УДК 004.05

**О. О. Ліневич, О. І. Лісовиченко**

# МАКРОПРОЦЕС ПРИСКОРЕННЯ РОЗРОБКИ ПРОЕКТІВ

*Анотація:* Починаючи з 1980-х років і до сьогодні, зберігається тенденція розробки надпланових і надбюджетних проектів програмного забезпечення. У 2023 році, згідно з річним звітом BCG, кількість таких проектів досягла захмарних 84% на рік. Висока тенденція зберігається, незважаючи на використання найновітніших процесів та методів розробки. Спершу в статті розглядається існуючий макропроцес розробки, його підходи розробки і аналіз. Далі описується новий макропроцес, який підвищує ефективність, гнучкість і надійність програмних систем завдяки новому методу, що поєднує систематизацію доменних знань, розрахунок і вбудову шляхів еволюції систем, а також інтеграцію систем з іншими системами на основі теорії систем і математики.

*Ключові слова:* система, фактори якості, гнучкість, ефективність, аналіз, дизайн, реалізація, рівень абстракції, специфікація, інтеграція, дерево

## Формулювання Проблеми

За звітом the Standish Group 2004 71% software проєктів перевищили запланований бюджет. У 2015 році тендція збереглась і за звітом Chaos Report відсоток майже не змінився і став 71% [1]. Проте у 2023 році за звітом Project Failure Stat цей показник виріс до 84% і планований(original) бюджет був перевищений на 189% [2]. У більше половини ситуацій бюджет було перевищено, тому що ПЗ розроблювалось довше, ніж було заплановано і на оплату позапланованої праці пішли додаткові витрати. За дослідженнями причинами довгої розробки було визначено: неефективну оцінку задач і складність рішення задач[4], [5]. Ці блокери активно аналізувались починаючи з 1990х по сьогодні[6]. Протягом цього періоду було створено десятки методів аналізу, дизайну, архітектурних стилей і патернів, проте це не пришвидшило розробку, а подовжило.

Описати фреймворк аналізу підходів. Проаналізувати існуючі підходи опису доменних задач і створення рішень, використовуючи фреймворк. Оцінити ці методи за факторами якості. Визначити, як підвищити якість підходів. Описати новий метод, який підвищить якість підходів, що підвищить швидкість розробки ПЗ і зменшить бюджет [17].

## Фреймворк Аналіза

Існуючі підходи розробки розділяються на підходи опису доменної і кодової систем - домену і коду ПЗ[7]. Ці системи створюються під час розробки ПЗ, а саме – макропроцесу розробки[8], який показаний на рис. 1.


ISSN 1560-8956


Рисунок 1. Існуючий Макропроцес Розробки

В статті аналізуються етапи аналізу, дизайну і реалізації, так як на цих етапах виконується створення складних рішень і в результаті цих етапів створюються системи, які складно оцінити для вбудови нових вимог.

Оцінка підходів етапів виконується на основі аналізу CRM/ERP систем. Системи оцінюються за такими критеріями:

- точність відображення домену у системі поза іншими системами;
- точність відображення домену системи в інтегрованій системі;
- повнота відображення домену у системі;
- ефективність вирішення поточної проблеми системою;
- ефективність вирішення еволюційних проблем системою;
- гнучкість системи в інтегрованій системі;
- надійність системи.

**Рішення Проблеми**

Було досліджено:

- 5 підходів до аналізу;
- 7 підходів до дизайну системи;
- 3 підходи патернів до дизайну(реалізації) компонентів.

Підходи було застосовано до 120 середніх і великих задачах і 24 системах.

Домени систем: 7 банківських, 1 нерухомість, 2 кримінальне дослідження, 1 судова, 4 електронна комерція, 3 освітні, 4 бухгалтерські, 2 медичні.

Підходи застосовувались у комбінаціях взалежності від задач. Оцінка методів - відображає тенденцію роботи підходів.

Підходи етапу аналіза

Досліджені підходи аналізу: варіантів використання[9], поведінковий[10], доменний[11], структурний аналіз і метод Аббота[12]. При застосуванні всіх 5 методів аналізу у комбінації було знайдено(табл.1) 65% точних назв даних і процесів. Не вистачало точних назв абстракцій даних і процесів, тобто повнота була 66%, а ще 34% потрібно було отримувати додатковим аналізом і розмовами з доменними експертками. Ефективність вирішення поточної проблеми була 58%.

