ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

КВАЛІФІКАЦІЙНА РОБОТА

Тема: «Гнучке управління створенням платформи «TransactChain» для блокчейн-транзакцій для фінансових установ»

Ступінь вищої освіти – магістр

Спеціальність – 073 «Менеджмент»

Освітня програма «Agile-технології розробки програмного забезпечення»

ПОЯСНЮВАЛЬНА ЗАПИСКА

Керівники: доцент кафедри комп'ютерних наук, к.т.н.
Олександр ПОЛІЩУК

викладач кафедри інформаційного менеджменту, математики та статистики
Олег МУШИНСЬКИЙ

Виконав: здобувач
групи МЕН/Agile-24м
Владислав ТУЗЕНКО

Засвідчую, що кваліфікаційна робота оформлена відповідно до ДСТУ 3008:2015 та не містить запозичень з праць інших авторів без відповідних посилань.
Здобувач: _____
(підпис)

Київ, 2026 р.

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

ЗАТВЕРДЖУЮ:
завідувач кафедри інформаційного менеджменту, математики та статистики
_____ Денис БАЛДИК
«___»_____20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
Тузенко Владислав Вячеславович

Тема роботи	Гнучке управління створенням платформи «TransactChain» для блокчейн-транзакцій для фінансових установ
Номер та дата наказу про затвердження теми	№ 109-2 від 14 жовтня 2025 року р.
Коротка постановка завдання	Обґрунтування бачення створюваного продукту для розв'язання проблеми в діяльності замовника на основі розробки моделі його організації. Детальний опис особливостей гнучкого управління розробкою платформи «TransactChain». Розкриття особливостей прийняття управлінських рішень для гнучкого управління розробкою платформи «TransactChain».
Посилання на джерела інформації (не більше п'яти найменувань, які рекомендує науковий керівник)	1. Балдик Д., Мушинський О, Спічак Р. Застосування NLP моделей в управлінні бізнес-процесами електронної комерції будівельних матеріалів: моделювання та статистичний аналіз. Вчені записки Університету "КРОК". 2025. № 2(78). С. 297-305. DOI https://doi.org/10.31732/2663-2209-2025-78-297-305 2. Mihus I. Possibilities of using blockchain technologies to protect fraud. Вчені записки Університету "КРОК". - 2022. - № 1(65). - С. 84-94. - DOI https://doi.org/10.31732/2663-2209-2022-65-84-94. https://dspace.krok.edu.ua/handle/krok/2309
Вимоги до кваліфікаційної роботи	Кваліфікаційна робота має містити теоретичне та/або практичне дослідження за темою роботи, яку слід розглядати як складне спеціалізоване завдання або практичну проблематику в галузі управління та адміністрування, яка характеризується комплексністю та невизначеністю умов і потребує застосування Agile-технологій.

Дата видачі завдання «16» жовтня 2025 р.

Керівник Олександр ПОЛІЩУК
Керівник Олег МУШИНСЬКИЙ
Здобувач Владислав ТУЗЕНКО

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання	Примітка
Підготовчий етап			
1	Вибір напряму дослідження та керівника.	01.09.2025 р.	виконано
2	Формування теми та призначення керівника.	22.09.2025 р.	виконано
3	Затвердження теми кваліфікаційної роботи.	14.10.2025 р.	виконано
4	Затвердження завдання на кваліфікаційну роботу.	16.10.2025 р.	виконано
Основний етап			
5	Розробка концепції та змісту кваліфікаційної роботи, погодження їх з науковим керівником	06.11.2025 р.	виконано
6	Підбір та вивчення джерел інформації з напряму дослідження.	08.11.2025 р.	виконано
7	Теоретико-методичний аналіз предметної області. Підготовка та подання керівнику розділу 1 кваліфікаційної роботи.	13.11.2025 р.	виконано
8	Реалізація гнучкого управління розробкою продукту. Підготовка та подання керівнику розділу 2 кваліфікаційної роботи.	20.11.2025 р.	виконано
9	Розробка рекомендацій щодо вдосконалення управління із застосуванням Agile-технологій. Підготовка та подання керівнику розділу 3 кваліфікаційної роботи.	27.11.2025 р.	виконано
10	Підготовка та подання керівнику першого варіанту всієї кваліфікаційної роботи.	01.12.2025 р.	виконано
11	Доопрацювання кваліфікаційної роботи з урахуваннямзауважень керівника та представлення керівнику доопрацьованого варіанту кваліфікаційної роботи	03.12.2025 р.	виконано
Завершальний етап			
12	Представлення рукопису для перевірки на плагіат.	08.12.2025 р.	*виконано*
13	Підготовка презентації та доповіді на передзахист.	22.12.2025 р.	*виконано*
14	Передзахист кваліфікаційної роботи.	23-24.12.2025 р.	*виконано*
15	Технічна самоекспертиза роботи на відповідність вимогам до оформлення та виправлення недоліків.	12-16.01.2026 р.	*виконано*
16	Експертиза роботи керівником та зовнішнім експертом (рецензентом).	20.01.2026 р.	*виконано*
17	Доопрацювання доповіді та презентації для захисту.	22.01.2026 р.	*виконано*
18	Захист кваліфікаційної роботи.	26-30.01.2026 р.	*виконано*

Керівник Олександр ПОЛІЩУК
Керівник Олег МУШИНСЬКИЙ
Здобувач Владислав ТУЗЕНКО

ЗМІСТ

АНОТАЦІЯ

Тузенко В.В. Гнучке управління створенням платформи «TransactChain» для блокчейн-транзакцій для фінансових установ.

Пояснювальна записка кваліфікаційної роботи за спеціальністю 073 – Менеджмент (освітня програма – Agile-технології розробки програмного забезпечення), СО Магістр. – ВНЗ «Університет економіки та права «КРОК», Навчально-науковий інститут інформаційних та комунікаційних технологій, кафедра інформаційного менеджменту, математики та статистики, Київ, 2026 р.

У кваліфікаційній роботі досліджено особливості застосування Agile-технологій у процесі створення платформи «TransactChain» для здійснення блокчейн-транзакцій у фінансових установах. Проведено аналіз ринку фінансових блокчейн-рішень, визначено ключові проблеми безпеки та регуляторні обмеження. Обґрунтовано вибір гнучких підходів до управління проєктом та розроблено адаптовану модель управління, що поєднує Scrum і Kanban з урахуванням вимог комплаєнсу та безпеки. Запропоновано рекомендації щодо підвищення ефективності управління проєктами у сфері FinTech.

Ключові слова: Agile, Scrum, Kanban, блокчейн, FinTech, управління проєктами, платформа, безпека, комплаєнс.

Табл. 14. Рис. 18. Бібліограф.: 47 найм.

АНОТАЦІЯ

Tuzenko V.V. Agile management of developing the «TransactChain» platform for blockchain transactions in financial institutions.

Qualification paper explanatory note by specialty 073 – Management (educational program – Agile software development technologies). – «KROK» University, Educational and Scientific Institute of Information and Communication Technologies, Department of Information Management, Mathematics and Statistics, Kyiv, 2026.

The qualification work examines the application of Agile technologies in the development of the «TransactChain» platform for blockchain transactions in financial institutions. The market of blockchain financial solutions is analysed, and key security challenges and regulatory constraints are identified. The choice of Agile approaches to project management is justified, and an adapted hybrid management model combining Scrum and Kanban is proposed, taking into account compliance and security requirements. Recommendations for improving project management efficiency in the FinTech domain are developed.

Keywords: Agile, Scrum, Kanban, blockchain, FinTech, project management, platform, security, compliance.

Tabl. 14. Fig. 18. Bibliography: 47 items.

ВСТУП

Актуальність теми. Сучасний фінансовий ринок переживає фундаментальну трансформацію, зумовлену переходом від традиційних централізованих моделей до децентралізованих екосистем (DeFi) на базі технології блокчейн. Забезпечення прозорості, швидкості транзакцій та захисту від шахрайства стає критичною вимогою для конкурентоспроможності фінансових установ. Технологія розподіленого реєстру (DLT) відкриває нові можливості для автоматизації фінансових операцій, зниження операційних витрат та мінімізації ризиків.

Водночас, процес розробки таких складних FinTech-продуктів, як платформа «TransactChain», стикається з унікальним набором викликів. З одного боку, ринок вимагає високої швидкості виведення продукту (Time-to-Market) та гнучкості, що диктує необхідність використання Agile-підходів (Scrum, Kanban). З іншого боку, фінансова сфера характеризується жорстким регулюванням, високими вимогами до безпеки та комплаєнсу (KYC/AML), що традиційно асоціюється з каскадними моделями управління (Waterfall).

Протиріччя між потребою у гнучкості та вимогами надійності створює науково-прикладну проблему: як адаптувати гнучкі методології для створення критично важливих фінансових систем, щоб забезпечити і швидкість розробки, і відповідність регуляторним нормам. Вирішення цієї проблеми шляхом розробки адаптованої моделі управління проєктом «TransactChain» зумовлює актуальність обраної теми магістерської роботи.

