**МІНІСТРЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКИЙ ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»**

НАВЧАЛЬНИЙ НАУКОВИЙ ІНСТИТУТ

ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І РОБОТОТЕХНІКИ

КАФЕДРА КОМП’ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

**Лабораторна робота 4**

з дисципліни «Програмне забезпечення комп'ютерних систем»

за темою «Процес розробки та тестування програмного забезпечення»

**Виконав:**

Студент 501-ТК

Коренівський Олег Григорович

**Перевірив:**

професор

Фесенко Тетяна Григорівна

**Полтава**

**2024Вступна частина**

У даній лабораторній роботі описується процес моделювання етапів розробки та тестування програмного забезпечення з використанням методології IDEF0 (SADT). Метою роботи є побудова функціональної моделі, що включає опис основних етапів розробки ПЗ, їх взаємозв'язків, а також визначення вхідних та вихідних даних, механізмів і обмежень.

# **Мета роботи**

* Створити функціональну модель розробки та тестування програмного забезпечення.
* Побудувати контекстну діаграму (A-0) і декомпозицію (A0, A1 тощо).
* Визначити вхідні дані, контрольні елементи, вихідні результати та механізми.
* Оформити звіт з описом моделі та її елементів.

# **Завдання до роботи**

1. Створити контекстну діаграму A-0.
2. Розробити декомпозицію контекстної діаграми (A0).
3. Визначити та описати вхідні дані, контрольні елементи, вихідні результати та механізми для кожного етапу розробки.
4. Побудувати дерево вузлів.
5. Оформити звіт про виконану роботу

# **1. Опис моделі**

## **1.1. Контекстна діаграма A-0**

Контекстна діаграма відображає весь процес розробки та тестування програмного забезпечення як єдиний блок. Вхідними даними є вимоги до програмного забезпечення, які обробляються у процесі розробки. Контрольні елементи включають технічне завдання та стандарти. Ресурси для виконання процесу представлені інструментами розробки та командою розробників. Вихідним результатом є протестоване програмне забезпечення.

