівня вищого керівництва («планувальника забудови») до технічного фахівця.

- **Стовпці можна міняти місцями, але не можна видаляти**
- **Кожному стовпчику відповідає власна модель.**
- **Кожна з моделей має бути унікальна.**
- **Кожен рівень (рядок) є описом системи з погляду групи користувачів (представляє окремий вид).**
- **Кожна з комірок унікальна.**
- **Кожна комірка містить опис аспекту реалізації системи у вигляді моделі й текстового документа.**
- **Заповнення комірок повинне відбуватись послідовно зверху вниз**

“**Перший рядок** матриці визначає контекст всіх інших й являє собою загальний погляд на організацію.

Другий рядок описує функціонування організації в бізнесах-термінах.

Третій рядок описує бізнес-процеси в термінах інформаційних систем.

Четвертий рядок дозволяється розподілити дані й виконувані над ними операції між конкретними апаратними і програмними платформами.

П’ятий рядок описує конкретні моделі устаткування, мережеві топології й програмний код.

Шостий рядок описує готову систему у вигляді інструкції користувачів, довідкових баз даних і т.д.” [9, 16]

Концептуальні ідеї схеми Захмана такі:

- рекурсивность логіки формування моделей і метамоделей на основі однієї узагальненої схеми;

- управління архітектурою і змінами підприємства на основі загального сховища;

- використання сховища для роботи з різними моделями і їх станами.

Схема Захмана дозволяє:

- використовувати одну концептуальну основу, єдину і зрозумілу як для бізнес-фахівців, так і для IT-фахівців;

- фокусуватися на окремих аспектах підприємства (аж до конкретної системи), не втрачаючи погляду на ціле;

- забезпечувати узгодженість бізнесу та IT за рахунок відповідності описів в осередках;

- зберігати незалежність від будь-якого програмного продукту (інструменту).

Основні принципи використання схеми Захмана:

- стовпці рівнозначні за статусом і порядку;

- кожен стовпець - проста базова модель підприємства;

- базова модель для кожного стовпця унікальна;

- кожен рядок - обмежений унікальний рівень або перспектива;

- кожна клітинка унікальна;

- сукупність всіх осередків одного рядка задає повну модель одного рівня підприємства.

Недоліки схеми Захмана:

- відсутні специфікації і опису процесів (інструкцій) щодо створення архітектури;

- є статичним описом, не містить «динаміки»;

- немає механізмів оцінки якісності побудови архітектури;

- не дозволяє визначити, чи потрібно вдосконалювати поточну архітектуру;

- в стовпчиках об'єднані рівні, що народжуються на різних етапах життєвого циклу проектування підприємства (т. е. з віссю часу).

Ключові переваги:

- універсальність, незалежність від методів і засобів;

- повнота таксономії.

Фреймворк TOGAF

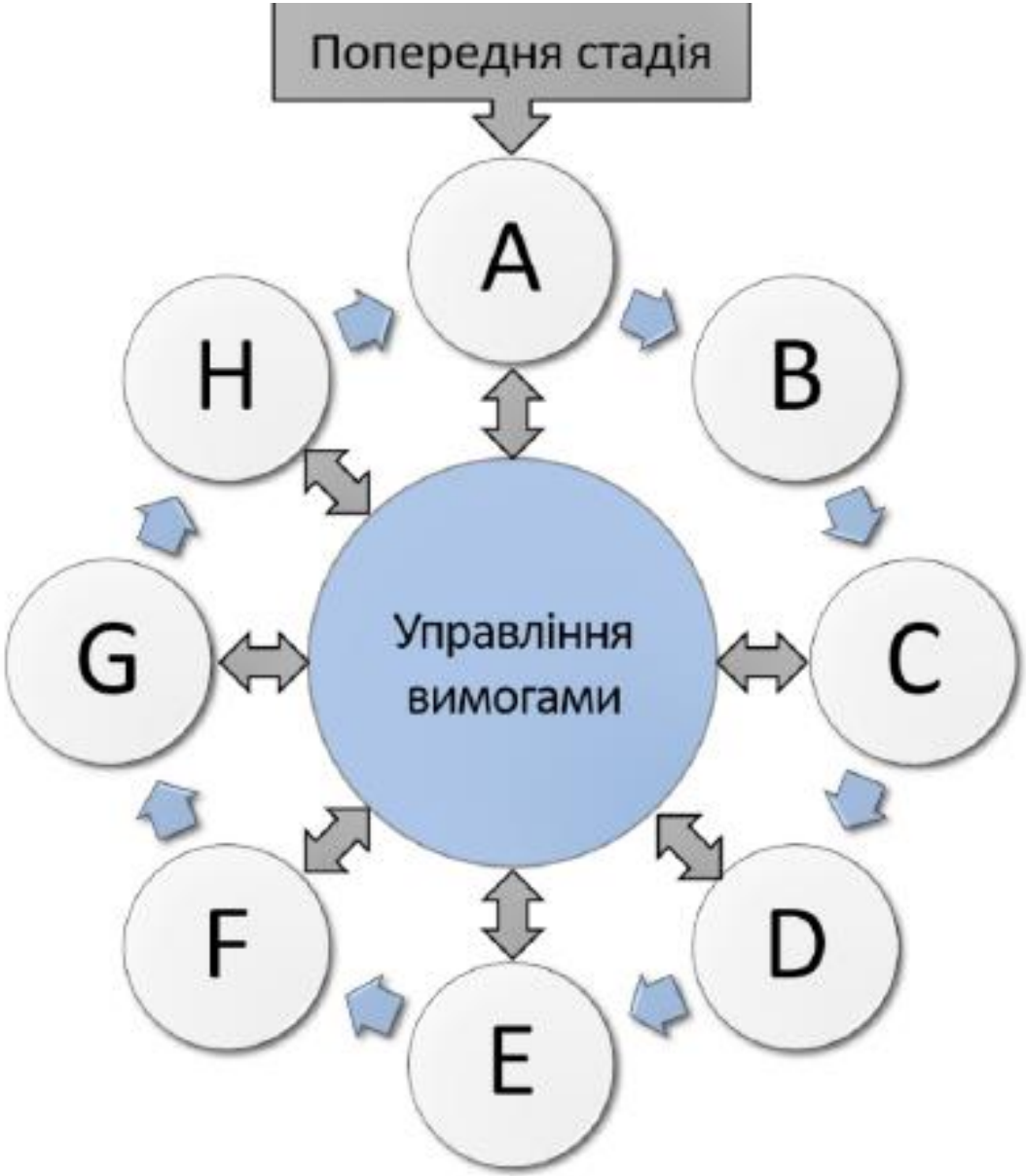
[9]

АРХІТЕКТУРНИЙ ФРЕЙМВОРК TOGAF (The Open GroupArchitecture Framework)

Розробник: The Open Group (відкритий консорціум).Рік формування 1991. Форма: стандарт в рамках консорціуму «The Open Group», розвинений інститут сертифікації фахівців. Тип: комплексний фреймворк.

Ключові особливості: TOGAF містить в своєму ядрі метод розробки архітектури - ADM. Метод детально описаний, що є відмінною рисою TOGAF.

TOGAF - найбільш широко поширений і відомий у світовій практиці фреймворк.



[9]

Методологія призначена для всіх типів підприємств, що розробляють, що реалізують і застосовують архітектурний підхід. Формально TOGAF може застосовуватися для об'єднання організацій і підприємств, проте в змісті методології наявність в однієї корпоративної архітектурі підприємств різних секторів, сегментів або рівнів управління не розглядається.

Основна відмінність від інших методів побудови архітектури - наявність методу розробки архітектури (Architecture Development Method -ADM), що відповідає на питання «ЯК?». Матеріали TOGAF містять велике кількість прикладів і шаблонів архітектурних артефактів, що створюються на різних етапах ADM.

Методологія оперує широким набором архітектурних продуктів: **концепції, бачення, референтні моделі, описи, діаграми, бази технічних стандартів конкретних рішень, каталоги** і ряд інших. TOGAF передбачає поділ на чотири домена для опису цілісної архітектури підприємства:

- бізнес-архітектура (Business Architecture);
- архітектура додатків (Application Architecture);
- архітектура даних (Data Architecture);
- технологічна архітектура (Technical Architecture).

“TOGAF містить

- метод розробки архітектури (Architecture Development Method -ADM);
- рекомендації до методу розробки архітектури (ADM Guidelines and Techniques);
- структуру / метамодель архітектурного контенту (Architecture Content Framework);
- континуум підприємства (The Enterprise Continuum);
- референтні (довідкові) моделі (TOGAF Reference Models);
- схему опису здібностей (The Architecture Capability Framework)” [9].

Під час роботи над **архітектурою підприємства** архітектори створюють документи, схеми, презентації, плани і т. Д. Весь цей обсяг інформації утворює інформаційний контент архітектури підприємства. **Метамодель контенту - це інструмент організації архітектурної інформації таким чином, щоб вона була сконцентрована навколо потреб зацікавлених сторін**.

В склад **методології** входить поняття «континуум підприємства» (Enterprise Continuum) - «віртуальний репозиторій» всіх архітектурних активів, що існують як всередині підприємства, так і в галузі в цілому. **Континуум підприємства** - це накопичувач таких ресурсів, як моделі, шаблони рішень, каталоги і архітектурні продукти, які можуть

використовуватися як «будівельні блоки» у всьому процесі адаптації та реалізації архітектури підприємства.

У класі архітектурних продуктів з'явився «Будівельний блок» (Building Block), він являє собою компонент бізнесу, IT або архітектурну сутність (потенційно багаторазового використання), який може бути скомбінований з іншими будівельними блоками для побудови архітектур і рішень. Відноситься до одного **з двох типів - ABB і SBB**:

- **архітектурні будівельні блоки** (Architecture Building Blocks - ABB) зазвичай **описують необхідну результативність і формують специфікації для будівельних блоків рішень** (наприклад, необхідна

результативність клієнтського сервісу може підтримуватися декількома SBB, такими як процеси, дані і ПО);

- **будівельні блоки рішень** (Solution Building Blocks - SBB) представляють собою **компоненти, які будуть використовуватися для забезпечення необхідної результативності** (наприклад, комп'ютерна мережа - будівельний блок, який може бути описаний через відповідний артефакт і використовуватися в конкретному рішенні для архітектури підприємства).

TOGAF володіє такою важливою властивістю розвитку, як **можливість розширення вихідної метамоделі шляхом додавання нових сутностей і архітектурних продуктів**. Таким чином, сама

методологія є адаптується під конкретні потреби підприємства. Це властивість - одне з корінних властивостей методології TOGAF, яка сама по собі (т. е. в стані «як є») не є абсолютно завершеною і однаково придатною для всіх підприємств методологією, але служить великим рамковим набором описів, який підприємству треба пристосовувати до своїх потреб. Тому **важливою властивістю методології є гнучкість**, що дозволяє виконувати етапи частково, пропускати їх, об'єднувати, змінювати порядок і вносити зміни відповідно до конкретних вимог.

Методологія не регламентує використання певного інструменту моделювання або підтримки. На практиці TOGAF підтримується рядом комп'ютерних інструментів.

До основних **переваг** можна віднести:

- універсальність, незалежність від засобів і постачальників послуг;

- наявність методу розробки (Architecture Development Method - ADM);

- велика кількість матеріалів і обговорень у відкритому доступі або за незначну оплату;

- масштабованість методу (можливість застосовувати як для організації в цілому, так для конкретного IT-проекту).

Серед **недоліків**:

- високий рівень абстракції і узагальнення;

- немає механізмів оцінки якісності побудови архітектури;

- не дозволяє визначити, чи потрібно вдосконалювати поточну архітектуру.

TOGAF ADM. Метод розробки архітектури (Architecture Development Method - ADM) методології TOGAF надає закінчений набір інструкцій для реалізації і виконання АП в організації. Цей процес складається з декількох послідовних фаз, замкнутих в цикл.

Метод розробки архітектури (ADM) TOGAF

Стадія процесу	Опис
Попередня фаза (Preliminary Phase)	**Визначення способів управління, меж розробки, принципів реалізації, уточнення моделі щодо специфіки .**
Фаза A, розробка загального подання (Architecture Vision)	**Створення загального уявлення про архітектуру, визначення меж проекту, затвердження плану робіт і завдань для виконавців**
Фаза B, розробка бізнес-архітектури (Business Architecture)	**Виявлення принципів функціонування організації, принципів управління проектом**
Фаза C, розробка інформаційної архітектури (Information Systems Architecture)	**Розробка архітектури даних і додатків**
Фаза D, розробка технологічної архітектури (Technology Architecture)	**Підбір апаратних засобів, засобів інфраструктури, механизмів їх взаємодії мережевої**
Фаза E, Можливості і рішення (Opportunities and Solutions)	**Реалізація розробленої архітектури (купити готове рішення або зробити власне)**
Фаза F, Планування переходу до нової архітектури (Migration Planning)	**Оцінка ризиків, аналіз деталей переходу і специфікація**
Фаза G, формування системи керування реалізацією (Implementation Governance)	**Моніторинг процесу впровадження і виникаючих проблем**
Фаза H, керування зміною архітектури (Architecture Change Management)	**Налагодження процесу керування змінами розробленої архітектури**

“Завдання **підготовчої фази** (Preliminary Phase) - виявлення **зацікавлених в процесі реалізації осіб** і обговорення з ними завдань АП. На цій фазі виробляються керівні принципи архітектури (Architecture Guiding Principles '), які ґрунтуються на бізнес-принципах організації і описують процеси і критерії для спостереження за процесом реалізації АП.

Фаза А цього процесу призначена для **вираження бачення АП**. Концепція архітектури (Architecture Vision) використовує рушійні сили бізнесу, щоб позначити мета дій по створенню АП і створити опису першого різу для базової і цільової середовища. Якщо завдання бізнесу не ясні, то частина завдання цієї фази - допомогти бізнесу ідентифікувати свої головні завдання і відповідні процеси, які повинна

підтримувати АП. На цій фазі створюється спеціальний документ під назвою «Завдання на розробку архітектури» (Statement of Architectural Work), який окреслює область дії і умови АП.

Фаза B призначена для **детальної розробки архітектури предметної області** бізнесу. І базова, і цільова архітектура, які окреслені в документі «Концепція архітектури», деталізуються, щоб отримати корисні вхідні дані для технічного аналізу. На цій фазі використовуються різні методики і нотації бізнес-моделювання.

Фаза C пов'язана зі **створенням архітектури предметних областей** «Додаток» і «Дані (Інформація)». На цій фазі виробляється опис поточної і розробка цільової архітектури інформаційних систем. Для

цього також використовуються спеціальні методики, підходи і нотації, пов'язані з даними і додатками.

Фаза D фокусується на описі і розробці **технологічної архітектури** (мережі, обчислювальні пристрої, вузли, технологічні інтерфейси).

Мета **фази E** - з'ясувати можливості, пропоновані цільової архітектурою, і створити **ескіз потенційного рішення**. Робота в цій фазі концентрується навколо застосовності і практичності альтернатив реалізації. На цій фазі створюється план реалізації та впровадження, а також визначаються основні проекти.

У **фазі F** розставляються **пріоритети проектів реалізації, виконуються деталізоване планування** і аналіз прорахунків

процесу міграції. У це завдання входить оцінка залежностей між проектами і мінімізація їх підсумкового впливу на функції підприємства. У цій фазі оновлюється список проектів, деталізується «План реалізації та впровадження».

Після затвердження специфікації проекту фокус переміщається на формулювання більш конкретних умов і рекомендацій для кожного з проектів реалізації. Протягом **фази G** встановлюється **зв'язок між керуючою архітектурою і організацією, що розробляє систему** (, наприклад, або яких-небудь ще методологій управління проектом), а обрані проекти реалізуються під управлінням формальної архітектури. На виході цієї фази ми маємо «Архітектурні контракти», які

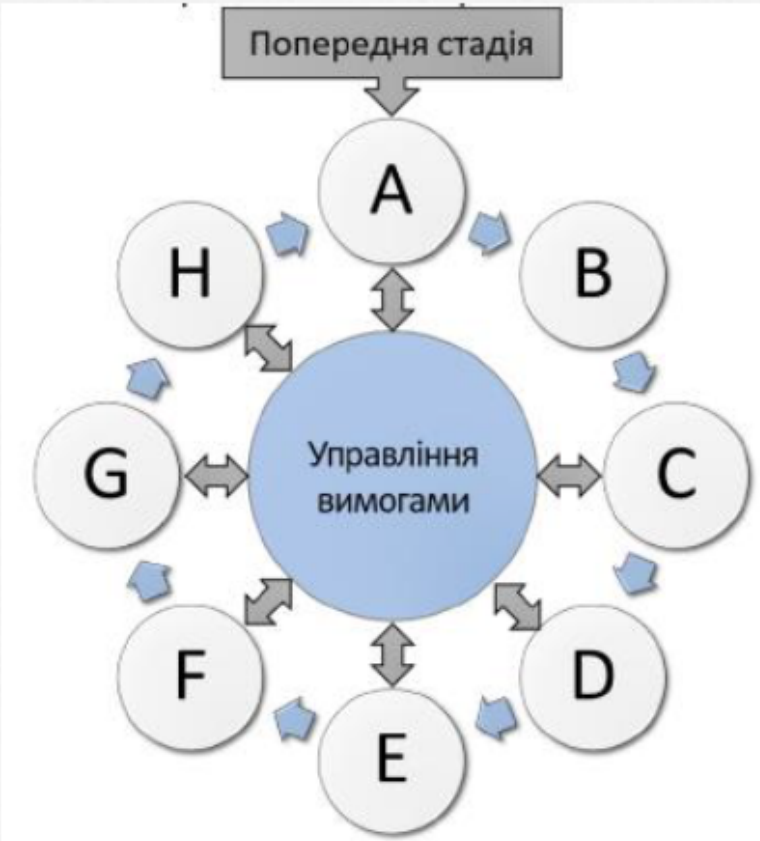
затверджуються організацією-розробником. Кінцевим виходом фази G є рішення, сумісні з архітектурою.

У **фазі H** акцент переноситься на **управління зміною архітектури**, яка досягається постачанням реалізованих рішень. У цій фазі може бути створено «Вимога до архітектурного завданням», що встановлює цілі для подальших циклів реалізації АП.

Метод є масштабованим і може використовуватися як для розробки архітектури компанії в цілому, так і для конкретного IT-рішення. Можливе використання в сукупності з іншими методами, більш спеціалізованими на конкретних завданнях, а також з галузевими методами і стандартами” [9].

Архітектурні домени:

- бізнес-архітектура (описує ключові бізнес-процеси, стратегію розвитку бізнесу і принципи управління);
- архітектура рівня додатків (описує інтерфейси додатків і способи їх застосування в термінах бізнес-сервісів);
- архітектура рівня даних (визначає логічну й фізичну структуру даних в організації);
- технологічна архітектура (визначає програмну, апаратну і мережеву інфраструктури).



Складові частини:

- ADM-методика (Architecture Development Method);
- керівництва і методики проектування для ADM;
- фреймворк архітектурного опису (Architecture Content Framework)
- архітектурний континуум організації (Enterprise Continuum),
- еталонні моделі TOGAF (TOGAF Reference Models):

TRM (Technical Reference Model);

III-RM (The Integrated Information Infrastructure Model)

- фреймворк, що описує структуру організації, її персонал, необхідні ролі і рівні відповідальності (Architecture Capability Framework).

5

[9]

Лекція 5

ТИПИ ФРЕЙМВОРКІВ. ФРЕЙМВОРК DODAF. ФРЕЙМВОРК FEA. ФРЕЙМВОРК GEAF

Департамент оборонної архітектури (DoDAF) є фреймворк архітектури для Міністерство оборони США (DoD), що забезпечує інфраструктуру візуалізації для конкретних зацікавлених сторін. Артефакти для **візуалізації, розуміння та асиміляції** широкого обсягу та складності опису архітектури: **табличний, структурний, поведінковий, онтологічний, зображальний, тимчасовий, графічний, імовірнісний, або альтернативний.** Особливо підходить для великих систем із складною інтеграцією та проблемами сумісності завдяки використанню "операційних поглядів". Ці погляди пропонують огляд та деталі, спрямовані на конкретних зацікавлених сторін у їхньому домені та у взаємодії з іншими доменами, в яких система буде працювати.

Фреймворк DoDAF

Елементи фреймворку

- моделі
- види
- точки зору

Орієнтація на дані

- Підвищена надійність збереження
- Повторне використання

Системи:

❑ системи збору, зберігання та аналізу даних для СППР

[9]

Фреймворк DoDAF

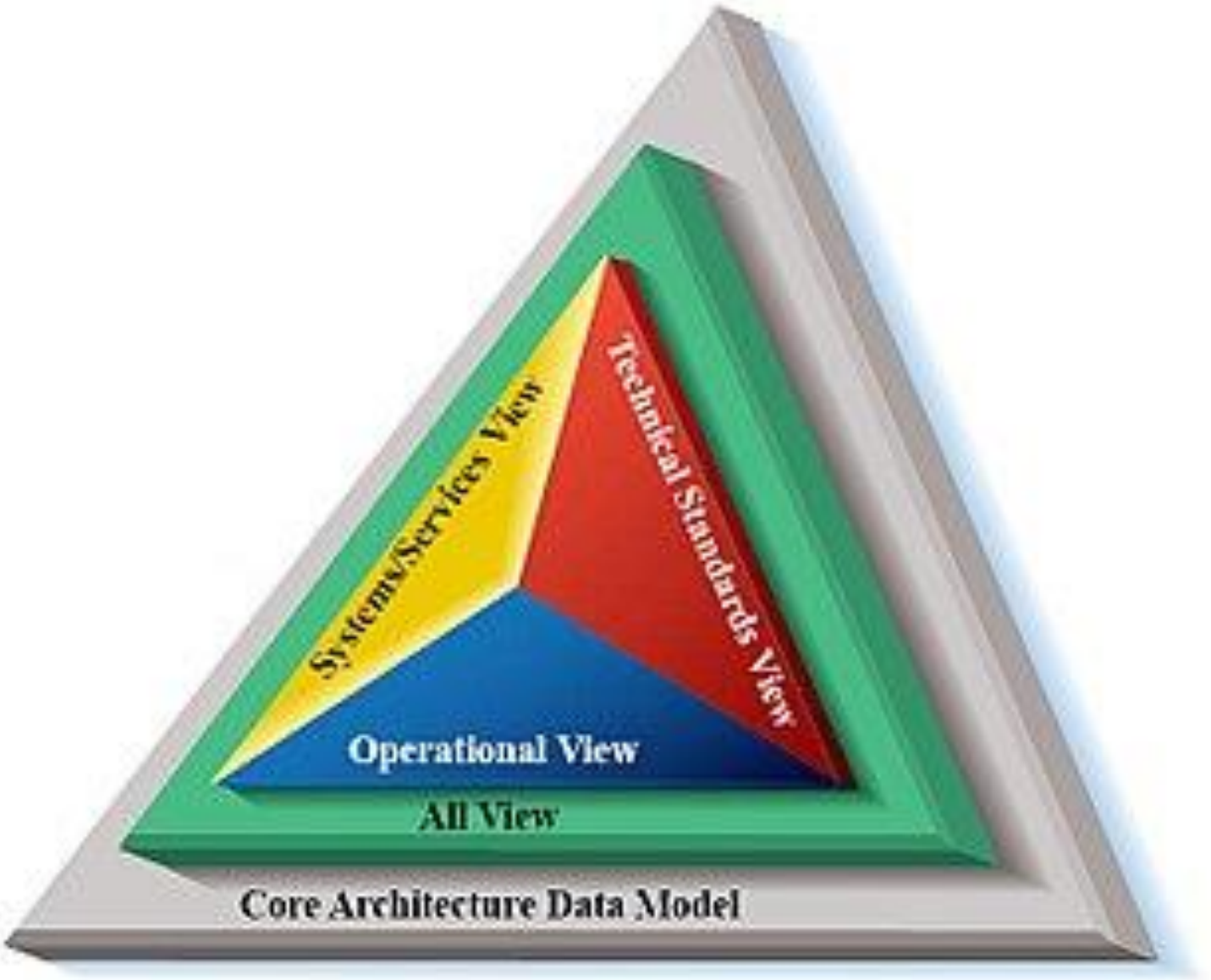
[9]

Мета-модель даних DoDAF

[9]

Базові принципи DoDAF

- ✓ Чітка орієнтація на цілі.
- ✓ Простота і зрозумілість.
- ✓ Полегшення процесу прийняття рішень.
- ✓ Використання для порівняння різних архітектур.
- ✓ Використання стандартних типів даних.
- ✓ Терміни даних.
- ✓ Організація для групової роботи.
- ✓ Використання в мережевому середовищі.


Systems/Services View
Technical Standards View
Operational View
All View
Core Architecture Data Model

Огляд

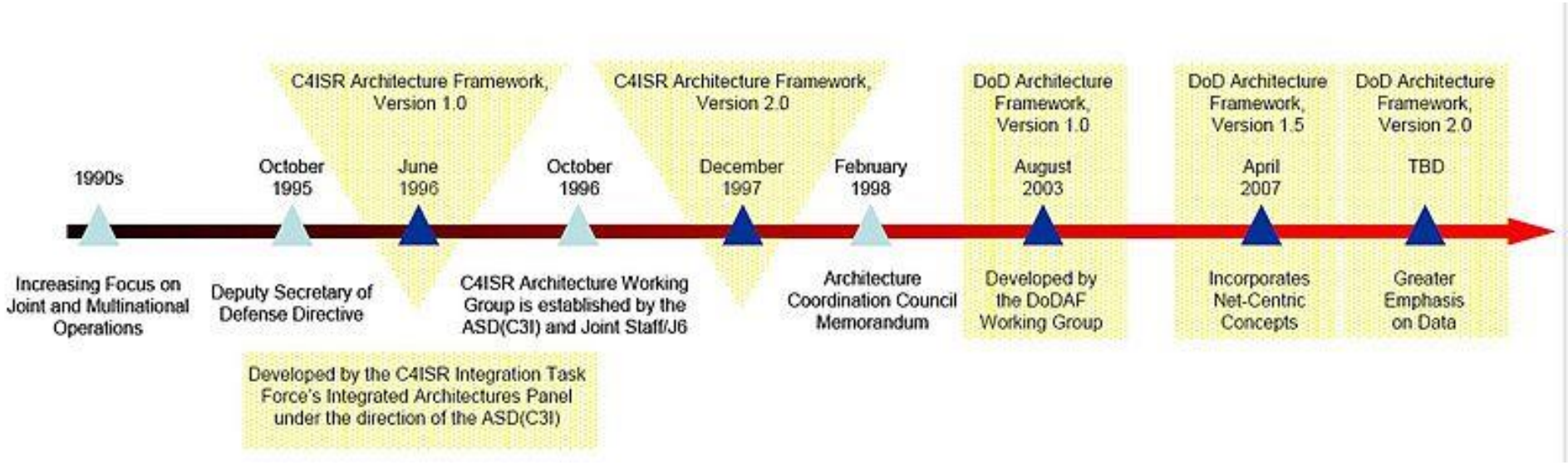
DoDAF забезпечує фундаментальну основу для розробки та представлення описів архітектури, що забезпечує спільний знаменник для розуміння, порівняння та інтеграції архітектур через організаційні, спільні та багатонаціональні кордони. Він встановлює визначення елементів даних, правила та взаємозв'язки та базовий набір продуктів для послідовного розвитку систем, інтегрованих або об'єднаних архітектур. Ці описи архітектури можуть включати сімейства систем (FoS), системи систем (SoS) та мережево-орієнтовані можливості взаємодії та взаємодії в не бойовому середовищі.

Очікується, що компоненти DoD максимально відповідають DoDAF при розробці архітектур у Департаменті. Відповідність гарантує, що повторне використання інформації, артефактів архітектури, моделей та точок зору може бути спільним для спільного розуміння. Усі основні придбання зброї Міністерства оборони США та систем інформаційних технологій повинні розробляти та документувати архітектуру підприємства (EA) з використанням поглядів, передбачених DoDAF. Незважаючи на те, що він явно спрямований на військові системи, DoDAF має широке застосування у приватному, державному та добровільному секторах у всьому світі і представляє одну з великої кількості систем архітектурних рамок.

- Мета DoDAF полягає у визначенні концепцій та моделей, які можна використовувати в шести основних процесах DoD:
    1. Інтеграція та розвиток спільних можливостей (JCIDS)
    2. Планування, програмування, складання бюджету та виконання (PPBE)
    3. Система придбання оборони (DAS)
    4. Системна інженерія (SE)
    5. Оперативне планування (OPLAN)
    6. Управління портфелем можливостей (CPM)
- Крім того, конкретними цілями DoDAF 2.0 були:[6]
    1. Встановити вказівки щодо змісту архітектури як функції цілі - "придатність до мети"

2. Збільште корисність та ефективність архітектур за допомогою суворої моделі даних - DoDAF Meta Model (DM2) - щоб архітектури могли бути інтегровані, проаналізовані та оцінені з більшою точністю.

Історія



Еволюція DoDAF з 1990-х років. DoDAF V2.0 вийшов у травні 2009 року. Перша версія розробки DoDAF була розроблена в 1990-х роках під назвою C4ISR Рамки архітектури. Перший C4ISR Architecture Framework v1.0, випущений 7 червня 1996 року, був створений у відповідь на проходження Закону Клінгера-Коена. Він звернувся до заступника Міністра оборони 1995 р. про те, що слід докласти зусиль для міністерства оборони для визначення та розробки кращих засобів та процесів для забезпечення того, щоб можливості C4ISR були сумісними та відповідали потребам військовослужбовця.

Що таке C4ISR?

C4 = Command and Control (C2)

– Communications

– Computers

– Intelligence, Surveillance, Reconnaissance.

Не назва IT системи — архітектура та концепція взаємодії складових.

Мета системи C4ISR

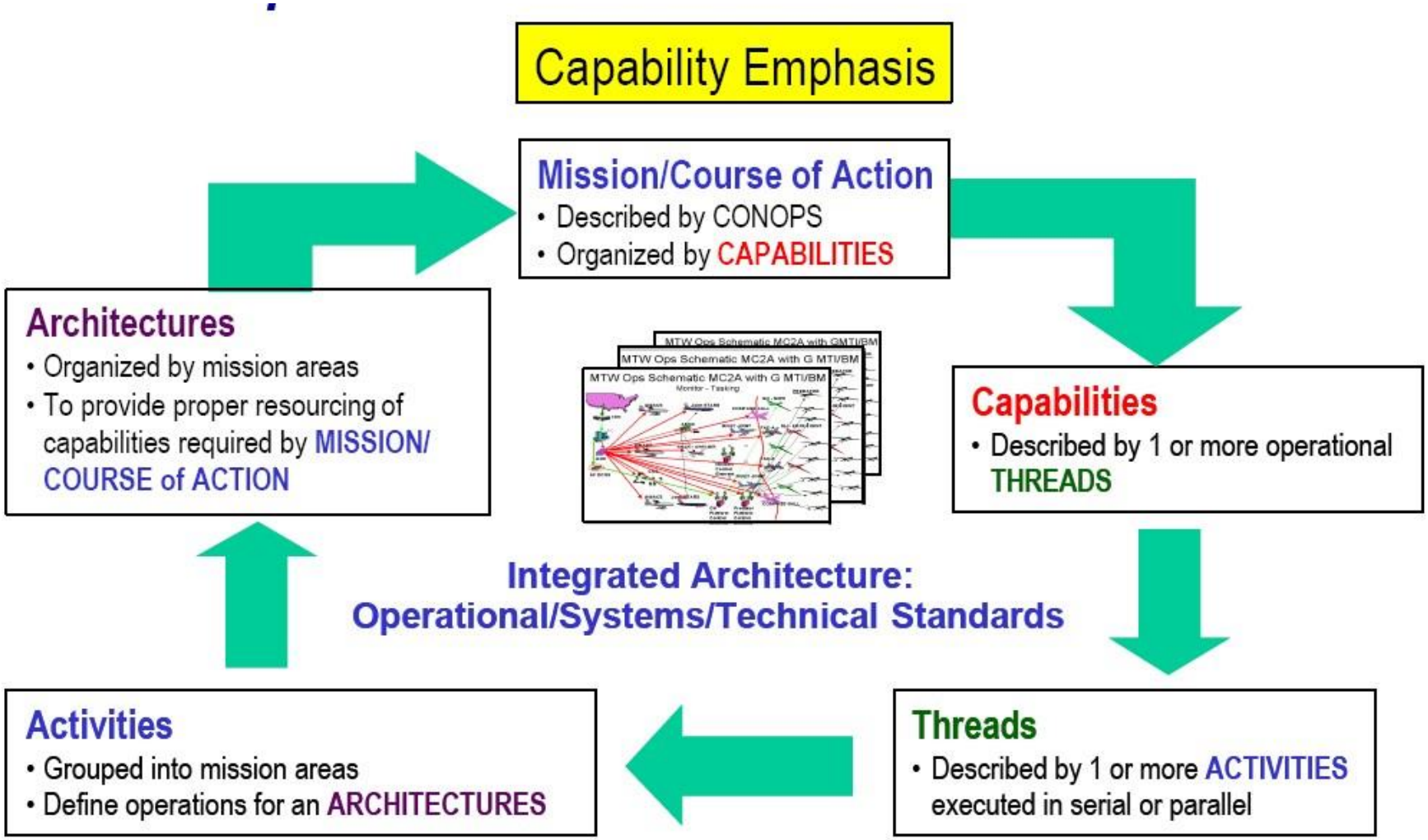
Досягнення інформаційної переваги над опонентом, яка трансформується у пригнічуючу бойову могутність завдяки поєднанню інтелектуальних об’єктів у єдиний бойовий простір.

Постійні зусилля з розробки призвели у грудні 1997 року до другої версії C4ISR Architecture Framework v2.0.

28 травня 2009 року Міністерство оборони затвердило DoDAF v2.0. Поточна версія - DoDAF 2.02 [1]DoDAF V2.0 опубліковано на загальнодоступному веб-сайті.

Інші похідні рамки, засновані на DoDAF, включають Архітектурну структуру НАТО (NAF) та. Наприклад, як і інші підходи до EA Архітектура Open Group Framework (TOGAF), DoDAF організований навколо спільного сховища для зберігання робочих продуктів. Сховище визначається загальною схемою бази даних Основна модель даних архітектури 2.0 та Архітектурна система реєстру DoD (DARS). **Ключовою особливістю DoDAF є взаємодія, яка організована у вигляді низки рівнів, які називаються Рівні взаємодії інформаційної системи (LISI).**

Можливості та місія



Можливості, описані в архітектурі

Міністерство оборони звернуло увагу на надання можливостей, які є причиною створення системи / послуги. Моделі спроможності описують таксономію та розвиток можливостей. Потік можливостей буде прирівнюватися до конкретних видів діяльності, правил та систем, які пов'язані з цією конкретною здатністю.

Концепція можливостей, визначена її Мета-моделлю групи даних, дозволяє відповісти на такі питання, як:

- Як конкретна здатність чи можливості підтримують загальну місію / бачення?
- Яких результатів передбачається досягти за допомогою певної можливості чи набору можливостей?

- Які послуги потрібні для підтримки можливостей?
- Який функціональний обсяг та організаційний діапазон можливостей чи набору можливостей?
- Який наш поточний набір можливостей, якими ми управляємо як частина портфеля?