Мета і завдання дослідження. Метою роботи є підвищення ефективності процесу створення платформи «TransactChain» шляхом розробки та обґрунтування адаптованої моделі гнучкого управління (Agile), яка враховує специфіку блокчейн-технологій та регуляторні обмеження фінансового сектору.

Для досягнення мети поставлено такі завдання:

1. Проаналізувати ринок фінансових блокчейн-рішень, виявити ключові проблеми безпеки та тенденції розвитку.
2. Здійснити порівняльний аналіз існуючих платформ-аналогів (RippleNet, Aave Arc) для визначення конкурентних переваг проєкту.
3. Визначити функціональні та нефункціональні вимоги до платформи «TransactChain», зокрема в аспекті комплаєнсу.
4. Обґрунтувати доцільність використання Agile-методологій для розробки FinTech-продуктів.
5. Розробити адаптовану гібридну модель управління проєктом, що поєднує гнучкість Scrum/Kanban з процедурами контролю якості та безпеки.
6. Спланувати життєвий цикл реалізації проєкту та оцінити ефективність запропонованої моделі.

Об’єкт дослідження – процес управління створенням програмного забезпечення для підприємств фінансового сектору з використанням технології блокчейн.

Предмет дослідження – моделі, методи та інструменти гнучкого управління (Agile) проєктом розробки платформи «TransactChain».

Методи дослідження. У роботі використано комплекс загальнонаукових та спеціальних методів: *системний аналіз* – для дослідження ринку блокчейн-рішень; *порівняльний аналіз* – для вивчення платформ-конкурентів; *моделювання* – для побудови адаптованої моделі управління; *метод експертних оцінок* – для аналізу ризиків проєкту; *графічний метод* – для візуалізації дорожньої карти та процесів розробки.

Наукова новизна одержаних результатів полягає в удосконаленні підходів до управління FinTech-проєктами шляхом розробки гібридної моделі, яка, на відміну від стандартних Agile-фреймворків, інтегрує етапи Compliance Review та Security Audit безпосередньо у життєвий цикл спринту, що дозволяє знизити регуляторні ризики без втрати гнучкості розробки.

Практична значущість одержаних результатів. Розроблена модель управління, дорожня карта проєкту та реєстр ризиків можуть бути використані проєктними командами для створення платформи «TransactChain» або аналогічних продуктів у сфері децентралізованих фінансів, забезпечуючи скорочення часу розробки та підвищення якості кінцевого продукту.

Структура та обсяг роботи. Робота складається зі вступу, трьох розділів, висновків та списку використаних джерел.

РОЗДІЛ 1
ДИЗАЙН БІЗНЕСУ ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ПРОЄКТУ

1.1. Характеристика ринку фінансових блокчейн-рішень: проблеми, тренди та виклики безпеки

Сучасна світова економіка перебуває на етапі фундаментальної трансформації, яку експерти характеризують як перехід від «Інтернету інформації» до «Інтернету цінностей» (Internet of Value) [24, 25]. Традиційна фінансова інфраструктура, побудована на базі централізованих банківських систем та клірингових палат, досягла межі своєї ефективності. В умовах глобалізації та цифровізації бізнес-процесів, класичні механізми транскордонних платежів (зокрема, система SWIFT) демонструють низку суттєвих недоліків: низьку швидкість проведення операцій (від 1 до 5 банківських днів), високу вартість транзакцій через наявність ланцюжка посередників, а також непрозорість процесів для кінцевого споживача [13].

Порівняльне зіставлення ефективності традиційних та блокчейн-транзакцій наведена на рис 1.1.

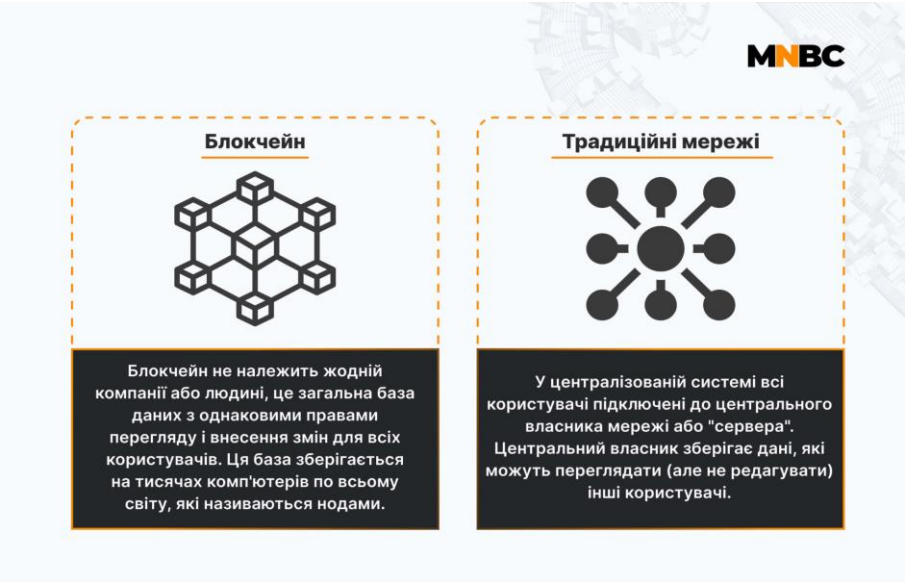

Рисунок 1.1 – Порівняльне зіставлення ефективності традиційних та блокчейн-транзакцій

Джерело: [34]

Відповіддю на ці системні виклики стала поява та стрімкий розвиток технології розподіленого реєстру (Distributed Ledger Technology – DLT) та блокчейну. За своєю сутністю, блокчейн є децентралізованою базою даних, яка забезпечує зберігання інформації про транзакції у вигляді ланцюжка блоків, захищених криптографічними методами [7]. Це дозволяє учасникам мережі досягати консенсусу щодо стану рахунків без необхідності залучення довіреної третьої сторони (центрального контрагента).

1.1.1. Проблематика економічного шахрайства у фінансовій сфері

Одним із ключових драйверів впровадження блокчейн-технологій є необхідність протидії фінансовому шахрайству, яке залишається глобальною проблемою для корпоративного сектору. Згідно з дослідженнями *Асоціації сертифікованих експертів із шахрайства (ACFE)*, на які посилається у своїх працях д.е.н. І.П. Мігус, типологія корпоративного шахрайства охоплює три основні категорії (рис 1.2), кожна з яких несе загрозу фінансовій стійкості установ [1, 12]:

1. Корупція та зловживання впливом. Сюди відносяться схеми хабарництва, конфлікт інтересів та незаконні винагороди. У традиційних системах виявлення таких схем ускладнене через можливість "ручного" втручання в бази даних та приховування слідів транзакцій.
2. Привласнення активів. Найпоширеніший вид шахрайства, що включає крадіжку грошових коштів, фальсифікацію чеків та маніпуляції з платіжними відомостями.
3. Шахрайство з фінансовою звітністю. Це найнебезпечніший вид зловживань, який, за даними звіту ACFE «Report to the Nations», завдає компаніям найбільших середніх збитків (медіанне значення збитків може сягати мільйонів доларів). Особливу загрозу становить латентність цього виду злочину: середній період від початку махінацій до їх виявлення становить близько 18 місяців [12].

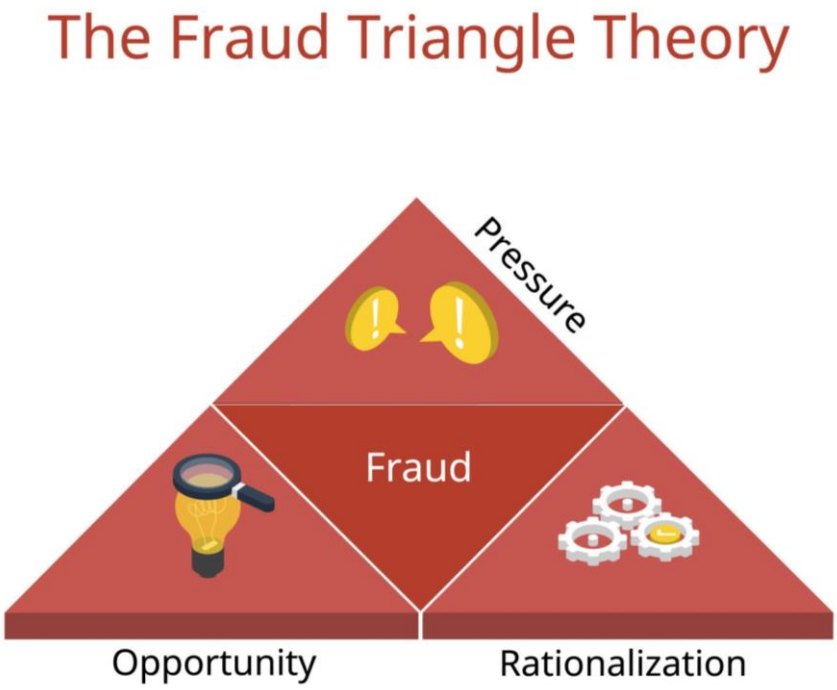

Рисунок 1.2 – «Трикутник шахрайства» (Fraud Triangle): тиск, можливість, виправдання

Джерело: [35]

Традиційні методи внутрішнього аудиту та контролю часто виявляються неефективними проти складних схем маніпулювання звітністю, оскільки базуються на вибірковій перевірці (sampling) та постфактум-аналізі. Натомість, інтеграція блокчейн-технологій дозволяє перейти до парадигми безперервного аудиту (Continuous Audit). Завдяки властивості незмінності (immutability), будь-яка транзакція, записана в блокчейн, не може бути змінена або видалена заднім числом без порушення цілісності всього ланцюга [1, 24].