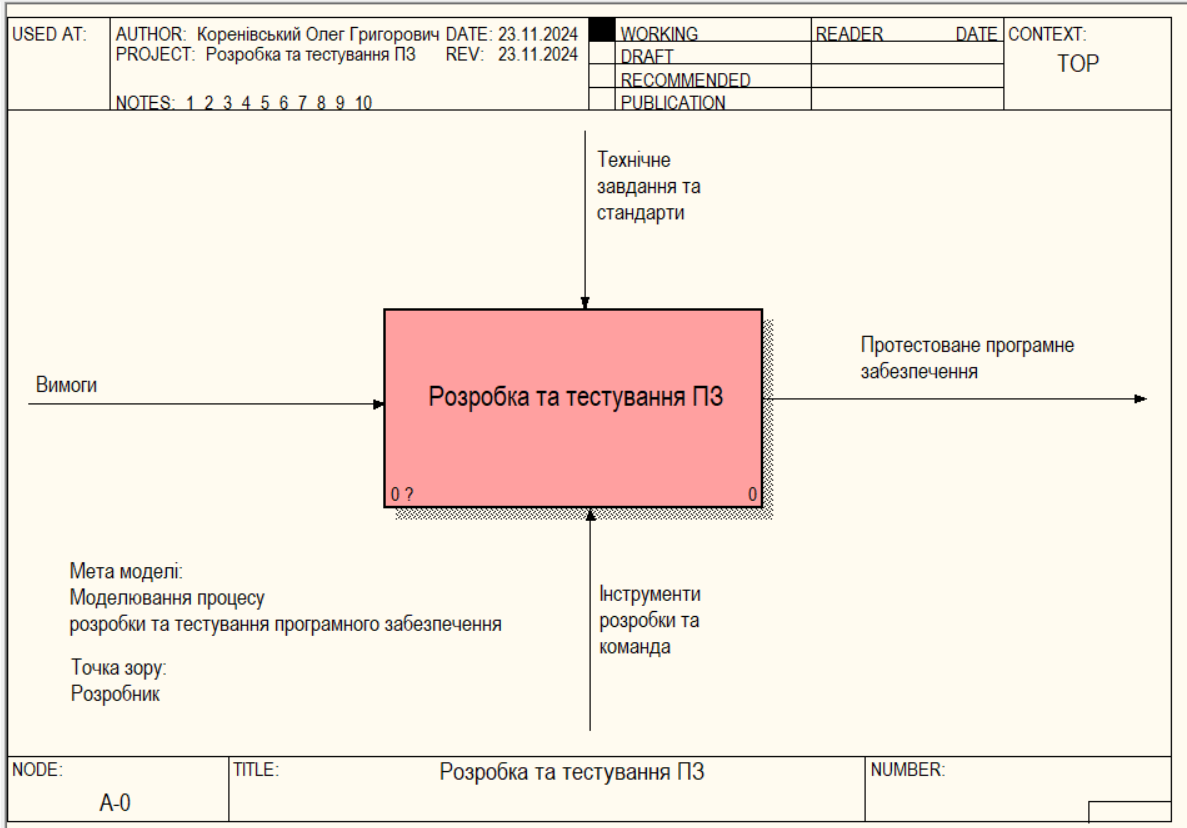


Рисунок 1 – Контекстна діаграма процесу розробки та тестування програмного забезпечення (Зроблено в програмі Process Modeler)

Таблиця 1.1 – Глосарій термінів та ключових слів процесу «Розробка та тестування програмного забезпечення»

|  |  |  |
| --- | --- | --- |
| **Ім’я стрілки (Arrow Name)** | **Визначення стрілки (Arrow Definition)** | **Тип стрілки (Arrow Type)** |
| Вимоги | Запит на створення або оновлення програмного забезпечення. | Вхідний потік даних (Input) |
| Технічне завдання та стандарти | Документація, яка регламентує процес розробки та тестування ПЗ. | Керуючий потік (Control) |
| Протестоване ПЗ | Програмний продукт, який пройшов перевірку та готовий до впровадження. | Вихідний потік даних (Output) |
| Інструменти розробки та команда | Ресурси, включаючи програмні засоби й спеціалістів, які виконують завдання. | Механізм (Mechanism) |

## **1.2. Декомпозиція A0**

Процес розробки та тестування програмного забезпечення розділений на такі підпроцеси:

1. **Аналіз вимог:** Оцінка вимог до програмного забезпечення, визначення технічних задач.
2. **Розробка коду:** Створення коду на основі технічного завдання.
3. **Тестування:** Перевірка функціональності та якості коду.
4. **Реліз:** Підготовка програмного забезпечення до впровадження та випуску.