Місія або курс дій описується Концепцією операцій (КОНОПС) та організовується "Можливостями".

- Можливості описуються Threads.
- Потоки описуються в розділі Діяльність, виконуваний послідовно або паралельно.

- Діяльність згрупована за напрямками місій. Діяльність визначає операції для архітектури.
- Архітектури організовані за напрямками місій. Архітектури забезпечують належне ресурсне забезпечення можливостей, необхідних місії або курсу дій.

Точки зору:

- узагальнена (All Viewpoint): інтегрує всі точки зору для створення архітектурного контексту;
- визначальна потенційні можливості (Capability Viewpoint): навчання персоналу, терміни поставок і т.д .;
- визначальна дані та інформацію (Data and Information Viewpoint): визначає способи подання та структури даних;
- операційна (Operational Viewpoint): розглядає сценарії роботи та активності системи;
- проектна (Project Viewpoint): розглядає необхідні характеристики і можливості системи;
- сервісна (Service Viewpoint): розглядає сукупність сервісів;
- враховує стандарти (Standards Viewpoint): розглядає технічні стандарти, обмеження, методики, керівництва і т.д.;
- системна (System Viewpoint): розглядає сукупність взаємодіючих систем та їх взаємодію

Рівні Data Meta-Model - DM2:

1. Концептуальна модель даних (Conceptual Data Model) - описує архітектуру в нетехнічних термінах.
2. Логічна модель даних (Logical Data Model) - розширення концептуальної моделі шляхом додавання атрибутів.
3. Специфікація обміну даними на фізичному рівні (Physical Exchange Specification) - засіб, що забезпечує обмін інформацією між моделями.

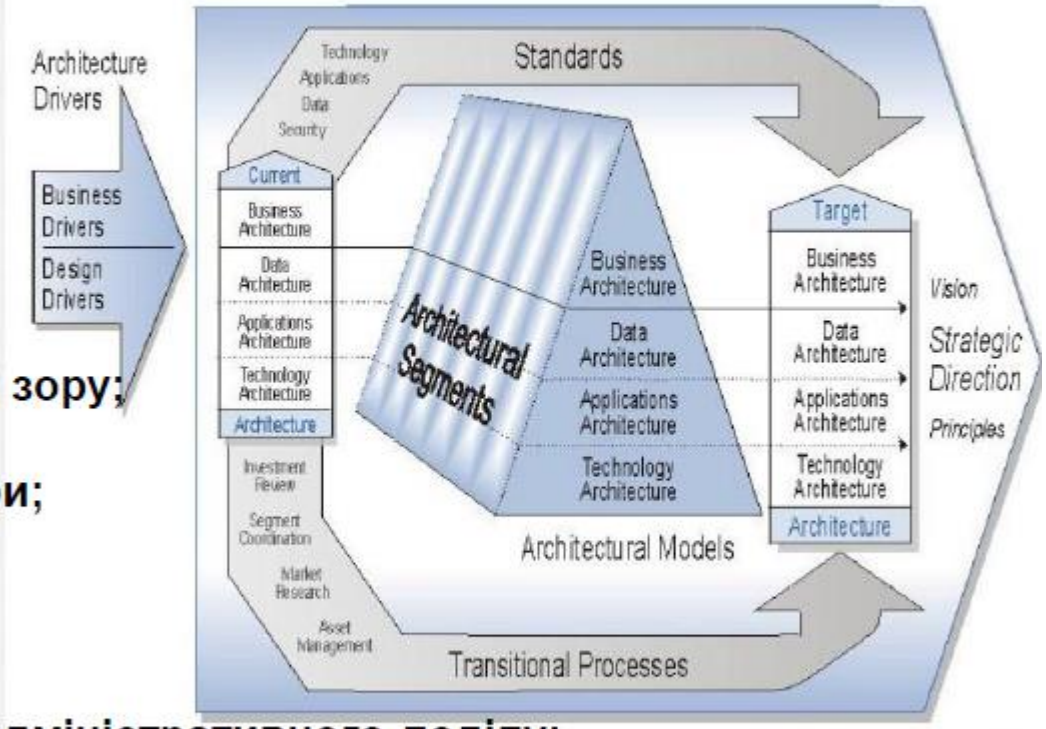
Принципи застосовування:

1. Архітектурний опис має бути чітко орієнтований на проголошені цілі.
2. Архітектурний опис має бути по можливості простим і зрозумілим, але не спрощеним.
3. Архітектурний опис повинен полегшувати, а не ускладнювати процес прийняття рішень.
4. Архітектурний опис має бути складений таким чином, щоб його можна було використовувати для порівняння різних архітектур.
5. При складанні архітектурного опису повинні в максимальному ступені використовуватися стандартні типи даних, що визначаються в DM2.
6. Архітектурний опис має виконуватися в термінах самих даних, а не інструментальних
засобів роботи з даними.
7. Архітектурні дані повинні бути організовані у вигляді, зручному для групової роботи.
8. Архітектурний опис має бути побудований таким чином, щоб його можна було використовувати в мережевому середовищі.

Фреймворк FEA

включає набір із п'яти еталонних моделей:

- модель бізнесу;
- модель обслуговування;
- модель компонентів;
- технологічна модель;
- модель даних.



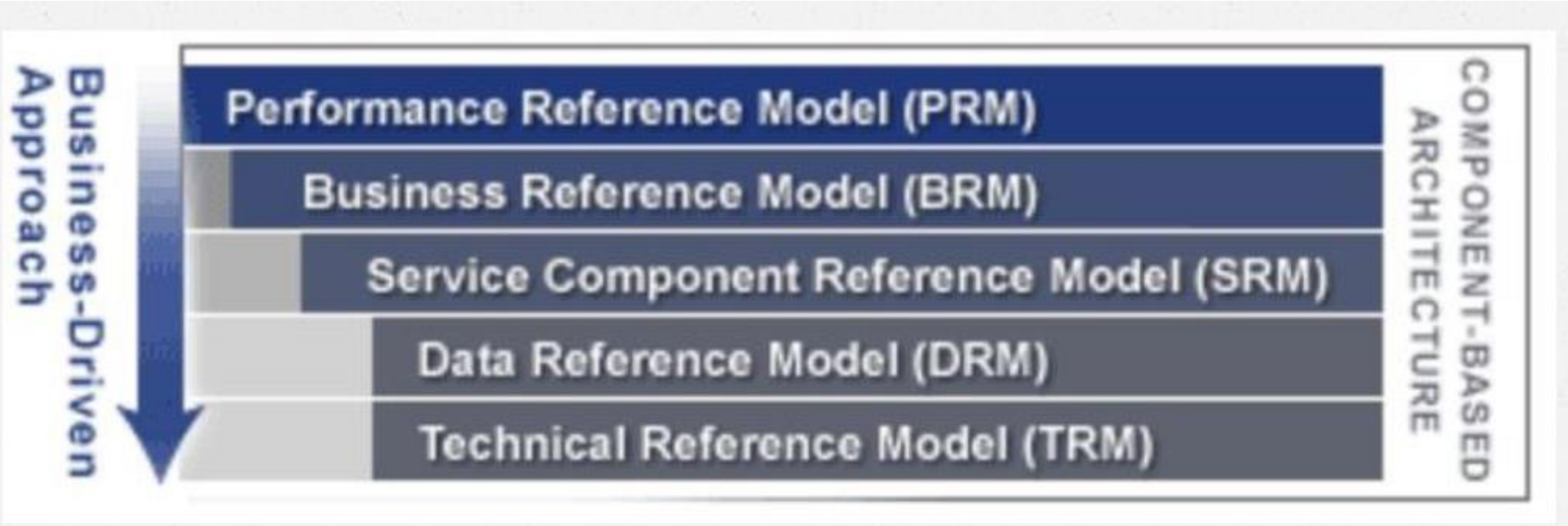
Повний опис FEA складається з:

- погляду на архітектури;
- еталонних моделей, що описують різні точки зору;
- процесу створення архітектури;
- процесу переходу від старої нової архітектури;
- таксономії для класифікації активів;
- методики оцінки успішності застосування.

Сегменти:

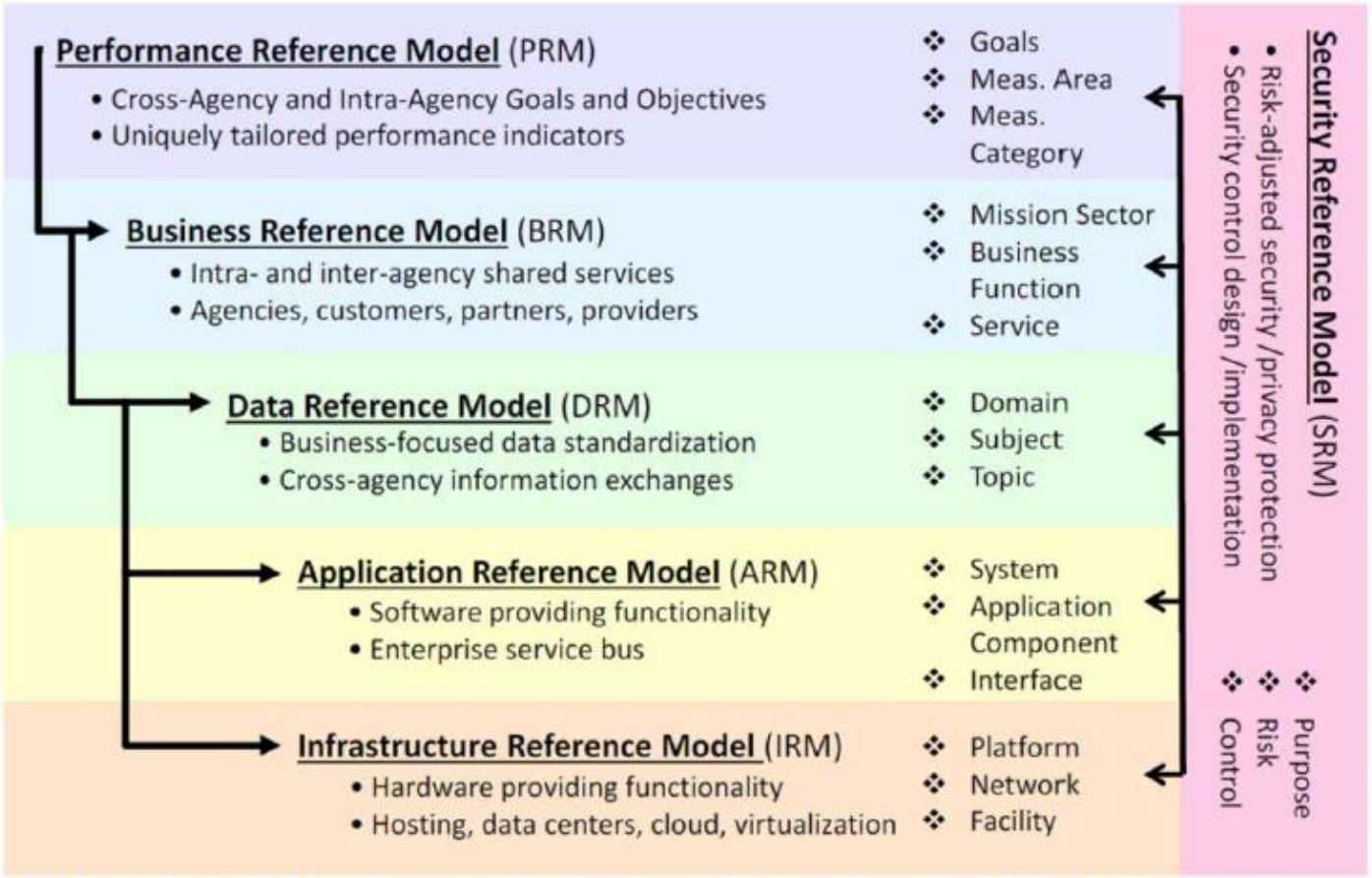
- базовий – аспект діяльності у межах політико-адміністративного поділу;
- службовий – сегмент, що є фундаментальним для більшості організацій

Служба – це чітко визначена функція у межах політико-адміністративного поділу



- Еталонна модель компонентів (CRM) дає IT-представлення систем, що підтримують бізнес.
- Технічна еталонна модель (TRM) визначає різні технології та стандарти, які використовуються при побудові IT-систем.
- Еталонна модель даних (DRM) визначає стандартні способи опису даних.
- Еталонна модель продуктивності (PRM) визначає стандартні способи опису корисності, що забезпечується архітектурами підприємств.
- Еталонна Бізнес-модель (BRM) — структура для опису ділових операцій федерального уряду незалежно від агенцій, які їх виконують.

FEAF-II



- **Еталонна модель додатків (ARM) класифікує стандарти та технології, пов'язані з системою та додатками, які підтримують надання можливостей обслуговування**
- **Еталонна модель інфраструктури (IRM) стандарти та технології, пов'язані з мережею/хмарою, для підтримки та забезпечення доставки компонентів і можливостей передачі голосу, даних, відео та мобільних послуг.**
- **Еталонна модель безпеки (SRM) забезпечує спільну мову та методологію для обговорення безпеки та конфіденційності в контексті бізнес-цілей федеральних агентств і їх ефективності**

Процес розробки архітектури сегмента:

1. Аналіз архітектури: формування уявлення сегмента, враховуючи план організації.
2. Архітектурне визначення: вказівка необхідного стану сегмента, документування показників продуктивності, розгляд альтернатив та розробка архітектури підприємства для сегмента (бізнесу, даних, служб, технологічної).
3. Стратегія інвестицій та фінансування: аналіз способів фінансування проекту.
4. План управління програмою та реалізація проектів: створення плану управління проектом, його реалізації (контрольні точки, показники продуктивності).

Категорії оцінювання:

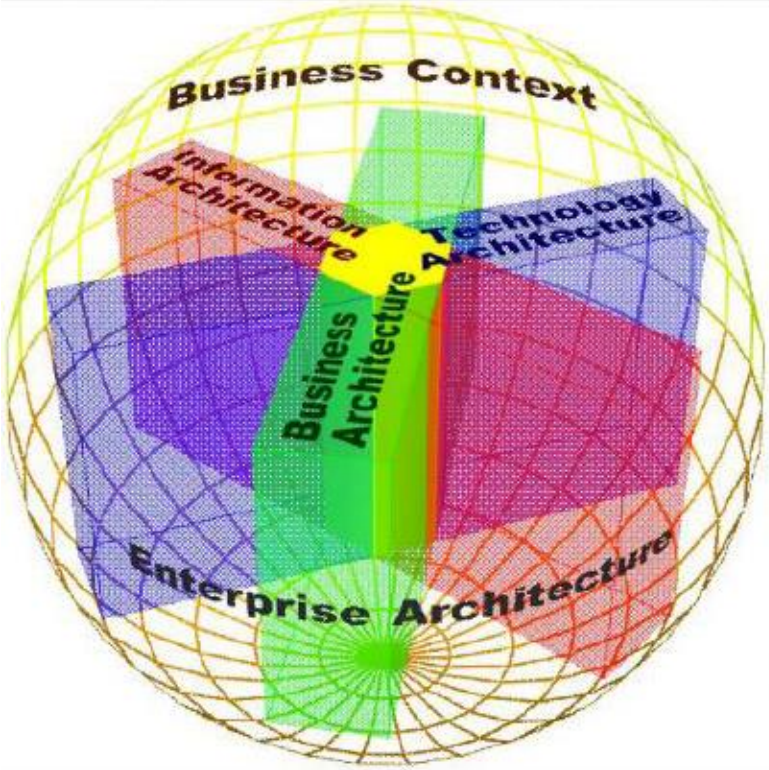
- завершеність архітектури – рівень готовності архітектури;
- використання архітектури - ефективність використання архітектури під час прийняття рішень;
- результати використання архітектури - переваги, досягнуті завдяки використанню архітектури.

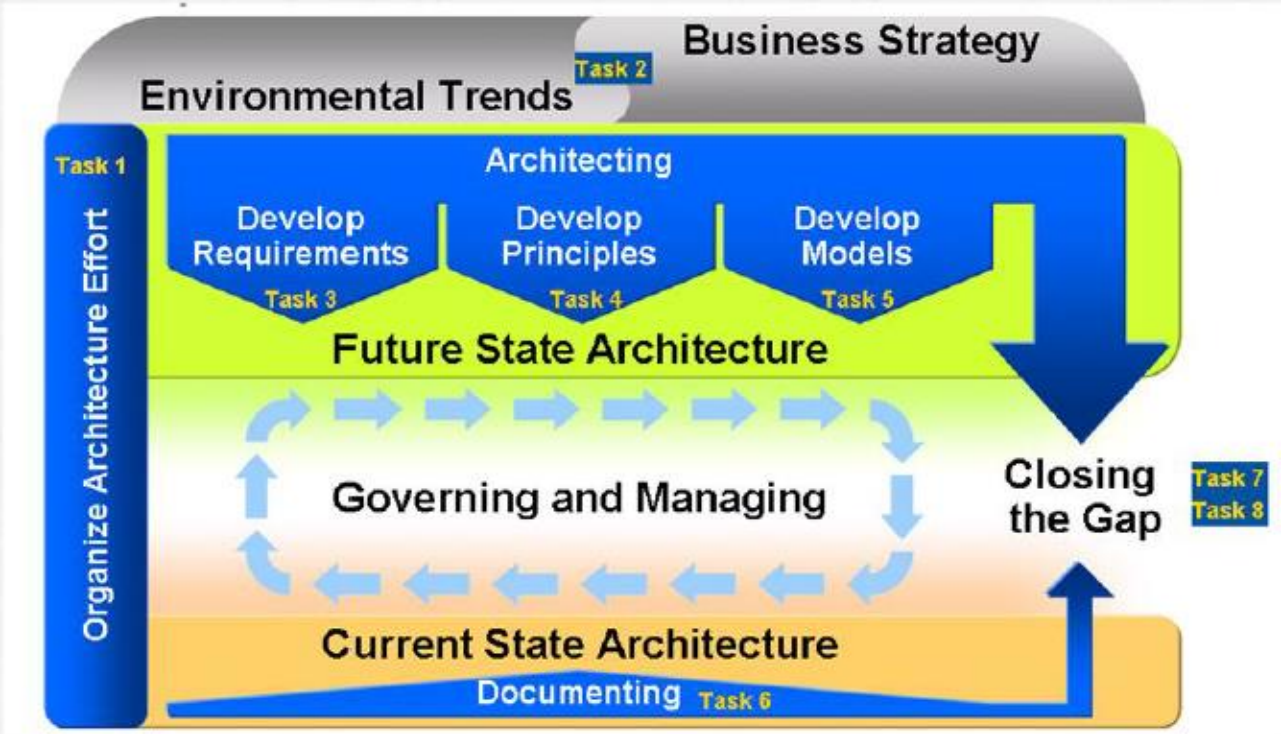
Рейтинг успіху на основі оцінок за кожною категорією та сукупним показником:

- зелений - агентство показало хороші результати за всіма трьома критеріями;
- жовтий — агентство показало хороші результати у сфері завершеності й у сфері використання, чи області результатів;
- червоний — агентство або завершило розробку архітектури, або неефективно використовує розроблену архітектуру.

GartnerEnterpriseArchitectureFramework (GEAF)

- Бізнес архітектура (Business Architecture) - описує бізнес-процеси та організаційну структуру підприємства.
- Інформаційна архітектура (Information Architecture) – моделює інформаційні потоки всередині підприємства.
- Технічна архітектура (Technology Architecture) - описує технічні рішення (фізично) та алгоритми їх експлуатації.





Фаза 1. Ініціалізація (Phase I - Initiation).

Крок 1. Організація архітектурного процесу (Organize Architecture Effort)

Крок 2. Аналіз ситуації для підприємства (Analyze Enterprise Context).

- Оточуючі тенденції (Environment Trends).
- Бізнес-стратегія (Business Strategy)

Фаза 2. Визначення цільової архітектури (Future State Architecture "Architecting").

Крок 3. Розробка вимог (Develop Requirements).

Крок 4. Розробка принципів (Develop Principles)

Крок 5. Розробка моделей (Develop Models)

Фаза 3. Розробка сучасної архітектури (Current State Architecture).

Крок 6. Документування (Documenting). Фаза 4. Проведення аналізу GAP (Closing the Gap).

Крок 7. GAP аналіз (Analyze Gaps):

- Проведення класифікації всіх існуючих елементів у категорії.
- Виділення різниці між поточним станом та цільовою архітектурою.
- Створення списку невідповідностей між поточною та цільовою архітектурою з розділенням за категоріями.
- Угруповання ідентифікованих невідповідностей щодо рівня їхнього впливу на підприємство.

Крок 8. План міграції (Plan Migration):

- Напрямок розвитку бізнес-процесів та інформаційних технологій у середньостроковий та довгостроковий періоди часу.
- Принципи реалізації, що визначають «правила» внесення змін до структури підприємства.
- Динамічність – планування змін з урахуванням постійної зміни технологій та вдосконаленням організаційної структури підприємства.

Порівняння підходів

Характеристика	Опис	ZF	TOGAF	FEA	Gartner
Повнота таксономії	Наскільки методологія придатна для класифікації різних архітектурних артефактів	4	2	2	1
Повнота процесу	Наскільки повно у методології представлено покроковий процес створення архітектури підприємства	1	4	2	3
Посібник по еталонних моделях	Корисність методології створення адекватного набору еталонних моделей. На цьому практично повністю зосереджено методологію FEA	1	3	4	1
Практична інструкція	Наскільки методологія дозволяє втілити у життя умоглядне уявлення про архітектуру підприємства та сформувати культуру, в якій ця архітектура використовуватиметься	1	2	2	4
Модель готовності	Наскільки методологія дозволяє оцінити ефективність використання архітектури підприємства у різних підрозділах	1	1	3	2
Орієнтованість на бізнес	Чи орієнтована методологія на використання технології підвищення цінності бізнесу, де цінність бізнесу визначається як зниження витрат або збільшення доходів	1	2	1	4
Інструкція по керуванню	Наскільки методологія корисна у розумінні та створенні ефективної моделі управління для архітектури підприємства	1	2	3	3
Інструкція по розбиттю	Корисність методології в ефективному розбиття підприємства на відділи, що дуже важливо при управлінні складністю	1	2	4	3
Наявність каталогу	Наскільки ефективно методологія дозволяє створити каталог архітектурних архівів, які можна буде використати надалі	1	2	4	2
Нейтральність по відношенню до постачальників послуг	Імовірність того, що при впровадженні методології ви виявитеся прив'язаними до конкретної консалтингової організації	2	4	3	1
Доступність інформації	Кількість та якість безкоштовних або відносно недорогих матеріалів за даною методологією	2	4	2	1
Термін окупності інвестицій	Тривалість періоду, протягом якого ви використовуватимете цю методологію, перш ніж зможете побудувати на її основі рішення, що забезпечують високу цінність бізнесу	1	3	1	4
РАЗОМ:		17	31	31	29

Лекція 6

СИСТЕМИ МАСОВОГО ОБСЛУГОВУВАННЯ (СМО). БАЗОВІ ПОНЯТТЯ І ПОКАЗНИКИ ЕФЕКТИВНОСТІ СМО

СТРУКТУРА СМО ТА РОЗПОДІЛ ПОТОКУ ВИМОГ

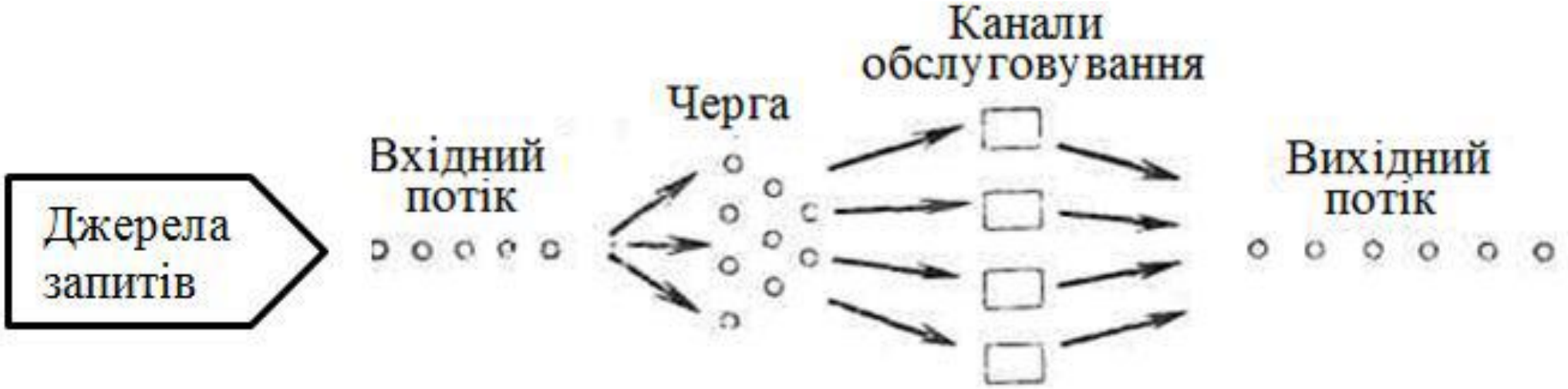
БАЗОВІ ПОНЯТТЯ СИСТЕМ МАСОВОГО ОБСЛУГОВУВАННЯ

“Сучасні комп’ютерні системи складаються з низки компонентів, а саме: процесори, оперативна пам’ять, пристрої вводу-виводу інформації (клавіатура, монітор, принтер, миша та ін.), модулі довготривалої пам’яті.

Ці компоненти можуть працювати взаємопов’язано й незалежно один від одного, що дає змогу організувати мультипрограмний режим роботи під управлінням операційної системи. Операційна система виконує низку функцій (робіт): управління виконанням програм користувачів, управління вводом-виводом, реагування на сигнали із зовнішнього середовища і надзвичайні ситуації. Здійснення кожної з цих функцій полягає у виконанні відповідних програм, які

викликаються за необхідності. Для робіт, які забезпечують базові функції операційної системи, використовують термін «процес». Поняття процесу можна розширити, включивши до нього забезпечення функцій користувачів. Тому виконання програм користувачів - процес. Для всіх таких компонентів комп'ютерної системи використовується термін «**ресурс**». Щоб запустити процес на будь-якому ресурсі, відповідна програма ініціює **програмний запит**. Зазвичай ресурс може обробляти (обслуговувати) в певний момент часу **тільки один запит**, і якщо під час обслуговування запиту від одного процесу надійде запит на обслуговування від іншого процесу, то він **ставиться в чергу** до відповідного ресурсу.

Під **системою масового обслуговування** (далі СМО) розуміють складну систему, що складається з одного або декількох джерел запитів (заявок, вимог) на виконання певних дій (обслуговування), декількох приладів обслуговування (ліній обслуговування, каналів обслуговування), що виконують ці дії відповідно до певних правил (дисципліни обслуговування) за запитами, що надійшли в систему.



Характерною особливістю СМО є ймовірнісність процесів, що відбувається, і можливість утворення черги запитів на обслуговування.

Всі процеси в системах масового обслуговування є ймовірнісними, для їхнього дослідження необхідно використовувати теорію випадкових процесів.

Теорія масового обслуговування **становить новий напрям у теорії ймовірностей, що сформувався як самостійна наукова дисципліну, внаслідок специфіки застосовуваного математичного апарата й важливості розв'язуваних практичних завдань.**

Розроблення практичних завдань масового обслуговування започаткував співробітник Копенгагенської телефонної компанії данський математик А. К. Ерланг (1878–1929 рр.) у період із 1908 по 1922 роки. У 1909 р. було опубліковано його робота «Теорія ймовірностей і телефонні переговори» та інші, у яких сформульовано перші прикладні завдання телефонії. Ці завдання були пов’язані з необхідністю впорядкувати роботу телефонної мережі й розробити методи оцінювання якості обслуговування споживачів залежно від кількості використовуваних пристроїв. Значний внесок зробив Л. Клейнрок, який використовував ТСМО для дослідження обчислювальних систем.

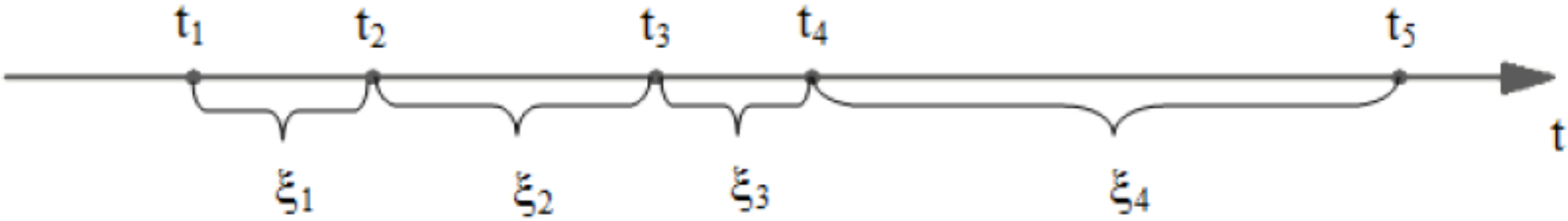
Теорія систем масового обслуговування – це комплексна дисципліни, яка базується на низці класичних математичних дисциплін, а саме: теорія ймовірностей, математична статистика, теорія випадкових процесів. Процеси масового обслуговування можуть описуватися системами лінійних рівнянь або системами диференціальних рівнянь.

Базові поняття СМО.

Джерело запитів визначається як зовнішня щодо СМО система, із якої запити надходять в неї для обслуговування. Джерело називають нескінченним або кінцевим залежно від того, нескінченна чи кінцева кількість запитів міститься в ньому. Якщо джерело містить кінцеву, але досить велику кількість запитів, то його зазвичай вважають нескінченним.

Наприклад, хоча кількість користувачів інформаційно-пошукової системи Google кінцева, припускають, що вони утворюють нескінченне джерело

Процес надходження запитів для обслуговування в СМО є ймовірнісним і утворює потік однорідних або неоднорідних подій, які настають через випадкові проміжки часу, де t_i – моменти надходження запитів, що утворюють потік випадкових подій; ξ_i – інтервали часу між сусідніми запитами).



Вхідний потік. Запити, що надходять з нескінченного джерела, прибувають в канал обслуговування в моменти часу $t_0 < t_1 < \ldots < t_k < \ldots$.

Інтервали часу $\xi_k = t_k - t_{k-1}\,(k > 1)$ між послідовними моментами надходження запитів є **випадковими величинами**. Передбачається, що вони утворюють послідовність незалежних і однаково розподілених випадкових величин з функцією розподілу

$$A(t) = P\{\xi_k < t\}.$$

Випадкові часові інтервали між настанням подій у потоці можуть характеризуватися різними законами розподілу. Однак у більшості робіт щодо теорії масового обслуговування, особливо прикладних, розглядається пуассонівський (найпростіший) потік, у якому

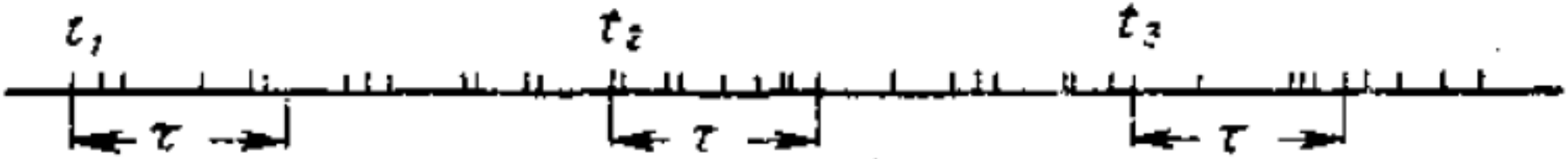
ймовірність того, що в проміжку часу t відбудеться рівно k вимог, задається формулою Пуассона

$$p_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}$$

Найпростіший потік характеризується трьома головними властивостями: стаціонарність, відсутність післядії й ординарність.

Випадковий потік називається **стаціонарним**, якщо ймовірність надходження певної кількості запитів протягом певного відрізка часу **залежить від його величини й не залежить від початку його відліку на осі часу**. Якщо на часовій осі відкласти рівні, які не перетинаються, інтервали часу τ, то ймовірність появи в цих інтервалах певної кількості запитів для цього потоку залежить від

величини τ й не залежить від розміщення цього інтервалу на часовій осі.



Базовим параметром потоку є інтенсивність потоку *λ(t)* – математичне сподівання кількості подій за одиницю часу

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{m_k(t, t+\Delta t)}{\Delta t}$$

тобто, це границя відношення математичного сподівання кількості подій на інтервалі $(t, t + \Delta t)$ до довжини цього інтервалу, що наближається до нуля.

Якщо величини ξ_i є незалежними і однаково розподіленими, то такий потік називається потоком з обмеженою післядією (потік Пальма).

Таким чином, базовим описом потоку є функція розподілу

$$A(t) = P\{\xi < t\}.$$

Такі потоки є головними в теорії масового обслуговування.

Кожен запит, що надійшов на обслуговування, перебуває на приладі обслуговування випадковий час η, що описується функцією розподілу $B(t) = P\{\eta < t\}$. Якщо припустити, що на вході до приладу обслуговування весь час перебувають запити, то на виході обслуговуючого приладу сформується потік обслужених запитів з функцією розподілу *B(t)*. Отже, обслуговування відбувається так, ніби на запит, що надійшов на обслуговування, спрямовується потік обслуговування. Така постановка завдання дуже зручна методично, оскільки уможливлює узагальнений розгляд процесів надходження й обслуговування. Таким чином, у системі масового обслуговування функціонують два видів потоків – надходження й обслуговування, що характеризуються різними функціями розподілу.

Таким чином, два найпростіші потоки різняться тільки своїми параметрами, тому для завдання найпростішого потоку досить задати лише його параметр λ.

Відсутність післядії полягає в тому, що ймовірність надходження за відрізок часу τ певної кількості вимог не залежить від того, скільки вимог вже надійшло в систему, тобто не залежить від передісторії досліджуваного явища. Відсутність післядії - незалежність перебігу процесу за проміжки часу, які не перекриваються.

Ординарність потоку вимог означає практичну неможливість появи двох і більше вимог в один і той самий момент часу. Таким чином, **найпростіший потік – це стаціонарний, ординарний потік без післядії**.

Процес обслуговування. Нехай η_k – тривалість обслуговування k-го запиту. Передбачається, що η_k (k = 1, 2, ...) – незалежні однаково розподілені випадкові величини з функцією розподілу $B(t) = P\{\eta_k < t\}$. Функція $B(t)$ називається розподілом тривалості обслуговування. Припускаємо, що існує щільність імовірності $b(t)=B'(t)$. Як і для опису вхідних потоків запитів у теорії масового обслуговування використовують обмежений набір законів розподілу для опису часу обслуговування. Найпоширенішим є експоненціальний закон розподілу з функцією розподілу

$$B(t)=1-\exp(-\mu t) .$$

Параметр μ інтерпретується як інтенсивність обслуговування. Середній час обслуговування дорівнює $1/\mu$. Крім експоненціального

розподілу, використовують ерлангівський і гіперекспоненціальний розподіли, які отримують шляхом перетворення експоненціального закону.

Канали обслуговування. СМО може мати один або більше каналів (ліній, приладів) обслуговування. СМО з одним каналом називаються одноканальними або однолінійними, тоді як системи обслуговування, що містять більше каналів обслуговування, називаються багатоканальними або багатолінійними. Прилади обслуговування можуть бути однорідними й неоднорідними. У СМО з однорідними приладами всі прилади обслуговують запити однаково. У СМО з неоднорідними приладами прилади відрізняються один від одного деякими параметрами, наприклад інтенсивністю обслуговування.