Для системного розуміння доцільності впровадження платформи «TransactChain», нами було проведено SWOT-аналіз використання блокчейн-технологій у фінансових операціях, адаптований на основі досліджень І.П. Мігус (табл. 1.1).

Таблиця 1.1 SWOT-аналіз впровадження блокчейн-технологій для фінансових операцій

Сильні сторони (Strengths)	*Слабкі сторони (Weaknesses)*
1. Можливість відстеження історії змін фінансових документів у реальному часі (Audit Trail). 2. Мінімізація затримок у підготовці звітності та проведенні транзакцій. 3. Зниження адміністративного навантаження на відділи внутрішнього контролю. 4. Покращення ділової репутації компанії як прозорого та технологічного партнера. 5. Неможливість підробки даних заднім числом.	1. Незрілість механізмів правового регулювання та малий досвід застосування в Україні. 2. Висока вартість початкової інтеграції з існуючими (legacy) банківськими системами. 3. Складність у розумінні технології персоналом та необхідність навчання. 4. Незворотність транзакцій (помилковий переказ неможливо скасувати без згоди отримувача).
Можливості (Opportunities)	*Загрози (Threats)*
1. Автоматизація комплаєнс-процедур (KYC/AML) через смарт-контракти. 2. Підвищення ефективності заходів із виявлення шахрайства (fraud detection) на ранніх етапах. 3. Інтеграція у глобальні цифрові екосистеми та доступ до ринків DeFi. 4. Відповідність загальним світовим трендам цифровізації економіки.	1. Відсутність універсального підходу до транскордонного обміну даними (різні стандарти). 2. Кіберзагрози: вразливості смарт-контрактів та атаки на інфраструктуру. 3. Ризик жорсткого регулювання криптовалют з боку державних органів. 4. Волатильність транзакційних комісій у публічних мережах.

1.1.2. Ринок DeFi та еволюція фінансових платформ

Паралельно з корпоративним сектором, активний розвиток демонструє ринок децентралізованих фінансів (DeFi). Екосистема DeFi пропонує відкриті фінансові інструменти (кредитування, обмін активів, деривативи), які функціонують на базі публічних блокчейнів (переважно Ethereum, Solana, Binance Smart Chain) без участі банків [8].

Динаміку розвитку цього сектору найкраще ілюструє показник TVL (Total Value Locked) – загальна вартість активів, заблокованих у смарт-контрактах (рис. 1.3).

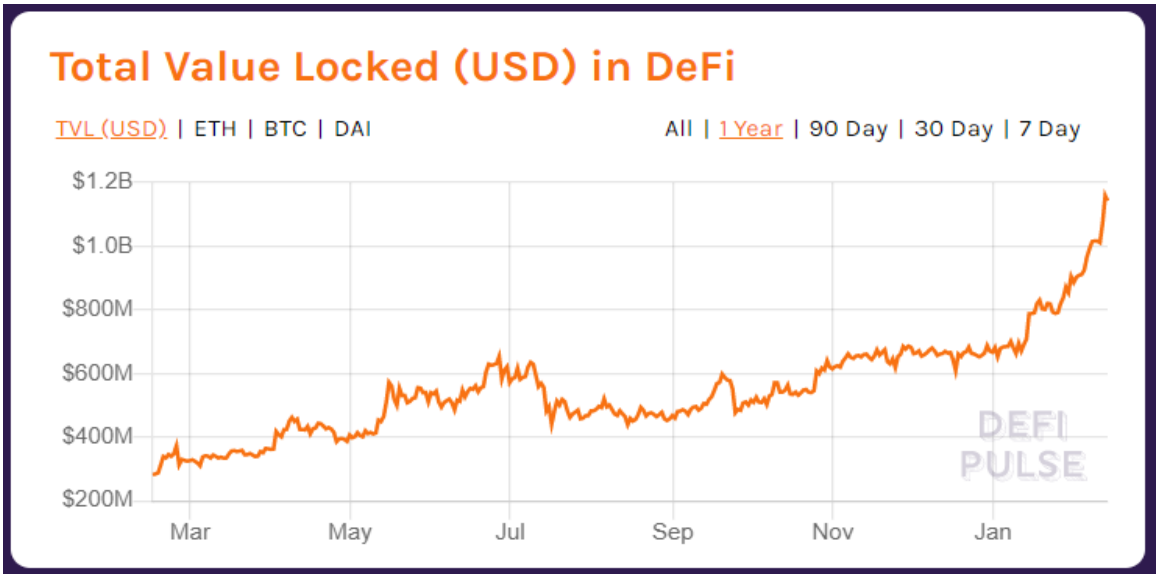

Рисунок 1.3 – Динаміка зміни TVL (Total Value Locked) на ринку DeFi (млрд дол. США)

Джерело: [36]

Як видно з діаграми, незважаючи на корекцію ринку у 2022 році, довгостроковий тренд залишається висхідним, що свідчить про довіру інституційних інвесторів до технології. Станом на 2024-2025 роки, ринок характеризується такими ключовими трендами [31]:

1. Токенізація реальних активів (RWA). Перенесення прав власності на фізичні активи (нерухомість, облігації) на блокчейн.
2. Інституціоналізація. Великі гравці (JPMorgan, BlackRock) створюють власні приватні мережі.
3. Інтероперабельність. Потреба у крос-чейн мостах для передачі ліквідності.

1.1.3. Виклики безпеки та класифікація ризиків

Попри значний потенціал, масове впровадження блокчейн-рішень стримується низкою суттєвих ризиків. Для проєкту «TransactChain» нами було розроблено класифікацію основних ризиків, які необхідно врахувати при побудові моделі управління (табл. 1.2).

Таблиця 1.2 Класифікація ризиків розробки платформи «TransactChain»

Тип ризику	*Опис проблеми*	*Вплив на проєкт*	*Стратегія Agile-команди*
Технічні ризики	Вразливості у коді смарт-контрактів (bugs), атаки типу Reentrancy, проблеми масштабованості мережі [8].	Критичний (втрата коштів клієнтів)	Впровадження обов'язкового етапу Security Audit у Definition of Done кожного спринту.
Регуляторні ризики	Невідповідність вимогам AML/KYC, зміни у законодавстві про віртуальні активи, GDPR [9, 11].	Високий (штрафи, блокування)	Залучення Compliance Officer до команди розробки, створення гнучкого механізму оновлення правил.
Операційні ризики	Втрата приватних ключів доступу, помилки персоналу, проблеми з інтеграцією API.	Середній (збої в роботі)	Використання мульти-підписів (Multi-sig), автоматизоване тестування, навчання персоналу [20].

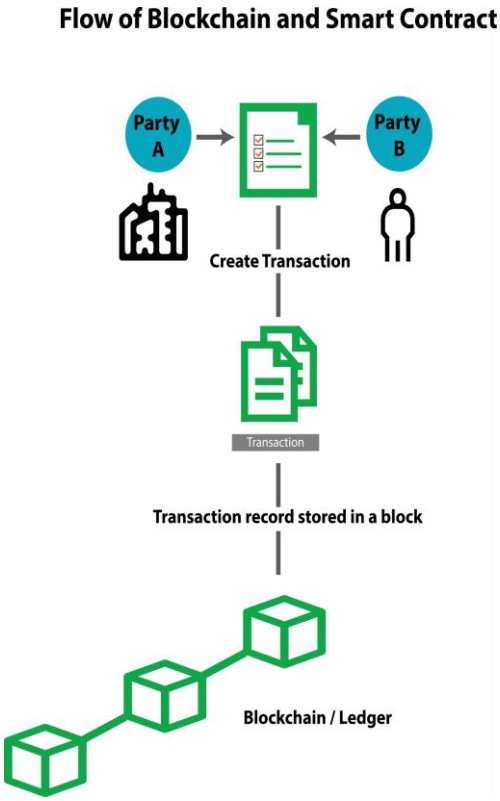

Рисунок 1.4 – Схема інтеграції комплаєнс-процедур у блокчейн-транзакцію

Джерело: [37]

1.2. Аналіз конкурентного середовища (на прикладі RippleNet та Aave Arc) та постановка цілей проєкту

Для формування стратегії розробки платформи «TransactChain» необхідно детально проаналізувати існуючі на ринку рішення, які намагаються вирішити проблему неефективності традиційних фінансових транзакцій. Ринок сьогодні поляризований між двома підходами: централізованими блокчейн-мережами для банків (Enterprise Blockchain) та адаптованими протоколами децентралізованих фінансів (Permissioned DeFi) [13, 14].

Як релевантні об'єкти для бенчмаркінгу (порівняльного аналізу) нами обрано RippleNet (лідер у секторі банківських транскордонних платежів) та Aave Arc (провідний протокол DeFi для інституційних інвесторів).