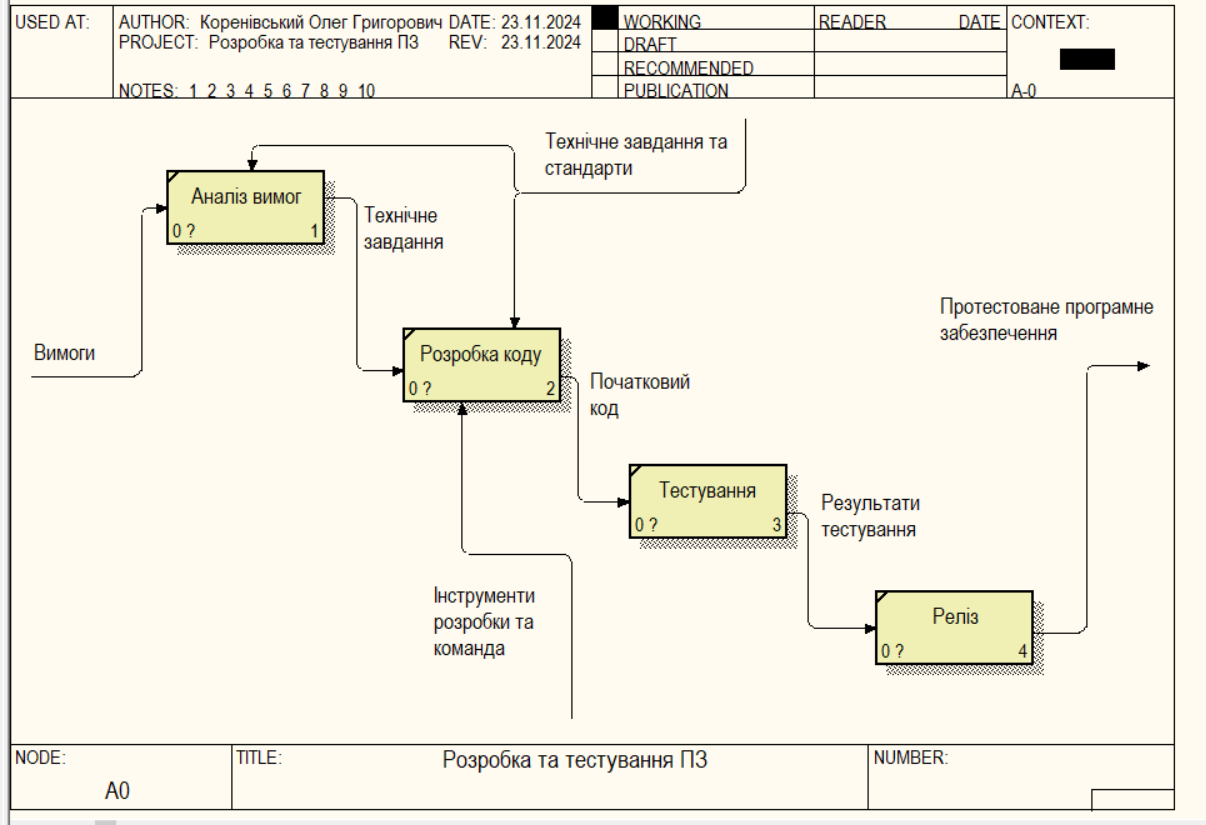


Рисунок 2 – Декомпозиція контекстна діаграма процесу «Розробки та тестування програмного забезпечення»

Глосарій етапів робіт діаграм декомпозицій процесів ремонту розробки та тестування програмного забезпечення IDEF0 (таблиця 1.2).

Таблиця 1.2 – Глосарій етапів робіт діаграм декомпозицій процесів

у стандарті IDEF0

|  |  |  |
| --- | --- | --- |
| ***Ім’я роботи***  ***(Activity Name)*** | ***Визначення (Definition)*** | ***Призначення елементу*** |
| Аналіз вимог | Визначення та уточнення вимог замовника, їх обробка та створення технічного завдання. | Процес |
| Розробка коду | Створення програмного коду на основі технічного завдання, його інтеграція та тестування. | Процес |
| Тестування | Перевірка працездатності системи та відповідності її вимогам. | Процес |
| Реліз | Завершення розробки ПЗ, його впровадження та підготовка до експлуатації. | Процес |
| Вимоги | Початкові дані, що визначають, яким має бути програмне забезпечення. | Вхідний потік даних (Input) |
| Технічне завдання та стандарти | Документовані правила та вимоги, які керують процесом розробки. | Керуючий потік (Control) |
| Технічне завдання | Результат процесу "Аналіз вимог", який стає основою для розробки коду. | Потік даних між процесами |
| Початковий код | Код, створений під час розробки, який передається на тестування. | Потік даних між процесами |
| Результати тестування | Звіт про якість програмного забезпечення після перевірки, переданий для релізу. | Потік даних між процесами |
| Протестоване ПЗ | Програмний продукт, готовий до впровадження після завершення тестування. | Вихідний потік даних (Output) |
| Інструменти розробки та команда | Фахівці та програмно-апаратне забезпечення, що використовуються для створення та впровадження ПЗ. | Механізм (Mechanism) |

## **1.3.** **Діаграма дерева вузлів**

Діаграма дерева вузлів відображає ієрархію процесу розробки та тестування програмного забезпечення:

* A-0: Розробка та тестування ПЗ
* A0: Аналіз вимог
* A1: Розробка коду
* A2: Тестування
* A3: Реліз