Дисципліна обслуговування. Правило, згідно з яким запити вибираються для обслуговування, складають дисципліну обслуговування.

Умовно всі дисципліни обслуговування по наданих переваг в обслуговуванні діляться на дві групи: безпріоритетні і пріоритетні. Кожна з цих груп ділиться на ряд підгруп.

Безпріоритетні дисципліни обслуговування поділяються на дисципліну обслуговування в порядку надходження, у зворотному порядку, із випадковим вибором з черги та циклічну дисципліну обслуговування.

Дисципліна обслуговування в порядку надходження (**FIFO – First Input First Output**) є найпоширенішою, більшість досліджень щодо теорії

масового обслуговування виконуються для цієї дисципліни. Вона широко використовується в операційних системах.

Дисципліна обслуговування в зворотному (інверсному) порядку (**LIFO – Last Input First Output**) використовується в операційних системах під час обробки переривань і організації стеків.

Циклічні дисципліни обслуговування використовуються під час обробці інформації в режимі поділу часу.

При **пріоритетних** дисциплінах обслуговування з черги на обслуговування спочатку вибираються заявки з вищим пріоритетом. Вони поділяються на дисципліни з фіксованими й динамічними пріоритетами.

При дисципліні обслуговування з відносним пріоритетом не дозволяється переривання обслуговування запиту на каналі.

Якщо в систему з дисципліною обслуговування з абсолютним пріоритетом надійде запит з пріоритетом вищим, ніж той що обслуговується, то він припинить обслуговування цього запиту й надійде на обслуговування.

Системи з абсолютним пріоритетом розрізняють за кількістю рівнів пріоритету, та за алгоритмами для обслуговування перерваних запитів.

При дисциплінах обслуговування з динамічним пріоритетом пріоритет конкретних запитів змінюється залежно від змінювання деяких величин, наприклад часу очікування в черги.

За наявністю певної ознаки системи масового обслуговування можна класифікувати так:

1. *За кількістю вимог, що надходить за одиницю часу,* на системи з *ординарним і неординарним потоками вимог*. Якщо імовірність надходження двох і більше вимог одночасно дорівнює нулю або має настільки мале значення, що ним можна знехтувати, то отримаємо систему з ординарним потоком вимог. Наприклад, потік вимог – літаки, що надходить на злітно-посадкову смугу аеродрому, можна вважати ординарним.

2. За *зв'язком між вимогами* – на системи *без післядії і з післядією.*

Якщо ймовірність надходження вимог у систему в деякий момент часу не залежить від того, скільки вимог уже надійшло, тобто не пов'язана з передісторією досліджуваного процесу, то отримаємо систему без післядії, в іншому разі – із післядією. Прикладом системи з післядією може слугувати потік студентів, що здають залік викладачу.

3. За *реакцією вимоги на зайнятість каналів* – на системи *з відмовами й очікуваннями.* Якщо вимога, яка надійшла на обслуговування, застала всі канали зайнятими і змушена залишити систему, то отримаємо систему з відмовами.

4. *Системи з очікуванням* розподіляються на системи з *обмеженим* і *необмеженим* очікуванням. Якщо вимога залишає систему, коли черга

набула певного розміру, то отримаємо систему з обмеженим очікуванням.

Прикладом може слугувати самоскид з розчином. Якщо час очікування настільки великий, що розчин може затвердіти, то самоскид доречно розвантажити в іншому місці. Якщо вимога, яка надійшла, застала всі канали зайнятими і змушена очікувати своєї черги доти, доки вона не буде обслужена, то отримаємо систему з очікуванням без обмежень. Приклад: літак, що перебуває на аеродромі й очікує звільнення злітної смуги.

5. *За способом вибору вимог на обслуговування* розподіляються так:

– із пріоритетом вимог;

– у процесі надходження вимог;

– із випадковим вибором вимог;

– остання вимога обслуговується першою.

Якщо система масового обслуговування охоплює декілька категорій вимог і за якимись ознаками визначається порядок їх вибору на обслуговування, то отримаємо систему з пріоритетом вимог. Приміром, під час надходження виробів на будівельний майданчик насамперед монтують ті, що обумовлені будівельною технологією.

Якщо канал, що звільнився, обслуговує вимогу, яка надійшла в систему раніше за інші, то отримаємо систему обслуговування вимог

у процесі їх надходження. Наприклад, покупець, що підійшов першим до продавця, обслуговується першим.

Якщо вимоги з черги в канал обслуговування надходить випадково, то отримаємо систему з випадковим вибором вимог на обслуговування.

Приклад: вибір слюсарем-сантехніком однієї з декількох заявок від мешканців, із часом надходження яких він не ознайомлений.

Якщо для обслуговування обирається остання вимога, що надійшла, то отримаємо систему з вибором «останній обслуговується першим». Приміром, під час укладання будівельних виробів штабелями зручніше обирати зі штабеля (черги) виріб, покладений останнім.

6. За *часом обслуговування* вимоги – на системи з *детермінованим* і *випадковим часом* обслуговування. Якщо інтервал часу між моментом надходження вимоги в канал обслуговування і моментом виходу вимоги з каналу постійний, то отримаємо систему з детермінованим часом обслуговування, в іншому разі – із випадковим. Наприклад, миття автомобілів становить систему обслуговування з детермінованим часом обслуговування.

7. За *кількістю каналів обслуговування* – на одно- й багатоканальні системи. Приміром під час монтажу будинку може використовуватися один підіймальний кран (один канал обслуговування) або декілька (багато каналів обслуговування).

8. За *кількістю етапів обслуговування* – на одно- й багатофазні системи.

Якщо канали обслуговування розташовуються послідовно й неоднорідні, то отримаємо багатофазну систему обслуговування. Прикладом такої системи може слугувати обслуговування автомобілів на СТО (миття, діагностика, заміна фільтрів тощо).

9. За *однорідністю вимог* – на системи з *однорідними* й *неоднорідними потоками* вимог. Приміром, якщо під навантаження прибувають фургони однієї вантажопідйомності, то отримаємо систему з однорідним потоком вимог, якщо різної – із неоднорідним.

10. *За завантаженістю каналів* – на *впорядковані* й *невпорядковані*

системи. У впорядкованих системах обслуговуючі канали завантажені нерівномірно. Вимога, що надійшла, обслуговується чіткого визначеним каналом із наявних вільних, а саме каналом з найменшим номером (вважається, що всі канали пронумеровані). У невпорядкованих системах усі канали однакові й вимога, що надійшла, обслуговується одним із вільних каналів без будь-яких переваг.

Позначення розімкнутих безпріоритетних невпорядкованих систем масового обслуговування: ***A/B/n/m***. ***A*** і ***B*** - закони розподілу інтервалів часу між подіями вхідного потоку запитів і часом обслуговування, ***n*** - кількість каналів обслуговування, ***m*** – кількість місць очікування. Закони розподілу: ***M*** – експоненціальний розподіл, ***E*** або E_k – ерланговський; ***H*** або H_r – гіперекспоненціальний; ***D*** – регулярний потік подій; ***G*** – розподіл загального вигляду. ***D/H2/3/10*** - система з 3 каналами обслуговування, постійним (детермінованим) часом між запитами вхідного потоку, час обслуговування розподілено за гіперекспоненціальним законом другого порядку й кількістю місць для очікування – 10. Якщо параметр ***m*** відсутній, то це означає, кількість місць для очікування нескінченна.

Показники ефективності систем масового обслуговування

Технічні показники

1) Імовірність *відмови обслуговування.* Імовірність того, що вимога, яка надходить у систему, відмовиться приєднатися до черги і *втратиться* системою – P_{vidm}. Цей показник для системи масового обслуговування з відмовами дорівнює імовірності того, що в системі міститься стільки вимог, скільки є каналів обслуговування:

$$P_{vidm} = P_n ,$$

де n – кількість каналів обслуговування.

Для системи з обмеженою довжиною черги P_{vidm} дорівнює імовірності

того, що в системі міститься $n + m$ вимог:

$$P_{vidm} = P_{n+m},$$

де m – допустима довжина черги.

Протилежним показником є імовірність обслуговування вимоги:

$$P_{obsl} = 1 - P_{vidm};$$

2) середня кількість вимог, що очікують на обслуговування:

$$L_q = \sum_{i=n+1}^{n+m} (i-n) p_i$$

де p_i – імовірність того, що в системі міститься i вимог;

3) відносна й абсолютна пропускні здатності системи:

– відносна пропускна здатність:

$$Q = 1 - P_{vidm};$$

– абсолютна пропускна спроможність:

$$A = \lambda Q;$$

4) середня кількість зайнятих обслуговуванням каналів для систем

G/G/n/m (тобто *n* – обслуговуючі пристрої, *m* – місця для чекання):

$$\overline{n} = \sum_{i=1}^{n-1} i p_i + n \sum_{i=n}^{n+m} p_i$$

Для систем з відмовами середню кількість зайнятих обслуговуванням каналів можна знайти за формулою:

$$\overline{n} = \sum_{i=1}^{n} i \cdot p_i$$

5) загальна кількість вимог, що містяться в системі L_s:

– для систем масового обслуговування з відмовами:

$$L_s = \overline{n}$$

– для систем з обмеженою довжиною черги:

$$L_s = \overline{n} + L_q$$

6) середній час очікування вимогами початку обслуговування. Якщо

відома функція розподілу ймовірності часу очікування вимогою початку обслуговування $W(t) = P(T_q < t)$, то середній час очікування вимогами початку обслуговування Tq визначається, як математичне очікування випадкової величини T_q :

$$\tau_q = M[T_q] = \int_0^\infty t dW(t)$$

При показовому (експоненціальному) законі розподіли вимог у вхідному потоці τ_q можна визначити за формулою

$$\tau_q = \frac{L_q}{\lambda}$$

Економічні показники ефективності систем масового обслуговування

Показники, що характеризують економічні особливості систем масового обслуговування, зазвичай формують відповідно до певного виду системи та її призначення. Одним із загальних показників є економічна ефективність системи:

$$G = \lambda P_{obsl} C_s T - E,$$

де C_s – середній економічний ефект, отриманий під час обслуговуванні однієї вимоги; T – розглянутий інтервал часу; E – величина втрат у системі:

– для систем з відмовами

$$E = (c_e \overline{n} + \lambda c_{zb} P_{vidm} + c_{prost} \overline{n}_{vi\ln})T$$

де c_e – вартість експлуатації одного каналу за одиницю часу; c_{zb} – вартість збитків унаслідок виходу вимог із системи за одиницю часу; c_{prost} – вартість одиниці часу простоювання каналу обслуговування; $\overline{n}_{vi\ln} = n - \overline{n}$ – середня кількість вільних каналів (що простоюють),

– для систем з очікуванням:

$$E = (c_e \overline{n} + c_q L_q \lambda + c_{zb} \overline{n}_{vi\ln})T$$

де c_q – вартість втрат, пов’язаних із простоюванням вимоги в черзі за одиницю часу” [20].

Лекція 7

МЕТОДИ ДОСЛІДЖЕННЯ СМО. МАТЕМАТИЧНА МОДЕЛЬ СМО З ОДНОКАНАЛЬНОЮ ЧЕРГОЮ І ПУАСОНІВСЬКИМ ПОТОКОМ ВИМОГ. ДОСЛІДЖЕННЯ СТАЦІОНАРНОГО СТАНУ СМО

Методи дослідження систем масового обслуговування

“Головним завданням дослідження СМО є встановлення залежностей базових характеристик, а саме: **середній час очікування, імовірність відмови обслуговування,** від параметрів системи – **інтенсивностей обслуговування на приладах, кількості приладів обслуговування, дисципліни обслуговування**. Різноманітність СМО унеможливлює використання для аналізу одного методу. Хоча на практиці для певної групи СМО зазвичай використовується один метод, що дає змогу мінімізувати обчислення й скоротити час визначення базових залежностей.

Експоненціальним (показниковим) називають розподіл ймовірностей неперервної випадкової величини *X*, яка описується щільністю розподілу *f(x)* та функцією розподілу *F(x)*:

$$f(x)=\begin{cases}0, & x<0,\\ \lambda e^{-\lambda x}, & x\geq 0,\end{cases} \qquad F(x)=\begin{cases}0, & x<0,\\ 1-e^{-\lambda x}, & x\geq 0.\end{cases}$$

де λ – постійна додатна величина.

Числові характеристики: математичне сподівання – $m_x=1/\lambda$,

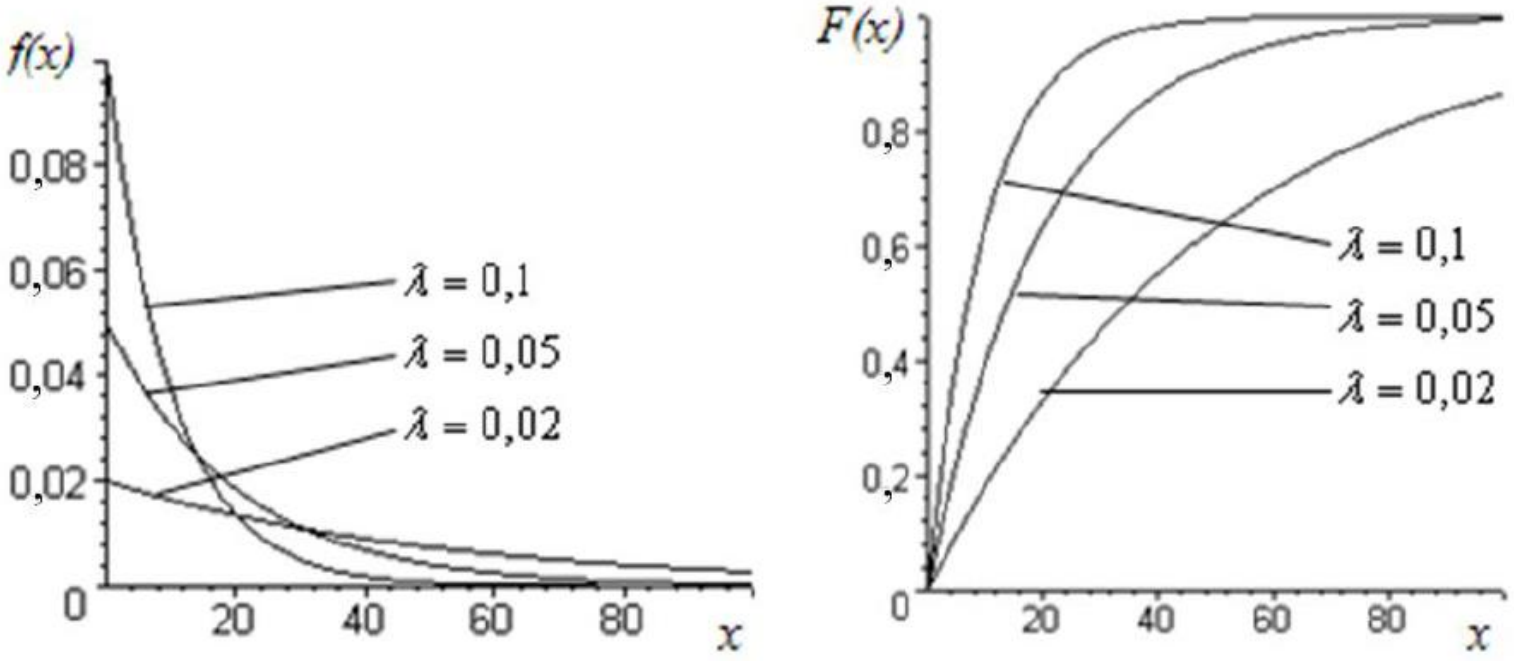
дисперсія – $D_x=1/\lambda^2$,

середньоквадратичне відхилення – $1/\lambda$,

мода – $M_0=0$, медіана – $M=\ln 2/\lambda$.

Для експоненціально розподіленої випадкової величини *X* імовірність того, що *X* набуде значення в інтервалі *(α,β)* дорівнює

$$P(\alpha < X < \beta) = \int_{\alpha}^{\beta} \lambda e^{-\lambda x} dx = e^{-\lambda\alpha} - e^{-\lambda\beta}$$



Рівномірним називають розподіл ймовірностей неперервної випадкової величини *X*, якщо на інтервалі *(a,b)*, до якого належать усі значення випадкової величини, щільність розподілу постійна:

$$f(x)=\begin{cases}0, & x<a,\\ \dfrac{1}{b-a}, & a\le x\le b,\\ 0, & x>b.\end{cases}\qquad F(x)=\begin{cases}0, & x<a,\\ \dfrac{x-a}{b-a}, & a\le x\le b,\\ 1, & x>b.\end{cases}$$

Числові характеристики: математичне сподівання – $m_x=(b+a)/2$,

дисперсія – $D_x=(b-a)^2/12$,

середньоквадратичне відхилення – $(b-a)/\,12^{1/2}$,

медіана – $M=(b+a)/2$.

У теорії масового обслуговування розглядається пуассонівський (найпростіший) потік, у якому ймовірність того, що в проміжку часу t відбудеться рівно k вимог, задається формулою Пуассона

$$p_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}$$

Функція розподілу інтервалів часу ξ між сусідніми подіями пуассонівського потоку задається експоненціальним розподілом

$$A(t) = 1 - e^{-\lambda t}$$

Щільність розподілу – $a(t) = A'(t) = \lambda e^{-\lambda t}$.

Найпростіший потік в теорії масового обслуговування відіграє таку саму роль, як і нормальний закон розподілу випадкових величин у теорії ймовірностей. У разі підсумовування (взаємного накладання) великої кількості ординарних стаціонарних потоків фактично з будь-якими розподілами формується потік, наближений до найпростішого. Найпростіший потік відіграє важливу роль в теорії моделювання. По-перше, найпростіші й близькі до найпростіших потоки подій дуже часто зустрічаються на практиці. По-друге, навіть при потоці подій, що відрізняється від найпростішого, зазвичай можна знайти задовільні за точністю результати, замінивши потік будь-якої структури найпростішим потоком із тією самою середньою інтенсивністю.

Крім експоненціального, застосовують такі розподіли:

Ерлангівський з функцією розподілу:

$$A(t) = 1 - \sum_{i=1}^{k} \frac{(\lambda t)^{i-1}}{(i-1)!} e^{-\lambda t}$$

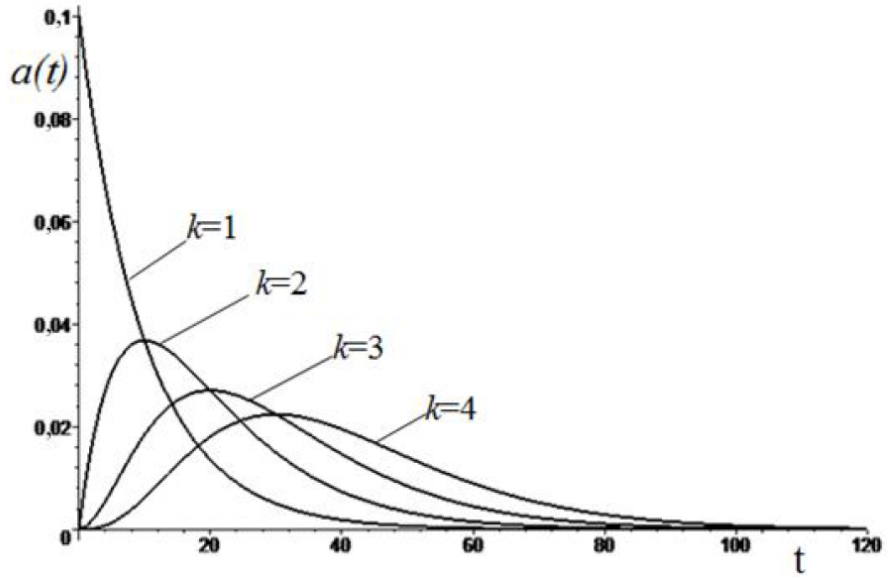
Ерлангівський розподіл формується шляхом підсумовування k інтервалів, кожен з яких розподілений за експоненціальним законом із параметром λ. Отже, потік подій, описуваний ерлангівським розподілом k-го порядку, формується з пуасонівського шляхом видалення всіх подій, що містяться між l-ю і $(l + k)$-ю подіями. При k=1 ерлангівський розподіл перетворюється на експоненціальний.

Щільність ерлангівського розподілу $a(t)$:

$$a(t) = \frac{1}{(k-1)!}\lambda(\lambda t)^{k-1}e^{-\lambda t}$$

Математичне сподівання $m=k/\lambda$,

середньоквадратичне відхилення $\sqrt{k}/\lambda$.



Гіперекспоненціальний з функцією розподілу і шільністю

$$A(t)=1-\sum_{i=1}^{k}a_i e^{-\lambda_i t},\ \sum_{i=1}^{k}a_i=1 \qquad a(t)=\sum_{i=1}^{k}a_i\lambda_i e^{-\lambda_i t},\ \sum_{i=1}^{k}a_i=1.$$

Рекомендовано використовувати гіперекспоненціональний розподіл другого порядку, що має тільки два параметри λ і φ із функцією розподілу $A(t)=1-\varphi e^{-2\varphi\lambda t}-(1-\varphi)e^{-2(1-\varphi)\lambda t},\ 0<\varphi\le 0{,}5$

Потік подій, який описується гіперекспоненціональним розподілом формується, якщо з імовірністю $a_i, \sum a_1=1$, черговий інтервал часу між настанням сусідніх подій обирається розподіленим за експоненціальним законом з параметром λ_i .

У математиці випадковим процесом називається функція двох аргументів, значення якої – випадкові величини

$$\zeta(t) = \varphi(n,t),\ n = 0,1,\ldots,N,\ t \in T$$

де *N-1* – кількість елементарних подій (воно може бути і нескінченним); *t* – параметр; *T* – множина значень параметра *T*.

Для кожного значення параметра *t* функція $\varphi(n,t)$ є функцією тільки від *n* і, отже, становить випадкову величину. Для кожного фіксованого значення аргументу *n* (тобто для кожної елементарної події) $\varphi(n,t)$ залежить тільки від *t* і є, таким чином, просто функцією одного дійсного аргументу. Кожна така функція є реалізацією випадкового процесу.

Випадкові процеси можна розподілити на чотири види:

- процеси з дискретними станами та дискретним часом;
- процеси з дискретними станами й безперервним часом;
- процеси з безперервними станами й дискретним часом;
- процеси з безперервними станами й безперервним часом.

Процеси систем масового обслуговування становлять процеси з дискретними станами й безперервним часом. **Станом системи можна вважати кількість вимог в системі**. Залежно від виду СМО воно може бути кінцевим (для систем з обмеженою чергою) або нескінченним. При конкретному значенні параметра *t* стан системи змінюється від реалізації до реалізації, а тому можна оцінити тільки його ймовірність або відповідні параметри розподілу.

Надійшовши в систему, запит перебуває деякий час у черги, на обслуговуванні та в системі загалом. Ці характеристики можна досліджувати, за допомогою **теорії випадкових процесів із дискретними станами й безперервним часом**.

Випадковий процес називається **марківським** (або процесом без післядії), якщо для кожного моменту часу *t* **ймовірність будь-якого стану системи в майбутньому обумовлюється тільки її станом на даний момент часу і не залежить від того, яким шляхом система набула цього стану**. Параметр часу *t* може розглядатися **і як дискретний, і як безперервний**, унаслідок чого використовуються **різні методи дослідження**.

Марківські випадкові процеси (ланцюги Маркова) з дискретним часом

Нехай досліджувана система може перебувати в станах

$$\varphi(n,t), n = 0, 1, 2, \ldots, N; t = 0, 1, 2, 3, \ldots.$$

Переходи зі стану $\varphi(i,t)$ в стан $\varphi(j,t)$ відбуваються в дискретні моменти часу t = 0, 1, 2, 3,.... Нехай $p_n(t)$ позначає ймовірність стану $\varphi(n,t)$ у момент часу t. Тоді вектор

$$[P(t)] = [p_0(t), p_1(t), \ldots, p_N(t)]$$

описує ймовірності перебування системи в тому чи іншому стані в дискретний момент часу t.

Позначимо через $p_{ij}(t), 0 \le i, j \le N$ умовну ймовірність того, що система, перебуваючи в момент t в стані $\varphi(i,t)$, перейде в стан $\varphi(j,t+1)$ в момент часу $t+1$ (так звані перехідні ймовірності), тоді

$$p_j(t+1) = \sum_{i=0}^{N} p_i(t) \cdot p_{ij}(t), j = 0,1,2,...,N$$

Це рівняння і початкові умови повністю описує ланцюг Маркова.

Якщо перехідні ймовірності не залежать від часу, то такий ланцюг називається однорідним (в іншому разі неоднорідним). Для однорідного ланцюга Маркова рівняння можна записати так:

$$p_j(t+1) = \sum_{i=0}^{N} p_i(t) \cdot p_{ij}, j = 0,1,2,...,N$$

Матриця

$$[\Pi]=\begin{bmatrix} p_{00} & p_{01} & p_{02} & \cdots & p_{0N} \\ p_{10} & p_{11} & p_{12} & \cdots & p_{1N} \\ p_{20} & p_{21} & p_{22} & \cdots & p_{2N} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{N0} & p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix} \qquad \sum_{j=0}^{N} p_{ij} = 1$$

називається однокроковою матрицею перехідних ймовірностей або стохастичною матрицею.

Рівняння переходу в матричній формі можна записати так

$$[P(t+1)]=[P(t)]\cdot[\Pi]$$

Ймовірності системи за *k* кроківвишлядає так:

$$[P(t+k)] = ([P(t)] \cdot [\Pi]) \cdot [\Pi] = [P(t)] \cdot [\Pi]^k$$

звідки випливає найважливіша характеристика ланцюга Маркова – матриця переходу зі стану $\varphi(i,t)$ в стан $\varphi(j,t+k)$ за *k* кроків

$$[P_{ij}^{k}] = [\Pi]^k$$

Приклад. Нехай є три альтернативні моделі смартфонів: *a*, *b*, *c*. Серед споживачів проведено опитування щодо їхнього ставлення до цих моделей. При цьому визначалася частка (частота) споживачів, які надають перевагу тим чи іншим моделям:

$$p_a(0) = 0{,}6;\ p_b(0) = 0{,}3;\ p_c(0) = 0{,}1$$

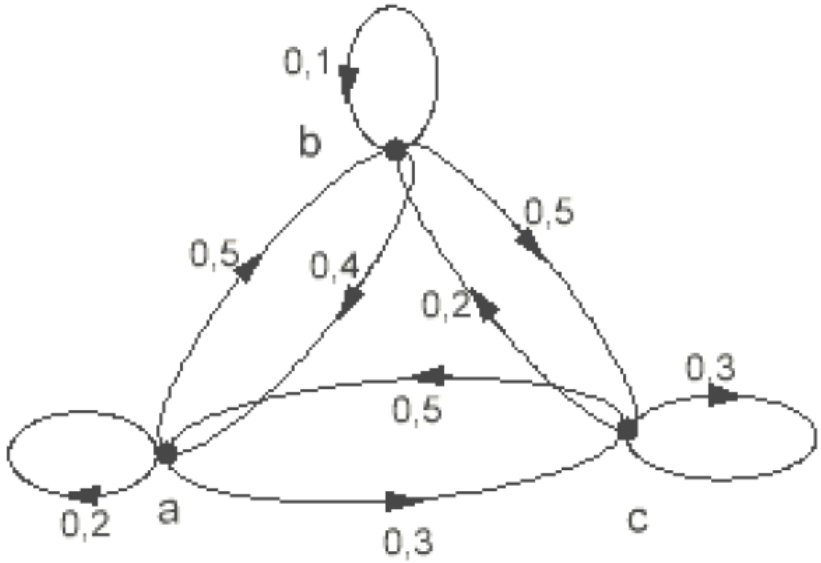
За місяць споживачів опитують щодо того, чи продовжують вони надавати перевагу обраним моделям, чи бажають змінити тип моделі. При цьому було отримано умовні ймовірності переходу:

$$p_{aa}=0{,}2;\ p_{ab}=0{,}5;\ p_{ac}=0{,}3;\ p_{ba}=0{,}4;\ p_{bb}=0{,}1;\ p_{bc}=0{,}5;\ p_{ca}=0{,}5;\ p_{cb}=0{,}2;\ p_{cc}=0{,}3$$

За місяць проводилися нове опитування і т. д. Припустивши, що переваги споживачів не змінюються в часі, отримали однорідний ланцюг Маркова з матрицею перехідних ймовірностей :

$$[\Pi]=\begin{bmatrix} 0{,}2 & 0{,}5 & 0{,}3 \\ 0{,}4 & 0{,}1 & 0{,}5 \\ 0{,}5 & 0{,}2 & 0{,}3 \end{bmatrix}$$

Досліджуваний процес для наочності зручно надати у вигляді графа



Визначимо переваги споживачів за місяць

$$[P(t+1)] = [P(t)] \cdot [\Pi] = [0,6;0,3;0,1] \cdot \begin{bmatrix} 0,2 & 0,5 & 0,3 \\ 0,4 & 0,1 & 0,5 \\ 0,5 & 0,2 & 0,3 \end{bmatrix} = [0,29;0,35;0,36]$$

За два місяці отримаємо

$$[P(t+2)] = [P(t+1)] \cdot [\Pi] = [0{,}29; 0{,}35; 0{,}36] \cdot \begin{bmatrix} 0{,}2 & 0{,}5 & 0{,}3 \\ 0{,}4 & 0{,}1 & 0{,}5 \\ 0{,}5 & 0{,}2 & 0{,}3 \end{bmatrix} = [0{,}378; 0{,}252; 0{,}37]$$

$$[P(t+3)] = [P(t)] \cdot [\Pi]^3 = [0{,}6; 0{,}3; 0{,}1] \cdot \begin{bmatrix} 0{,}2 & 0{,}5 & 0{,}3 \\ 0{,}4 & 0{,}1 & 0{,}5 \\ 0{,}5 & 0{,}2 & 0{,}3 \end{bmatrix}^3 = [0{,}361; 0{,}288; 0{,}351]$$

$$[P(t+6)] = [P(t)] \cdot [\Pi]^6 = [0{,}6; 0{,}3; 0{,}1] \cdot \begin{bmatrix} 0{,}2 & 0{,}5 & 0{,}3 \\ 0{,}4 & 0{,}1 & 0{,}5 \\ 0{,}5 & 0{,}2 & 0{,}3 \end{bmatrix}^6 = [0{,}362; 0{,}281; 0{,}357]$$

Можна зробити висновок про те, що ймовірності прагнуть до своїх границь при $k \to \infty$. Ця властивість називається ергодичною, згідно з нею система, яка описується ланцюгом Маркова, з часом переходить у сталий, стаціонарний стан, для існування якого необхідні певні умови. Стохастична матриця називається розкладною, якщо вона має вигляд (*A,B* – квадратні матриці, в іншому разі – нерозкладною)

$$\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}$$

Система, що перебуває в стані, *A*, ніколи не перейде в стан, *B* і навпаки.

Матриця називається періодичною (*C,D* – квадратні матриці, в іншому разі – неперіодичною), якщо вона має вигляд

$$\begin{bmatrix} 0 & C \\ D & 0 \end{bmatrix}$$

Якщо ланцюг Маркова є нерозкладним або неперіодичним, то для системи, яку він описує, існує стаціонарний стан.

$$\lim_{k\to\infty}\left[P(t+k)\right]=\lim_{k\to\infty}\left[p_0(t+k),p_1(t+k),\ldots,p_N(t+k)\right]=\left[\pi_0,\pi_1,\pi_2,\ldots,\pi_N\right]$$

$$\left[p(\infty)\right]=\left[\pi_1,\pi_2,\ldots,\pi_N\right] \qquad \left[p(\infty)\right]=\left[p(\infty)\right]\cdot\left[\Pi\right]$$

$$\left[p(\infty)\right]=\left[p(\infty)\right]\cdot\left[\Pi\right]\Rightarrow 0=\left[p(\infty)\right]\cdot\left[\Pi\right]-\left[p(\infty)\right]=0\Rightarrow\left[p(\infty)\right]\left(\left[\Pi\right]-I\right)=0$$

де *I* одинична матриця.

$$[\pi_0, \pi_1, \ldots, \pi_N] \cdot \begin{bmatrix} p_{00}-1 & p_{01} & \cdots & p_{0N} \\ p_{10} & p_{11}-1 & \cdots & p_{1N} \\ \cdots & \cdots & \cdots & \cdots \\ p_{N0} & p_{N1} & \cdots & p_{NN}-1 \end{bmatrix} = 0$$

Остаточно отримаємо таку систему рівнянь

$$\begin{aligned}
&(p_{00}-1)\pi_0 + p_{10}\pi_1 + p_{20}\pi_2 \ldots + p_{N0}\pi_N = 0, \\
&p_{01}\pi_0 + \quad (p_{11}-1)\pi_1 + p_{21}\pi_2 \ldots + p_{N1}\pi_N = 0, \\
&p_{02}\pi_0 + \quad\quad p_{12}\pi_1 + (p_{22}-1)\pi_2 \ldots + p_{N2}\pi_N = 0, \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&p_{0N}\pi_0 + \quad p_{1N}\pi_{12} + p_{2N}\pi_2 \ldots + (p_{NN}-1)\pi_N = 0.
\end{aligned}$$

Одне з рівнянь необхідно замінити на так звану умову нормування

$$\pi_0 + \pi_1 + \pi_2 + \ldots + \pi_N = 1$$

Для приклада

$$(0,2-1)\pi_a + 0,4\pi_b + 0,5\pi_c = 0,$$
$$0,5\pi_a + (0,1-1)\pi_b + 0,2\pi_c. = 0,$$
$$0,3\pi_a + 0,5\pi_b + (0,3-1)\pi_c = 0.$$

$$-0,8\pi_a + 0,4\pi_b + 0,5\pi_c = 0,$$
$$0,5\pi_a - 0,9\pi_b + 0,2\pi_c. = 0,$$
$$1\cdot\pi_a + 1\cdot\pi_b + 1\pi_c = 1.$$

Розв’язок: (0,363 0,281 0,356)

Імовірності, обчислені на шостому кроці прикладу, практично співпадають зі стаціонарними ймовірностями під час дослідження процесу змінювання уподобань споживачів у часі.

Процеси з дискретними станами й безперервним часом

Такі процеси описують функціонування систем, які набувають в процесі роботи кінцевого числа станів $\varphi(n,t), n=0,1,2,...,N$ і переходять з одного стану в інший – $\varphi(k,t), k = 0,1,2,...,N$ випадково в довільний момент часу t. Отже, **час перебування системи в будь-якому стані становить безперервну випадкову величину**.

Випадковий процес з безперервним часом називається **марківським процесом з дискретними станами й безперервним часом**, якщо поведінка системи після довільного моменту часу t обумовлюється тільки станом процесу в момент часу t і не залежить від передісторії процесу, що передує моменту часу t.

Найважливішою характеристикою системи є $p_i(t)$ - ймовірність того, що в момент t система буде перебувати в стані $\varphi(n,t), n=0,1,2,...,N$.

Для будь-якого моменту часу t сума всіх ймовірностей дорівнює 1.