 ISSN 1560-8956

*Таблиця 1.*

**Якість Підходів Аналізу**

| Критерій Якості | Значення |
|---|---|
| 1. Точність відображення домену поза системами | 54% |
| 2. Точність відображення домену в інтегрованій системі | 0% |
| 3. Повнота відображення домену | 50% |
| 4. Ефективність вирішення поточної проблеми | 40% |
| 5. Ефективність вирішення еволюційних проблем | 0% |
| 6. Гнучкість | 0% |
| 7. Надійність | 0% |

**Підходи етапу дизайну систем**

Досліджені підходи дизайну: OOD, SOA, DDD, BDD, TDD, RDD, API-Design First Method [13].

Підходи дизайну застосовувались поверх застосованих до систем комбінованих підходів аналізу і підбирались за схожими доменними проблемами. Підходи ODD, DDD, BDD та TDD підвищували точність відображення поза системами на 10-15%. Підходи SOA, RDD, API-Design робили систему гнучкою до 40% за рахунок ділення доменного функціоналу на часто використовувані сервіси. Повна оцінка критеріїв якості подана нижче (табл.2).

*Таблиця 2.*

**Якість Підходів Дизайну Архітектури**

| Критерій Якості | Значення |
|---|---|
| 1. Точність відображення домену поза системами | 56% |
| 2. Точність відображення домену в інтегрованій системі | 0% |
| 3. Повнота відображення домену | 55% |
| 4. Ефективність вирішення поточної проблеми | 60% |
| 5. Ефективність вирішення еволюційних проблем | 0% |
| 6. Гнучкість | 45% |
| 7. Надійність | 0% |

**Підходи етапу реалізації**

Досліджені підходи реалізації: патерни дизайна[14], інтеграційні[15] і мікро-сервісні[16]. Патерни застосовувались поверх застосованих до систем методів дизайну і підбирались за схожими доменними проблемами. Патерни підвищували гнучкість (tab.3) до 52% так як описували рішення специфічних проблем і були ефективними у тих випадках, коли патерни були застосовані доречно і система розвивалась відповідно до них. Точність відображення домену в інтегрованій системі була нижчою 65% за точність незалежної системи і знижувалась з вбудовуванням нових вимог. Ефективність вирішення поточної проблеми збільшувалась до 82% проте зменшувалась з вбудовуванням в систему нових вимог. Ефективність вирішення еволюційних проблем сягала 50%, перший рік це значення варіювалось під час формування системи, а в наступних роках починала знижуватись. Надійність сягала 60% і підвищувалась після тестування і користування в проді проте також падала після 1 року користування системами і надалі зберігалась тенденція зниження.

*Таблиця 3.*

**Якість Підходів Реалізації**

| Критерій Якості | Значення |
|---|---|
| 1. Точність відображення домену поза системами | 70% |
| 2. Точність відображення домену в інтегрованій системі | 65% |
| 3. Повнота відображення домену | 65% |
| 4. Ефективність вирішення поточної проблеми | 82% |
| 5. Ефективність вирішення еволюційних проблем | 50% |
| 6. Гнучкість | 52% |
| 7. Надійність | 60% |

**Аналіз Існуючих Рішень**

Існуючий макропроцес і його підходи зосереджені на описі рішення поточного варіанта використання без врахування еволюції домену через що точність відображення домену, ефективність і гнучкість не сягали 90% і з часом зменшувались через ускладнення системи, що показано на рис. 2. Ці підходи у 9 з 10 випадків під час розробки підбираються командами за схожістю доменної проблеми, але не на основі розрахунків використання даних і процесів.

Загальний бал системи: 70 + 65 + 65 + 82 + 50 + 52 + 60 = 444 із 700.

 ISSN 1560-8956

Тоді, якщо за Деніелом Канеманом вирахувати загальну оцінку по критеріям якості, то 444/700 = 0.63 балів із 1, тобто зазвичай системи були зроблені на 60% із 100%. З часом % функціональної системи знижувався, так як існуючий функціонал переставав працювати чи/та знижувалась ефективність і можливість розширювати. Це відбувалось через еволюціонування системи без врахування 100% доменних правил внаслідок чого система розширювалась не за правилами роботи домену, ринку і бізнеса.