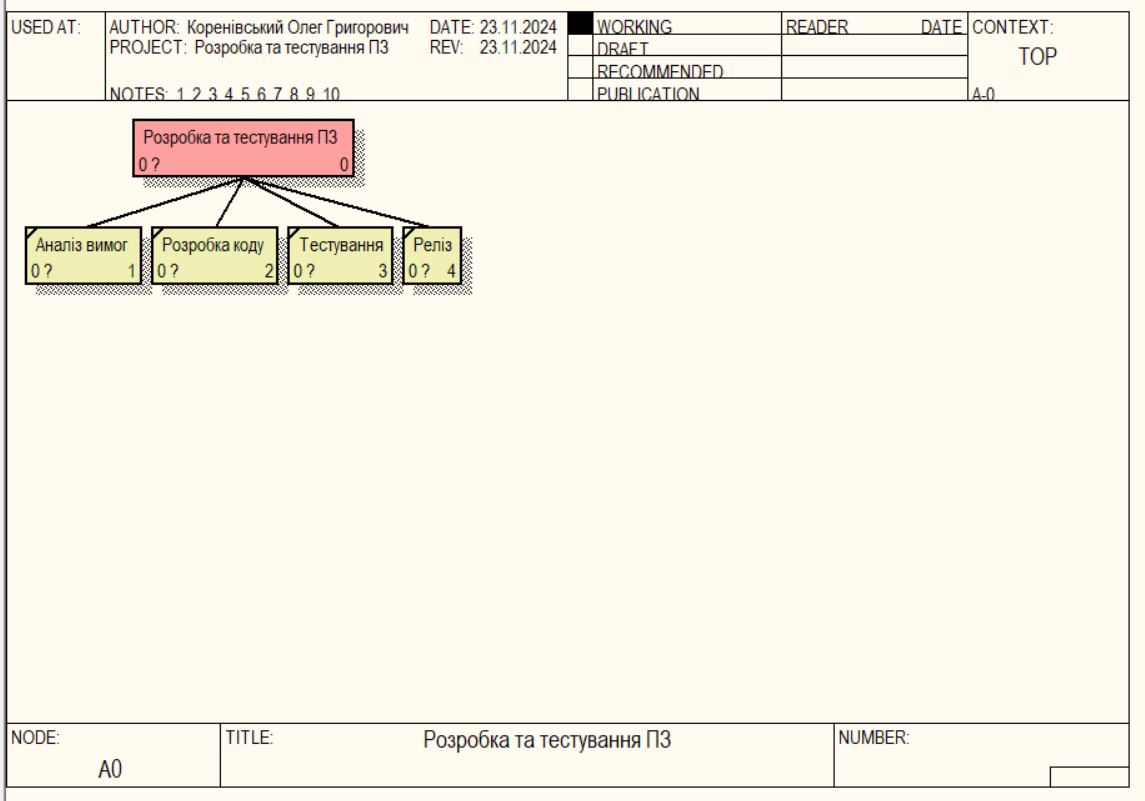


Рисунок 3 – Діаграма дерева вузлів процесу «Розробки та тестування програмного забезпечення»

# **Властивості моделі**

**Основні властивості моделі та її контекст**

Властивості розділу "General" відображають основну інформацію про модель, зокрема її назву, автора, назву проєкту та часову рамку (AS-IS). Ці дані формують основу для ідентифікації моделі та її контексту.

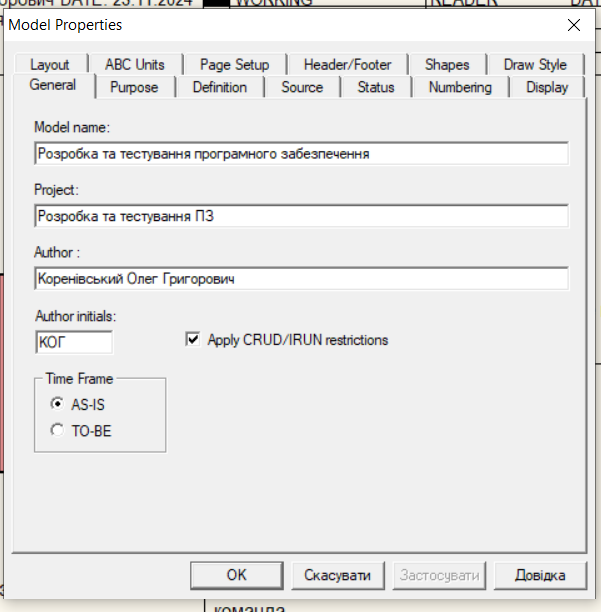


Рисунок 8 – Загальні властивості

**Мета моделі та точка зору на процес розробки**

У властивостях "Purpose" визначається мета моделі, яка полягає у формалізації процесу розробки та тестування програмного забезпечення. Також вказано точку зору (Viewpoint) — вона відображає підхід розробників, що акцентує увагу на технічних аспектах реалізації..