Позначимо через $p_{nk}(t, \Delta t)$ імовірність переходу системи зі стану $\varphi(n,t), n=0,1,2,...,N$ у стан $\varphi(k,t), k=0,1,2,...,N$ за нескінченно малий проміжок часу Δt . Позначимо через $\lambda_{nk}(t)$ таку границю:

$$\lambda_{nk}(t) = \lim_{\Delta t \to 0} \frac{p_{nk}(t, \Delta t)}{\Delta t}$$

Ця характеристика називається інтенсивністю переходу або щільністюймовірності переходу і загалом залежить від t.

Імовірність переходу з точністю до нескінченно малих вищих порядків

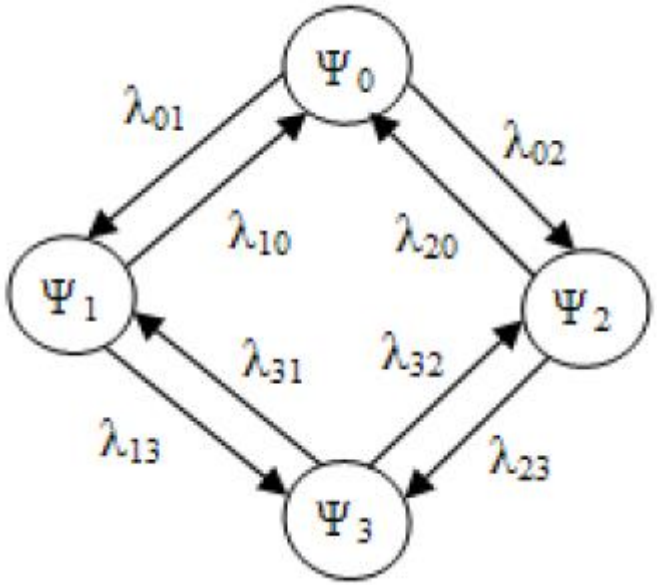
$$p_{nk}(t,\Delta t) = \lambda_{nk}(t)\Delta t, \Delta t \to 0$$

Якщо усі $\lambda_{nk}(t) = \lambda_{nk}$ не залежать від часу, то марківський процес називається **однорідним**, в іншому разі неоднорідним.

Кожен стан зображується у вигляді вершини графа; між вершинами позначаються дуги, що відображають можливість переходу системи з n-го стану в k-й за нескінченно малий інтервал Δt; над дугами проставляють величини інтенсивностей переходів λ_{nk}. Такий граф прийнято називати графом станів і переходів, на підставі цього графа можна вивести диференціальні рівняння Колмогорова, які повністю описують поведінку системи.

Приклад Система S складається з двох вузлів – А і В, кожен із яких у процесі роботи може відмовити. Можливими є такі стани:

- Ψ_0 – працюють обидва вузла;
- Ψ_1 – вузол А відмовив і відновлюється, вузол В – працює;
- Ψ_2 – вузол А працює, вузол В відмовив і відновлюється;
- Ψ_3 – відновлюються обидва вузла, до того ж час закінчення відновлення вузла *А* не збігається з часом закінчення відновлення вузла *В*.



Відомо інтенсивності переходів між станами.

Правило складання диференціального рівняння для довільного стану Ψ_i – $p_i(t)$: *похідна ймовірності будь-якого стану Ψ_i дорівнює добутку суми інтенсивностей переходу зі стану Ψ_i в інші стани та ймовірності стану Ψ_i, узятого з від'ємним знаком, плюс сумі добутків інтенсивностей переходу з інших станів у стан Ψ_i на ймовірності відповідних станів*.

$$p_0'(t) = -(\lambda_{01} + \lambda_{02}) p_0(t) + \lambda_{10} \cdot p_1(t) + \lambda_{20} \cdot p_2(t)$$

$$p_1'(t) = -(\lambda_{10} + \lambda_{13}) p_1(t) + \lambda_{01} \cdot p_0(t) + \lambda_{31} \cdot p_3(t),$$

$$p_2'(t) = -(\lambda_{20} + \lambda_{23}) p_2(t) + \lambda_{02} \cdot p_0(t) + \lambda_{32} \cdot p_3(t),$$

$$p_3'(t) = -(\lambda_{31} + \lambda_{32}) p_3(t) + \lambda_{13} \cdot p_1(t) + \lambda_{23} \cdot p_2(t).$$

У початковий момент часу система була працездатною, тому

$$p_0(0) = 1, p_1(0) = 0, p_2(0) = 0, p_3(0) = 0.$$

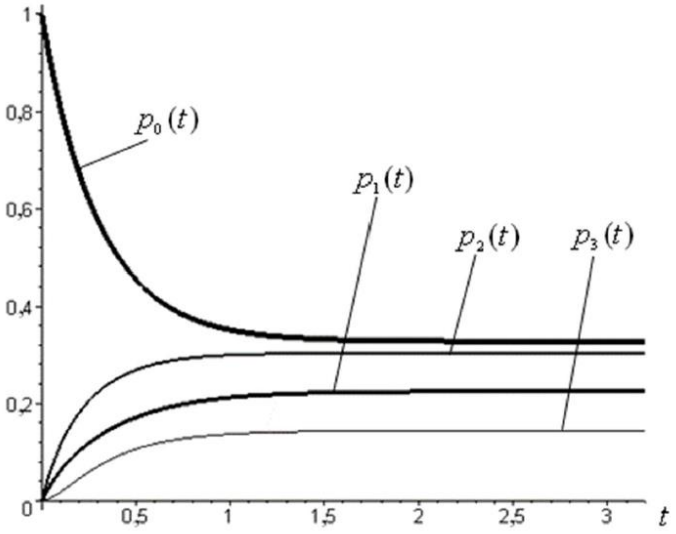
При $\lambda_{01}=0{,}8;\ \lambda_{02}=1{,}5;\ \lambda_{10}=1{,}6;\ \lambda_{13}=2{,}0;\ \lambda_{20}=1{,}3;\ \lambda_{23}=2{,}5;\ \lambda_{31}=3{,}8;\ \lambda_{32}=4{,}6$

$$p_0(t)=0{,}8544\sin(0{,}2541t+0{,}8943)e^{-3{,}4892t}+0{,}01e^{-11{,}12t}+0{,}32788,$$

$$p_1(t)=-0{,}9290\sin(0{,}2541t+0{,}2094)e^{-3{,}48t}-0{,}03e^{-11{,}12t}+0{,}22471,$$

$$p_2(t)=-0{,}5642\sin(0{,}2541t+2{,}6592)e^{-3{,}48t}-0{,}041e^{-11{,}12t}+0{,}30371,$$

$$p_3(t)=-0{,}1244\sin(0{,}2541t+1{,}0199)e^{-3{,}48t}+0{,}062e^{-11{,}12t}+0{,}14385.$$



Імовірності станів системи в стаціонарному режимі можна обчислити більш простим методом.

$$\lim_{t\to\infty} p_i'(t) = \lim_{t\to\infty} p_i = 0$$

$$\begin{aligned}
-(\lambda_{01}+\lambda_{02})p_0 + \lambda_{10}\cdot p_1 + \lambda_{20}\cdot p_2 = 0,\\
-(\lambda_{10}+\lambda_{13})p_1 + \lambda_{01}\cdot p_0 + \lambda_{31}\cdot p_3 = 0,\\
-(\lambda_{20}+\lambda_{23})p_2 + \lambda_{02}\cdot p_0 + \lambda_{32}\cdot p_3 = 0,\\
-(\lambda_{31}+\lambda_{32})p_3 + \lambda_{13}\cdot p_1 + \lambda_{23}\cdot p_2 = 0,
\end{aligned}$$

Система перейде в систему лінійних алгебраїчних рівнянь відносно p_i

Необхідно розв'язати систему лінійних алгебраїчних рівнянь, замінивши одне з рівнянь умовою нормування.

$$p_0 + p_1 + p_2 + p_3 = 1$$

$$-(\lambda_{01}+\lambda_{02})p_0+\lambda_{10}\cdot p_1+\lambda_{20}\cdot p_2=0,$$
$$-(\lambda_{10}+\lambda_{13})p_1+\lambda_{01}\cdot p_0+\lambda_{31}\cdot p_3=0,$$
$$-(\lambda_{20}+\lambda_{23})p_2+\lambda_{02}\cdot p_0+\lambda_{32}\cdot p_3=0,$$
$$p_0+p_1+p_2+p_1=1.$$

$$p_0=0{,}327\,89,\ p_1=0{,}224\,70,\ p_2=0{,}303\,56,\ p_3=0{,}143\,85$$

Більшість часу експлуатації система перебуває в стаціонарному стані. Подібні системи зазвичай досліджують при стаціонарному стані. Якщо складено розмічений граф станів і переходів, то для нього можна скласти систему лінійних алгебраїчних рівнянь щодо стаціонарних ймовірностей.

$$-(\lambda_{ie}+\lambda_{id}+\ldots+\lambda_{is})\cdot p_i+\sum_k \lambda_{ki}\cdot p_k=0 \qquad \sum_k p_k=1$$

Диференціальний метод для дослідження СМО з пуассонівськими потоками подій (час обслуговування й інтервали часу між надходженням запитів підпорядковані експонентному розподілу). Процес змінювання кількості запитів у системі є марківським.

Послідовність дослідження:

– із аналізу системи обслуговування відокремлюють стани системи, можливі переходи між станами та їх інтенсивності і складають граф станів і переходів;

– із кожним станом зв'язується відповідна ймовірність і складається система диференціальних рівнянь для ймовірностей станів системи. - складену систему доцільно розв'язують й досліджують за однією з систем комп'ютерної математики" [20].

Лекція 8

АНАЛІТИЧНІ МОДЕЛІ НАЙБІЛЬШ ПОШИРЕНИХ СМО З ОЧІКУВАННЯМ. ФОРМУЛИ ЛІТТЛА

ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ. ПРИНЦИПИ СТАТИСТИЧНОГО ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

Формули Літтла

“Для СМО, які функціонують у **стаціонарному режимі**, найсприйнятливішими є формули Літтла. Нехай СМО, на яку поступає потік запитів з інтенсивністю λ, функціонує в стаціонарному режимі досить великий інтервал часу T. Тоді за цей час у систему надійде в середньому λT запитів. Кожен запит перебуває в черзі середній час τ_q; тоді середній час W, який перебуватимуть у черзі всі λT запитів, обчислюється за формулою:

$$W = \lambda \tau_q T$$

З іншого боку, якщо L_q – середня кількість запитів, які очікують обслуговування, то сумарний середній час W, який вони перебуватимуть у черзі за час T, обчислюється за формулою:

$$W = L_q T \quad \Rightarrow \quad L_q = \lambda \tau_q$$

Кожен запит перебуває в системі середній час τ_s; тоді середній час V, який перебувають у системі всі λT запитів, обчислюється за формулою:

$$V = \lambda \tau_s T$$

З іншого боку, якщо L_s – середня кількість запитів у системі, то середній час τ_s, який вони перебуватимуть у системі за час T, обчислюється за формулою:

$$V = L_s T \quad \Rightarrow \quad L_s = \lambda \tau_s$$

Ці формули відіграють важливу роль у теорії масового обслуговування, оскільки дають змогу за певних умов унеможливити складні перетворення під час виведення характеристик системи.

Аналітичні моделі найбільш поширених СМО з очікуванням

В таких СМО вимоги, що надійшли в момент, коли всі обслуговуючі канали зайняті, ставляться в чергу і обслуговуються в міру звільнення каналів.

Загальна постановка задачі полягає в наступному. Система має n обслуговуючих каналів, кожен з яких може одночасно обслуговувати тільки одну вимогу. У систему надходить **найпростіший (пуассоновский) потік** вимог з параметром λ. Якщо в момент надходження чергової вимоги в системі на обслуговуванні вже знаходиться не менше n вимог (тобто всі канали зайняті), то це вимога стає в чергу і чекає початку обслуговування.

Час обслуговування кожної вимоги $t_{об}$ - випадкова величина, яка підпорядковується експоненціальним законом розподілу з параметром μ.

СМО з очікуванням можна розбити на дві великі групи: **замкнуті і розімкнуті**. До **замкнутих** відносяться системи, в яких надходить потік вимог виникає **в самій системі і обмежений**. Наприклад, майстер, завданням якого є налагодження верстатів в цеху, повинен періодично їх обслуговувати. Загальне число циркулюючих вимог найчастіше **постійно**.

Якщо джерело живлення має **нескінченне числом вимог**, то системи називаються **розімкнутими**. Розрахунок характеристик роботи СМО різного виду може бути проведений на основі розрахунку ймовірностей станів СМО (так звані формули Ерланга).

Алгоритми розрахунку показників якості функціонування розімкнутої системи масового обслуговування з очікуванням.

Введемо в розгляд параметр $\alpha = \lambda / \mu$. Зауважимо, що якщо $\alpha < 1$, то черга не може рости безмежно. Ця умова має наступний зміст: λ - середня кількість вимог, що надходять за одиницю часу, $1/\mu$ - середній час обслуговування одним каналом однієї вимоги, тоді $\alpha = \lambda \cdot 1/\mu$ - середнє число каналів, яке необхідно мати, щоб обслуговувати в одиницю часу всі вступники вимоги. Тому умова $\alpha / n < 1$ означає, що число обслуговуючих каналів повинно бути більше середнього числа каналів, необхідних для того, щоб за одиницю часу обслужити усі вимоги.

Найважливіші характеристики роботи СМО:

1. Імовірність того, що всі канали вільні від обслуговування:

$$P_0 = \left[\sum_{k=0}^{n-1} \frac{\alpha^k}{k!} + \frac{\alpha^n}{n!(1 - \alpha/n)} \right]^{-1}$$

2. Імовірність того, що зайнято рівно k обслуговуючих каналів за умови, що загальне число вимог, що знаходяться на обслуговуванні, не перевершує числа обслуговуючих каналів:

$$P_k = \frac{\alpha^k}{k!} P_0 \quad 1 \le k \le n$$

3. Імовірність того, що в системі знаходиться k вимог у разі, коли їх число більше числа обслуговуючих каналів:

$$P_k = \frac{\alpha^k}{n! n^{k-n}} P_0 \qquad k \ge n$$

4. Імовірність того, що всі обслуговуючі канали зайняті:

$$P_k = \frac{\alpha^k}{n!(1 - \alpha/n)} P_0 \qquad \alpha/n < 1$$

5. Середній час очікування вимогою початку обслуговування в системі:

$$\bar{t}_{ож} = \frac{P_n}{\mu(n - \alpha)} \qquad \alpha/n < 1$$

6. Середня довжина черги:

$$\bar{L}_{оч} = \frac{\alpha P_n}{n(1 - \alpha/n)} = \frac{\alpha^{n+1}}{n!\,n(1 - \alpha/n)^2} P_0 \qquad \alpha/n < 1$$

7. Середнє число вільних від обслуговування каналів:

$$\bar{N}_0 = \sum_{k=0}^{n-1} \frac{n - k}{k!} \alpha^k P_0$$

8. Коефіцієнт простою каналів:

$$K_{пр} = \frac{\overline{N}_0}{n}$$

9. Середнє число зайнятих обслуговуванням каналів:

$$\overline{N}_з = n - N_0$$

10. Коефіцієнт завантаження каналів:

$$K_з = \frac{\overline{N}_з}{n}$$

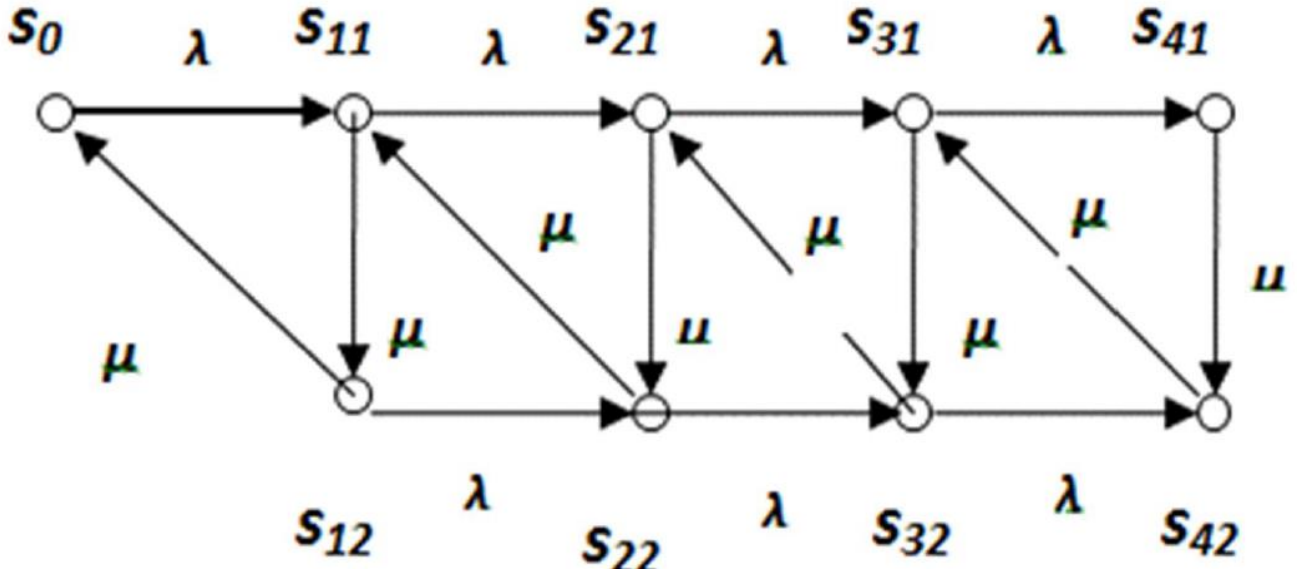
Метод фаз

Метод фаз дає змогу досліджувати СМО **більш загального вигляду**, порівняно з пуассонівським за допомогою введення **додаткових станів**. Це дає змогу використати для такої розширеної системи методику складання рівнянь для додаткових станів. Метод фаз зазвичай використовується, коли потоки описуються ерлангівським або гіперекспоненціальним розподілами.

Наприклад, якщо час обслуговування розподілено за **ерлангівським** розподілом k-го порядку з параметром μ. У цьому разі канал обслуговування умовно замінюється k окремими фіктивними каналами (фазами), розташованими послідовно. Кожен запит, прийнятий на обслуговування, послідовно проходить через ці k фаз, перебуваючи на кожній фазі період часу, що підпорядковується експоненціальному розподілу з параметром μ.

Новий запит не обслуговується до тих пір, допоки попередній не пройде всі k фаз обслуговування. СМО в цьому разі описується марківським процесом із станами s_{ij}, $1 \le j \le k$. Стан s_{ij} означає, що в системі міститься i запитів, а черговий запит обслуговується на j-й фазі. Цьому процесу відповідають імовірності станів $p_{ij}(t)$.

$$M / E_2 / 1$$



ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ ІНФОРМАЦІЙНИХСИСТЕМ

Імітаційне моделювання – це метод дослідження, при якому досліджувана система замінюється моделлю, яка з достатньою точністю описує реальну систему, і з нею **проводяться експерименти** з метою отримання інформації про цю систему. Експериментування з моделлю - імітація (збагнення суті явища, не вдаючись до експериментів на реальному об'єкті).

Імітаційне моделювання – це окремий випадок **математичного моделювання**. Існує клас об'єктів, для яких з різних причин не розроблені аналітичні моделі, або не розроблені методи розрахунку

отриманої моделі. В цьому випадку математична модель замінюється імітатором або імітаційною моделлю.

Імітаційне статистичне моделювання являє собою чисельний метод проведення на ЕОМ обчислювальних експериментів з математичними моделями, що імітують поведінку реальних об'єктів, процесів і систем у часі протягом заданого періоду.

Імітаційне моделювання – це сукупність методів алгоритмізації функціонування об'єктів досліджень, програмної реалізації алгоритмічних описів, організації, планування та виконання на ЕОМ обчислювальних експериментів з математичними моделями.

Переваги імітаційного моделювання:

- висока адекватність між суттю описуваного процесу і моделлю;
- можливість описати складну систему на високому рівні деталізації;
- більше областей дослідження, ніж аналітичне моделювання;
- відсутність обмежень на залежності між параметрами моделі;
- можливість оцінки функціонування системи не тільки в стаціонарних станах, але і в перехідних режимах (процесах) ;
- одержання значної кількості даних про досліджуваний об'єкт (закон розподілу, числові значення абсолютні та відносні) ;
- найбільш раціональне відношення «результат - витрати» у порівнянні з аналітичним і фізичним моделюванням.

Недоліки імітаційного моделювання:

- розробка хорошої моделі часто обходиться дорожче, ніж аналітична і вимагає більше часу на створення і налагодження
- складно оцінити ступінь точності моделі, її адекватність;
- відносно високі вимоги до кваліфікації дослідника;
- спільність застосування та індивідуальність реалізації.

Комбіновані методи моделювання:

- модель представляється у комбінації методів моделювання;
- найбільш широко застосовуються імітаційно-аналітичні моделі;

- ступінь застосування методів моделювання визначає дослідник, виходячи з поставлених завдань, наявних ресурсів і часу на проведення дослідницької роботи.

Області застосування імітаційного моделювання:

- інформаційні системи та мережі, IT-інфраструктура;
- бізнес процеси, виробництво, ринок і конкуренція;
- бойові дії;
- динаміка населення;
- математичне моделювання історичних процесів;
- логістика (ланцюги поставок) ;
- вуличний рух;
- сервісні центри;
- управління проектами;
- економіка охорони здоров'я;
- екосистеми.

Імітація отримала початковий розвиток у зв'язку із створенням ЕОМ в 1950-х – 1960-х роках (Станіслав Улам). Одним з основних видів імітаційного моделювання є **статистичне імітаційне моделювання** – імітують взаємодію елементів системи, що носять **імовірнісний характер**.

При реалізації на ЕОМ статистичного імітаційного моделювання виникає задача отримання на ЕОМ випадкових числових послідовностей із заданими ймовірнісними характеристиками. Чисельний метод, який вирішує задачу генерування послідовності випадкових чисел із заданими законами розподілу, отримав назву "**метод статистичних випробувань**" або "**метод Монте-Карло**".

Метод **статистичного імітаційного моделювання** – це спосіб вивчення складних процесів і систем, що піддаються випадковим збуренням, за допомогою імітаційних моделей.

Методика складається з таких **етапів**:

1) Моделювання на ЕОМ псевдовипадкових послідовностей із заданою кореляцією і законом розподілу ймовірностей (метод Монте-Карло), що імітують на ЕОМ випадкові значення параметрів при кожному випробуванні.

2) Перетворення отриманих числових послідовностей на імітаційних математичних моделях.

3) Статистична обробка результатів моделювання.

Адекватність (від лат. Adaequatus – прирівняний, рівний) моделі – збіг властивостей (функцій / параметрів / характеристик і т. п.) моделі і відповідних властивостей модельованого об'єкта. Адекватністю називається збіг моделі з модельованою системою у відношенні мети моделювання.

Універсального загального способу побудови адекватних моделей не існує.

Методи імітаційного моделювання

Агентне моделювання (agent-based model, ABM)– напрямок в імітаційному моделюванні, який використовується для дослідження децентралізованих систем, динаміка функціонування яких визначається не глобальними правилами і законами, а навпаки коли ці правила і закони є результатом індивідуальної активності членів групи. Аналітик визначає поведінку агентів на індивідуальному рівні, а глобальна поведінка виникає як результат діяльності багатьох агентів (моделювання «знизу вгору»).

Дискретно-подієве моделювання – підхід до моделювання, що пропонує абстрагуватися від неперервної природи подій і розглядати тільки основні події модельованої системи, такі як: «очікування», «обробка замовлення», «рух з вантажем», «розвантаження» та інші. Дискретно-подієве моделювання найбільш розвинене і має величезну сферу застосувань – від логістики та систем масового обслуговування до транспортних і виробничих систем. Цей вид моделювання найбільш підходить для моделювання процесів в інформаційних мережах. Заснований Джеффрі Гордоном в 1960х роках.

В **системі масового обслуговування** протікають наступні процеси:

- надходження заявок;
- вибір обслуговуючого пристрою;
- обслуговування;
- звільнення.

Для відображення цих процесів мають бути імітовані:

- вхідний потік заявок;
- управління / розподілом заявок;
- обслуговування;
- вихідний потік заявок;
- статистичну обробку вхідних і вихідних параметрів.

ПРИНЦИПИ СТАТИСТИЧНОГО ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

Моделювання статистичних розподілів випадкових величин

Імітація вхідного потоку заявок

Потік заявок – це послідовність заявок (викликів), що надходять в систему обслуговування у визначені моменти часу:

$t_1, t_2..., t_i, ..., t_c, ...,$

де t_i - це вимірюваний параметр, який може приймати визначені або довільні значення.

Детермінований потік – потік заявок у **фіксовані** моменти часу.

Стохастичний (випадковий) потік – потік заявок у **випадкові** моменти часу.

Для визначення потоку заявок необхідно описати інтервал часу між сусідніми заявками:

$$\Delta t_k = t_k - t_{k-1}$$

Для моделювання випадкового потоку заявок необхідно задати функцію розподілу *F(Δt)* інтервалу часу між сусідніми заявками. Для найпростішого потоку викликів розподіл числа викликів, що надходять за час *t* визначається за формулою Пуассона:

$$P_i(t) = \frac{(\lambda t)^i}{i!} e^{-\lambda t} \qquad P(\Delta t) = P(\tau < \Delta t) = 1 - \lambda e^{-\lambda \Delta t} \qquad p(\Delta t) = \lambda e^{-\lambda \Delta t}$$

Це ймовірність надходження в точності *i* викликів найпростішого потоку за відрізок часу *t*.

Властивості найпростішого потоку:

- стаціонарність;
- ординарність;
- відсутність післядії.

Ординарність потоку означає практичну неможливість одночасного надходження двох і більше вимог.

Стаціонарним називається потік, для якого математичне очікування числа вимог, що надходять у систему в одиницю часу (позначимо λ), не змінюється в часі. Таким чином, імовірність надходження в систему певної кількості вимог протягом заданого проміжку часу Δt залежить від його величини і не залежить від початку його відліку на осі часу.

Відсутність післядії означає, що число вимог, що надійшли в систему до моменту t, не визначає того, скільки вимог надійде в систему за проміжок часу від t до $t+\Delta t$.

Для отримання значень випадкових величин, які характеризують модельований потік заявок використовується метод, заснований на наступній **теоремі**:

якщо випадкова величина ρ має щільність розподілу $f(\rho)$, то розподіл випадкової величини ε є рівномірним на інтервалі [0,1]

$$\varepsilon = \int_0^{\rho} f(x)d(x)$$

Для функції експоненціального розподілу, визначеної на дискретному часі t_i:

$$f(x)=p(t_i)=\lambda e^{-\lambda t}$$

Вирішуючи дане рівняння відносно детермінованих величин $\{\rho\}$ можна отримати формулу для розрахунку значень випадково величини ρ_i:

$$\rho_i = -\frac{1}{\lambda}\ln \varepsilon_i$$

де ε_i – рівномірно розподілена на інтервалі [0; 1] випадкова величина. Користуючись цією формулою можна отримати множину значень ρ_i, які будуть відповідати експоненціальній щільності розподілу.

На основі послідовності випадкових величин ρ_i можна отримати послідовність моментів надходження викликів в потоці:

$$t_1=\rho_1;$$

$$t_2=t_1+\rho_2=\rho_1+\rho_2;$$

$$t_3=t_2+\rho_3=\rho_1+\rho_2+\rho_3.$$

Загальний вираз для розрахунку моментів надходження заявок в потоці при експоненціальному розподілі інтервалу часу між сусідніми заявками має вигляд:

$$t_i = t_{i-1} + \rho_i = \sum_{j=1}^{i} \rho_j = \sum_{j=1}^{i} -\frac{1}{\lambda} \ln \varepsilon_j$$

Формування часу зайняття каналів.

Сукупність часу звільнення каналів може бути визначена наступним чином:

$$t_{обсл} = t_i + \tau_i$$

де τ_i – час обслуговування заявки (заняття каналу), що надійшла в момент часу t_i.

У припущенні, що час обслуговування розподілено по експоненціальному закону, щільність розподілу має такий вигляд:

$$\varphi(t) = \mu e^{-\mu t}$$

де μ – параметр потоку обслуговуваних викликів.

Параметр потоку обернено пропорційний середньому часу обслуговування:

$$\mu = \frac{1}{\bar{\tau}}$$

Аналогічно, тривалість обслуговування можна визначати у вигляді:

$$\tau_i = -\frac{1}{\mu} \ln \varepsilon_i$$

Для $i \geq 1$ тривалість обслуговування розрахувується за формулою:

$$\tau_i = \tau_{i-1} + \rho_i = \sum_{j=1}^{i} \rho_j = \sum_{j=1}^{i} -\frac{1}{\mu} \ln \varepsilon_j$$

де ε_i – рівномірно розподілена на інтервалі [0; 1] випадкова величина.

Метод розрахунку значень випадкових величин, підпорядкованих **заданому розподілу** на основі генерації **рівномірно розподілених** випадкових величин в інтервалі [0; 1], дозволяє задати потоки заявок у вигляді імітації дискретних моментів часу їх виникнення. По-суті, в такий спосіб імітується процес надходження і обслуговування заявок в каналі, гілці, тощо.

Приклад імітаційної моделі M/M/1

Модель імітує роботу одноканальної системи обслуговування з явними втратами при умовах:

- вхідний потік викликів – найпростіший з параметром 1;
- час обслуговування має експоненціальний розподіл з параметром *m*;
- час – дискретний;
- система має 2 стани каналу: 1) вільний; 2)зайнятий;

Зміна стану відбувається при надходженні і завершенні обслуговування заявки. Тобто, система обслуговування відображає дискретно-подієвий принцип моделювання. Система обслуговування відображає *дискретно-подієвий принцип моделювання*.

Масив випадкових чисел рівномірно розподілених на інтервалі [0 ; 1]			Вхідний потік			Вихідний потік		
		параметр потоку	1		Стан каналу обслугову-вання	1		заявки, що були обслуговані
для вхідного потоку	для часу обслуговування	№ заявки в потоці	t між сусідніми заявками у вхідному потоці	T - час надходження заявок на обслугову-вання	0 - вільний / 1 - зайнятий	Тривалість обслугову-вання заявки	T - час вивільнення каналу обслугову-вання	
0,449966563	0,034569003	1	0,798582004	0,798582004	1	3,364797875	4,16337988	1
0,399012501	0,486572825	2	0,918762533	1,717344537	0	0,720368697	4,16337988	0
0,425216868	0,790931955	3	0,855155963	2,5725005	0	0,234543338	4,16337988	0
0,203503703	0,176394894	4	1,59207108	4,16457158	1	1,735030083	5,899601663	1
0,361451752	0,56620045	5	1,017626713	5,182198292	0	0,568807111	5,899601663	0
0,26448398	0,915773275	6	1,329974596	6,512172889	1	0,087986462	6,60015935	1
0,069255409	0,415544812	7	2,669954026	9,182126915	1	0,878164819	10,06029173	1
0,05986412	0,579422826	8	2,815677956	11,99780487	1	0,545722798	12,54352767	1
0,434979545	0,591988453	9	0,832456273	12,83026114	1	0,52426815	13,35452929	1
0,281611343	0,204845967	10	1,267227376	14,09748852	1	1,585496963	15,68298548	1
0,707028348	0,945984825	11	0,346684517	14,44417304	0	0,055528751	15,68298548	0
0,8387859	0,976367112	12	0,17579979	14,61997283	0	0,023916624	15,68298548	0
0,006697944	0,489901858	13	5,005954716	19,62592754	1	0,713550197	20,33947774	1
0,186605128	0,363290035	14	1,678760511	21,30468805	1	1,012553768	22,31724182	1
0,167204858	0,864563165	15	1,788535526	23,09322358	1	0,145530911	23,23875449	1
0,69462348	0,694473672	16	0,364385335	23,45760891	1	0,364601027	23,82220994	1
0,219880072	0,019677654	17	1,514673009	24,97228192	1	3,928271593	28,90055352	1
0,944163746	0,86886128	18	0,057455668	25,02973759	0	0,140571799	28,90055352	0
0,575364934	0,345318873	19	0,552750771	25,58248836	0	1,063287018	28,90055352	0
0,44867489	0,152118963	20	0,801456729	26,38394509	0	1,883092414	28,90055352	0
0,612700639	0,463628852	21	0,489878817	26,87382391	0	0,768670934	28,90055352	0
0,033456162	0,18805175	22	3,397519281	30,27134319	1	1,671038086	31,94238128	1
0,888993474	0,776639421	23	0,117665384	30,38900857	0	0,252779102	31,94238128	0
0,605670245	0,029390709	24	0,501419591	30,89042816	0	3,52707666	31,94238128	0
0,568164148	0,318257637	25	0,565344909	31,45577307	0	1,144894046	31,94238128	0
0,981723057	0,548586101	26	0,01844603	31,4742191	0	0,600411036	31,94238128	0
0,027487627	0,917098114	27	3,594019313	35,06823842	1	0,086540818	35,15477923	1
		27			14			14
					52%			

Переваги:

- простота реалізації (на основі вбудованих функцій);
- невеликі витрати часу на створення моделі;
- дозволяє оцінювати основний показник – ймовірність втрати заявок через зайнятість обслуговуючого пристрою;

Недоліки:

- модель працює в ручному режимі, не працює самостійно;
- кількість «прогонів» обмежено;
- статистична обробка виконується вручну.

Приклад демонструє ручний режим прогону експериментів імітаційної моделі процесів надходження та обслуговування заявок“ [20].

Лекція 9

МЕРЕЖІ ПЕТРІ

ЗАВДАННЯ АНАЛІЗУ МЕРЕЖ ПЕТРІ

Мережі Петрі (МП)

“Мережі Петрі (МП) — математичний апарат для моделювання динамічних дискретних систем. МП розроблялися спеціально для моделювання тих систем, які містять взаємодіючі паралельні компоненти. Вперше МП запропонував Карл Петрі у своїй докторській дисертації «Kommunikation mit Automaten» («Зв'язок автоматів»). Він сформулював основні поняття теорії зв'язку асинхронних компонент обчислювальної системи. Зокрема, детально розглянув опис причинних зв’зків між подіями. Його дисертація присвячена, головним чином, теоретичній розробці основних понять, з яких почали розвиток МП.