1.2.1. Аналіз екосистеми RippleNet

RippleNet – це глобальна децентралізована мережа банків та платіжних провайдерів, яка використовує технологію розподіленого реєстру для здійснення платежів у реальному часі. На відміну від Bitcoin чи Ethereum, які використовують енергоємний майнінг (Proof-of-Work) [7], Ripple використовує унікальний алгоритм консенсусу RPCA (Ripple Protocol Consensus Algorithm) (рис 1.5).

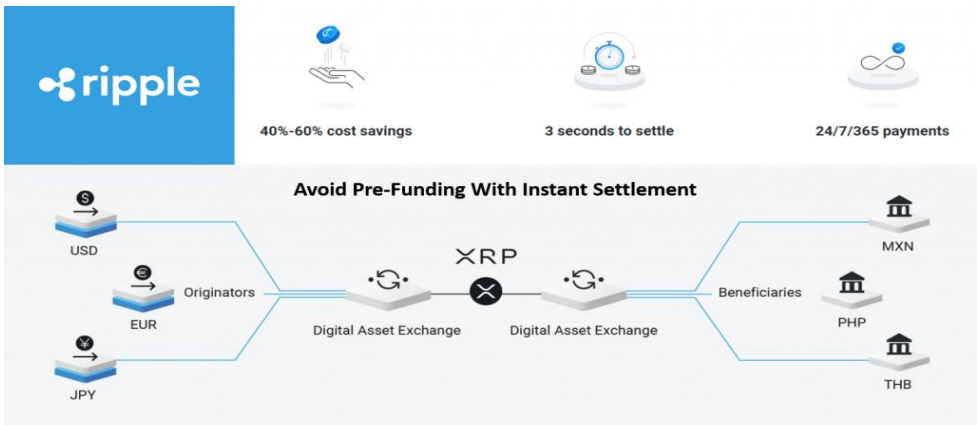

Рисунок 1.5 – Архітектура здійснення транзакцій у мережі RippleNet [13]

Джерело: [38]

Ключовим елементом системи є використання криптовалюти XRP як "моста" для забезпечення ліквідності (On-Demand Liquidity – ODL).

Механізм роботи: Замість традиційної системи кореспондентських рахунків (Nostro/Vostro), де банки змушені заморожувати значні обсяги капіталу на рахунках партнерів по всьому світу, RippleNet пропонує наступний алгоритм:

1. Банк-відправник (наприклад, у США) конвертує фіатну валюту (USD) в цифровий актив XRP.

2. XRP миттєво переміщується через розподілений реєстр XRP Ledger до банку-отримувача (наприклад, у Мексиці).

3. Банк-отримувач конвертує XRP у місцеву валюту (MXN). Весь процес займає 3-5 секунд, що є революційним порівняно з 3-5 днями у SWIFT [13].

Переваги RippleNet:

1. Швидкість та масштабованість: Мережа здатна обробляти 1500 транзакцій на секунду (TPS) з фінальністю (незворотністю) до 4 секунд.

2. Низька вартість: Комісія за транзакцію становить долі цента (0.00001 XRP), що робить економічно доцільними навіть мікроплатежі.

3. Усталена клієнтська база: Понад 300 фінансових установ-партнерів, включаючи Santander, SBI Holdings та American Express.

Недоліки та вразливості (Opportunities для TransactChain):

1. Централізація управління. Попри декларацію децентралізації, компанія Ripple Labs контролює значну частину вузлів-валідаторів (UNL – Unique Node List) та близько 50% емісії токенів XRP. Це створює ризики цензури транзакцій та залежності всієї екосистеми від одного комерційного вендора.

2. Регуляторні ризики. Тривалий судовий процес із Комісією з цінних паперів і бірж США (SEC), розпочатий у 2020 році, продемонстрував вразливість моделі, де інфраструктура критично залежить від юридичного

статусу одного активу. Невизначеність статусу XRP відлякує консервативні банки.

3. Жорсткість (Rigidity) розробки. Як велика корпоративна структура, Ripple має довгі цикли оновлення протоколу. Впровадження нових функцій (наприклад, смарт-контрактів) відбувається повільно, що є характерним для Waterfall-підходів до управління продуктом. Це обмежує можливості кастомізації для окремих клієнтів [28].

1.2.2. Аналіз протоколу Aave Arc

Aave Arc (рис 1.6)– це спеціалізована версія (permissioned pool) популярного DeFi-протоколу кредитування Aave, розроблена спеціально для фінансових установ, які прагнуть отримати доступ до ринку DeFi, але обмежені регуляторними вимогами [14].

Рисунок 1.6 – Механізм "Permissioned Pool" (дозволеного пулу) в Aave Arc [14]
Джерело: [39]

Механізм роботи: Aave Arc створює ізольовані пули ліквідності (Private Pools), які працюють паралельно з основним публічним протоколом Aave.

Ключова відмінність – наявність ролі "Вайтлістера" (Whitelister). Це ліцензовані організації (наприклад, кастодіан Fireblocks), які проводять перевірку KYC/KYB (Know Your Business) кожного учасника перед тим, як надати йому дозвіл на взаємодію зі смарт-контрактом [10].

Переваги Aave Arc:

1. Прозорість та автоматизація DeFi. Використання смарт-контрактів Ethereum гарантує автоматичне виконання умов кредитування та нарахування відсотків без участі посередників-людей [23].

2. Відповідність регуляціям (Compliance). Механізм "білих списків" гарантує, що всі учасники пулу є ідентифікованими та перевіреними, що знімає ризик взаємодії з "брудними" коштами (AML ризик) [10].

3. Дохідність. Дозволяє інституційним інвесторам отримувати вищу дохідність, характерну для крипторнку, порівняно з традиційними банківськими депозитами.

Недоліки та вразливості:

1. Фрагментація ліквідності. Пули Arc ізольовані від основної, багатомільярдної ліквідності публічного ринку DeFi. Це знижує ефективність капіталу та може призводити до менш вигідних відсоткових ставок.

2. Залежність від публічного блокчейну. Aave Arc працює на базі Ethereum, що означає залежність від пропускної здатності цієї мережі та високих комісій (gas fees) у періоди пікового навантаження.

3. Ризики смарт-контрактів. Як і будь-який протокол на Ethereum, Aave схильний до ризиків злому коду [8]. Необхідність внесення змін у смарт-контракти вимагає складного процесу голосування в DAO (децентралізованій автономній організації), що робить реакцію на загрози повільною.

1.2.3. Порівняльний аналіз та визначення ніші «TransactChain»

На основі детального аналізу технологічних та операційних аспектів діяльності конкурентів, нами сформовано порівняльну характеристику існуючих рішень та проєктованої платформи «TransactChain» (табл. 1.3).

Таблиця 1.3 - Порівняльний аналіз платформ для фінансових транзакцій

Критерій порівняння	*RippleNet (Enterprise)*	*Aave Arc (Institutional DeFi)*	*TransactChain (Проєкт)*
Архітектура мережі	Централізований приватний реєстр (Permissioned Ledger)	Публічний блокчейн (Ethereum) з надбудовою доступу	Гібридна архітектура (Consortium Blockchain)
Механізм консенсусу	RPCA (Federated Byzantine Agreement)	Proof-of-Stake (Ethereum)	IBFT 2.0 (Proof-of-Authority)
Управління даними та приватність	Закрите (офф-чейн повідомлення), дані приховані	Повністю відкрите (он-чейн), транзакції видимі	Гнучке (Zero-Knowledge Proofs для приватності балансів)
Модель управління розробкою	Корпоративна (Waterfall/Hybrid) – довгі релізи [28]	Децентралізована (DAO Voting) – складна координація	Адаптивна (Agile: Scrum + Compliance Integration)
Швидкість змін (Time-to-Market)	Низька (цикли оновлення 6-12 місяців)	Середня (залежить від швидкості голосування спільноти)	Висока (спринти 2 тижні) [22]
Пропускна здатність (TPS)	~1500 TPS	~15-30 TPS (Layer 1)	>10 000 TPS (Layer 2 / Sidechain)
Фокус безпеки	Юридичні гарантії та SLA	Аудит коду спільнотою та Bug Bounty	Continuous Security & Compliance Integration (DevSecOps) [17]

Як видно з таблиці, RippleNet пропонує швидкість, але страждає від централізації та жорсткості управління. Aave Arc забезпечує прозорість, але обмежений технічними можливостями Ethereum та складністю DAO-управління. Платформа «TransactChain» позиціонується у "блакитному океані" між цими двома крайнощами.

1.2.4. Постановка цілей проєкту «TransactChain»

Аналіз конкурентного середовища дозволив виявити критичну ринкову потребу: фінансові установи потребують платформи, яка була б більш гнучкою та адаптивною, ніж Ripple, але більш контрольованою та масштабованою, ніж Aave Arc.