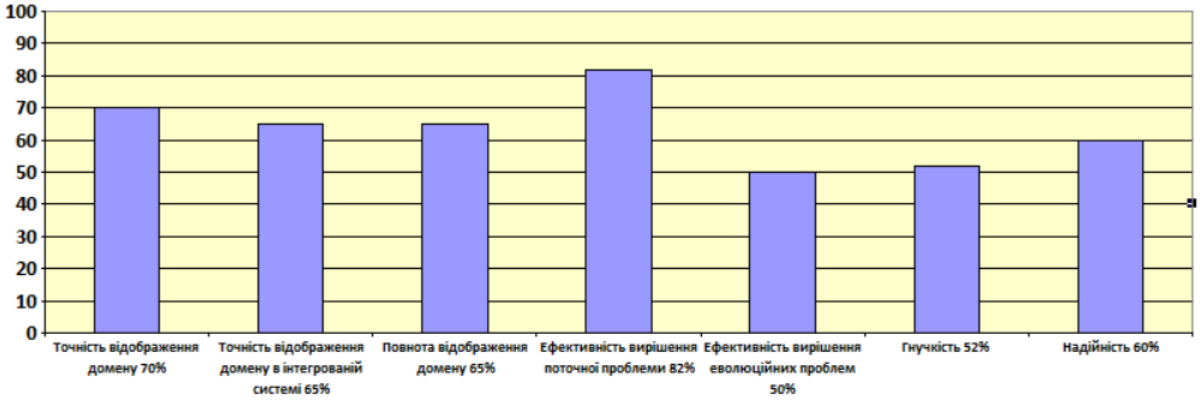


*Рисунок 2*. Результуюча Якість Існуючого Макропроцесу

**Нове Рішення**

Новий підхід має змінені етапи і кроки макропроцесу розробки, що підвищує показники критеріїв якості систем і показаний на рис. 3. В основі підходу лежить систематизоване дослідження домену, еволюція систем за допомогою теорії систем та графів, а також визначення ефективності організацій доменів і даних, за допомогою теорії вірогідності і теорії алгоритмів. В результаті роботи підходу будується система рівнянь, яка однозначно описує рішення варіанта використання із врахуванням еволюційних потреб.

Нове рішення було застосовано на 77 задачах 5 CRM систем(2 банківські, 1 електронна комерція, 1 нерухомість, 1 рекомендаційна телекомунікаційна система).

Створення початкової системи

Ціллю етапа є створення системи, яка ефективно вирішує поточну проблему варіанта використання.

Кроки

1. Визначити назви даних і процесів.

Назви визначаються у варіанті використання за допомогою метода Аббота.

2. Визначити порядок процесів.

Порядок кроків виражається у формі дерева процесів із детальним описом.

3. Визначити вхідні, проміжні і вихідні дані.

Дані розписуються у дереві процесів.

4. Нормалізувати назви за логікою домену.