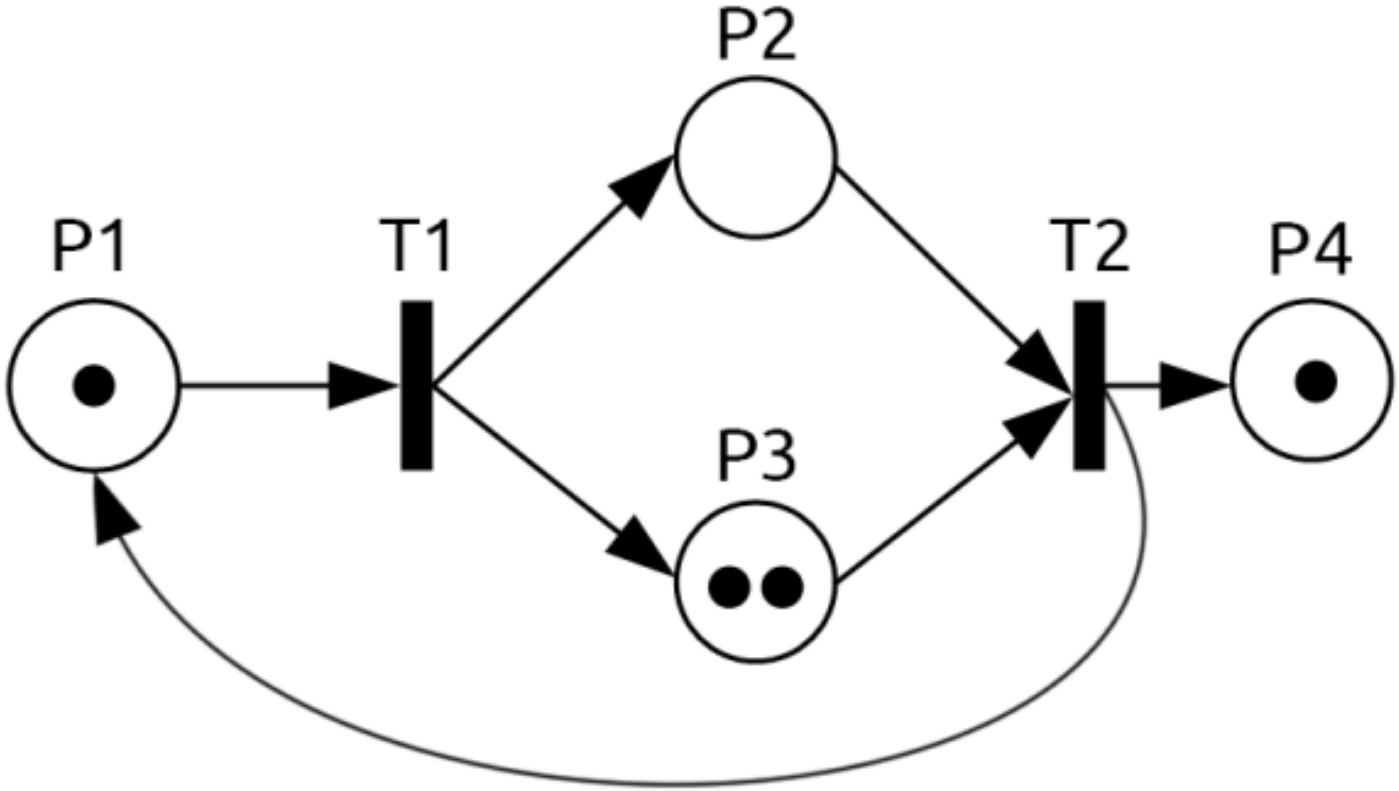

Carl Adam Petri

МП використовуються для моделювання асинхронних систем, що функціонують як сукупність паралельних взаємодіючих процесів. Аналіз МП дозволяє отримати інформацію про структуру та динамічну поведінку модельованої системи.

Причинно-наслідковий зв'язок подій в асинхронних системах задається множиною відношень вигляду «умови-події». У МП умови — це позиції, а події — переходи. Відповідно до цього граф МП є двочастковим орієнтованим мультиграфом. Орієнтовані дуги можуть сполучати лише позиції і переходи в прямому і зворотному напрямі. МП є мультиграфом, оскільки допускається кратність дуг між позиціями і переходами.

В графах МП кількісні характеристики умов (числа натурального ряду) прийнято задавати числом міток у відповідних позиціях.

Приклад мережі Петрі. Білими колами позначені позиції, смужками — переходи, чорними колами — мітки.



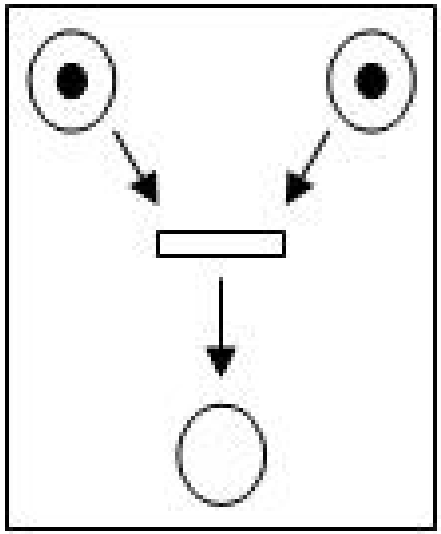
Прості мережі Петрі

Мережа Петрі є орієнтованим дводольним графом, який має чотири базових елементи: вузли, або **місця (places), переходи (transitions), дуги (arcs)** і **маркери (tokens).**

Вузли позначаються кружками і визначають стан, в якому може знаходитись мережа або її частина.

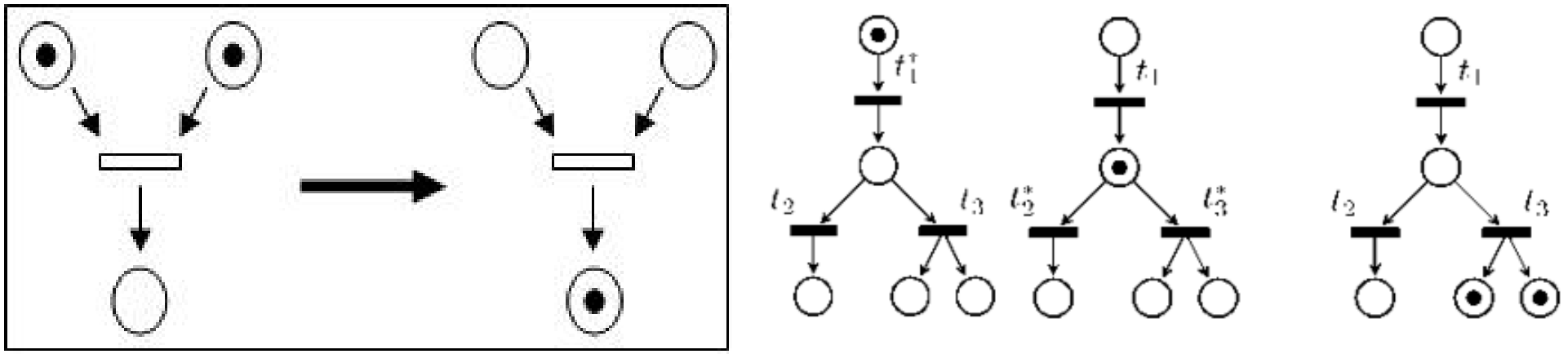
Переходи-це активні елементи мережі, які позначають дії, виконувані під час спрацювання переходів. Для того щоб перехід міг спрацювати, необхідне виконання певних умов, які визначаються наявністю маркерів у вузлах мережі, з'єднаних з переходом. Якщо умови настання подій виконано, то вважають, що перехід збуджений. Переходи позначаються короткими вертикальними або горизонтальними лініями. Вузли та переходи

з'єднуються орієнтованими ребрами (дугами). Два вузли або два переходи з'єднуватись дугами не можуть.



Функціонування мережі Петрі можна описати так: **вузли як певні умови, а переходи-як події**. Таким чином, стан мережі в кожний момент часу задається системою умов. Для зручності задання умов мережі Петрі вводяться **маркери (фішки)**, які зображуються крапками всередині вузлів. Виникнення певної комбінації маркерів у вузлах приводить до настання

деякої події, яка у свою чергу викликає зміну стану умов мережі. Стан маркування або стан мережі Петрі визначається сукупністю маркерів кожного окремого вузда мережі.



Графічне зображення спрацювання переходу

Перехід, в якого всі вхідні вузли містять маркери, називається збудженим. Збуджений перехід може спрацювати, після чого всі маркери із вхідних вузлів переходу перемістятся у вихідні. Таким чином, настає подія, яка змінює стан мережі.

Якщо одночасно збуджуються кілька переходів мережі, виникає **невизначеність**, тому одночасне спрацювання кількох переходів у мережі Петрі неможливе, тобто **переходи спрацьовують послідовно, миттєво**. Незважаючи на те, що маркери змінюють своє положення у вузлах, **прості мережі Петрі- це статичні моделі, в яких не враховується динаміка в часі** (зміна станів мережі не залежить від моментів часу). Для того щоб за допомогою мережі Петрі відтворити динаміку роботи деякої детермінованої динамічної системи в часі, необхідно зазначати моменти спрацьовування переходів. Такі можливості мають тільки розширення мереж Петрі, в яких спрацювання переходів здійснюється в задані моменти модельного часу з деяким постійним кроком.

Моделювання систем за допомогою мереж Петрі

Прості мережі Петрі містять лише три основних елементи: вузли, переходи та маркери. Тому побудувати за їх допомогою моделей складних динамічних систем, в яких протікає велика кількість взаємодіючих паралельних і асинхронних процесів та існує багато інформаційних і матеріальних потоків, стає досить складною та громіздкою процедурою. Це помітно звужує клас моделей систем, які можна побудувати на основі простих мереж Петрі які дають можливість значно спростити побудову складних моделей і їх графічне зображення.

Розширення мережі Петрі - це така їх модифікація, яка збільшує можливості мережі стосовно опису та моделювання систем. Існують

різні розширення мереж Петрі, орієнтовані на моделювання систем різних типів: стохастичних, динамічних, предикатних та ін. Кольорові мережі Петрі дають змогу значно зменшити розміри мереж, які використовуються, наприклад, для опися моделей складних паралельних обчислювальних систем. На практиці широко використовуються проблемно-орієнтовані розширення мереж Петрі, серед яких найбільш відомі Е-мережі, комбі-мережі, FIFI-мережі, М-мережі та ін.

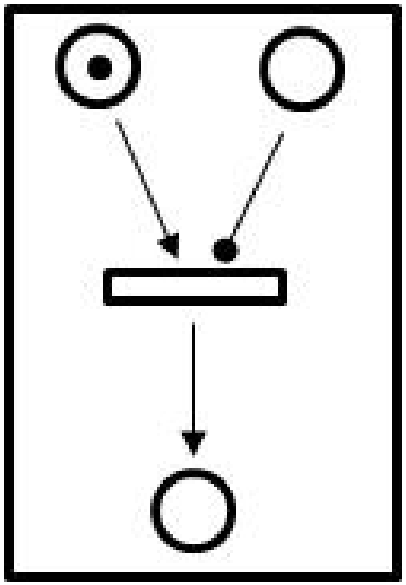
У простих мережах Петрі допускається наявність у вузлі лише одного маркера, тоді як урозширених мережах кожний вузол може містити кілька маркерів у вузлі позначає число поряд з вузлом.

Відповідно для цих вузлів змінюються і правила маркування:

- перехід збуджується тільки тоді, коли число, яке визначає кількість маркерів у кожному вхідному вузлі, більше або дорівнює одиниці;
- якщо збуджений перехід спрацьовує, то число маркерів у всіх вхідних вузлах, які містять маркери, зменшуються на одиницю, а в усіх вихідних вузлах - збільшується на одиницю. Певна річ, кількість маркерів не може біти від'ємним значенням.

Мережі Петрі достатньо для опису причинно-наслідкових подій, які виникають у системах, однак мережі не забезпечують повну спільність з логічними операціями. Зрозуміло, що проста мережа Петрі відтворює роботу тільки логічного елемента "І" ("AND"), тому за допомогою цієї

мережі не можна змоделювати збудження переходу, коли вхідний вузол не має маркерів (логічний оператор заперечення "НІ"). Для цього в розширених мережах усводяться **дуги заперечення**, які зображуються у вигляді лінії з кружечком на кінці замість стрілки і не мають дугової ваги (дугова вага визначає пропускну здатність дуги).

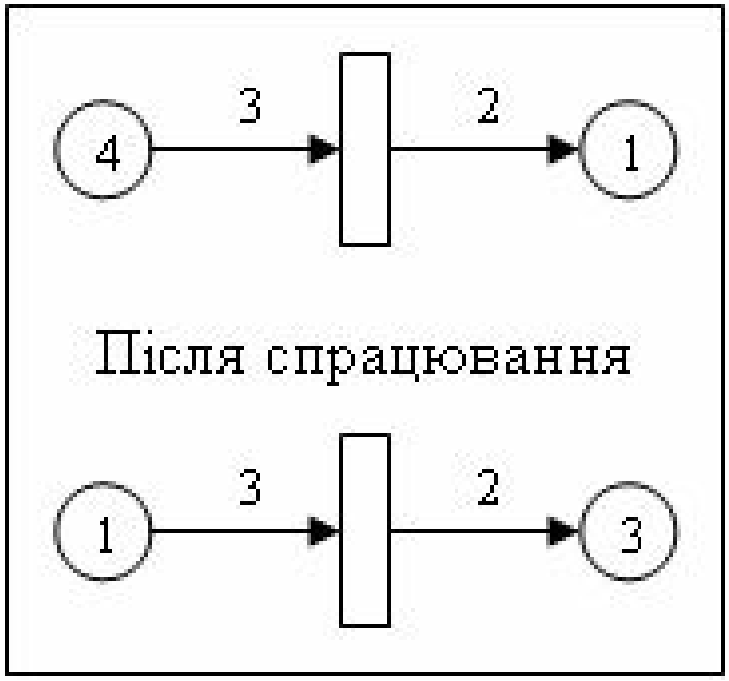


Дуги, визначені раніше, розглядаються як позитивні. Вони завжди направлені від деякого вузла до переходу і не можуть мати

зворотнього напрямку. Наявність дуги заперечення змінює правила маркування на такі:

- перехід збуджується лише тоді, коли число, яке визначає кількість маркерів кожного вхідного вузла з позитивною дугою, більше або дорівнює одиниці, та коли кількість маркерів кожного вхідного вузла з дугою заперечення дорівнює нулю;
- якщо збуджений перехід спрацьовує, то число маркерів усіх вхідних вузлів з позитивною дугою зменшується на одиницю, у той час як кількість маркерів вхідних вузлів з дугами заперечення залишається незмінною. Кількість маркерів усіх вихідних вузлів збільшується на одиницю.

Для кожної позитивної дуги можна задати певний ваговий коефіцієнт (або вагу), рівний одиниці або більший за неї. За замовчуванням ваговий коефіцієнт дуги дорівнює одиниці.



Тоді збудження та спрацьовування переходу відбувається за такими правилами:

- прехід збуджується тільки тоді, коли кількість маркерів у кожному вхідному вузлі більше ваги дуги або дорівнює їй, а для дуги заперечення дорівнює нулю;
- у разі перемикання переходу кількість маркерів кожного вхідного вузла зменшується на відповідну вагу вхідної дуги та залишається незмінною для дуг заперечень. Кількість маркерів кожного вихідного вузла збільшується на вагу відповідної вихідної дуги” [12-19].

Особливості мереж Петрі та систем, що моделюються за їх допомогою:

- паралелізм, або одночасність (мережі Петрі зручні для моделювання систем з розподіленим керуванням).

- асинхронність (у мережі Петрі відсутній вимір часу чи перебіг часу. Мережа Петрі містить необхідну інформацію визначення можливих послідовностей подій).

- недетермінованість виконання мережі Петрі (порядок появи подій одна з можливих, вибір переходу, що запускається, при кількох дозволених здійснюється недетерміновано, тобто випадково).

Запуск переходу та відповідної події розглядається як миттєва подія, і виникнення двох подій одночасно неможливе. Моделювана таким чином подія називається примітивною, примітивні події миттєві та неодноразові.

Примітивні та непримітивні події

Непримітивними називаються такі події, тривалість яких **не дорівнює 0**. Вони є неодночасними і, отже, можуть перетинатися у часі.

Непримітивну подію можна у вигляді двох примітивних подій: ”початок не примітивної події”, “кінець не примітивної події”, та умови “не примітивна подія відбувається”.

Мережа Петрі **C** є четвіркою, **C=(P,T,I,O)**.

P={p_1, p_2, ... p_i, p_n} - кінцева множина позицій, **n≥0**.

T={ t_1, t_2, ... t_j, t_m } - кінцева множина переходів, **m≥0**.

Множина позицій і множина переходів не перетинаються, тобто перетин **P** і **T** дорівнює порожній множині.

I: $T \rightarrow P^y$ є вхідною функцією - відображенням з переходів в комплекти позицій.

O: $P^y \rightarrow T$ є вихідна функція – відображення з комплектів позицій у переходи.

Довільний елемент **P** позначається символом **p_i , i=1, ..., n**, а довільний елемент **T** символом **t_j, j=1, ..., m**..

Правила виконання мереж Петрі

Подією називають спрацювання переходу, у якому мітки з вхідних позицій цього переходу переміщуються у вихідні позиції. Події відбуваються миттєво, чи різночасно, у виконанні деяких умов.

Виконанням мережі Петрі керують кількість та розподіл фішок у мережі. Мережа Петрі виконується у вигляді запусків переходів. Перехід запускається видаленням фішок з його вхідних позицій та утворенням нових фішок, що розміщуються в його вихідних позиціях.

Перехід називається дозволеним, якщо кожна з його вхідних позицій має число фішок принаймні рівне числу дуг з позиції переходу.

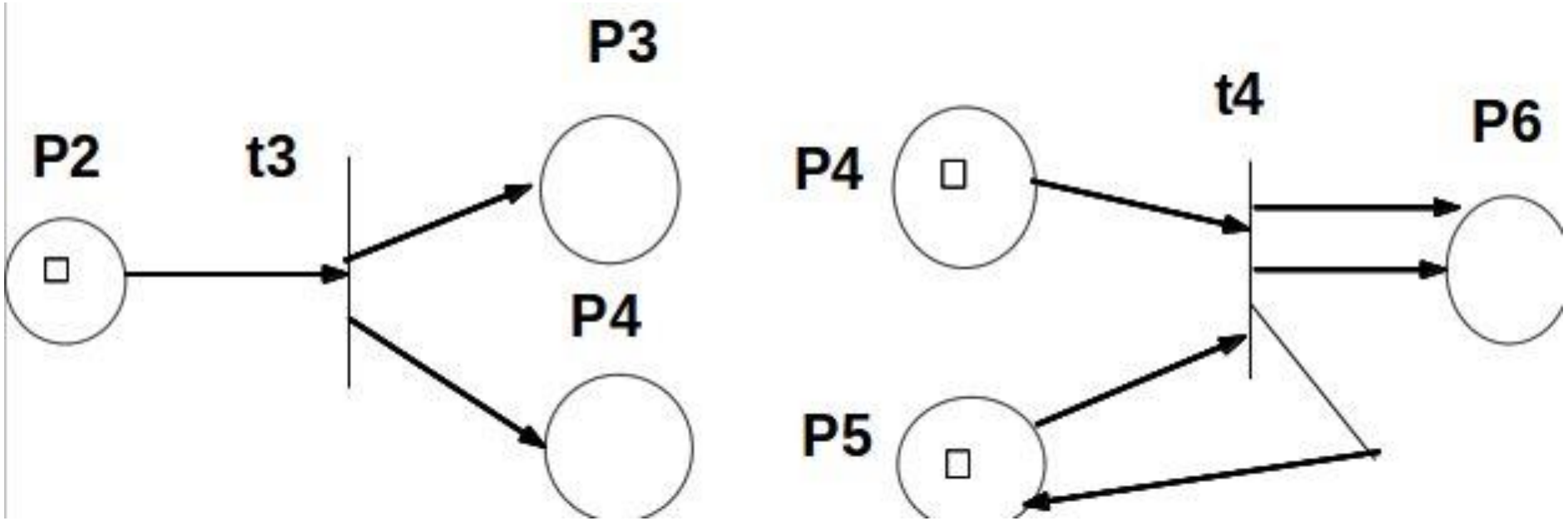
Перехід запускається, якщо його дозволено. Фішки у вхідній позиції, які дозволяють перехід, називаються його роздільними фішками. Наприклад, якщо позиції p_1 і p_2 є входами для переходу t_1, тоді t_1 дозволений, якщо p_1 і p_2 мають хоча б по одній фішці. Для переходу t_3 із вхідним комплектом $\{p_3,p_3,p_3\}$ позиція p_3 повинна мати не менше 3 фішок для дозволу переходу t_3.

Визначення. Перехід $t_j \in T$ маркованої мережі Петрі $C = (P, T, I, O, \mu)$ з маркуванням μ, дозволений, якщо для всіх $p_j \in P$ $\mu(p_i) \geq \#(p_i, I(t_j))$.

Перехід запускається видаленням дозволяючих фішок, з усіх його вхідних позицій (кількість віддалених фішок для кожної позиції відповідає числу дуг, що йдуть з цієї позиції в перехід), з наступним розміщенням фішок в кожну з його вихідних позицій (кількість дуг,

що вміщаються, в позицію відповідає кількості дуг, що входять цю позицію з переходу).

Перехід t3 I(t3) = {p2} і O(t3) = {p3,p4} дозволений щоразу, коли в p2 буде хоча б одна фішка. Перехід t3 запускається видаленням однієї фішки з позиції p2 та поміщенням однієї фішки в позицію p3 та p4 (його виходи). Перехід t4, в якому I(t4) = {p4,p5} і O(t4) = {p5,p6,p6} запускається видаленням по одній фішці з позицій p4 і p5, при цьому одна фішка міститься в p5 і дві p6

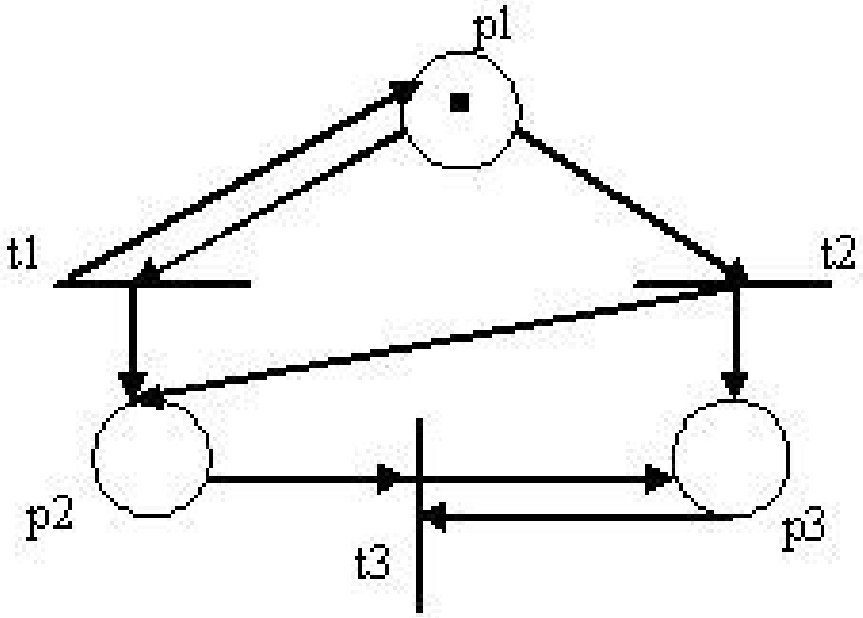

P3
P2
t3
P4
t4
P6
P4
P5

Методи аналізу

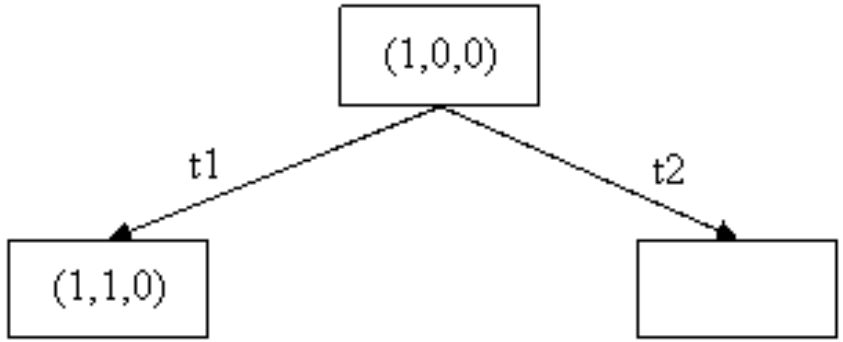
“Існує два основних методи аналізу мереж Петрі, які описують механізми розв’язку наведених вище задач. Перший метод аналізу полягає у побудові і дослідженні **дерева досяжності**, другий метод – пов’язаний з **матричними рівняннями**.

Дерево досяжності

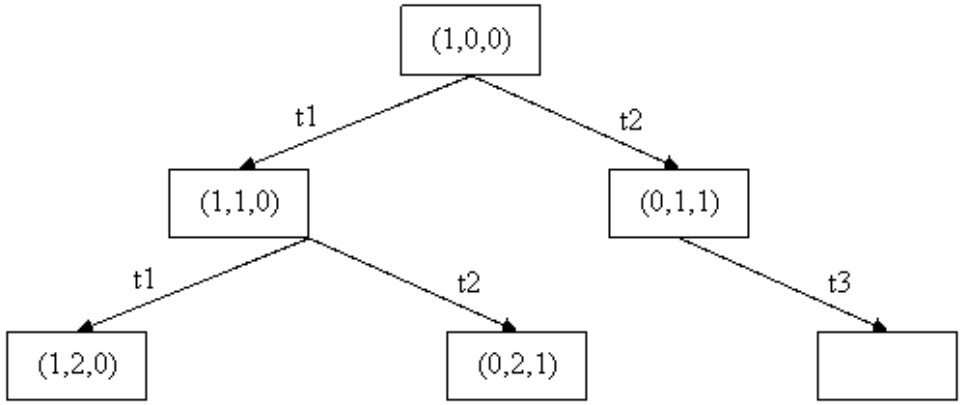
Дерево досяжності являє собою множину досяжності мережі Петрі. Для прикладу, розглянемо марковану мережу Петрі

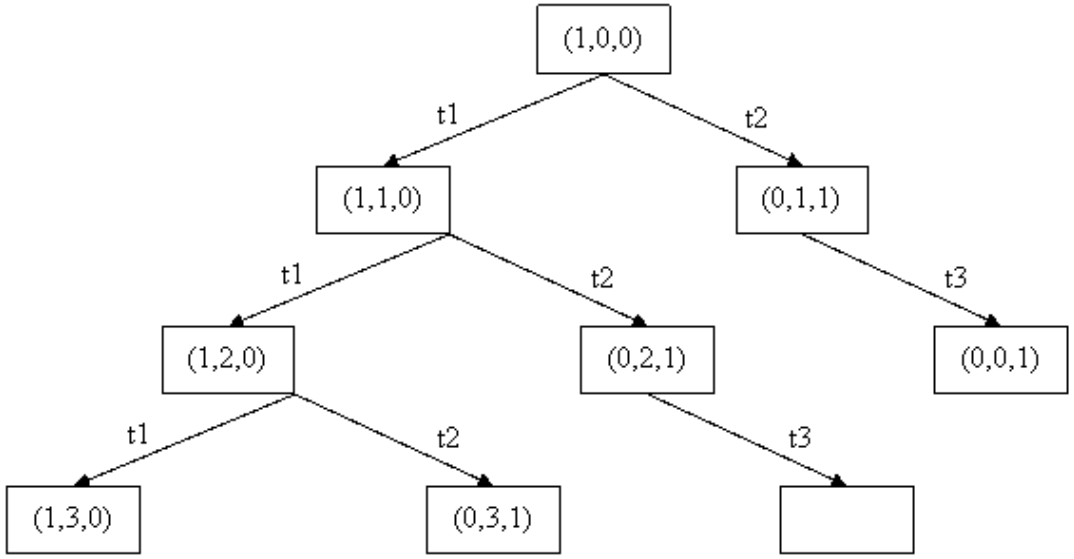


Початкове маркування мережі має вигляд – $(1,0,0)$. Для цього маркування дозволеними є два переходи $t1$ і $t2$. Нехай коренем дерева буде вершина, що відповідає початковому маркуванню. Визначимо нові вершини дерева досяжності для (досяжних) маркувань, які можна отримати в результаті запуску цих двох переходів. Дуга, що позначена переходом, який запускається, з початкового маркування приводить до нового маркування. Часткове дерево показує всі маркування, які безпосередньо є досяжними з початкового маркування.



Тепер розглянемо всі можливі маркування які можна отримати з цих нових маркувань.





Маркування (0,0,1) є пасивним, жодний перехід в ньому не являється дозволеним, тому в дереві досяжності вона не буде породжувати

інших маркувань. Крім того, необхідно відзначити, що маркування $(0,1,1)$, що породжується запуском $t3$ у $(0,2,1)$, було вже породжене безпосередньо з початкового маркування запуском $t2$.

Якщо описану процедуру продовжувати далі, то виявиться, що будь-яке досяжне маркування вже було породжене. Але бувають випадки мережі Петрі коли дерево досяжності виявляється нескінченним навіть тоді, коли множина досяжності є скінченною. Це обумовлено тим, що дерево представляє всі можливі послідовності запусків переходів. Будь-який шлях дерева, починаючи з кореня, відповідає припустимій послідовності переходів. Для того що б дерево стало корисним інструментом аналізу, необхідно знайти засоби його обмеження до скінченного розміру.

У графічному зображенні множини досяжності в дереві досяжності видно, що рівні маркування, які відповідають різним послідовностям переходів, відображаються різними вершинами дерева. В цьому випадку у кожну вершину дерева буде вести єдиний орієнтований шлях від вершини кореня, якому відповідає єдина послідовність переходів. Якщо з метою скорочення розмірів дерева досяжності рівним маркуванням ставити у відповідність одну вершину, то отриманий у результаті граф буде називатись **діаграмою досяжності** маркувань мережі Петрі або **діаграмою станів** мережі Петрі. Для останнього випадку від кореневої вершини діаграми у будь-яку іншу вершину може вести декілька орієнтованих шляхів, кожному з яких відповідає своя послідовність переходів. Зведення до кінцевого

представлення можна виконати декількома способами. Для цього необхідно визначити засоби для обмеження введення нових маркувань (вони називаються граничними вершинами) на кожному кроці побудови. Тут можна застосовувати **пасивні маркування (маркування у яких немає дозволених переходів)**. Такі пасивні маркування ще називають термінальними вершинами. Інший клас маркувань – це маркування, що вже зустрічались у структурі дерева. Такі маркування називаються дублюючими, а відповідні вершини – **дублюючими вершинами**. Для таких вершин немає потреби розглядати наступні маркування, бо всі вони будуть рівними, породженими з місця **першої появи** дублюючої вершини" [12-19].

Маркування мереж Петрі. Правила виконання мереж Петра. Простір станів мереж Петрі.

Маркування (розмітка) M - це надання фішок позиціям мережі Петрі. Фішка - це базове поняття мереж Петрі (подібно до позицій і переходів). Фішки використовуються визначення поняття "виконання мережі Петрі" . Фішки присвоюються позиціям, проте їх кількість та положення фішок при виконанні мережі Петрі можуть змінюватися.

Маркування мережі Петрі C=(P,T,I,O) є функція m, що відображає множину позицій P у множину невід'ємних цілих чисел N: m: P→N.

Маркування m можна визначити як n-вектор: $m=(m_1, m_2,..., m_n)$, де $n=|P|$ і кожне $m_i \in N$, $i=1,...,n$. Вектор m визначає для кожної позиції p_i мережі Петрі кількість фішок в цій позиції: кількість фішок в позиції pi є mi, тобто. $m(p_i) = m_i$. Маркована мережа Петрі M=(C, m) є сукупність структури мережі Петрі C=(P,T,I,O) та маркування m.

Виконанням мережі Петрі керують кількість та розподіл фішок мережі. Фішки перебувають у колах і керує виконанням переходів мережі. Мережа Петрі виконується у вигляді запусків переходів. Перехід запускається видаленням фішок з його вхідних позицій та утворенням нових фішок, що розміщуються в його вихідних позиціях.

Перехід може запускатися лише у тому випадку, коли він дозволений. Перехід називається **дозволеним**, якщо кожна з його вхідних позицій має число фішок, принаймні, рівну дуг з позиції в перехід. Кратні фішки потрібні для кратних вхідних дуг. Фішки у вхідній позиції, які дозволяють перехід, називаються його **дозволяючиими фішками**.

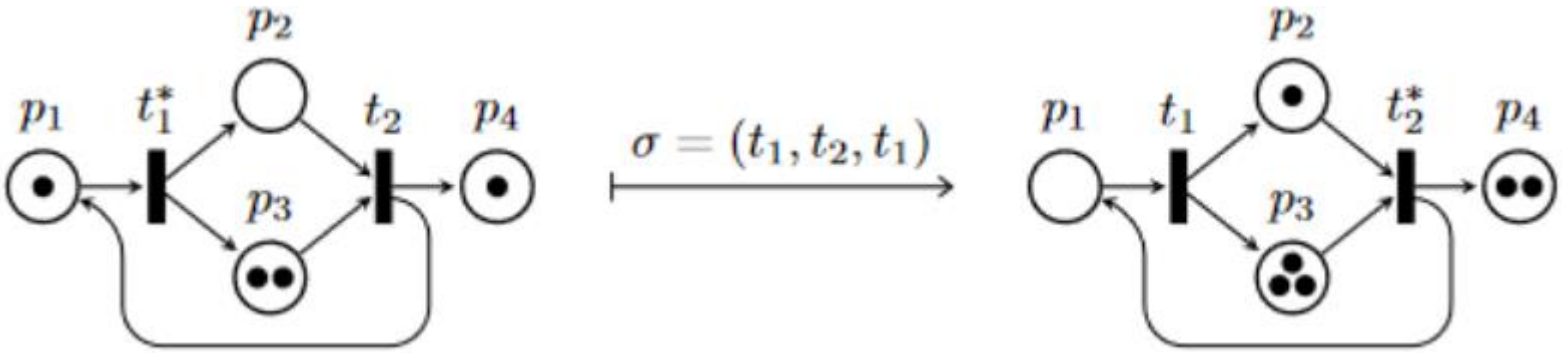
Стан мережі Петрі визначається її маркуванням. Запуск переходу змінює стан мережі Петрі шляхом зміни маркування мережі Петрі. Простір станів мережі Петрі, що має n позиціями, є багато всіх маркувань, тобто. Nn. Зміна у стані, викликане запуском переходу, визначається функцією зміни b, яку ми назвемо **функцією наступного стану.**

Мережа Петрі можна задати і в компактній **векторно-матричній формі**. У цьому випадку структура мережі Петрі (тобто її граф) описується двома матрицями W^+ та W^- розміру n×m, що визначають набори дуг, що ведуть від місць до переходів (матриця W^-) та назад (W^+): w^-_{ik} = числу дуг, що ведуть з i-го місця в k-ий перехід; w^+_{ik} = числу дуг, що ведуть в i-е місце з k-го переходу. З матриць W^+ і W^- складається ще одна матриця

$$W = W^+ - W^-,$$

яка зазвичай використовується для обчислення нового стану (розмітки) мережі після застосування до неї заданої послідовності спрацьовувань. Початкова розмітка мережі задається цілим вектором μ_0 довжини n.

Наприклад, для мережі



матриці W^+, W^- та W дорівнюють

$$W^- = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad W^+ = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad W = \begin{bmatrix} -1 & 1 \\ 1 & -1 \\ 1 & -1 \\ 0 & 1 \end{bmatrix}$$

а її розмітка (у лівій частині малюнка) визначається вектором

$$\mu = [1, 0, 2, 1]$$

Для розмітки μ перехід $t_k \in T$ є активним, якщо виконується умова

$$w_k^- \leq \mu.$$

В останній формулі символ w^-_k позначає k-ий стовпець матриці W^-, а порівняння двох векторів виконується поелементно (тобто вважається, що a ≤ b, якщо кожен елемент a менше або дорівнює відповідного елемента b: $a_i \leq b_i$ для всіх можливих i).

$$w_1^- \leq \mu: \ [1,0,0,0] \leq [1,0,2,1] \text{ и } w_2^- \not\leq \mu: \ [0,1,1,0] \not\leq [1,0,2,1].$$

Нехай для деякої мережі Петрі задано коректну послідовність спрацьовувань σ. Позначимо за v(σ) вектор, k-ий елемент якого дорівнює числу входжень символу t_k у σ. Наприклад, для мережі з двома переходами та послідовності σ = (t1, t2, t1) вектор v = [2, 1].