Виходячи з цього, основною метою проєкту є створення платформи для фінансових транзакцій, процес розробки якої базується на адаптованій Agile-методології. Це дозволить досягти наступних стратегічних цілей, які декомпозуються на три рівні:

1. Технологічні цілі:

– забезпечити пропускну здатність понад 10 000 транзакцій на секунду (TPS) з фінальністю (незворотністю) до 2 секунд;

– реалізувати підтримку стандарту ISO 20022 для безшовної інтеграції з існуючими банківськими системами [18];

– забезпечити приватність даних клієнтів за допомогою протоколів доведення з нульовим розголошенням (Zero-Knowledge Proofs);

2. Комплаєнс-цілі:

– інтегрувати автоматизовану перевірку AML/KYC безпосередньо у протокол транзакції (Transaction-level compliance), що унеможливить проведення незаконних операцій на рівні коду [10];

– забезпечити відповідність вимогам регуляторів ЄС (MiCA - Markets in Crypto-Assets) та США [9];

Управлінські цілі (Ключові для магістерської роботи):

– побудувати процес розробки таким чином, щоб цикл внесення змін до смарт-контрактів (від ідеї до безпечного деплою) займав не більше 2 тижнів (один спринт), попри жорсткі вимоги до безпеки [4];

– створити модель управління, яка дозволяє ефективно координувати роботу розподіленої команди, що включає розробників, юристів та спеціалістів з кібербезпеки.

Саме досягнення управлінських цілей є критичним фактором успіху. Традиційні підходи не здатні забезпечити необхідну швидкість реакції на зміни ринку та регуляторного середовища, а стандартні Agile-методики несуть неприпустимі ризики безпеки. Це вимагає розробки унікального, гібридного підходу до менеджменту, який буде детально розглянуто та обґрунтовано у наступних розділах роботи.

1.3. Визначення функціональних вимог до платформи «TransactChain» та регуляторних обмежень (комплаєнс)

Успіх платформи «TransactChain» залежить не лише від обраного технологічного стеку, а й від чіткого визначення вимог, які задовольняють потреби стейкголдерів (банків, регуляторів, клієнтів) та відповідають жорстким нормам законодавства. Враховуючи специфіку проєкту, процес формування вимог має базуватися на принципах Compliance-by-Design (відповідність нормам на рівні архітектури) та Privacy-by-Design (пріоритетність захисту персональних даних) [11].

1.3.1. Аналіз регуляторних обмежень (Regulatory Landscape)

Перед формулюванням функціональних вимог необхідно окреслити правове поле, в якому функціонуватиме платформа. На відміну від нерегульованих DeFi-протоколів, «TransactChain» орієнтується на інституційних клієнтів, а отже, мусить відповідати трьом ключовим міжнародним стандартам:

1. MiCA (Markets in Crypto-Assets Regulation). Це нове законодавство ЄС, яке набуває чинності у 2024-2025 роках. Воно вимагає від емітентів токенів та провайдерів послуг (CASP) забезпечення прозорості, наявності резервів та захисту прав споживачів. Для «TransactChain» це означає необхідність впровадження процедур аудиту смарт-контрактів та публікації Whitepaper з чітким описом ризиків [9].

2. GDPR (General Data Protection Regulation). Вимога "права на забуття" (Right to be Forgotten) вступає в конфлікт з природою блокчейну (незмінність даних). Для вирішення цієї колізії платформа повинна використовувати архітектуру, де персональні дані зберігаються "офф-чейн" (на захищених серверах), а в блокчейн записуються лише їхні хеші (криптографічні відбитки) [11].

3. FATF Travel Rule. Вимога Групи з розробки фінансових заходів боротьби з відмиванням грошей. Вона зобов'язує провайдерів (VASP) обмінюватися даними про відправника та отримувача транзакції, якщо сума перевищує певний поріг [10].

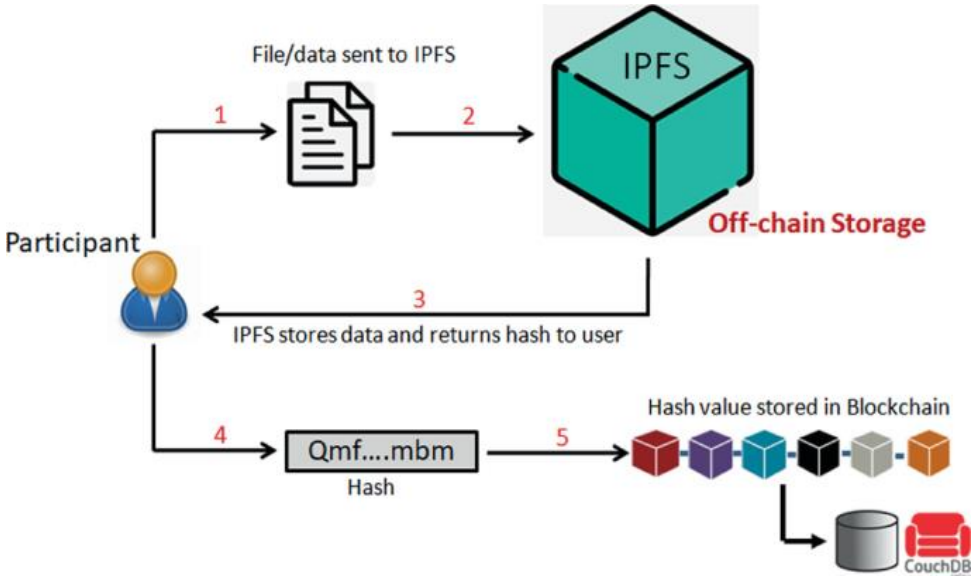

Рисунок 1.7 – Архітектурне вирішення конфлікту між GDPR та Blockchain: зберігання хешів замість даних

Джерело: [40]

1.3.2. Функціональні вимоги до платформи (Functional Requirements)

На основі аналізу потреб користувачів та регуляторних норм, нами сформовано перелік основних функцій системи, згрупованих за ролями користувачів (табл. 1.4). Ці вимоги стануть основою для формування Product Backlog у наступних розділах.

Таблиця 1.4 Матриця функціональних вимог до платформи «TransactChain»

Роль користувача	*ID Вимоги*	*Опис функції (User Story)*	*Критерії приймання (Acceptance Criteria)*
Клієнт (Бізнес/Банк)	FR-01	Реєстрація та онбординг (KYB)	Користувач може завантажити корпоративні документи. Система автоматично перевіряє їх через реєстри та надає доступ до гаманця [12].
	FR-02	Створення транзакції	Користувач може ініціювати переказ активів, вказавши суму та отримувача. Система розраховує комісію та час виконання.
	FR-03	Управління ліквідністю	Користувач може надавати активи у пули ліквідності для отримання доходу під заставу.
Compliance Officer	FR-04	Моніторинг транзакцій (AML)	Система автоматично блокує транзакції, що пов'язані з адресами із "чорних списків" (Sanction Lists) [10].
	FR-05	Верифікація користувачів	Офіцер має інтерфейс для ручного затвердження складних кейсів KYC, які не пройшли автоматичну перевірку.
Аудитор	FR-06	Вивантаження звітів	Система генерує незмінний звіт (Immutable Log) усіх операцій за обраний період для зовнішнього аудиту [1].
Адміністратор	FR-07	Управління смарт-контрактами	Можливість оновлення логіки смарт-контрактів через механізм мульти-підпису (Multi-sig) без зупинки системи.

1.3.3. Нефункціональні вимоги (Non-Functional Requirements)

Для забезпечення конкурентоспроможності порівняно з RippleNet, платформа повинна відповідати високим стандартам якості обслуговування (QoS).

1. Продуктивність та масштабованість (Performance):

– Throughput (Пропускна здатність): Система повинна підтримувати мінімум 10 000 TPS у піковому навантаженні, щоб конкурувати з традиційними платіжними шлюзами [25];

– Latency (Затримка): Час фіналізації блоку (підтвердження транзакції) не повинен перевищувати 2 секунди.

2. Безпека (Security):

– Криптографія: Використання стандартів шифрування, стійких до квантових обчислень (Post-Quantum Cryptography) для критичних вузлів [8];

– Multi-Signature: Усі операції з управління коштами платформи (Treasury) вимагають схвалення мінімум 3 з 5 уповноважених осіб;

– Інтероперабельність (Interoperability);

– Підтримка стандарту обміну фінансовими повідомленнями ISO 20022. Це критична вимога для інтеграції з банківськими системами (Core Banking Systems).