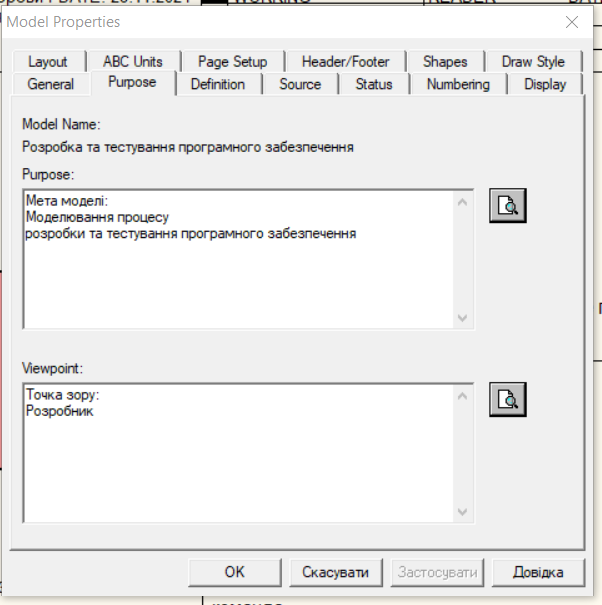


Рисунок 9 – Властивості «Purpose»

**Опис структури моделі та взаємозв’язків між процесами**

На рисунку наведено властивості "Definition", які описують загальну структуру моделі. Тут вказано ключові процеси, такі як аналіз вимог, розробка коду, тестування та реліз. Також зазначено взаємозв’язки між процесами через потоки даних, механізми й обмеження. Цей розділ допомагає зрозуміти основну логіку моделі та її відповідність цілям проєкту. Властивості спрямовані на деталізацію кожного етапу для забезпечення прозорості моделювання.

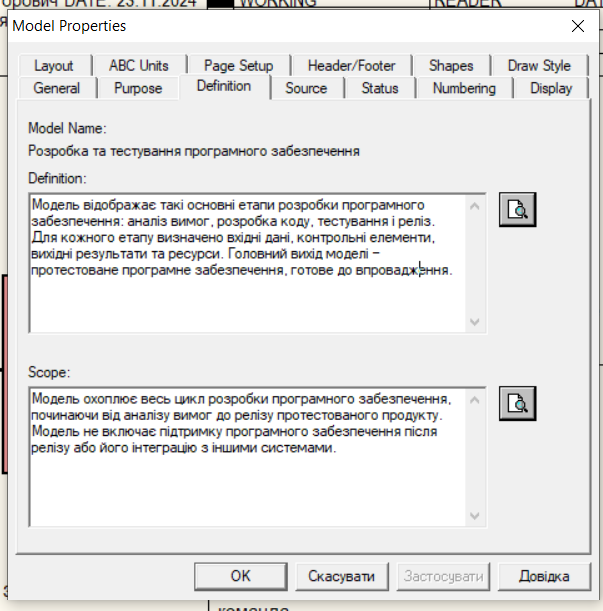


Рисунок 10 – Властивості «Definition»

**Опис одиниць роботи та їх ролі в процесі розробки**

На рисунку зображено властивості "Unit of Work", які деталізують основні одиниці роботи в моделі. Включено такі етапи: аналіз вимог, розробка коду, тестування та реліз. Для кожного етапу вказано їхній взаємозв’язок через потоки даних та сховища, що забезпечує послідовність ієрархії процесів. Властивості визначають, як кожна одиниця роботи інтегрується у загальну структуру моделі, формуючи чіткий і логічний робочий процес. Це важливо для візуалізації залежностей між процесами.