Тоді легко перевірити, що застосування послідовності σ до початкової розмітки μ призводить до розмітки μ′, векторне подання якої обчислюється за такою формулою:

$$\mu' = \mu + W v(\sigma)$$

$$\mu' = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 & 1 \\ 1 & -1 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 3 \\ 2 \end{bmatrix}$$

Видно, что результат такого векторно-матричного вычисления полностью соответствует разметке сети в правой части рисунка.

Властивості мереж Петрі

Місця та переходи можуть мати різні властивості, чим визначаються властивості самих мереж і їх класифікація.

Властивістю місця є кількість міток, які можуть бути в ньому. Якщо у будь-якій досяжній розмітці число міток у заданому місці буде не більше однієї (0 або 1), то таке місце називається **безпечним**. **Мережа Петрі називається безпечною, якщо її місця безпечні**. У безпечних мережах стан кожного місця описується лише одним бітом, тому такі мережі можуть бути легко реалізовані апаратно, використовуючи ті чи інші види перемикачів (тригерів). Початковий варіант визначення мережі Петрі, даний самим Адамом Петрі, мав на увазі, що мережа є безпечною.

Однак, для більшості додатків вимога безпеки мережі є надто суворою. Його можна послабити: місце називається **k-обмеженим**, якщо в будь-якій досяжній розмітці в даному місці буде не більше k міток. Очевидно, що 1-обмежене місце є безпечним. Місце називається обмеженим, якщо є таке k, що це місце є k-обмеженим. Нарешті, мережа Петрі є k-обмеженою, якщо будь-яке його місце є k-обмеженим і просто обмеженою, якщо всі його місця обмежені. Обмежені мережі також допускають ефективну апаратну реалізацію, де кожне місце представляється вже лічильником (наприклад, регістром) деякої заданої ємності. Необмежені мережі мають, як правило, лише теоретичний інтерес.

Ще однією властивістю є властивість **консервативності**. Мережа називається консервативною, якщо число міток у будь-якій досяжній розмітці зберігається рівним числу міток у початковій розмітці. Така модель використовується, наприклад, у випадках, коли мітки є деякі ресурси системи, які знищуються і створюються. Ці ресурси можуть переходити від однієї частини системи до іншої, але їхня сумарна кількість у процесі роботи системи не змінюється. Будь-який перехід, який зустрічається хоча б в одній досяжній розмітці, повинен мати однакову кількість вхідних і вихідних дуг, скільки він вибрав міток, стільки він повинен їх і поставити.

ЗАВДАННЯ АНАЛІЗУ МЕРЕЖ ПЕТРІ.

Аналіз полягає у вивченні основних властивостей мереж Петрі: **безпеки, обмеженості, збереження, активності, досяжності та покриваності.**

обмеженість мережі Петрі - властивість мережі, число міток якої будь-якої позиції мережі неспроможна перевищити деякого значення K;

безпека мережі Петрі - є окремий випадок обмеженості, K = 1;

зберігання мережі Петрі - є сталість завантаження ресурсів, коли $\Sigma A_i N_i$ постійна. Де N_i – число маркерів у і-тій позиції, A_i – ваговий коефіцієнт;

досяжність мережі Петрі - можливість переходу мережі з одного заданого стану (що характеризується розподілом міток) до іншого;

жвавість мережі Петрі - можливість спрацьовування будь-якого переходу при функціонуванні об'єкта, що моделюється.

В основі дослідження цих властивостей лежить **аналіз досяжності**. Методи аналізу властивостей мереж Петрі ґрунтуються на використанні графів досяжних (покриваючих) маркувань, вирішенні рівняння станів мережі та обчисленні лінійних інваріантів позицій та переходів. Застосовуються також допоміжні методи редукції, що дозволяють зменшити розмір мережі Петрі зі збереженням її властивостей та декомпозиції, що розділяють вихідну мережу на підмережі.

Безпека. Позиція $p_i \in P$ мережі $C=(P, T, I, O, M_0)$ є безпечною, якщо $m(p_i) \in I$ для будь-якої $M \in R(C, M_0)$. Мережа Петрі безпечна, якщо безпечна кожна її позиція.

Безпека – важливе властивість апаратної реалізації. Безпечна позиція має число міток 0 або 1 і може бути реалізована одним тригером.

Мережі, в яких позиції розглядаються (інтерпретуються) як передумови подій, маркування кожної позиції має бути безпечним.

Обмеженість. Безпека – це окремий випадок більш загальної якості обмеженості. Безпека дозволяє реалізувати позицію тригером, а більш загальному випадку можна використовувати лічильник. Будь-який лічильник обмежений за максимальним

числом К. Відповідна позиція також є К-безпечною або К-обмеженою, якщо кількість міток у ній не може перевищити ціле число К. Позиція $p_i \in P$ мережі $C=(P, T, I, O, M0)$ є безпечною, якщо $m(p_i) \in К$ для всіх $M \in R(C, M0)$. Позиція називається обмеженою, якщо вона К безпечна для деякого К. Мережа Петрі обмежена, якщо її позиції обмежені.

Збереження. У мережах Петрі, що моделюють запити, розподілу та звільнення ресурсів, деякі позиції можуть представляти стан ресурсів. Наприклад, якщо три мітки в позиції представляють три пристрої (однотипні) в обчислювальній системі, то інтерес представляє властивість збереження міток. Тобто мітки, що становлять ресурси, ніколи не створюються і не знищуються.

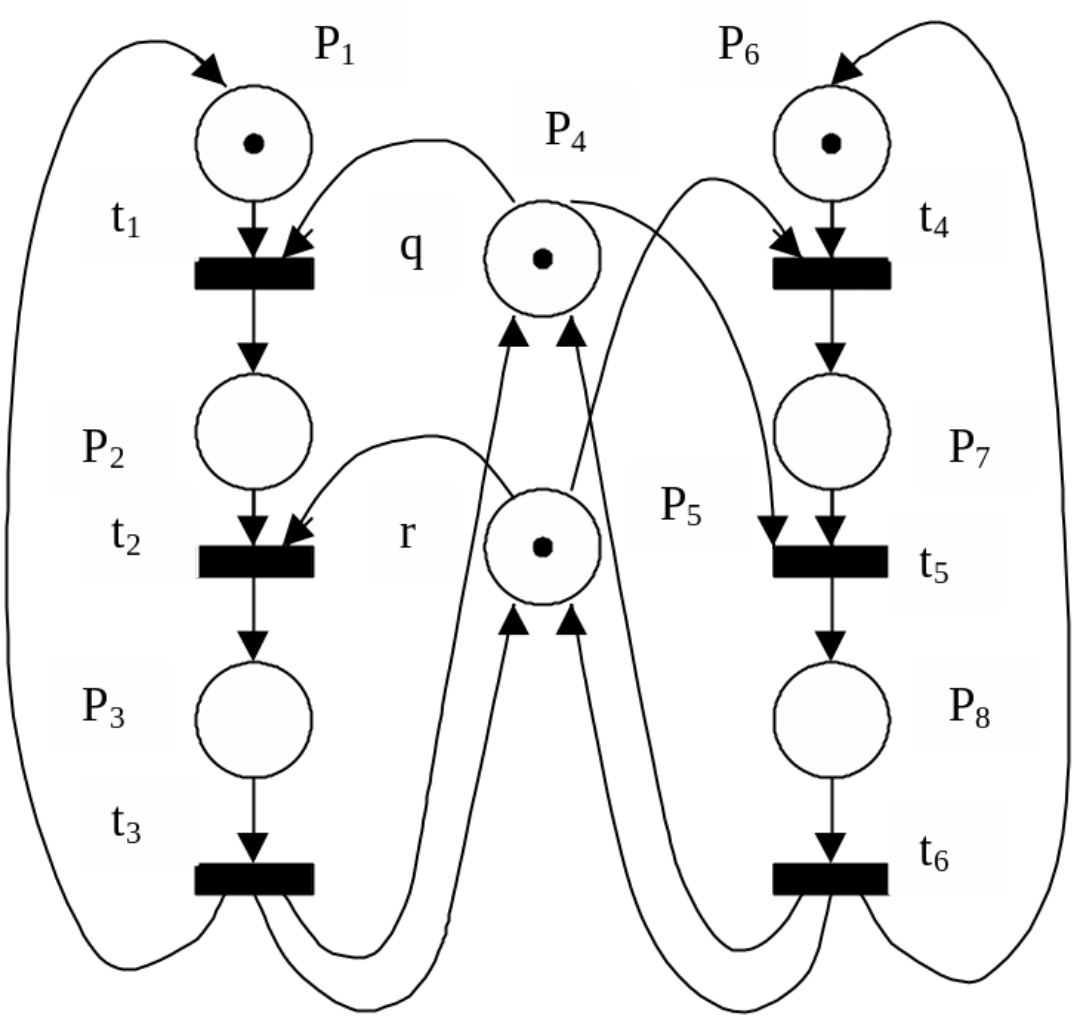
Мережа Петрі називається суворо зберігає, якщо всім $M \in R(C, M0)$ виконується умова:

$m(pi) = m_0(pi)$, $pi \in P$

Мережа Петрі повинна зберігати ресурси, що вона моделює. Однак не завжди є однозначна відповідність між міткою та кількістю чи кількістю ресурсів. У цьому випадку мітка використовується для створення кратних міток (по одній на ресурс) шляхом запуску переходу з більшим числом виходів, ніж входів. Тому вводяться виважені мітки, а умова збереження визначається через виважену суму міток.

Активність. Іншим завданням, що виникає при розподілі ресурсів, є виявлення тупиків. Розглянемо систему, що включає два різні

ресурси q і r і два процеси а) і в), які потребують обох ресурсів. Кожен процес вимагає ресурс, а потім звільняє його. Процес а) спочатку запитує ресурс q, потім ресурс r і звільняє їх. Процес в) працює аналогічно, але запитує спочатку ресурс r, а потім q.



Початкове маркування позначає ресурси вільними та готовність процесів до роботи. Виконання мережі в послідовності t1, t2, t3, t4,

t5, t6 або t4, t5, t6, t1, t2, t3 не призводить до безвиході. Якщо почати з переходів t1, t4 то виконання заблоковане, процес а) має ресурс q і хоче отримати r, процес в) має r і хоче отримати q.

Безвихідь в мережі Петрі - це перехід (або множина переходів), які не можуть бути запущені. Переходи t2 та t5 є тупиковими. Перехід називається активним, якщо не заблокований, тобто потенційно запустимым.

Досяжність та покриваність. Завдання досяжності полягає у визначенні маркування M_0 маркування $M \in R(C, M_0)$. До цього завдання можуть зводитися багато перерахованих вище завдань. Наприклад, глухий кут у мережі на малюнку може виникнути, якщо досяжним є стан (0, 1, 0, 0, 0, 0, 1, 0).

Завдання покриваності. Для мережі C з початковим маркуванням M_0 і маркування M визначити, чи існує таке досяжне маркування $M' \in R(C, M0)$, що $M' \in M$.

Завдання досяжності і покриваності можуть розглядатися щодо деяких цікавих для нас підмножин

Фізична інтерпретація мережі Петрі під час моделювання комп'ютерних мереж

Позиції місце зберігання даних;

Мітки – дані, що передаються;

Переходи – обробка даних.

Позиції – обробка даних;

Мітки – дані, що передаються;

Переходи – місце зберігання даних.

Недоліки мереж Петрі

Як зазначає професор Массачусетського технологічного інституту, США, Карл Хьюїт (англ. Carl Hewitt), мереж Петрі притаманні такі недоліки:

вони моделюють керування потоком, але не сам потік даних;

складність опису одночасних дій, що відбуваються під час обчислювального процесу;

фізична інтерпретація переходу у мережах Петрі дуже сумнівна.

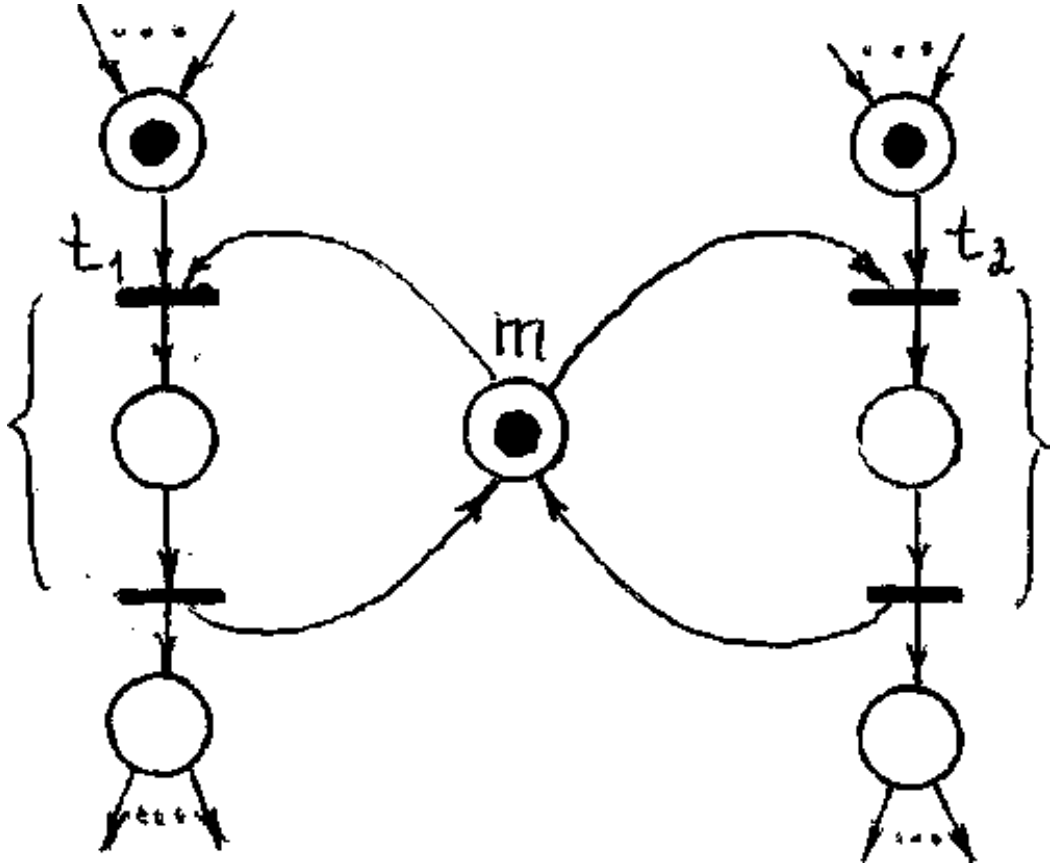
Громіздкість мережі Петрі для моделювання паралельних процесів

Завдання про взаємне виключення

Це завдання виникає в умовах, коли кілька процесів поділяють загальну змінну. Кожен процес може спочатку вважати значення змінної, потім обчислити нове значення і записати його те саме місце. Якщо два процеси одночасно виконуватимуть зазначену послідовність дій, може виникнути неправильна робота системи. Наприклад, обидва процеси вважають по черзі старе значення змінної, а при записі одне з нових значень записане першим буде знищено. Таким чином, результат роботи одного з двох процесів буде втрачено.

Для правильної роботи системи необхідно забезпечити механізм взаємного виключення, при якому одночасно не більше ніж один

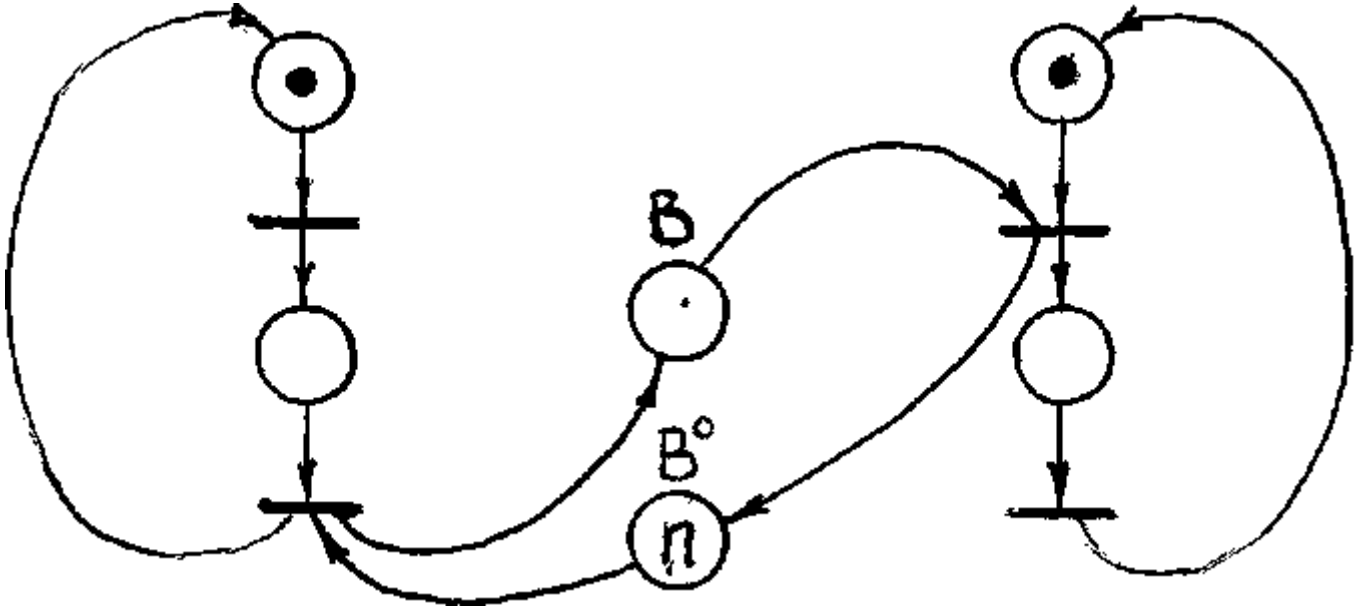
процес має доступ до змінної, що розділяється. Частина програмного коду процесу, в якому виконується доступ до змінної, що розділяється і при цьому потрібен захист від доступу до розділюваної змінної іншого процесу, називається критичною секцією. Механізм взаємного виключення працює таким чином, що при виконанні критичної секції будь-якого процесу критичні секції інших процесів блокуються.



Переходи і перебувають у конфлікті за мітку у позиції. Запуск, наприклад, переходу, змушує перший процес чекати, поки другий процес вийде зі своєї критичної секції і поверне мітку в позицію.

Завдання про виробника/споживача

У цій задачі об'єктом, що розділяється, є буфер. Процес виробник породжує компонент інформації і розміщує її в буфер. Процес споживач чекає, поки компонента інформації розміститься в буфері, після цього він може її видалити з буфера і використовувати. Використовується буфер обмеженої ємності. В цьому випадку швидкість заповнення буфера процесами-виробниками та швидкість вилучення даних процесами - споживачами повинні бути зіставні. Буферу зіставлені дві позиції: B – вказує на кількість компонентів даних розміщених у буфері, але ще не використаних, B^0 – кількість вільних місць у буфері для розміщення компонентів даних.

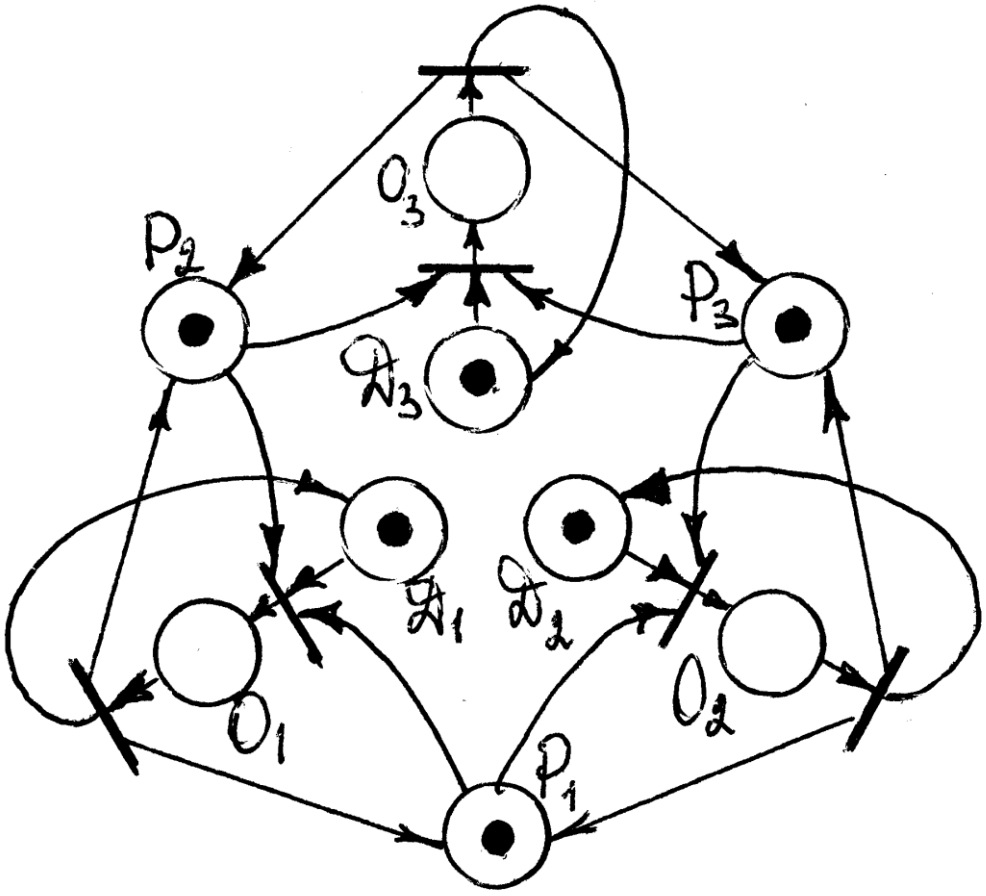


Спочатку позиція B має n міток, а позиція B^0 не має. Якщо буфер буде повністю заповнений, то позиції B буде n міток, а B° − 0 міток. Таким чином, доступ процесу-виробника в заповнений буфер неможливий, так як в позиції буде 0 міток.

Завдання про мудрих, що обідають.

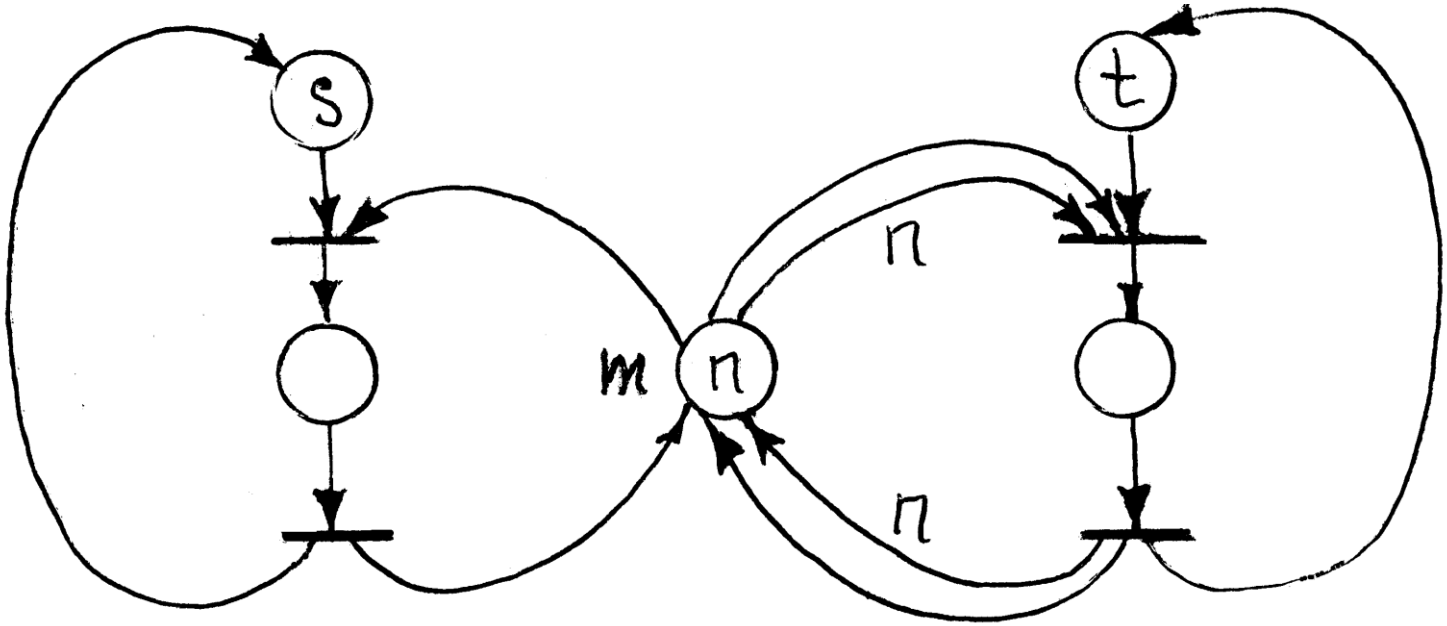
Завдання про обідаючих мудреців було запропоновано Дейкстрой і пов'язане мудрецями сидять за круглим столом і кожен з них може перебувати в одному з двох станів: думає або обідає. На столі страви китайської кухні, а між мудрецями лежить по одній паличці. Для їжі мудрець повинен взяти паличку зліва і паличку праворуч. У цьому випадку сусіди мудреця, що обідає, можуть тільки думати і чекати, коли звільняться палички, щоб приступити до обіду. Мережею Петрі представлено взаємодію трьох процесів (трьох мудреців), стан яких відбито позиціями і : O_i –обідає, D_i – думає. Позиції представляють палички для їжі (розділені ресурси). У вихідному маркуванні мудреці думають, а всі палички вільні. При

такому вихідному маркуванні будь-який із трьох мудреців може приступити до обіду, захопивши палички зліва та праворуч.



Завдання про читання/запис

Розглядається взаємодія процесів двох типів: процеси читання та процеси запису. Усі процеси спільно використовують спільний файл або змінну чи елемент даних. Процеси читання не змінюють об'єкт, а процеси запису змінюють. Тому процеси запису повинні взаємно виключати (всі інші процеси читання та запису, тоді як кілька процесів читання можуть мати доступ до даних, що розділяються одночасно. Взаємодія процесів необхідно організовувати так, щоб не могла виникнути тупикова ситуація та було забезпечено механізм взаємного виключення з боку процесів запису.



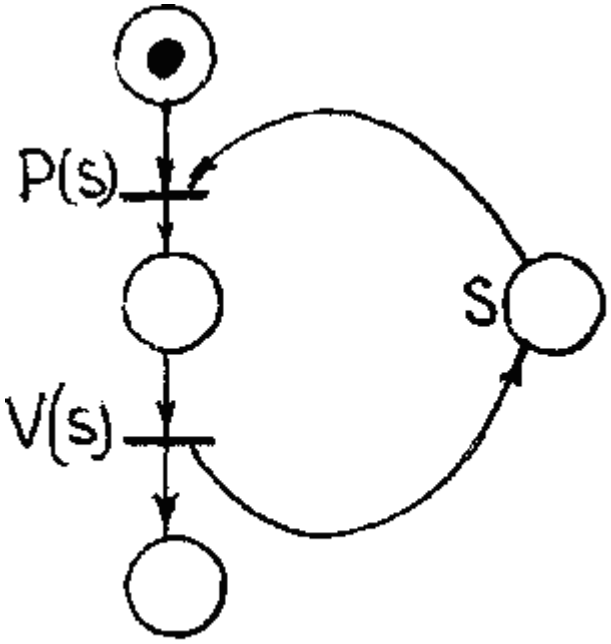
Представлена мережа Петрі, що імітує взаємодію процесів читання та запису за умови, що кількість процесів читання та запису обмежена величиною n. Це означає, що і при необмеженому числі процесів читання одночасно можуть виконуватися не більше процесів. Початкове маркування мережі передбачає наявність процесів читання та процесів запису. З цієї мережі видно, що у

конфлікті за мітку у позиції . При захопленні міток процесом запису позиція залишається без міток, тим самим блокується запуск процесів читання, доки процес запису не поверне міток у позицію . Слід звернути увагу, що у цій мережі позиція після завершення процесів читання завжди матиме міток і цим дозволяє запуск процесів записи.

P - і V - операції над семафорами

Одним із поширених механізмів синхронізації процесів є P- та V-операції над семафорами. Семафор – це змінна, яка може набувати лише невід'ємних цілей. V – операція збільшує значення на одиницю, а P-операція зменшує його на одиницю. P-операція може виконуватися тільки у тому випадку, коли отримане значення семафору залишається невід'ємним. Таким чином, якщо значення семафору дорівнює 0, то P-операція повинна чекати, поки якийсь інший процес не виконає V-операцію. P- і V-операції визначені як примітивні, тобто ніяка інша операція не може змінювати значення семафору одночасно з ними. На рис.5.8 показано, як такі операції моделюються мережею Петрі. Кожен семафор моделюється

позицією, кількість міток r позиції показує значення семафора. P-операції використовують позицію семафора як вход, а V-операції як вихід.



Лекція 10

МЕТОД АНАЛІЗУ ІЄРАРХІЙ. ПОНЯТТЯ ПРО ІЄРАРХІЇ, ПОБУДОВА ІЄРАРХІЙ

МЕТОД АНАЛІЗУ ІЄРАРХІЙ. ШКАЛА СААТІ. РОЗРАХУНОК ЛОКАЛЬНИХ ПРІОРИТЕТІВ. МЕТОД РАНЖУВАННЯ ФАКТОРІВ ІНФОРМАЦІЙНИХ ПРОЦЕСІВ

Метод аналізу ієрархій

“Метод аналізу ієрархій (MAI) — це структурований метод організації та аналізу складних рішень, заснований на математиці та психології. Він був розроблений Томасом Л. Сааті в 1970-х роках, який співпрацював з Ернестом Форманом для розробки вибору експертів у 1983 році, і з тих пір він широко вивчався та вдосконалювався. Він представляє точний підхід для кількісної оцінки ваги критеріїв прийняття рішень.

Для оцінки відносної величини факторів за допомогою парних порівнянь використовується досвід окремих експертів. Кожен з респондентів повинен порівняти відносну важливість між двома пунктами відповідно до спеціально розробленої анкети (хоча більшість опитувань прийняли п'ятибальну шкалу Лікерта, анкета MAI становить від 1 до 9).

MAI має особливе застосування в груповому прийнятті рішень, і використовується у всьому світі в широкому спектрі ситуацій прийняття рішень у таких галузях як уряд, бізнес, промисловість, охорона здоров'я, суднобудування та освіта.

Замість того, щоб призначити **«правильне» рішення**, MAI допомагає особам, які приймають рішення, знайти рішення, яке **найкраще відповідає їх цілі та їхньому розумінню проблеми**. Він забезпечує всеосяжну та раціональну основу для структурування проблеми прийняття рішень, для представлення та кількісної оцінки її елементів, для зв'язку цих елементів із загальними цілями та для оцінки альтернативних рішень.

Користувачі MAI спочатку **розкладають свою проблему** прийняття рішень **на ієрархію легших для сприйняття підпроблем**, кожна з яких може бути проаналізована незалежно. Елементи ієрархії можуть стосуватися будь-якого аспекту проблеми прийняття рішення — матеріального чи нематеріального, ретельно виміряного або грубо

оціненого, добре або погано зрозумілого — будь-чого, що стосується відповідного рішення.

Після того, як ієрархія побудована, особи, що приймають рішення, систематично оцінюють різні її елементи, порівнюючи їх один з одним попарно, з огляду на їх вплив на елемент над ними в ієрархії. Здійснюючи порівняння, особи, що приймають рішення, можуть використовувати конкретні дані про елементи, але вони, як правило, використовують свої судження щодо відносного значення та важливості елементів. **Суть MAI полягає в тому, що при проведенні оцінок можна використовувати людські судження, а не лише основну інформацію**.

MAI перетворює ці оцінки на числові значення, які можна обробити та порівняти протягом усієї проблеми. Числова вага або пріоритет виводиться для кожного елемента ієрархії, що дозволяє порівнювати між собою різні та часто неспівставні елементи раціональним та послідовним способом. Ця можливість відрізняє MAI від інших методів прийняття рішень.

На завершальному етапі процесу обчислюються числові пріоритети для кожної з альтернатив рішення. Ці цифри відображають відносну здатність альтернатив досягти цілі прийняття рішення, тому вони дозволяють прямо розглянути різні напрямки дій.

Незважаючи на те, що його можуть використовувати особи, які працюють над прямолінійними рішеннями, MAI є найбільш корисним, коли групи людей працюють над складними проблемами, особливо тими, де є високі ставки, за участю людського сприйняття та суджень, де рішення мають довгостроковий характер, наслідки. Він має унікальні переваги, коли важливі елементи рішення важко кількісно визначити або порівняти, або коли спілкуванню між членами команди перешкоджають їхні різні спеціалізації, термінології чи перспективи.

Ситуації прийняття рішень, до яких може застосовуватися MAI, включають:

Вибір — вибір однієї альтернативи із заданого набору альтернатив, зазвичай там, де задіяно кілька критеріїв прийняття рішення.

Ранжування — Введення набору альтернатив у порядку від найбільш до найменш бажаного.

Пріоритетність — Визначення відносних достоїнств членів набору альтернатив, на відміну від вибору одного або просто ранжування їх

Розподіл ресурсів — Розподіл ресурсів між набором альтернатив

Бенчмаркинг — Порівняння процесів у власній організації з процесами інших організацій з найкращими практиками

Управління якістю — Розгляд багатовимірних аспектів якості та вдосконалення якості

Вирішення конфліктів — Вирішення суперечок між сторонами з, імовірно, несумісними цілями або позиціями

Застосування MAI до складних ситуацій прийняття рішень налічують тисячі прикладів, і дали значні результати у проблемах, пов'язаних із плануванням, розподілом ресурсів, встановленням пріоритетів та вибором серед альтернатив. Інші сфери включають прогнозування, загальне управління якістю, розробку бізнес-процесів, розгортання функції якості та збалансовану систему показників. Про багато програм MAI ніколи не повідомляють у всьому світі, оскільки вони мають місце на високих рівнях великих організацій, де міркування безпеки та конфіденційності забороняють їх розголошення. Але деякі способи використання MAI обговорюються в літературі.

Нещодавно вони включали:

Вибір типу ядерного реактора (Міланський політехнічний університет)

Вирішення, як найкраще зменшити вплив глобальної зміни клімату(Fondazione Eni Enrico Mattei)

Кількісна оцінка загальної якості програмних систем (Microsoft Corporation)

Вибір університетського факультета (Bloomsburg University of Pennsylvania)

Вирішення питання про місцезнаходження офшорних виробництв (Кембриджський університет)

Оцінка ризику при експлуатації міжнародних нафтопроводів (Американське товариство цивільних інженерів)

Вирішення, як найкраще керувати вододілами США (Міністерство сільського господарства США)

Визначення та оцінка найбільш ефективних підходів до впровадження SAP (SAP Experts)

Прискорене будівництво мостів — інструмент прийняття рішень, який допомагає визначити життєздатність прискореного будівництва мостів у порівнянні з традиційними методами будівництва та у виборі відповідних стратегій будівництва та укладання договорів для кожного конкретного випадку.