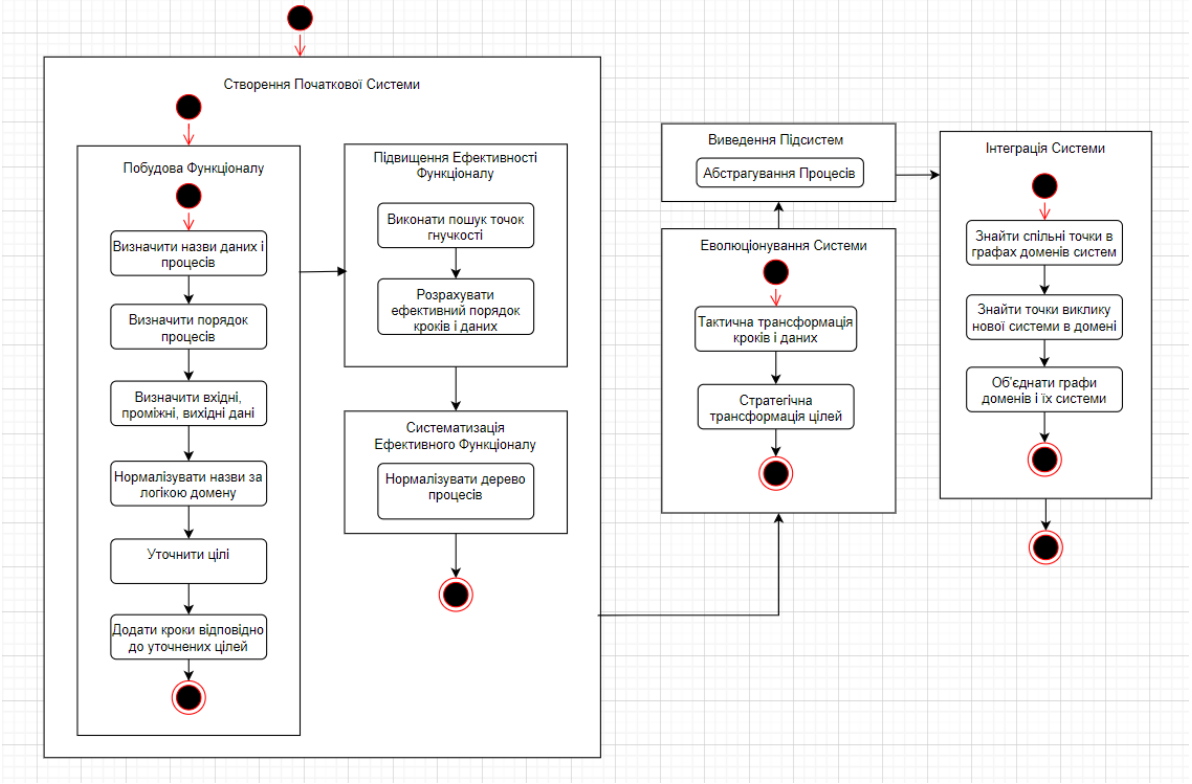


*Рисунок 3.* Нова Схема Макропроцесу

5. Уточнити цілі процесів.

Цілі уточнюються на основі аналізу пост/передумов процесів та вимог.

6. Додати кроки процесів відповідно до уточнених цілей.

Кроки додаються у вигляді мат операцій або псевдокоду CRUD операцій до дерева процесів.

Так як дерево процесів було розширене покроковою інформацію зміни даних, то дерево стало описувати реалізацію функціональних вимог. Тепер підвищується ефективність реалізованих функціональних вимог:

7. Виконати пошук точок гнучкості. Точки шукаються в дереві процесів за рахунок аналізу обов’язковості порядку кроків в кожному процесі

8. Розрахувати ефективний порядок кроків і даних. Для розрахунку порядка застосовується теорія складності алгоритмів.

В цьому дереві процесів доменно коректний і ефективний порядок обробки даних, проте не погоджені між собою (специфіка)назви і (абстракція)категорії процесів/даних.

9. Нормалізувати дерево процесів. Для цього аналізуються і співставляються кроки і дані у логічні доменно зв’язані структури - уніфікуються назви даних/процесів і категоризуються по рівням вкладеності дерево процесів.

 ISSN 1560-8956

Тепер дерево процесів описує порядок кроків і даних із узгодженою структурою, тобто дерево описує систему процесів, їх кроків і даних - **систему доменних знань.**

**Аналіз Результата Етапа**

Побудована система доменних знань виконує вимоги **ефективно** на 60%. Система представлена набором рівнянь вкладених функцій із вхідними, проміжними і вихідними параметрами. Рівняння побудовані на термінах домену, мат операціях(чи/та псевдокоді), тому вони швидко перекладається в код. Такий код - це робочий прототип системи. Проте така система не є **гнучкою і надійною**, тому потребує підвищення кількісних характеристик цих факторів якості.

**Еволюціонування системи**

Ціллю етапа є підвищення **гнучкості** і **надійності** системи за рахунок визначення найбільш вірогідних шляхів еволюції і адаптування до них систему доменних знань.

Опис кроків:

1. тактична Трансформація Системи. Для того, щоб визначити найвірогідніші шляхи еволюції виконується трансформація існуючих цілей і кроків системи. Обійти всі кроки процесів у системі рівнянь процесів. Порядок обходу кроків виконуєтья за рахунок використання алгоритма Depth-First Search (DFS) у зворотньому порядку. Назва процесу - це ціль для досягення якої виконується обробка даних на кроках. При обході крок за кроком трансформуються дані і дії кроків. Вибираються трансформації за теорією вірогідності на основі існуючих потреб із вимог, доменних знань, курсу бізнеса і маркету. Детальніше: розглядається можливість і потреба у різних типах даних вищого і нищого рівня абстракції, а також відповідна зміна дій та навпаки зміна даних через зміну дій.
2. Стратегічна Трансформація. Обійти всі процеси і їх кроки у порядку Depth-First Search (DFS). При обході змінюється ціль, тобто назва процеса, а потім аналізуються наслідки змін, а саме як змінюються вкладені в процес дані і кроки. Виконується обхід найвірогідніших процесів за теорією вірогідності на основі існуючих потреб із вимог, домену, бізнеса і маркета.