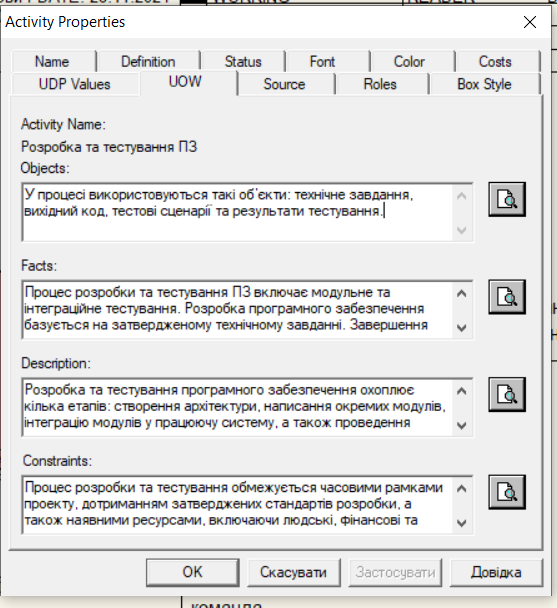


Рисунок 11 – Властивості «UOW»

# **Відповіді на контрольні питання**

**1. Що визначає контекстна діаграма?**

Контекстна діаграма визначає загальну мету системи, основні функції, які вона виконує, та її взаємодію із зовнішнім середовищем. Вона показує вхідні дані, контрольні елементи, механізми та виходи системи. Контекстна діаграма створюється як найзагальніша частина моделі.

**2. Які основні поняття використовуються при створенні функціональної діаграми IDEF0?**

Основними поняттями при створенні функціональної діаграми IDEF0 є:

* Блок (Activity): Зображає окрему функцію або процес.
* Стрілки ICOM (Input, Control, Output, Mechanism): Позначають входи, виходи, обмеження і ресурси, пов’язані з процесом.

**3. З якою метою будуються діаграми для експозиції (FEO)? Що означає поява "тунелів" на діаграмі?**

Діаграми для експозиції використовуються для демонстрації зовнішніх аспектів моделі або системи. "Тунелі" на діаграмі позначають стрілки, які виходять за межі діаграми, але зберігають свій зв’язок з іншими частинами моделі .

**4. Що показує діаграма дерева вузлів?**

Діаграма дерева вузлів відображає ієрархічну структуру моделі, включаючи взаємозв'язки між контекстною діаграмою (A-0) та всіма рівнями декомпозиції (A0, A1 тощо). Вона допомагає зрозуміти, як процес розподіляється на підпроцеси​(Лабораторна 4) .

**5. Які стрілки називаються стрілками механізму (Mechanism)?**

Стрілки механізму (Mechanism) позначають ресурси або засоби, необхідні для виконання процесу. На діаграмі IDEF0 ці стрілки зображуються вхідними знизу до блоку процесу .

**6. Що входить в визначення контексту моделі?**

Контекст моделі включає:

* Мету моделювання.
* Межі моделі (що входить у систему і що залишається поза нею).
* Точку зору, яка визначає, з якої позиції розглядається система .

**7. Які основні поняття використовуються при створенні функціональної діаграми IDEF0?**

Основними поняттями є:

* Блоки (Activity), що відображають функції.
* Стрілки ICOM для позначення входів, управління, виходів і механізмів.
* Декомпозиція, яка деталізує процеси на рівні підпроцесів.

# **Висновок**

У ході виконання лабораторної роботи було побудовано функціональну модель процесу розробки та тестування програмного забезпечення за методологією IDEF0. Модель відобразила основні етапи цього процесу: аналіз вимог, розробка коду, тестування та реліз.

Було створено контекстну діаграму, яка визначає взаємозв’язок системи із зовнішнім середовищем, і виконано декомпозицію процесу, що дозволило детально описати кожен етап. Побудована діаграма дерева вузлів допомогла візуалізувати ієрархію моделі та забезпечити її структурованість.

У результаті виконання роботи вдалося досягти поставленої мети – зрозуміти процеси, що лежать в основі розробки програмного забезпечення, визначити взаємозв’язки між етапами, а також вхідні, вихідні дані, ресурси та обмеження. Модель є зручною для подальшого використання в управлінні розробкою ПЗ і може бути базою для оптимізації процесів.Ця робота сприяла кращому розумінню методології IDEF0, яка дозволяє формалізувати складні процеси, роблячи їх наочними, логічними та структурованими. Отримані знання можна застосовувати як у навчанні, так і на практиці в розробці інформаційних систем.