Виконання

Як видно з наступного матеріалу, використання MAI передбачає математичний синтез численних суджень про рішення розв'язуваної проблеми. Нерідкі випадки, коли ці судження нараховуються десятками чи навіть сотнями. Хоча підрахунки можна робити вручну або за допомогою калькулятора, набагато частіше використовується один із декількох комп'ютеризованих методів введення та синтезу суджень.

Найпростіші з них включають стандартне програмне забезпечення для електронних таблиць, в той час як найскладніші використовують спеціальне програмне забезпечення, часто доповнене спеціальними пристроями для отримання суджень тих, хто приймає рішення, зібраних у залі засідань.

Процедуру використання MAI можна коротко описати так:

Змоделювати проблему як ієрархію, що містить мету прийняття рішення, альтернативи для її досягнення та критерії оцінки альтернатив.

Встановити пріоритети серед елементів ієрархії, зробивши ряд суджень на основі попарного порівняння елементів. Наприклад, порівнюючи потенційні закупівлі комерційної нерухомості, інвестори

можуть сказати, що віддають перевагу розташуванню над ціною та ціні перед термінами.

Синтезувати (об'єднати) ці судження, щоб отримати набір загальних пріоритетів для ієрархії. Поєднати наприклад судження інвесторів про місце розташування, ціну та терміни для об'єктів A, B, C та D у загальних пріоритетах для кожного об'єкта нерухомості.

Перевірити узгодженість суджень.

Підійти до остаточного рішення за результатами цього процесу.

Для представлення результатів оцінок у кількісному виразі Т.Сааті вводить шкалу парних порівнянь. Згідно з цією шкалою нас не цікавитиме відсутність фізичних чи об'єктивних одиниць виміру. Основною перевагою цього методу є те, що він є безрозмірним і не виникає проблем при приведенні до однакових одиниць виміру.

Правомірність цієї шкали доведена теоретично і практично при порівнянні з багатьма іншими відомими даними. Досвід показав, що при проведенні парних порівнянь, в основному, ставляться запитання: "Який з елементів є важливішим? Який най вірогідніший? Який з них найпривабливіший?".

Відносна важливість (бали)	**Визначення**	**Пояснення**
1	однакова важливість	обидва елементи вносять однаковий вклад
3	один елемент трохи важливіший за другий	досвід дозволяє поставити один елемент трохи вище за другий
5	суттєва перевага	досвід дозволяє встановити безумовну перевагу одного над другим
7	значна перевага	один елемент настільки важливіший за другий, що є практично значимим
9	абсолютна перевага одного над другим	очевидність переваги підтверджується більшістю
2,4,6,8	проміжні оцінки між сусідніми твердженнями	компромісне рішення
обернені величини чисел, наведених вище	якщо при порівнянні одного елемента з другим, отримане одне з вищевказаних чисел (1-9), то при порівнянні другого з першим, матимемо обернену величину	

MAI є систематичною процедурою ієрархічного представлення елементів, що визначають суть будь-якої проблеми. Існує кілька видів ієрархій:

• домінантні - схожі на перевернуте дерево;

• холархії - з оберненим зв'язком;

• модулярні — від простого до складного.

MAI полягає в декомпозиції (розкладанні) проблеми на все більш прості складові частини і подальшій обробці послідовності тверджень особи, яка приймає рішення, за допомогою парних порівнянь. В результаті може бути виражений відносний ступінь взаємодії в ієрархії. Ці твердження потім виражаються чисельно.

Визначення ієрархії

Ієрархія — це стратифікована система ранжування та організації людей, речей, ідей тощо, де кожен елемент системи, крім найвищого, підпорядкований одному або декільком іншим елементам. Хоча поняття ієрархії легко сприймається інтуїтивно, його також можна описати математично. Діаграми ієрархій часто мають форму піраміди, але крім необхідності мати один елемент у верхній частині, ієрархії необов'язково мати форму піраміди.

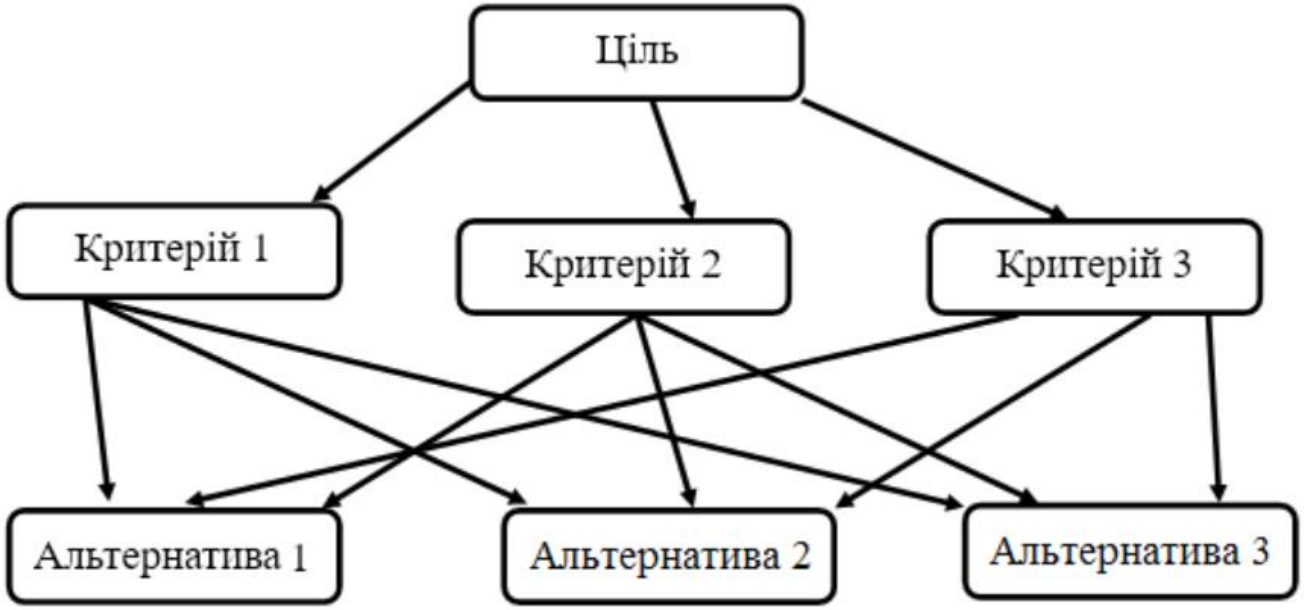
Людські організації часто структуровані як ієрархії, де ієрархічна система використовується для розподілу відповідальності, здійснення керівництва та полегшення спілкування.

У світі ідей ми використовуємо ієрархію, щоб допомогти собі отримати детальне знання складної реальності: ми структуруємо реальність на її складові частини, а вони, в свою чергу, на свої складові частини, рухаючись вниз по ієрархії на стільки рівнів, скільки нам цікаво. На кожному кроці ми зосереджуємось на розумінні окремого компонента цілого, тимчасово нехтуючи іншими компонентами на цьому та всіх інших рівнях. Проходячи цей процес, ми збільшуємо наше глобальне розуміння складної реальності.

Подібним чином, коли ми підходимо до складної проблеми прийняття рішення, ми можемо використовувати ієрархію, щоб інтегрувати великі обсяги інформації в наше розуміння ситуації.

Ієрархії в MAI

Ієрархія MAI є структурованим засобом моделювання відповідного рішення. Вона складається із загальної мети, групи варіантів або альтернатив для досягнення мети та групи факторів або критеріїв, які пов'язують альтернативи з метою.



Критерії можуть бути далі розбиті на підкритерії, під-критерії тощо на стільки рівнів, скільки вимагає проблема. Критерій може застосовуватись не однорідно, але, може мати градуйовані відмінності, як-от трохи солодощі приємно, але занадто багато солодощі може завдати шкоди. У цьому випадку критерій поділяється на підкритерії, що вказують на різну інтенсивність критерію, такі як: мала, середня, висока, і ці інтенсивності мають пріоритет через порівняння за батьківським критерієм, солодкість. Опубліковані описи програм MAI часто включають схеми та описи їх ієрархій; Складні ієрархії MAI були зібрані та передруковані щонайменше в одній книзі. Структура будь-якої ієрархії MAI буде залежати не тільки від характеру розглядуваної проблеми, а й від знань, суджень, цінностей, думок,

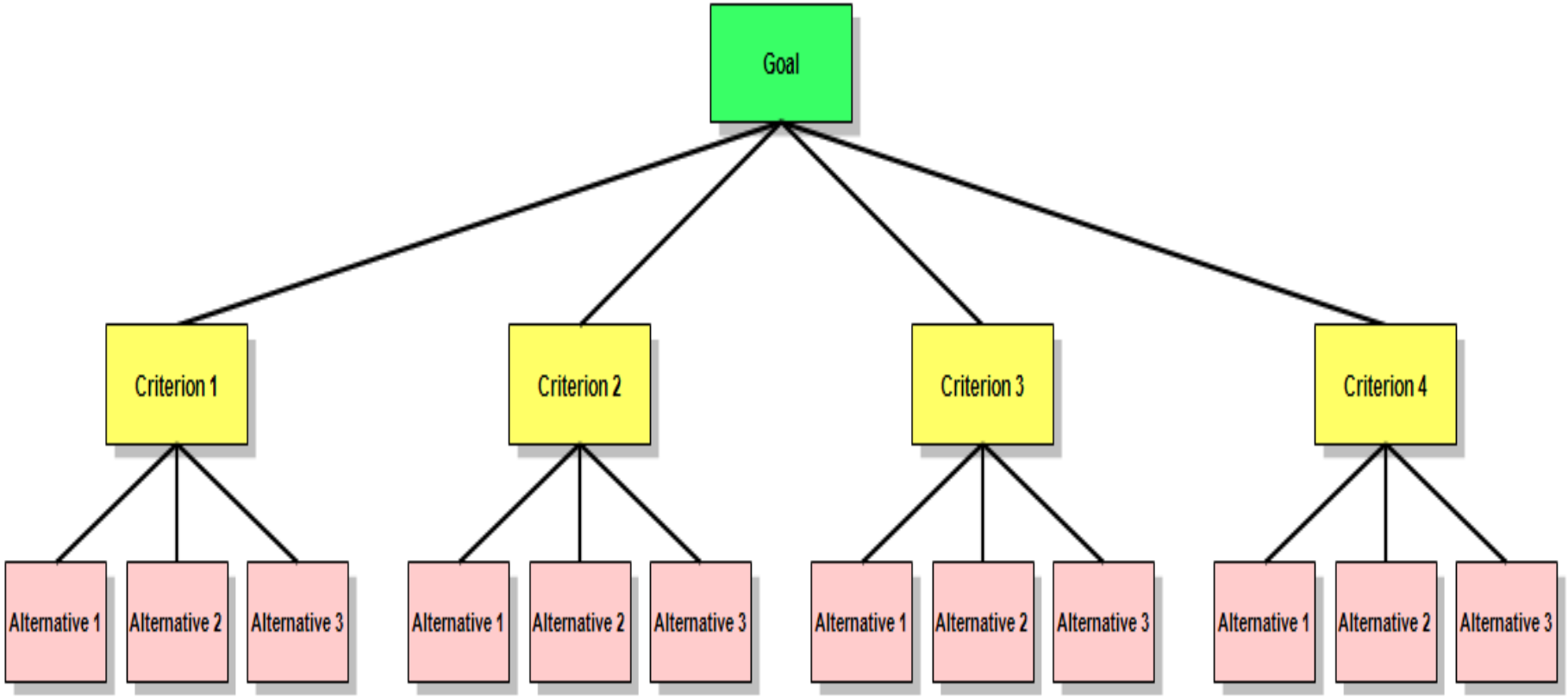
потреб, потреб тощо учасників процесу прийняття рішень. Побудова ієрархії, як правило, передбачає значні обговорення, дослідження та відкриття учасниками. Навіть після початкової побудови його можна змінити з урахуванням нещодавно продуманих критеріїв або критеріїв, які спочатку не вважалися важливими; альтернативи також можна додавати, видаляти або змінювати.[

Щоб краще зрозуміти ієрархію MAI, необхідно розглянути проблему прийняття рішення з метою, яку потрібно досягти, трьома альтернативними способами досягнення цілі та чотирма критеріями, за якими альтернативи потрібно вимірювати.

Таку ієрархію можна візуалізувати як схему, подібну до тієї, що знаходиться безпосередньо внизу, з ціллю вгорі, трьома

альтернативами внизу та чотирма критеріями між ними. Існують корисні терміни для опису частин таких діаграм: Кожне поле називається вузлом. Вузол, який підключений до одного або декількох вузлів на рівні нижче нього, називається батьківським вузлом. Вузли, до яких він так зв'язаний, називаються його дочірніми.

Застосовуючи ці визначення до наведеної нижче схеми, мета є батьківською для чотирьох критеріїв, а чотири критерії є дочірніми для цілі. Кожен критерій є батьківським для трьох альтернатив. Зверніть увагу, що є лише три альтернативи, але на схемі кожна з них повторюється під кожним із батьківських вузлів.


Goal
Criterion 1
Criterion 2
Criterion 3
Criterion 4
Alternative 1
Alternative 2
Alternative 3
Alternative 1
Alternative 2
Alternative 3
Alternative 1
Alternative 2
Alternative 3
Alternative 1
Alternative 2
Alternative 3

Оцінка ієрархії

Після побудови ієрархії учасники аналізують її через серію попарних порівнянь, які отримують числові шкали вимірювань для вузлів. Критерії попарно порівнюються по важливості для цілі. Альтернативи попарно порівнюються за кожним із критеріїв. Порівняння обробляються математично, а пріоритети виводяться для кожного вузла.

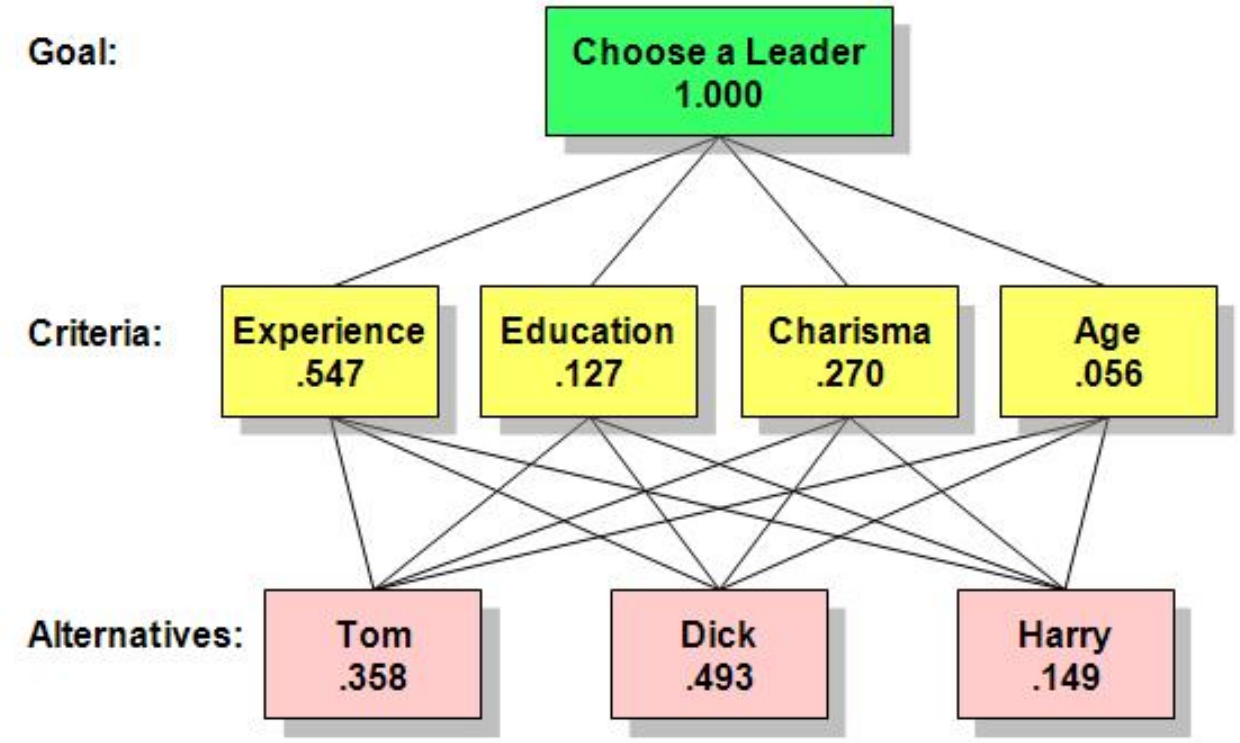
Розглянемо приклад «Вибір лідера». Важливим завданням осіб, які приймають рішення, є визначення ваги, яку слід надати кожному критерію. Іншим важливим завданням є визначення ваги, яку слід додати кожному кандидату з урахуванням кожного з критеріїв. MAI не лише дозволяє це робити, але і дозволяє надати значуще та об'єктивне числове значення кожному з чотирьох критеріїв.

AHP: Choosing a Leader


Tom


Dick


Harry



Встановлення пріоритетів

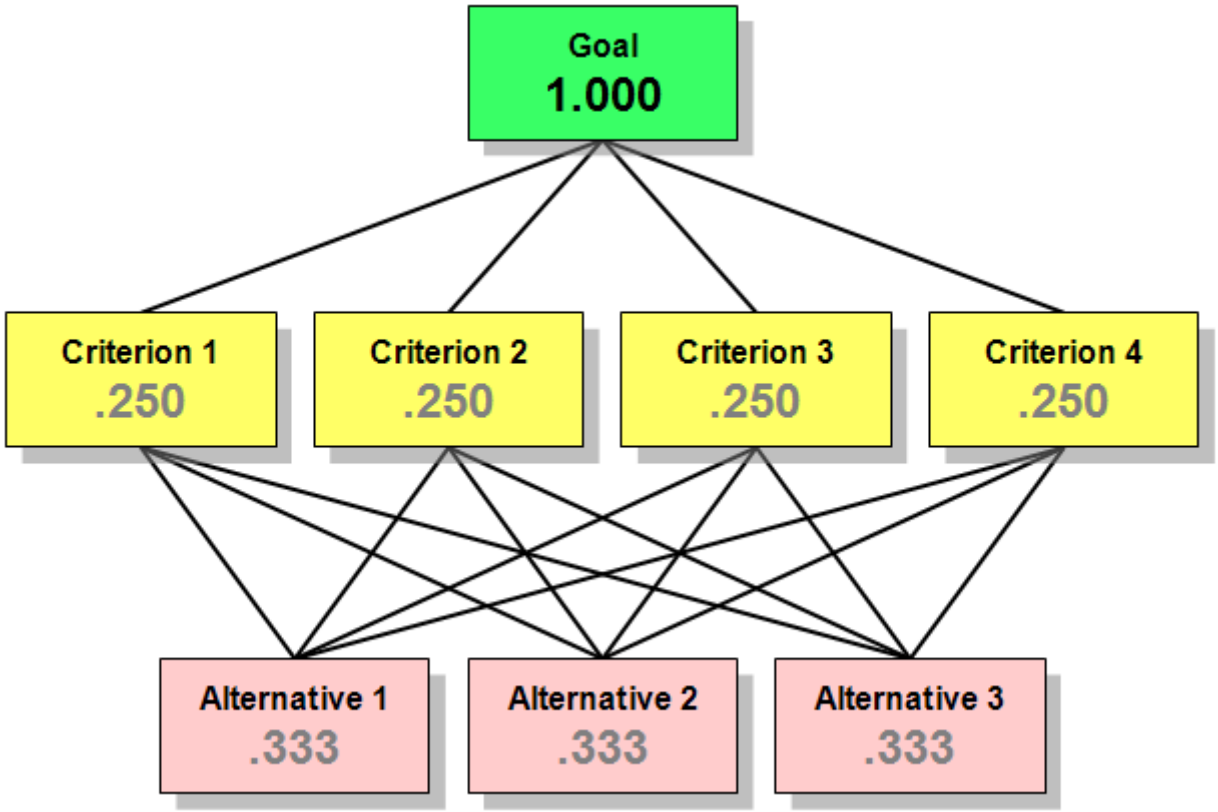
Визначення та пояснення пріоритетів

Пріоритетами є числа, пов'язані з вузлами ієрархії MAI. Вони представляють відносну вагу вузлів будь-якої групи.

Як і ймовірності, пріоритетами є безрозмірні величини між нулем та одиницею, без одиниць виміру та розмірів. Вузол з пріоритетом 0,200 має вдвічі більшу вагу для досягнення цілі, як той, що має пріоритет 0,100, у десять разів перевищує вагу з пріоритетом 0,020 тощо. Залежно від розглянутої проблеми, «вага» може означати важливість, перевагу, вірогідність або будь-який фактор, який розглядається особами, що приймають рішення.

Пріоритети розподіляються по ієрархії відповідно до її архітектури, і їх значення залежать від інформації, що вводиться користувачами процесу. Пріоритети Цілі, Критеріїв та Альтернативи тісно пов'язані, але їх слід розглядати окремо.

За визначенням, пріоритетом Цілі є 1. Пріоритети альтернатив завжди складають до 1. Речі можуть ускладнитися з кількома рівнями критеріїв, але якщо існує лише один рівень, їх пріоритети також додаються до 1,000.



Зауважте, що пріоритети на кожному рівні прикладу — мета, критерії та альтернативи — складають до 1.

Показані пріоритети — це ті, які існують до введення будь-якої інформації про ваги критеріїв чи альтернатив, тому пріоритети на

кожному рівні рівні. Їх називають пріоритетами ієрархії за замовчуванням. Якби до цієї ієрархії було додано п'ятий критерій, пріоритетом за замовчуванням для кожного критерію було б 0,200. Якби було лише дві Альтернативи, кожна мала б пріоритет за замовчуванням 0,500.

Дві додаткові концепції застосовуються, коли ієрархія має більше одного рівня критеріїв: місцеві пріоритети та глобальні пріоритети.

Правило таке: у рамках ієрархії глобальні пріоритети дочірніх вузлів завжди складаються із загальним пріоритетом їх батьків. У групі дітей місцеві пріоритети складають до 1,000.

Поки що ми розглядали лише пріоритети за замовчуванням. По мірі просування процесу аналізу ієрархії пріоритети змінюватимуться від

значень за замовчуванням, оскільки особи, що приймають рішення, вводять інформацію про важливість різних вузлів. Вони роблять це, роблячи серію попарних порівнянь.

Результатом попарних порівнянь альтернатив є матриця попарних порівнянь альтернатив виду:

$$\begin{array}{c} \\ A_1 \\ \dots \\ A_n \end{array} \begin{array}{c} \begin{array}{ccc} A_1 & \dots & A_n \end{array} \\ \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ & & \\ a_{n,1} & \dots & a_{n,n} \end{bmatrix} \end{array}$$

де по головній діагоналі розташовано одиниці,

$$a_{j,i} = \frac{1}{a_{i,j}} \text{ та } a_{j,k} = \frac{a_{i,k}}{a_{i,j}}$$

Визначення вагових коефіцієнтів з попарних порівнянь

Процедура визначення вагових коефіцієнтів з попарних порівнянь може бути використана як для оцінки пріоритетів самих критеріїв, так і для оцінки альтернатив за результатами попарних порівнянь.

У якості вхідних даних приймається матриця попарних порівнянь. Для матриць попарних порівнянь значенням вагових коефіцієнтів буде власний вектор матриці, такий що

$$\begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ & & \\ a_{n,1} & \dots & a_{n,n} \end{bmatrix} \begin{bmatrix} W_1 \\ \dots \\ W_n \end{bmatrix} = n \begin{bmatrix} W_1 \\ \dots \\ W_n \end{bmatrix}$$

або скорочено $(A - nI)W = 0$, де I – одинична матриця.

Розрахунок коефіцієнтів вектору здійснюється за наступною формулою

$$W_i = \frac{\sqrt[n]{\prod_{j=1}^{n} a_{i,j}}}{\sum_{i=1}^{n} \sqrt[n]{\prod_{j=1}^{n} a_{i,j}}}$$

Тобто - це **нормоване середнє геометричне значень попарних порівнянь по кожному рядку**. Слід зауважити, що наведена формула розрахунку власного вектору не є універсальною і спирається на властивості матриці порівнянь.

Перевірка узгодженості матриці порівнянь

На практиці внаслідок помилок експертів, обмеженості простору можливих значень переваг при попарних порівняннях можливе

$$a_{j,i} = \frac{1}{a_{i,j}} \text{ та } a_{j,k} = \frac{a_{i,k}}{a_{i,j}}$$

недотримання властивості

та навіть транзитивності відношення переваги між альтернативами. Тобто матриця попарних порівнянь стає неузгодженою. Для перевірки узгодженості матриці порівнянь із значень матриці порівнянь та значень власного вектору будується матриця розміром *nxn*, кожен елемент якої

$$\epsilon_{i,j} = a_{i,j}\frac{W_j}{W_i}, \text{ де } i, j = 1....n.$$

Далі обчислюється сума усіх елементів матриці по кожному рядку, обирається максимальне значення

$$\lambda_{max} = \sum_{j=1}^{n} \epsilon_{i,j}$$

Якщо матриця попарних порівнянь була повністю узгоджена, то $\lambda_{max}=n$, у іншому випадку $\lambda_{max}>n$. Далі обчислюється індекс узгодженості

$$\mu = \frac{|\lambda_{max} - n|}{n-1}$$

Відношення узгодженості (*consistency ratio*) обчислюється як відношення обчисленого індексу узгодженості μ до табличного індексу μ_τ

Значення табличного індексу узгодженості															
Розмір матриці (*n*)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Табличний індекс (μ_τ)	0	0	0,52	0,89	1,11	1,25	1,35	1,40	1,45	1,49	1,52	1,54	1,56	1,58	1,59

Матриця попарних порівнянь вважається узгодженою, якщо $\mu/\mu_\tau<0,1$

Якщо матриця попарних порівнянь не є узгодженою, то необхідно у матриці *E* знайти максимальне значення

$$\epsilon_{i,j} = a_{i,j}\frac{W_j}{W_i}, \text{ де } i,j = 1....n.$$

і переглянути порівняння альтернатив A_i та A_j

Розрахунок пріоритетів альтернатив

Першим кроком у розрахунку пріоритетів альтернатив є попарне порівняння альтернатив за кожним критерієм. Тобто для *K* критеріїв повинно бути сформовано *K* матриць попарного порівняння.

З матриць попарного порівняння розраховується *K* векторів ваг з *n* елементів кожен де *(k)* означає *k*-й критерій, *n* - кількість альтернатив.

$$W^{(k)} = \begin{bmatrix} W_1^{(k)} \\ \dots \\ W_n^{(k)} \end{bmatrix}$$

З векторів $W^{(k)}$ формується матриця виду

$$\begin{matrix} A_1 \\ \dots \\ A_n \end{matrix} \begin{bmatrix} W_1^{(1)} & \dots & W_1^{(k)} \\ & & \\ W_n^{(1)} & \dots & W_n^{(k)} \end{bmatrix}$$

де кожному рядку відповідає одна альтернатива. Зазначена матриця множиться на вектор вагових коефіцієнтів критеріїв.

$$\begin{bmatrix} W_1^{(1)} & \dots & W_1^{(k)} \\ & & \\ W_n^{(1)} & \dots & W_n^{(k)} \end{bmatrix} \begin{bmatrix} W^{(1)} \\ \dots \\ W^{(k)} \end{bmatrix}$$

Результатом буде вектор пріоритетів альтернатив. **Альтернатива з найбільшим значенням пріоритету має найбільшу перевагу”** [12-19] .

Лекція 11

ОЦІНКА ЯКОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ. СТАНДАРТИ ОЦІНКИ ЯКОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ

МОДЕЛІ ПРОГНОСТИЧНОГО ОЦІНЮВАННЯ ЯКОСТІ ПРОЦЕСУ ФУНКЦІОНУВАННЯ ІС

ОЦІНКА ЯКОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ

Визначення ІС (ДСТУ 2874-94): **ІС – система, яка організовує накопичення і маніпулювання інформацією щодо проблемної сфери**.

З позиції **ділового бачення** ІС – сукупність інформації, апаратно-програмних і технологічних засобів, баз та банків даних, методів процедур обробки даних, персоналу управління, які організовують процес збирання, передавання, оброблення і накопичування інформації для підготовки і прийняття управлінських рішень.

З **технічної точки зору** ІС визначається як набір взаємозалежних компонентів, що збирають, обробляють, зберігають і розподіляють інформацію, щоб підтримувати процес прийняття управлінських рішень і управління організацією в цілому.

Із **семантичної точки зору** ІС – сукупність взаємопов'язаних або взаємозалежних відомостей про стан об'єкта управління та процеси, що відбуваються в ньому, які в показниках та інших інформаційних сукупностях, зібраних та оброблених за допомогою технічних засобів за визначеною методикою та заданими алгоритмами.

"ІС не тільки відображає функціонування об'єкта управління, а й впливає на нього через органи управління. Вона є сукупністю інформаційних процесів для задоволення потреби в інформації різних рівнів прийняття рішень. Її метою є продукування інформації для використання управлінським апаратом. Відповідно вона забезпечує нагромадження, передачу, збереження, оброблення та узагальнення інформації "знизу вгору", і конкретизацію інформації "зверху вниз".

Місія ІС – виробництво інформації для забезпечення ефективного управління всіма ресурсами організації, створення інформаційного і технічного середовища для здійснення управління організацією.

Інформаційна технологія – методи оброблення інформації та організаційно-управлінські концепції її формування і споживання, а також сукупність усіх видів інформаційної техніки; єдність процедур щодо збирання, накопичення, зберігання, оброблення та передачі даних із застосуванням вибраного комплексу технічних засобів. В ІС можуть використовуватись багато таких технологій. ІС є середовищем для реалізації технології, проте інформаційна технологія ширша від ІС, вона може існувати поза нею.

Властивості вхідної інформації

Вид властивості	Характеристика
Організаційноструктурна	Відповідає структурі системи управління
Організаційноекономічна	*Надійність* – ступінь безперебійного функціонування. *Потужність* – кількість операцій за одиницю часу. *Пропускна здатність* – обсяг інформації, що проходить за одиницю часу, та обсяг результатної інформації, яка видається за одиницю часу. *Усталеність* – здатність збереження ІС у заданих режимах. *Економічність* – собівартість операцій оброблення, термін окупності. *Ефективність* – рівень комплексності, рівень автоматизації.
Функціональна	Порядок функціонування, змінюваність у зв'язку з розвитком об'єкта управління, надмірність інформації
Споживча	Порядок взаємодії зі споживачами інформації, своєчасність її доставки, взаємозв'язок і взаємозалежність елементів інформації.

Базовими складовими інформаційних технологій є:

• компоненти технічного забезпечення для збору, передачі, обробки, збереження і видачі (представлення) даних;

• системне і прикладне програмне забезпечення;

• інформаційні послуги, телекомунікації, електронна комерція і банки.

Ці складові об’єднуються і взаємодіють, впливаючи на формування ринку інформаційних продуктів, і при цьому вони самі перебувають в залежності від стану ринку. Їх основу складають такі досягнення:

• поява можливості автоматизованої обробки інформації за допомогою комп’ютерів за заданими алгоритмами;

• поява середовища для компактного зберігання і швидкого доступу до великих об’ємів інформації;

• розвиток засобів зв’язку, які забезпечують доставляння інформації практично в будь-яку точку без суттєвих обмежень у часі та відстані;

• розробка програмного забезпечення, орієнтованого на непідготовленого споживача.

Інформаційні технології класифікуються за такими ознаками:

1) за способом реалізації в інформаційних системах (традиційні, нові);

2) за ступенем охоплення завдань управління (електронна обробка даних, автоматизація функцій управління, підтримка прийняття рішень, електронний офіс, експертна підтримка);

3) за класом технологічних операцій, що реалізуються (робота з текстовим редактором, робота з електронними таблицями, робота із

системами управління базами даних, робота з графічними об’єктами, мультимедійні системи, гіпертекстові системи);

4) за типом інтерфейсу користувача (пакетні, діалогові, мережеві);

5) за способом побудови мережі (локальні, багаторівневі, розподілені);

6) за видом предметної області (бухгалтерський облік, банківська діяльність, податкова діяльність, страхова діяльність тощо).

Автоматизована ІС реалізує інформаційні технології у сфері управління шляхом спільної роботи управлінського персоналу і комплексу технічних засобів. Вона повинна забезпечувати:

• постійне спостереження за поточним станом об’єкта управління та його характеристиками;

• адаптацію, тобто пристосування до прийнятої практики бізнесу та модифікації, якщо така практика змінюється;

• підтримку професійної діяльності управлінських працівників;

• взаємодію з управлінським персоналом;

• збирання та аналіз даних для управління й автоматичного виконання програм при настанні заданого часу з формуванням звітності;

• реалізацію системи підказок і рекомендацій для користувачів;

• ефективне збереження даних у БД і можливість доступу до них будь-якого кінцевого користувача зі свого робочого місця;

• взаємодію користувачів між собою на основі безпаперової технології.

Організаційне забезпечення ІС – сукупність документів, що описують технологію функціонування ІС, методи вибору і застосування

користувачами технологічних прийомів для одержання конкретних результатів при функціонуванні ІС.

Інформаційне забезпечення ІС – сукупність інформації, інформаційних ресурсів, засобів та методів ведення інформаційної бази.

Технічне забезпечення ІС – сукупність усіх технічних засобів, використовуваних при функціонуванні комп'ютерної ІС.

Математичне забезпечення ІС – сукупність математичних методів, моделей і алгоритмів розв'язування задач, які застосовуються в ІС.

Програмне забезпечення ІС – сукупність програм на носіях даних і програмних документів, які призначені для налагодження, функціонування і перевірки працездатності ІС.

Лінгвістичне забезпечення ІС – сукупність засобів і правил для формалізації природної мови, які використовуються при спілкуванні користувачів та експлуатаційного персоналу ІС з комплексом засобів автоматизації при функціонуванні ІС.

Правове забезпечення ІС – сукупність правових норм, які регламентують правові відносини при функціонуванні ІС та юридичний статус результатів такого функціонування.

Методичне забезпечення ІС – сукупність документів, які описують технологію функціонування ІС, методи вибору і застосування користувачами технологічних прийомів для одержання результатів.

Ергономічне забезпечення ІС – сукупність засобів і методів, які створюють найсприятливіші умови праці людини в ІС, умови для взаємодії

людини і ЕОМ. Тобто це сукупність реалізованих рішень в ІС по узгодженню психологічних, психофізіологічних, антропометричних, фізіологічних характеристик і можливостей користувачів ІС з технічними характеристиками комплексу засобів автоматизації ІС та параметрами робочого середовища на робочих місцях персоналу ІС.

Оцінка рівня якості і конкурентоздатності інформаційних продуктів

Оцінка рівня якості інформаційних продуктів і послуг ІПП - нова сфера діяльності в інформаційній індустрії. Специфіка ІПП зумовлює суб'єктивний характер оцінки їх якості. Користувач, якого задовольняє важлива інформація, що надійшла своєчасно, поблажливо ставиться до її низької якості (незадовільна поліграфія, недостатня ілюстрованість

тощо). Коли ж він одержує недостовірну інформацію, то це може завдати йому економічних збитків.