В результаті тактичної і стратегічної трансформації система поповнюється новими абстракціями даних і дій. Така система містить необхідні процеси, дані і кроки, а також їх варіанти розширення. Підвищується більше 70% точність і повнота відображення домену, ефективність вирішення поточних задач до 90%, ефективність вирішення еволюційних проблем до 75% і надійність до 80%. При цьому гнучкість

сягає 75%, проте може бути підвищена за рахунок методу абстрагування процесів в контексті інших процесів.

**Виведення підсистем**

Ціллю етапа є підвищення гнучкості і портативності систем методом абстрагування.

Опис кроку:

1. абстрагування процесів. Процеси абстрагуються у класи за рахунок об’єднання за цілями і видимістю у 4 види класів: manager class with strict order, provider class with strict order, provider class without strict order, provider with independent functions. Ці 4 види класів - це 4 методи організації доменних процесів. Для визначення виду класа розраховується видимість кожного процесу класа(модифікатор). Видимість процесів визначається за рахунок аналізу використання перед- і постумов процесів іншими системами.

В результаті абстрагування процесів формуються класи, які описують процеси згруповані по пов’язаним цілям, які представляються у вигляді графа доменів. Граф доменів описує систему високорівнева без деталей, що описує можливості гнучкості системи і пришвидшує наступний етап “Інтеграцію систем”. На цьому етапі формується точність відображення

Інтеграція систем

Ціллю етапа є підвищення точності відображення домену в інтегрованій системі, ефективності вирішення еволюційних проблем та гнучкості.

Опис кроків:

1. знайти спільні точки доменів систем. Із нової і нтегруючої систем доменних знань виводяться графи доменів - тобто залишається структура із доменних точок без процесів, кроків і даних. В цих графах виконується пошук однакових точок за спільною назвою домена або через аналіз споріднених доменів. Нова система може бути підсистемою інтегруючої або незалежною. Підсистема вбудовується в інтегруючу, а незалежна поєднується у нейтральному домені. Нейтральний домен може бути об’єднанням існуючих або новим доменом.

2. Знайти точки виклику нової системи. Визначити нова чи/та існуюча системи викликають одна одну, коли і за яких умов.

3. Об’єднати графи доменів і їх системи за рівнями абстракцій.

**Аналіз Макропроцесів**

Новий макропроцес(таб.4) має такі значення якості:

 ISSN 1560-8956

*Таблиця 4.*

**Етапи і Якість Нового Макропроцесу**

| Критерій Якості | Створення початкової системи | Еволюціонування системи | Виведення підсистем | Інтеграція систем |
|---|---|---|---|---|
| Точність відображення домену поза системами | 65 | 75 | 95 | 95 |
| Точність відображення домену в інтегрованій системі | 0 | 0 | 90 | 90 |
| Повнота відображення домену | 60 | 72 | 80 | 85 |
| Ефективність вирішення поточної проблеми | 60 | 90 | 95 | 95 |
| Ефективність вирішення еволюційних проблем | 0 | 75 | 80 | 85 |
| Гнучкість | 0 | 75 | 80 | 92 |
| Надійність | 0 | 80 | 80 | 90 |

В результаті існуючого і нового макропроцесу виводиться опис домену, дизайн коду і реалізація коду. Існуючий макропроцес продукує ці результати на окремих мало зв’язаних етапах, а новий процес продукує на кожному етапі нову нормалізовану версію домену, архітектури і коду при цьому ці 3 аспекти зв’язані в одну математичну систему, кількісні властивості якої ітераційно підвищуються.

Новий макропроцес має вищу точність і повноту доменних знань на 30% за рахунок того, що на кожному етапі збирається інформація про домен і поєднуєтья із існуючою інформацію методом нормалізації. Більша кількість точних знань домену - більш точний код, який відображає домен - вища швидкість читання коду, що важливо для еволюції систем. Існуючий макропроцес при отриманні нової інформації на етапах поза Аналізом, вбудовує із врахуванням локальних зв’язків процесів і даних без врахування вплиу на глобальні.