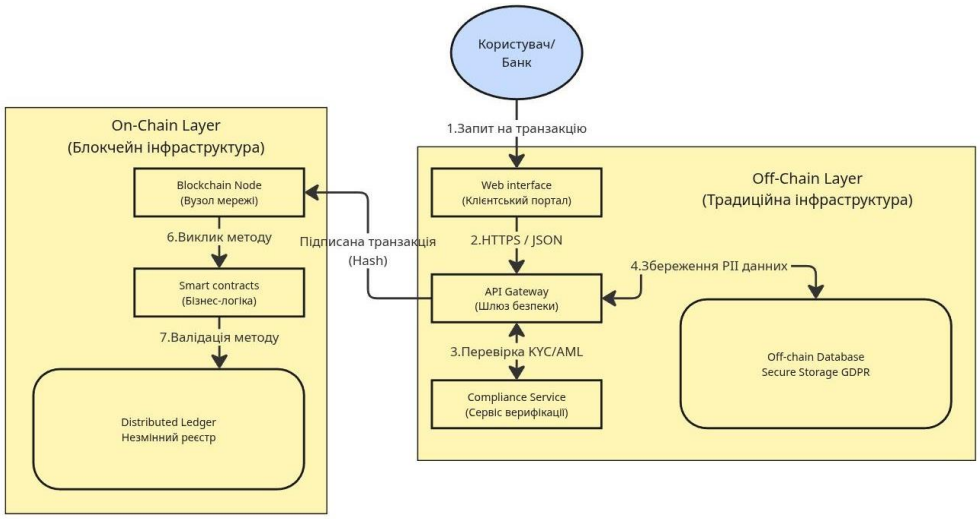

Рисунок 1.8 – Високорівнева архітектура платформи «TransactChain»

Джерело: розроблено автором

РОЗДІЛ 2
ГНУЧКЕ УПРАВЛІННЯ СТВОРЕННЯМ ПРОДУКТУ (AGILE-ПІДХОДИ)

2.1. Обґрунтування вибору методології Agile (Scrum/Kanban) для розробки FinTech-продуктів

У сучасному світі розробки програмного забезпечення, особливо у фінансовому секторі, спостерігається фундаментальний зсув парадигми управління проєктами. Жорстка конкуренція, стрімкий розвиток технологій та мінливість регуляторного середовища вимагають від компаній не просто створення якісного продукту, а здатності швидко адаптуватися до змін [26]. Створення платформи «TransactChain» характеризується високим рівнем невизначеності, що робить вибір правильної моделі управління критичним фактором успіху.

2.1.1. Еволюція методологій: від Waterfall до Agile

Традиційно, банківські та фінансові установи використовували каскадну модель (Waterfall) (рис 2.1) [18]. Цей підхід передбачає сувору послідовність етапів: системний аналіз -> проєктування -> розробка -> тестування -> впровадження. Перехід на наступний етап можливий лише після повного завершення та документального затвердження попереднього.

Для проєкту «TransactChain» використання Waterfall несе неприпустимі ризики:

1. Ефект "тунелю". Команда може витратити 6-9 місяців на розробку функціоналу, який на момент релізу стане неактуальним через зміну вимог регулятора (наприклад, впровадження нових норм MiCA). У Waterfall зміни на пізніх етапах є надзвичайно дорогими або неможливими [28].
2. Вартість помилки. У Waterfall тестування відбувається в самому кінці. Якщо архітектурну помилку буде виявлено на етапі тестування, вартість її виправлення може перевищувати бюджет усього проєкту.

3. Бюрократичне навантаження. Надмірна увага до детальної документації на етапі ініціації уповільнює комунікацію та прийняття рішень, що є критичним для стартапу, який працює в умовах "гонки озброєнь" на ринку DeFi [27].

Альтернативою є Agile (гнучка розробка) – філософія, що базується на ітеративному підході. Основна ідея Agile полягає в тому, щоб постачати цінність клієнту невеликими частинами (інкрементами), постійно отримуючи зворотний зв'язок та коригуючи курс проєкту відповідно до реальних потреб ринку [19, 28].

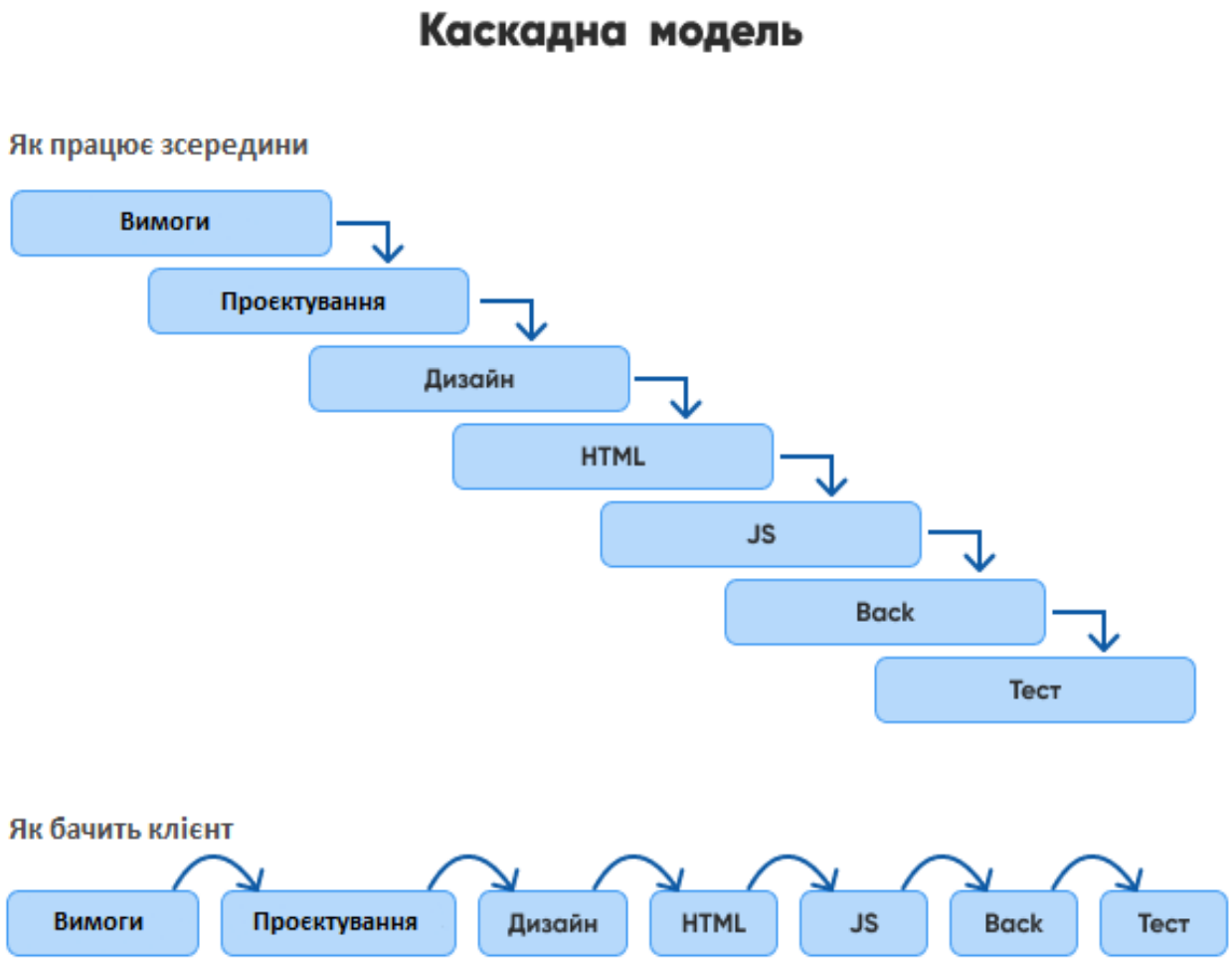

Рисунок 2.1 – Етапи життєвого циклу каскадної моделі (Waterfall)

Джерело: [41]

2.1.2. Аналіз принципів Agile Manifesto в контексті блокчейн-розробки

Для глибшого розуміння доцільності Agile, проаналізуємо ключові цінності Agile Manifesto [2, 30] крізь призму створення платформи «TransactChain»:

1. Люди та взаємодія важливіші за процеси та інструменти. В контексті блокчейн-проєкту це означає, що успіх залежить від ефективної горизонтальної комунікації між вузькопрофільними спеціалістами (криптографами, блокчейн-інженерами, бекенд-розробниками) та юристами (Legal/Compliance). Жоден регламент не може передбачити всі нюанси взаємодії коду смарт-контракту з нормами GDPR. Тільки жива співпраця дозволяє знаходити нестандартні рішення [4].

2. Працюючий продукт важливіший за вичерпну документацію. Інвесторам та стейкголдерам важливо бачити працюючий MVP (Minimum Viable Product) для тестування гіпотез, а не стоси технічних завдань [6]. Смарт-контракт, який працює у тестовій мережі, є кращим доказом прогресу.

3. Співпраця із замовником важливіша за узгодження умов контракту. Оскільки вимоги до продукту формуються в процесі (наприклад, деталі інтеграції з банківськими API), постійний діалог з Product Owner (представником бізнесу) є критичним [2].

4. Готовність до змін важливіша за дотримання плану. Ринок криптовалют працює 24/7 і змінюється миттєво. Команда повинна бути готова змінити пріоритети (наприклад, терміново імплементувати новий стандарт токенів) посеред розробки [5].

2.1.3. Порівняльний аналіз фреймворків: Scrum, Kanban, Lean, XP

Agile – це "парасольковий" термін, який об'єднує різні методології. Для вибору оптимального підходу нами було проаналізовано чотири найпопулярніші фреймворки (табл. 2.1).