Оцінка якості традиційних ІПП, зокрема оглядів, інформаційних листків тощо має експертний характер. Логічно оцінювати її залежно від рівня запитів видань з довідково-інформаційних фондів і бібліотек.

Цікавий аспект оцінки якості і конкурентноздатності електронної інформації, зокрема, електронних баз даних. Оскільки названі послуги коштують дорого, дуже відчутна економічна ефективність їх використання для кінцевих споживачів.

Багато дослідників дійшли висновку, що низька якість бази даних негативно впливає на ефективність усіх інформаційних процесів. Тому

підвищення якості продукції - одне із важливих завдань світового бізнесу в цілому.

Якість інформаційної продукції визначає багато факторів, серед яких - технічні засоби, програмне забезпечення для обробітку і пошуку інформації, система телекомунікації, документація, що її надають користувачу для роботи з базою даних, інструктивно-методичні матеріали тощо. Якість бази даних користувачі оцінюють з двох точок зору: якість інформації (даних), введених до бази даних і ефективності ІПП, що забезпечує оперативний доступ до інформації.

Інститут стандартів Великобританії розробив три різних визначення поняття якості, кожне із яких можна застосувати до банку даних:

- 1. Відносна якість, що дає змогу ранжувати подібні види продукції за їх якістю.
- 2. Абсолютна якість, яка передбачає кількісну оцінку якості продукції за певною шалою, наприклад, за середньою кількістю помилок на один запас у банку даних.
- 3. Відповідність призначенню, що допомагає оцінити можливість задовольняти різні потреби, використовуючи даний вид продукції.

Контроль за якістю бази даних передбачає комплекс методів, засобів і механізмів, потрібних для створення продукції або надання послуг, що задовольняють вимоги споживачів.

Особливу увагу приділяють якості програмного забезпечення, зокрема, її гарантуванню.

З поняттям якості тісно пов'язане поняття цілісності інформації. Останнє включає збирання, оброблення, зберігання і розповсюдження інформації такими оперативними і точними методами, що виключають випадковий або заздалегідь передбачений відступ від запропонованих вимог.

Із поняттям якості безпосередньо збігається поняття **санкціонованості, дозволеності, визначення точності інформації**. Підкреслюється важливість встановлення індикаторів якості, що дають змогу визначити якість інформації на різних рівнях (файлу, запису, окремого поля) з урахуванням її специфіки. Будь-які помилки в базі даних коштують дуже дорого. Так, пропуск трьох нулів у запису обійшлися страховій компанії Прадентел інсуеренс в 11 млн. дол.

Як свідчить аналіз, низька якість даних характерна для переважної більшості бази даних. Вона зумовлена орфографічними і синтаксичними помилками, дублюванням записів, невідповідністю індексних термінів, вживаних у системі дескрипторів.

Відповідальність за якість і надійність інформації - цн обов'язок її творців забезпечувати точність розповсюджуваних відомостей. Іноді фірми мають значні фінансові збитки через помилки в даних.

Щоб забезпечити ефективний контроль за якістю, треба застосувати різні традиційні і автоматизовані засоби:

- 1) навчання і підвищення кваліфікації користувачів, які повинні мати чіткі уявлення про елементи даних, формати, різні стандарти, процедури вводу і виводу інформації;

- 2) редагування вхідного інформаційного потоку, під час якого виявляють різні помилки;
- 3) зворотний зв'язок із споживачами інформації, що допомагає фіксувати виявлені в записах помилки.

Розроблено багато програм для виявлення і виправлення помилок на мікро-еом у інтерактивному режимі. Одним з елементів такої програми є словник. Стандартні словники призначено для орфографічного контролю, вони містять здебільшого близько 100 тис. слів. В основі різних підходів до виявлення орфографічних помилок - такі, як словникова перевірка, аналіз слів, що рідко вживаються тощо" [12-19].

Якість ІС пов'язано з дефектами, закладеними на етапі проектування і проявляються в процесі експлуатації. Будь-які властивості ІС, в тому числі і дефектологічні, можуть проявлятися лише у взаємодії із зовнішнім середовищем, що включає технічні засоби, персонал, інформаційне та програмне оточення.

Залежно від цілей дослідження і етапів життєвого циклу ІС дефектологічні властивості поділяють на дефектогенность, дефектабельность і дефектоскопічность [7].

Дефектогенность визначається впливом наступних факторів:

- • чисельністю розробників ІС, їх професійними і психофізіологічними характеристиками;
- • умовами і організацією процесу розробки ІС;

- • характеристиками інструментальних засобів і компонент ІС;
- • складністю завдань, що вирішуються ІС;
- • ступінь агресивності зовнішнього середовища (потенційною можливістю зовнішнього середовища вносити навмисні дефекти, наприклад, вплив вірусів).

Дефектабельность характеризує наявність дефектів ІС і визначається їх кількістю і місцезнаходженням. Іншими факторами, що впливають на дефектабельность є:

- • структурно-конструктивні особливості ІС;
- • інтенсивність і характеристики помилок, що призводять до дефектів.

Дефектоскопічность характеризує можливість прояву дефектів у віце відмов і збоїв в процесі налагодження, випробувань або експлуатації. На дефектоскопічность впливають:

- • кількість, типи і характер розподілу дефектів в ІС;
- • стійкість ІС до прояву дефектів;
- • характеристики засобів контролю та діагностики дефектів;
- • кваліфікація обслуговуючого персоналу.

Оцінка якості ІС є вкрай складним завданням з огляду на різноманіття інтересів користувачів. Тому неможливо запропонувати одну універсальну міру якості і доводиться використовувати ряд характеристик, що охоплюють весь спектр пропонованих вимог. Найбільш близькі до завдань оцінки якості ІС моделі якості програмного забезпечення, що

є однією з важливих складових частин ІС. В даний час використовується кілька абстрактних моделей якості програмного забезпечення, заснованих на визначеннях характеристики якості, показника якості, критерію і метрики.

Критерій може бути визначений як незалежний атрибут ІС або процесу її створення. За допомогою такого критерію може бути виміряна характеристика якості ІС на основі тієї чи іншої метрики. Сукупність декількох критеріїв визначає показник якості, що формується виходячи з вимог, що пред'являються до ІС. В даний час найбільшого поширення набула ієрархічна модель взаємозв'язку компонент якості ІС. На початку визначаються характеристики якості, в числі яких можуть бути, наприклад, загальна корисність, вихідна корисність, зручність

експлуатації. Далі формуються показники, до числа яких можуть бути віднесені: практичність, цілісність, коректність, зручність обслуговування, оцінюваність, гнучкість, адаптованість, мобільність, можливість взаємодії. Кожному показнику якості ставиться у відповідність група критеріїв. Для зазначених вище показників нижче наведені можливі критерії. Треба відзначити, що один і той же критерій може характеризувати кілька показників;

практичність - працездатність, можливість навчання, комунікативність, обсяг введення, швидкість введення-виведення; *цілісність* - регулювання доступу, контроль доступу; *ефективність* - ефективність використання пам'яті, ефективність функціонування;

коректність - трасируємість, завершеність, узгодженість;

надійність - точність, стійкість до помилок, узгодженість, простота;

зручність обслуговування - узгодженість, простоту, стисліть, інформативність, модульність;

оцінюваність - простоту, наявність вимірювальних засобів, інформативність, модульність;

гнучкість - розповсюджуваність, спільність, інформативність, модульність;

адаптованість - спільність, інформативність, модульність, апаратну незалежність, програмну незалежність;

мобільність - інформативність, модульність, апаратну незалежність, програмну незалежність;

можливість взаємодії - модульність, уніфікуючих процедур зв'язку, уніфікуючих даних.

За допомогою метрик можна дати кількісну або якісну оцінку якості ІС.

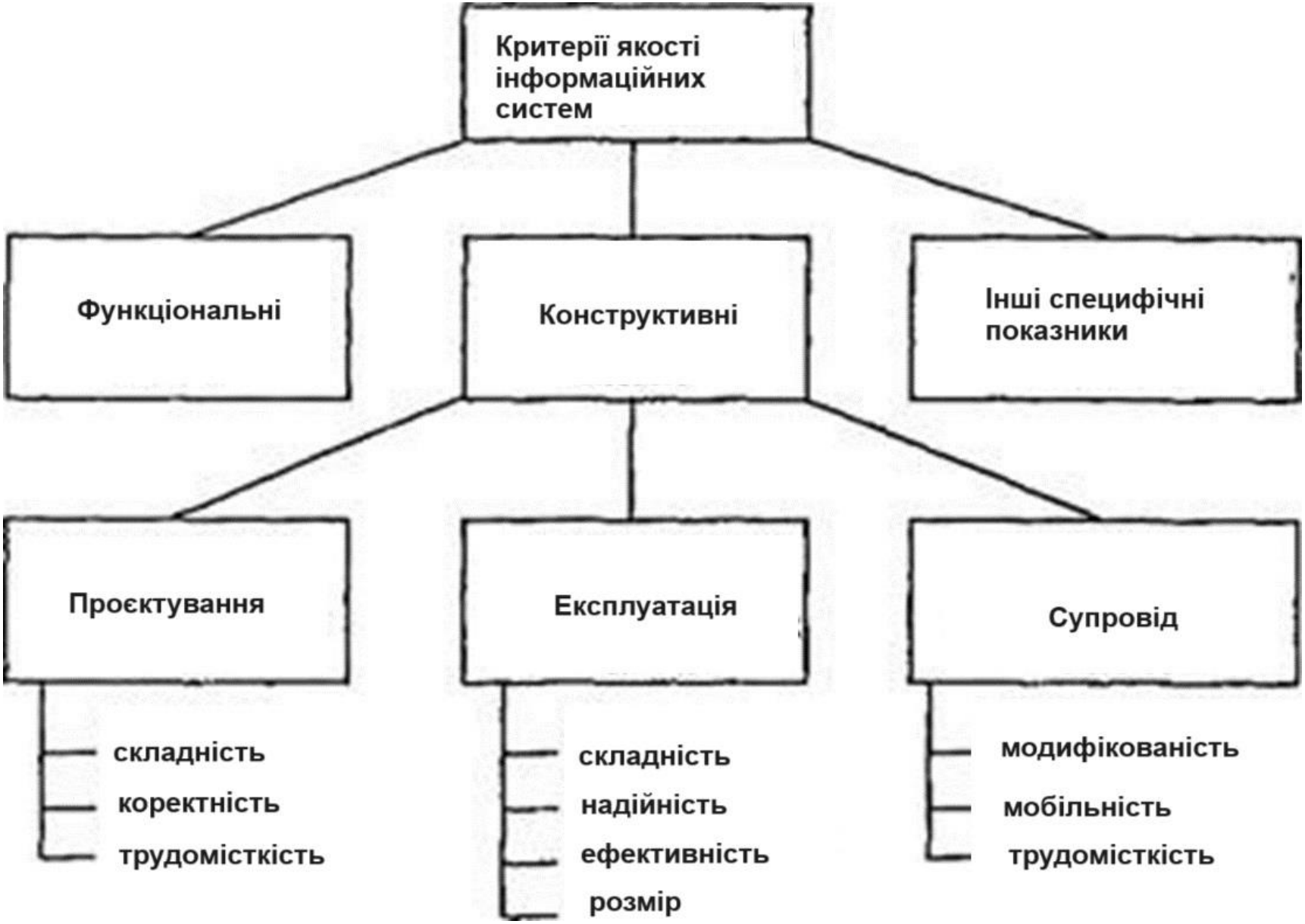
Розрізняють такі види метрик і шкал для вимірювання критеріїв.

Перший тип - метрики, які використовують интервальную шкалу, яка характеризується відносними величинами або реально вимірюваними фізичними показниками, наприклад, часом напрацювання на відмову, ймовірністю помилки, об'ємом інформації та ін.

Другий тип - метрики, яким відповідає порядкова шкала, що дозволяє ранжувати характеристики шляхом порівняння з опорними значеннями.

Третій тип - метрики, яким відповідають номінальна або категорійних шкала, яка визначає наявність даної властивості або ознаки у даного об'єкту без затишку градацій за цією ознакою. Так, наприклад, інтерфейс може бути "простим для розуміння", "помірно простим", "складним для розуміння".

Розвитком ієрархічного підходу є представлена модель класифікації критеріїв якості інформаційних систем. За допомогою функціональних критеріїв оцінюється ступінь виконання ІС основних цілей або завдань. конструктивних


Критерії якості інформаційних систем
Функціональні
Конструктивні
Інші специфічні показники
Проєктування
Експлуатація
Супровід
складність
коректність
трудомісткість
складність
надійність
ефективність
розмір
модифікованість
мобільність
трудомісткість

ні критерії призначені для оцінки компонент ІС, що не залежать від цільового призначення.

Одним із шляхів забезпечення якості ІС є сертифікація. У США Радиотехническая комісія з аеронавтики в своєму керівному документі визначає процес сертифікації в такий спосіб: "Сертифікація - процес офіційного затвердження державним повноважним органом ... виконуваної функції системи ... шляхом посвідчення, що функція ... задовольняє всім вимогам замовника, а також державним нормативним документам ". На жаль, в даний час не існує стандартів, повністю задовольняють оцінці якості ІС. У західно-європейських країнах є ряд стандартів, що визначають основи сертифікації програмних систем.

Стандарт Великобританії (BS750) описує структурні побудови програмних систем, при дотриманні яких може бути отриманий документ, який гарантує якість на державному рівні. Є міжнародний аналог зазначеного стандарту (ISO 9000) і аналог для країн членів НАТО (AQAPI). Існуюча в нашій країні система нормативно-технічних документів відносить програмне забезпечення до "продукції виробничо-технічного призначення", яка розглядається як матеріальний об'єкт. Однак програмне забезпечення є скоріше абстрактної нематеріальної сферою. Існуючі ГОСТи (наприклад, ГОСТ 28195-89. "Оцінка якості програмних засобів. Загальні положення") явно застаріли і є неповними.

Оцінка якості інформаційної системи

Якість інформаційних систем обов'язково пов'язують із цілим комплексом дефектів, які показують свої властивості у процесі проектування, а потім уже під час експлуатації. Якості інформаційної системи виявляються лише за наявності компонентів довкілля. Особливо важливими є властивості, що взаємодіють з елементами технічного характеру, персоналом, інформаційним оточенням. У тому числі важливі властивості дефектів, що визначаються такими факторами:

- кількість розробників інформаційних систем;
- умови забезпечення процесу опрацювання;
- характеристики спеціальних засобів та елементів;

- складність виконуваних з допомогою інформаційних систем процедур;
- рівень агресивності та складності зовнішнього середовища.

На дефектабельність впливають структурні та конструктивні особливості представленої інформаційної системи, характеристики помилок, їх ймовірність та інтенсивність, що призводить до виникнення дефектів.

При оцінці дефектоскопічності проявляється наявність дефектів як відмови чи збою при налагодженні, експлуатації чи під час випробувань.

Оцінка якості інформаційних систем є досить складним видом роботи, так як інтереси і ціла купа вимог користувачів можуть бути

найрізноманітнішими. Тому сформувати універсальний підхід оцінки якості інформаційних систем неможливо. Основні завдання оцінка бере із завдань, поставлених для програмного забезпечення, оскільки багато стандартів у разі є ідентичними. В даний час застосовується лише кілька моделей якості для оцінки програмного забезпечення, що базуються на метриці, показують характеристики, що базуються на комплексі показників та критеріїв якості.

Для критеріїв є визначення, яке показує, що вони є незалежною частиною інформаційної системи. За допомогою критеріїв можна виміряти характеристики якості ІС.

Критерії оцінки інформаційної системи останнім часом визначають ієрархічну модель, де компоненти пов'язані між собою. Спочатку визначають характеристики якості, серед яких можуть бути: зручність експлуатації, загальна та вихідна користь. Потім формують стандарти показників, яких відносять практичність, коректність, комфорт функціонування, можливість взаємодії, адаптації. Для кожного показника є своя окрема група критеріїв.

Оцінка якості інформаційних систем визначається з допомогою метрики. Такі методи контролю якості інформаційних систем дозволяють скласти якісні та кількісні оцінки.

Перший вид - **метрики, що застосовують інтервальну шкалу**. За допомогою такої шкали можна побачити час напрацювання вщерть, ймовірність помилок, обсяг інформації, що має.

Другий вид – **метрики із порядковою шкалою**. Завдяки такій шкалі можна визначити та вибудувати характеристики порівняно з опорними значеннями.

Третій вид представляють **метрики з номінальною та категоровану шкалою**. За допомогою цього методу можна визначити властивості та ознаки об'єкта без градації. Наприклад, інтерфейс може бути простим, складним, помірковано складним. Конструктивні критерії якості інформаційної системи використовуються для компонентів інформаційної інфраструктури і не залежать від цільового призначення.

Існує спеціальна модель класифікації критеріїв якості інформаційних систем. Саме модель класифікації критеріїв якості інформаційних під-систем в ієрархічному підході дає можливість оцінити рівень основних цілей та поставлених завдань.

Для забезпечення надійності ІВ застосовують сертифікацію. В даний час відсутні стандарти, які повністю відповідали б якісній оцінці якості. Сьогодні існують ГОСТи, але вони морально застаріли, тому багато хто використовує в оцінці системи міжнародні стандарти 1SO9000.

Стандарти оцінки якості інформаційних систем

Оцінка якості інформаційної системи за допомогою міжнародних стандартів 1SO9000 дозволяє виконувати роботи, в яких головним критерієм є можливість управління якістю продукту. Варто зазначити, що такі стандарти для інформаційної системи доповнюють ISO14000, де відображаються екологічні вимоги безпосереднього виробництва продукції.

Але важливість таких критеріїв полягає не тільки в ідеї управління якістю, а й на його контроль. Стандартно контроль інформаційних систем ґрунтується на вимірі показників за допомогою таких інструментів як методи контролю якості інформаційної системи. Це операції технологічного спрямування та вибракування виробів, що не відповідають

заявленим стандартам. Варто відзначити, що контроль інформаційного середовища є більш дієвим методом, оскільки проводить контроль процесу технологічного виробництва, що дозволяє надалі знизити ймовірність появи відбракованої продукції.

Такі методи дають можливість покращити ефективність заходів, що проводяться. Немає потреби проводити постійний контроль якості продукції. У цьому попереджається поява шлюбу, що знижує виробничі витрати. Саме такий контроль є стандартами ISO9000. Не варто думати, що вони морально застаріли, оскільки були ухвалені ще 1987 року. Такі стандарти проходять оновлення кожні п'ять років.

Дані стандарти дають можливість регламентувати та визначити всі питання, що стосуються створення, функціонування, експлуатації системи якості у промисловій сфері. Вони представлена форма вимог у структурі якості, щоб постачальник міг продемонструвати свої можливості, а учасники довкілля змогли оцінити показники якості ІС.

Американське суспільство, яке відстежує контроль якості, визначило основні цілі 1SO9000. До них можна віднести допомогу розвитку міжнародного обміну послуг та продукції. У цьому враховуються сфери наукового, інтелектуального, ділового, технічного характеру.

Стандарти оцінки ІС – це сукупність всіх властивостей, у яких обумовлена можливість її застосування задоволення потреб відповідно до її

призначенням. Саме кількісні характеристики кожного такого властивості визначають основні види показників.

Що стосується моделі якості програмного забезпечення, то на сьогоднішній день найбільшого поширення набула багаторівнева модель. Вона представлена стандартом ISO 9126. Спочатку вона виділяє шість основних характеристик якостей програмного забезпечення, а далі їх атрибути, які включають відповідні метрики для подальшої оцінки всієї ІС.

Основні види показників, за яких можлива оцінка якості інформаційного середовища, критерії ІС – це **безпека, достовірність та надійність**.

Якщо говорити про **безпеку**, то інформаційна інфраструктура контролю цю властивість визначає як здатність системи забезпечити максимальну конфіденційність, і навіть цілісність всієї наявної інформації. Тобто відбувається захист інформації від будь-якого несанкціонованого доступу.

До **достовірності** функціонування слід зарахувати такий набір властивостей, який визначає безпомилковість перетворень інформації. Достовірність функціонування визначається достовірністю вихідної інформації.

До **надійності** відносять набір властивостей, що зберігають час всіх параметрів, які необхідні виконання певних функцій у відповідних умовах і заданих режимах.

Оскільки основними показниками якості інформаційних систем є надійність, достовірність, безпека, такі моменти потрібно розглянути докладніше. Також слід зазначити, що **якість та ефективність інформаційних систем** йдуть поруч, тому окремо розглядати характеристики даних показників неможливо. Вони є модульністю і спільністю, які дуже важливі, якщо застосовуються різні методи контролю якості інформаційних систем.

Ефективність є такий набір властивостей структури інформації, який дає можливість виконувати поставлену мету в певних умовах з конкретною якістю. Для показників ефективності важливим є ступінь пристосованості системи до здійснення поставлених цілей. При цьому вони мають на увазі модульність показників для оптимального функціонування ІС. Найважливішим із показників вважатимуться економічну ефективність системи. Саме вона дає можливість показати доцільність виконаних витрат на створення та весь процес функціонування та експлуатації системи.

Такі показники якості інформаційних систем, як надійність настільки важливі для характеристики системи, що в даний час розроблена навіть спеціальна теорія надійності. Надійність можна віднести до

комплексу властивостей, оскільки цей показник якості є набором простіших за структурою властивостей. До них належать **безвідмовність, можливість проводити ремонтні роботи, довговічність.** Якщо говорити про **безвідмовність**, то контроль інформаційних систем дозволяє зрозуміти, наскільки система може зберігати свій працездатний стан протягом певного періоду або процесу напрацювання.

До **ремонтопридатності** відносять пристосованість системи до того, щоб попереджати та виявляти причини прояву відмови чи ушкодження. Також сюди входить можливість технічного обслуговування та виконання ремонтів.

Довговічність – одна з властивостей системи, завдяки якій, за наявності відповідного технічного обслуговування, вона зберігає належний стан максимально довго. Тобто має наступити такий момент, коли у системи не залишається ресурсів для подальшого виконання роботи та функціонування.

Показники якості інформаційних систем також включають **достовірність**. Це такі властивості інформації, що відображають реальні об'єкти з певною точністю. Інформаційні системи контролю якості як достовірність передбачають наявність такого комплексу показників: поодинокі, коректованості ІС та комплексні. Звичайно, для забезпечення достовірності інформації необхідний контроль.

Також до цих якостей можна віднести **портативність, зручність використання та зручність супроводу**. Завдяки яким ПЗ, а надалі і всю ІС можна аналізувати, оцінювати, переносити та змінювати у разі виникнення дефектів.

Оцінка якості інформаційної системи

Методи визначення показників якості ІС різняться:

- за способами отримання інформації про ІС – **вимірювальний, реєстраційний, органолептичний, розрахунковий**;
- за джерелами отримання інформації – **традиційний, експертний, соціологічний**.

Вимірювальний метод заснований на отриманні інформації про властивості та характеристики ІС з використанням інструментальних засобів. Наприклад, з використанням цього методу визначається обсяг ІС - число рядків вихідного тексту програм і число рядків - коментарів, число операторів та операндів, число виконаних операторів, число

гілок у програмі, число точок входу (виходу), час виконання гілки програми, час реакції та інші показники.

Реєстраційний метод заснований на отриманні інформації під час випробувань або функціонування ІС, коли реєструються та підраховуються певні події, наприклад, час та кількість збоїв та відмов, час передачі управління іншим модулям, час початку та закінчення роботи.

Органолептичний метод заснований на використанні інформації, одержуваної в результаті аналізу сприйняття органів чуття (зір, слуху), і застосовується визначення таких показників як зручність застосування, ефективність тощо.

Розрахунковий метод заснований на використанні теоретичних та емпіричних залежностей (на ранніх етапах розробки), статистичних даних, що накопичуються при випробуваннях, експлуатації та супроводі ІС. За допомогою розрахункового методу визначаються тривалість та точність обчислень, час реакції, необхідні ресурси.

Визначення значень показників якості ІС **експертним методом** здійснюється групою експертів-фахівців, компетентних у вирішенні даного завдання, на базі їхнього досвіду та інтуїції. Експертний метод застосовується у випадках, коли завдання не може бути вирішене жодним іншим з існуючих способів або інші способи значно трудомісткішими. Експертний метод рекомендується застосовувати щодо показників

наочності, повноти і доступності програмної документації, легкості освоєння, структурності.

Соціологічні методи засновані на обробці спеціальних анкет-запитань.

1. Оцінка якості ІС проводиться на фазах життєвого циклу та включає вибір номенклатури показників, їх оцінку та зіставлення значень показників, отриманих у результаті порівняння з базовими значеннями.

2. Показники якості об'єднані у систему із чотирьох рівнів. Кожен вищестоящий рівень містить як складові показники нижчих рівнів. Допускається вводити додаткові показники кожному рівні.

2.1. Для забезпечення можливості отримання інтегральної оцінки за групами показників якості використовують фактори якості (1-й рівень): **надійність ІС, супровідність, зручність застосування, ефективність, універсальність (гнучкість) та коректність**.

2.2. Кожному фактору якості відповідає певний набір критеріїв якості (комплексні показники - 2-й рівень): **стійкість функціонування, працездатність, структурність, простота конструкції, наочність, повторюваність, легкість освоєння, доступність експлуатаційних програмних документів, зручність експлуатації та обслуговування, рівень автоматизації, тимчасова ефективність , ресурсомісткість, гнучкість, мобільність, модифікованість, повнота реалізації, узгодженість, логічна коректність**

2.3. Критерії якості визначають однією чи кількома метриками (3-й рівень). Якщо критерій якості визначається однією метрикою, рівень метрики опускається.

2.4. Метрики складаються з оціночних елементів (поодиноких показників - 4-й рівень), що визначають задану в метриці якість. Кількість оцінювальних елементів, що входять до метрики, не обмежена.

2.5. Вибір оціночних елементів у метриці залежить від функціонального призначення оцінного елемента і з урахуванням даних, отриманих під час проведення випробувань різних видів, і навіть за результатами експлуатації ИС.

2.6. Для накопичення інформації про оцінні елементи формується довідник оціночних елементів на основі раніше отриманих даних про якість аналогічних ІС.

3. Оцінка якості ІС проводиться у певній послідовності.

3.1. На фазі аналізу проводиться вибір показників та його базових значень.

3.2. Для показників якості всіх рівнях (чинники, критерії, метрики, оціночні елементи) приймається єдина шкала оцінки від 0 до 1.

3.3. Показники якості на кожному вищому рівні (крім рівня оціночних елементів) визначаються показниками якості нижчестоящого рівня.

3.4. У процесі оцінки якості ІВ кожному рівні (крім рівня оціночних елементів) проводяться обчислення показників якості ІВ, тобто. визначення кількісних значень абсолютних показників

3.5. Кожен показник якості 2-го та 3-го рівнів (критерій та метрика) характеризується двома числовими параметрами - кількісним значенням та ваговими коефіцієнтами.

3.6. Сума вагових коефіцієнтів показників рівня є постійною величиною. Сума вагових коефіцієнтів приймається рівно й 1.

3.7. Загальна оцінка якості ІС загалом формується експертами з набору отриманих значень оцінок чинників якості.

3.8. Для оцінки якості ІС різного призначення методом експертного опитування складається таблиця значень базових показників якості ІС.

4. Якість ІС визначається шляхом порівняння отриманих розрахункових значень показників з відповідними базовими значеннями показників існуючого аналога або розрахункового ІС, який приймається за еталонний зразок.

4.1. Значення базових показників ІС мають відповідати значенням показників, що відображають сучасний рівень якості та прогнозований світовий рівень.

4.2. Як аналоги вибираються реально існуючі ІС того ж функціонального призначення, що і порівнюване, з такими ж основними параметрами, подібної структури та застосовувані в умовах експлуатації.

Отже, з характеристик, відповідальних якість інформаційної системи, визначимо коефіцієнт якості інформаційної системи, розробленої під час виконання дипломного проекту. Виходячи з того, що «ідеальна система» має 26 балів, визначимо ступінь якості за такими критеріями:

Оцінка якості:

Висока, якщо кіт 0,7 до 1;

Середня при кіт 0,5 до 0,7;

Низька при менш ніж 0,5.

Характеристика	Проміжна характеристика	Детальна характеристика	Наявність(1) Відсутність (0)
1. Функціональні можливості	1.1. Функціональна придатність	1.1.1- відповідність програмних засобів цілям їх застосування	1
		1.1.2- відповідність складу та змісту вихідної інформації вимогам користувачів	1
		1.1.3- відповідність вихідної інформації, що використовується в організації, вимогам ІС	1
2. Надійність и безпека	2.1 Захищеність	2.1.1- відповідність ІС вимогам захисту від навмисних загроз безпеці	0
		2.1.2- забезпечення ефективності оперативних методів захисту та відновлення при реалізації загроз	1
	2.2. Стійкість функціонування	2.2.1- наявність засобів відновлення при помилці на вході	0
		2.2.2- наявність засобів відновлення при збоях обладнання	0

		2.2.3- ймовірність працездатного функціонування протягом місяця	1
3. Практичність та зручність застосування	3.1. Легкість освоєння	3.1.1- можливість освоєння ІС з документації	1
		3.1.2- можливість освоєння ІС на контрольному прикладі	1
	3.2. Доступність експлуатаційних документів	3.2.1- повнота та зрозумілість документації для освоєння	1
		3.2.2- достатність документів для запуску ІС в експлуатацію	1
	3.3 Простота використання	3.3.1- комфортність експлуатації	1
		3.3.2- простота експлуатації ІВ	1
4. Ефективність	4.1. Часова ефективність	4.1.1- задоволення часом виконання програм та часом видачі відповідей на запити	1
		4.1.2- задоволення часом підготовки вхідних даних	1
	4.2. Економічна ефективність	4.2.1- задоволення витратами на захист даних	1
		4.2.2- задоволення співвідношенням загальних витрат на експлуатацію ІВ та одержуваним прибутком	1

		4.2.3- задоволення співвідношенням витрат на захист даних та одержуваним прибутком	1
5. Супроводжуваність	5.1. Внесення поточних змін до ІС у процесі експлуатації	5.1.1- наявність документів, що містять терміни внесення поточних змін до ІС	0
		5.1.2- повнота документів, що відображають порядок внесення поточних змін до ІС	0
		5.1.3- наявність системи контролю за внесенням поточних змін до ІС	0
	5.2. Навчання персоналу в період впровадження та після внесення змін до ІС	5.2.1- наявність системи навчання персоналу у процесі впровадження ІВ	1
		5.2.2- наявність тестів для контролю рівня знань учнів	0
		5.2.3- наявність системи навчання після внесення змін до ІС	1
		5.2.4- наявність вимог до знань персоналу допущеного до експлуатації ІВ	1

k= $\frac{19}{26} = 0{,}731$, оцінка якості «висока»” [12-19].

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1 Стандарт вищої освіти України: перший (бакалаврський) рівень, галузь знань 12 – Інформаційні технології, спеціальність 126 – Інформаційні системи та технології : Затверджено Наказом Міністерства освіти і науки України 12.12.2018 р. 2018. № 1380. 17 с.

2 Буката Л. М., Шаповаленко В. А., Трофименко О. Г. Чисельні методи та моделювання на ОЕМ : метод. посіб. Одеса : Державна адміністрація зв'язку Одеська Національна Академія зв'язку ім. О.С.Попова, 2010. 72 с.

3 Табунщик Г. В., Каплієнко Т. І., Петрова О. А. Проектування та моделювання програмного забезпечення сучасних інформаційних систем : навч. посіб. Запоріжжя, 2016. 259 с.

4 Петрик М. Р., Петрик О. Ю. Моделювання програмного забезпечення : наук.-метод. посіб. Тернопіль : ТНТУ, 2015. 200 с.

5 Schmidt D. C., Stal M., Rohnert Hans., Buschmann F. Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects. USA : JohnWilye&Sons, 2013. 450 p.

6 Антоненко В. М., Мамченко С. Д., Рогушина Ю. В. Сучасні інформаційні системи і технології: управління знаннями : навч. посіб. Ірпінь : Нац. університет ДПС України, 2016. 212 с.

7 Bilash Kanti Bala, Fatimah Mohamed Arshad, Kusairi Mohd Noh. System Dynamics: Modelling and Simulation. Springer; Softcover reprint of the original 1st ed. 2017 edition, 2017. 291 p.

8 Michael Pidd. Systems Modelling: Theory and Practice. Department of Management Science the Management School Lancaster University, 2006. 236 p.

9 Козубцов І. М. Методологія наукових досліджень: Конспект лекцій. Луцьк: ЛНТУ, 2022. 242 с.

10 Зацерковний В. І., Тішаєв І. В., Демидов В. К. Методологія наукових досліджень : навч. посіб. Ніжин : НДУ ім.М.Гоголя, 2017. 236 с.

11 Кудерметов Р. К., Луценко Н. В., Польська О. В. Методичні вказівки до виконання лабораторних робіт з дисципліни «Моделювання систем». Запоріжжя : ЗНТУ, 2019. 58 с.

12 Антонюк А. О. Опорний конспект з курсу Моделювання систем, НУДПСУ. URL: https://ua.kursoviks.com.ua/metodychni_vkazivky/article_post/648.
13 Задачин В. М., Конюшенко І. Г. Моделювання систем: конспект лекцій. Харків : Вид. ХНЕУ, 2010. 268 с.
14 Самсонкін В. М., Ніколаєнко І. В., Булгакова Ю. В. та ін. Інжиніринг криз та ризиків транспортних послуг / за ред. В. М. Самсонкіна та І. В. Ніколаєнко. Київ : Талком, 2021. 312 с.
15 Соболь О. М., Писклакова О. О. Лекція № 1 з навчальної дисципліни «Моделювання у сфері цивільного захисту». URL: http://univer.nuczu.edu.ua/tmp_metod/4089/Kurs_lekcij.pdf.
16 Бойко Ю. П. Опорний конспект лекцій з дисципліни «Моделювання в цифровій економіці». Київ : НАУ, 2019. 45 с.
17 Квєтний Р. Н., Богач І. В., Бойко О. Р., Софина О. Ю., Шушура О. М. Комп'ютерне моделювання систем та процесів. Методи обчислень. Частина 1 : навч. посіб. / за заг. ред. Р. Н. Квєтного. Вінниця : ВНТУ, 2012. 193 с.
18 Павленко П. М., Філоненко С. Ф., Чередніков О. М., Трейтяк В. В. М34 Математичне моделювання систем і процесів : навч. посіб. К. : НАУ, 2017. 392 с.
19 Моделі і методи проектування інформаційних систем. СумДУ, 2016. URL: https://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20160217112601/165292/index.html#p3.
20 Литвинов А. Л. Л64 Теорія систем масового обслуговування : навч. посіб. Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. Харків : ХНУМГ ім. О. М. Бекетова, 2018. 141 с.

Олевський Віктор Ісаакович
Олевська Юлія Борисівна
Соколова Наталя Олегівна

Конспект лекцій з дисципліни «Моделювання інформаційних систем»
для здобувачів ступеня бакалавра
спеціальності 126 Інформаційні системи та технології

За редакцією авторів

Електронний ресурс

Національний технічний університет «Дніпровська політехніка»
49005, м. Дніпро, просп. Д. Яворницького, 19.