Новий макропроцес має вищу ефективність, тому що у кожен процес перевіряється на точки гнучкості і за рахунок теорії алгоритмів підвищується швидкість виконання процесу. Існуючий макропроцес використовує готові рішення, де швидкість розраховується не для поточних вимог або на етапі тестування з’язовується, що процес реалізований в коді працює довго і тоді підвищується ефективність коду, проте із врахування впливу тільки на явно пов’язані процеси і дані.

Новий макропроцес має вищу гнучкість на 40% і надійність майже на 30% вище, так як структура даних, їх класів і процесів виводиться для кожного варіанта використання за допомогою повторного аналізу домену на кожному етапі макропроцесу. Вища гнучкість

пришвидшує процес еволюції систем. Існуючий макропроцес підвищує надійність до 60% на етапі тестування, проте надійність підвищується локально і для явно зв’язаних процесів або процесів визначеними стейкхолдерами.

*Таблиця 5.*

**Порівняння Якості Нового і Існуючого Макропроцесів**

| # | Критерій | Новий метод | Існуючі |
|---|---|---|---|
| 1 | Точність відображення домену поза системами | 95% | 70% |
| 2 | Точність відображення домену в інтегрованій системі | 90% | 65% |
| 3 | Повнота відображення домену | 85% | 65% |
| 4 | Ефективність вирішення поточної проблеми | 95% | 82% |
| 5 | Ефективність вирішення еволюційних проблем | 85% | 50% |
| 6 | Гнучкість | 92% | 52% |
| 7 | Надійність | 90% | 60% |
| Сумарний показник якості (у балах) | | 0.9 із 1 | 0.6 із 1 |

## Висновок

Новий макропроцес, порівняно із існуючим, підвищує точність на 25% і повноту відображення домена на 20%, ефективність поточного рішення на 13% і його еволюціонування на 35%, гнучкості на 40% і надійності на 30%. Ці критерії якості підвищуються поступовим нарощуванням інформації і розвитком за цією інфою актуальних ефективних механізмів гнучкості. В такій системі домен і код розвиваються одночасно, внаслідок чого точність/повнота відображення домену і надійність перманентно оновлюються в коді, а зміни в ефективності і гнучкості коду перманентно оновлюються в домені.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Johnson J.* CHAOS Report Project Outcome Results, 2015. URL: https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf (дата звернення: 09.08.2024)

2. *Boston Consulting Group.* Progress in a Year of Adversity, 2021.

3. *Weaver P.* Do 80% of organizations average a project failure rate of 80%?. 2023.

4. *Flyvbjerg B.* The Empirical Reality of IT Project Cost Overruns: Discovering A Power-Law Distribution. 2022.

5. *Tamburri D. A.* Success and Failure in Software Engineering: Journal “IEEE Transactions on Engineering Management”. 2020. С. 1-13.

6. *Pinto J.* The Causes of Project Failure December. 1990.

7. *Shaw M.* Software Architecture: Perspectives on an Emerging Discipline. 1996.

8. *Booch G.* Object-Oriented Analysis and Design with Applications. 2007.

9. *Jacobson I.* Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley. 1992.

*10. Shlaer S., Mellor S. J.* The Shlaer-Mellor Method. Project Technology, Inc. 1988.

*11. Abbott R.* A laboratory for teaching object-oriented thinking. 1989.

12. *Evans E.*. Domain-Driven Design: Tackling Complexity in the Heart of Software. Addison-Wesley Professional. 2003.

13. *Linevych O., Lisovychenko O.* Software evolution from system perspective / Interdepartmental scientific-technical journal «Adaptive systems of automatic control».- 2024.- № 1 (44).-P.103-110.

14. Gamma E. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional. 1994.

15. Hophe G., Woolf B. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley Professional. 2003.

16. Higginbotham J. Principles of Web API Designing. Pearson Addison-Weasley. 2021.

17. Progress in a Year of Adversity 2020 Annual Sustainability Report April 2021. https://web-assets.bcg.com/40/84/80b567044409b74c32806275a3c1/bcg-2020-annual-sustainability-report-apr-2021-r2.pdf (дата звернення: 08.07.2024)