Таблиця 2.1 Порівняльний аналіз Agile-фреймворків для розробки «TransactChain»

Критерій	*Scrum*	*Kanban*	*Lean (Ощадливе виробництво)*	*XP (Екстремальне програмування)*
Основа	Ітерації (Спринти) фіксованої довжини [2, 22]	Безперервний потік завдань (Flow) [3]	Усунення втрат (Muda) [21]	Інженерні практики якості коду [5]
Планування	Перед кожним спринтом	По мірі надходження (Just-in-Time)	Стратегічне	Щоденне
Ролі	Product Owner, Scrum Master, Team	Не регламентовані	Лідер, команда	Coach, Tracker, Customer
Зміни	Небажані під час спринту	Вітаються в будь-який момент	Постійне вдосконалення	Вітаються
Застосування для «TransactChain»	Ідеально для розробки ядра (Core)	Ідеально для підтримки та DevSecOps	Як філософія оптимізації процесів	Як набір технік (TDD, Pair Programming)

Scrum (рис 2.2) забезпечує ритм та дисципліну. Наявність чітких подій (Sprint Planning, Review, Retrospective) змушує команду регулярно показувати результат. Це критично важливо для стартапу, якому потрібно демонструвати прогрес інвесторам та підтримувати високий темп розробки (Velocity) [22].

Kanban фокусується на візуалізації робочого процесу та обмеженні кількості незавершеної роботи (WIP – Work In Progress). Він ідеально підходить для тих частин проєкту, які важко запланувати наперед: виправлення багів, технічна підтримка, реакція на інциденти безпеки. Kanban дозволяє виявити "вузькі місця" (bottlenecks) у процесі розробки [3].

XP (Extreme Programming) пропонує конкретні інженерні практики: парне програмування (Pair Programming), розробка через тестування (TDD), безперервна інтеграція (CI). Для блокчейн-проєкту, де ціна помилки в коді є надзвичайно високою, елементи XP є обов'язковими [5].

Рисунок 2.2 – Життєвий цикл розробки за методологією Scrum

Джерело: [42]

2.1.4. Обґрунтування гібридного підходу (ScrumBan)

Аналіз показує, що жоден із "чистих" фреймворків не покриває всі потреби проєкту «TransactChain».

Чистий Scrum може бути занадто жорстким для етапу активного аудиту безпеки, коли помилки треба виправляти негайно, не чекаючи нового спринту.

Чистий Kanban може призвести до втрати фокусу на стратегічних цілях та "розмивання" дедлайнів релізів.

Тому ми пропонуємо використати гібридну модель (ScrumBan), яка інтегрує найкращі практики обох фреймворків [15]:

1. Структура Scrum: Ми зберігаємо ролі (Product Owner, Scrum Master), артефакти (Backlog) та події (Planning, Review, Retro) для забезпечення прозорості та ритму.

2. Гнучкість Kanban: В середині спринту ми використовуємо візуалізацію на дошці та ліміти WIP для управління потоком задач. Критичні баги (Blockers) можуть додаватися в спринт позачергово (Expedite lane).

3. Практики XP: Впровадження обов'язкового Code Review та автоматизованого тестування (TDD) для забезпечення якості смарт-контрактів [15].

2.2. Специфіка застосування гнучких підходів в умовах розподілених команд та високих ризиків

Впровадження Agile-методологій у розробку FinTech-продуктів, таких як «TransactChain», суттєво відрізняється від класичної розробки веб-сайтів чи мобільних додатків. Специфіка індустрії Web3 диктує дві фундаментальні умови, які визначають вибір управлінського інструментарію: глобальна розподіленість команд (Remote-first culture) та незворотність транзакцій, що підносить ціну помилки до критичного рівня [31].

2.2.1. Управління розподіленими (Distributed) Agile-командами

Сучасний ринок блокчейн-розробки не має кордонів. Команда проєкту «TransactChain» потенційно складатиметься з фахівців, що знаходяться в різних часових поясах: розробники ядра (Core Devs) можуть перебувати у Східній Європі, спеціалісти з безпеки – в Ізраїлі або США, а менеджери та комплаєнс-офіцери – у Західній Європі (через регуляцію MiCA). Така географія створює серйозні виклики для класичного Scrum, який історично створювався для колокованих команд (collocated teams), що сидять в одній кімнаті [29].

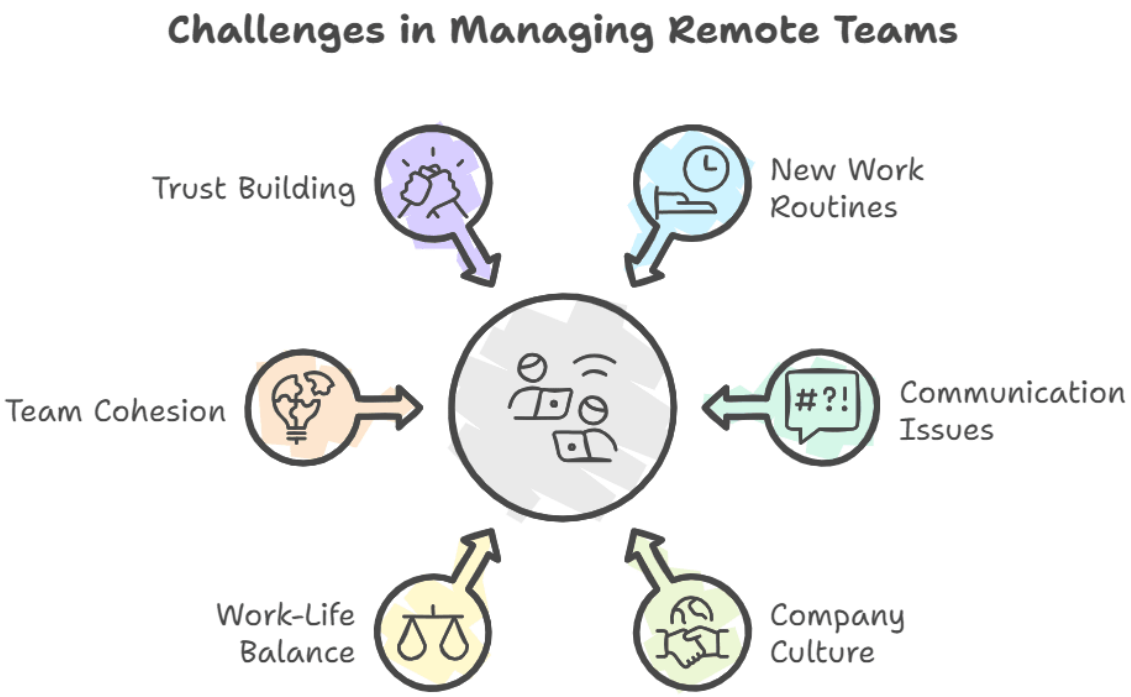

Рисунок 2.3 – Ключові виклики управління розподіленими командами

Джерело: [43]

Для нівелювання цих викликів та забезпечення ефективної співпраці, ми пропонуємо адаптувати Agile-церемонії та комунікаційні процеси наступним чином:

1. Перехід до асинхронної комунікації (Async-first) У розподіленій команді спроба зібрати всіх на онлайн-мітинг часто призводить до блокування роботи частини команди (хтось вже спить, хтось ще не прокинувся) та виникнення феномену "Zoom fatigue" (втоми від відеодзвінків). Тому пріоритет надається текстовій комунікації.

– Daily Scrum: Трансформується з усного мітингу в автоматизований звіт у Slack/Discord боті (наприклад, Geekbot). Кожен член команди відповідає на три стандартні питання Scrum у зручний для нього час до 11:00 за своїм часовим поясом. Це створює прозорий лог активності без переривання потоку роботи (Flow).

– Прийняття рішень: Використовується практика RFC (Request for Comments). Ідея описується в документі (Notion/Confluence), команда має 24-

48 годин на коментування. Рішення приймається після обробки всіх коментарів, а не на емоціях під час відеодзвінка.

2. Правило "Золотих годин" (Golden Hours) Для збереження відчуття єдиної команди необхідно визначити інтервал перетину (overlap) тривалістю 2-3 години, коли всі учасники доступні онлайн. Цей час використовується виключно для синхронних активностей: Sprint Planning, Retrospective, парного програмування або вирішення блокерів [29].

3. Крос-культурна комунікація В інтернаціональних командах важливо враховувати різницю в комунікаційних стилях. Наприклад, розробники з країн "низького контексту" (США, Німеччина) схильні говорити прямо, тоді як колеги з країн "високого контексту" (Азія, Південна Європа) можуть уникати прямої критики. Scrum Master має виступати фасилітатором, створюючи безпечне середовище для висловлювання думок [31].

4. Інструментальна підтримка розподіленої роботи Для забезпечення прозорості процесів (Transparency) та можливості інспекції (Inspection) – стовпів скраму – необхідно використовувати спеціалізований стек інструментів (табл. 2.2).

Таблиця 2.2 – Аналіз інструментів для управління розподіленою блокчейн-командою

Категорія	*Інструмент*	*Функція в «TransactChain»*	*Переваги*	*Недоліки*
Task Tracking	Jira	Візуалізація беклогу та спринтів.	Потужна звітність, інтеграція з GitHub.	Складність налаштування, високий поріг входу.
Documentation	Confluence	Єдине джерело правди (SSOT).	Гнучкість, можливість вбудовувати код та діаграми.	Ризик створення хаосу без чіткої структури.
Communication	Slack	Оперативна комунікація.	Чудово підходить для корпоративної взаємодії.	Постійні сповіщення відволікають від глибокої роботи.

Whiteboarding	Miro	Візуалізація архітектури та ретроспективи.	Заміна фізичної дошки зі стікерами для віддалених брейнштормів.	Може "гальмувати" при великій кількості елементів.

2.2.2. Управління високими ризиками: Zero Failure Culture

Головний парадокс розробки фінансових блокчейн-систем полягає у конфлікті між філософією Agile ("Fail Fast" – помиляйся швидко, вчись на помилках) та вимогами фінансової безпеки ("Zero Failure" – жодних помилок). Якщо у звичайному стартапі баг на продакшені можна виправити хотфіксом (hotfix) за годину, то вразливість у смарт-контракті, який вже задеплоєний у Mainnet і управляє мільйонами доларів ліквідності (TVL), виправити неможливо. Блокчейн пам'ятає все, а код часто є незмінним (immutable) [8].

Для систематизації цього процесу нами розроблено Матрицю управління ризиками Agile-трансформації (табл. 2.3).

Таблиця 2.3 Матриця ризиків та стратегій їх мітигації при Agile-розробці «TransactChain»

Категорія ризику	*Опис загрози*	*Стратегія Agile-мітигації*	*Інструментарій*
Технічні (Smart Contracts Risk)	Критична помилка в логіці контракту (Reentrancy, Overflow, Logic Error), що призводить до втрати коштів.	Shift-Left Testing: Впровадження тестування на якомога ранніх стадіях. Аудит безпеки стає частиною Definition of Done (DoD) для кожного PBI (Product Backlog Item) [17].	MythX, Slither, Hardhat, Testnet deployment, Bug Bounty
Регуляторні (Compliance Risk)	Зміна вимог регулятора (наприклад, оновлення списків санкцій FATF) робить поточну архітектуру нелегальною.	Compliance-as-Code: Залучення юриста як стейкхолдера на Sprint Review. Автоматизація перевірок комплаєнсу (KYT - Know Your Transaction) в CI/CD пайплайні [10].	Chainalysis, Elliptic, автоматизовані оракули

Операційні (Bus Factor)	Ключовий розробник (наприклад, головний криптограф) залишає проєкт, забираючи унікальні знання про архітектуру.	Pair Programming & Code Review: Обов'язкова ротація знань. Жоден рядок коду не потрапляє в майстер-гілку без перевірки мінімум двома іншими розробниками [5].	GitHub Pull Requests, документація в Notion
Кризові (Security Incident)	Злам протоколу або атака на інфраструктуру в реальному часі.	War Room Protocol: Заздалегідь розроблений план дій (Playbook), який дозволяє команді миттєво мобілізуватися, поставити контракт на паузу (Emergency Stop) та випустити патч [26].	Multisig-гаманці (Gnosis Safe), Emergency DAO

2.2.3. Інтеграція DevSecOps та культура психологічної безпеки

Для забезпечення надійності платформи, Agile-процес має бути підсилений інженерною культурою DevSecOps (Development, Security, Operations) (рис 2.4) [17]. Це означає, що безпека не є окремим етапом "після розробки" (як у Waterfall), а інтегрована у кожен крок створення цінності.

У рамках проєкту «TransactChain» пропонується наступний алгоритм DevSecOps [16]:

1. Pre-Commit: Локальні перевірки коду на комп'ютері розробника (Linter, статичний аналіз) перед відправкою в репозиторій. Це дозволяє відсіяти до 40% простих помилок ще до того, як вони потраплять у спільний код.

2. Continuous Integration (Build & Test): При кожному Pull Request (запиті на злиття коду) автоматично запускається набір Unit-тестів та інструментів сканування вразливостей (наприклад, Slither для Solidity). Якщо хоча б один тест не проходить, злиття коду блокується [16].

3. Testnet Deployment: Автоматичне розгортання успішної збірки в тестовій мережі (Goerli або Sepolia). На цьому етапі проводиться імітація реальних транзакцій та навантажувальне тестування.

4. External Audit: Перед великими релізами (Major Release) залучаються зовнішні аудиторські фірми (наприклад, Certik або Hacken). В Agile-процесі це враховується як окремий етап (Hardening Sprint).

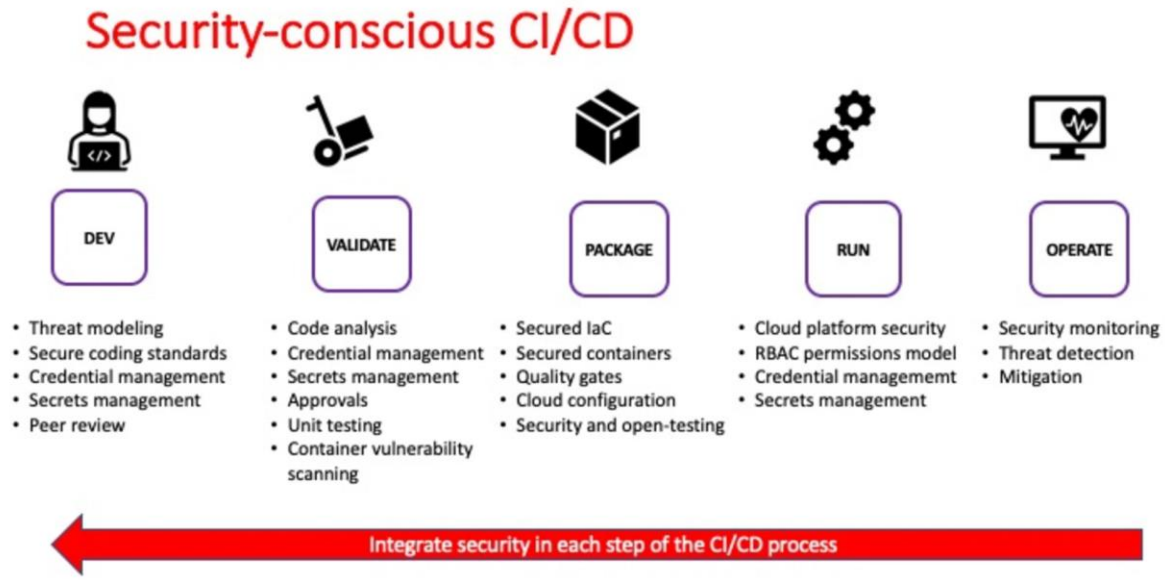

Рисунок 2.4 – Інтеграція етапів безпеки (Security) у безперервний цикл розробки (CI/CD)

Джерело: [44]

Кризовий менеджмент. Протокол "War Room" Навіть найкращі практики не гарантують 100% безпеки. Тому Agile-команда повинна мати чіткий протокол дій на випадок інциденту:

1. Виявлення. Автоматизовані системи моніторингу (наприклад, Tenderly) сигналізують про аномальну активність.

2. Мобілізація. Створення екстреного каналу в Discord ("War Room"), куди додаються всі ключові розробники та стейкхолдери.

3. Локалізація. Використання "аварійного гальма" (Circuit Breaker) у смарт-контракті для зупинки транзакцій.

4. Виправлення. Розробка та тестування патчу в прискореному режимі.

5. Комунікація. Прозоре інформування спільноти про інцидент (Post-Mortem) [26].

Важливою складовою управління ризиками є створення культури психологічної безпеки (Blameless Culture). У фінансових проєктах страх покарання за помилку може змусити розробника приховати потенційну вразливість. Менеджмент «TransactChain» має заохочувати відкрите обговорення помилок через проведення Blameless Post-Mortems (аналіз інцидентів без пошуку винних), фокусуючись на вдосконаленні системи, а не на покаранні особистості [17].

2.3. Інструментарій менеджера проєкту для забезпечення якості та дотримання термінів розробки

Ефективне управління створенням високотехнологічного продукту «TransactChain» неможливе без використання спеціалізованого інструментарію. В умовах Agile роль менеджера проєкту трансформується з "контролера", який роздає завдання, в "фасилітатора" та "лідера-слугу" (Servant Leader), який усуває перешкоди та забезпечує команду необхідними засобами для продуктивної роботи [2]. У цьому підрозділі ми детально розглянемо комплекс інструментів та управлінських практик, необхідних для дотримання балансу між швидкістю розробки (Speed), якістю (Quality) та обсягом (Scope).

2.3.1. Система управління завданнями та пріоритезацією (Task Management)

Основним інструментом управління потоком робіт для проєкту обрано Jira Software. Для реалізації гібридної моделі ScrumBan, описаної в пункті 2.1, необхідно налаштувати середовище наступним чином:

1. Налаштування Workflow (Робочого процесу) Стандартний процес "To Do -> In Progress -> Done" є недостатнім для блокчейн-проєкту [20]. Ми пропонуємо розширений Workflow, який гарантує якість на кожному етапі:

– To Do: Завдання сформульоване, але ще не взяте в роботу;

– In Progress: Розробник працює над кодом;

– Code Review: Створено Pull Request, код перевіряється іншими розробниками;

– In QA (Testnet): Код розгорнуто в тестовій мережі, тестувальник перевіряє функціонал;

– Security Audit: (Тільки для смарт-контрактів) Код перевіряється на вразливості;

– Done: Код влито в основну гілку (main) і він готовий до релізу.

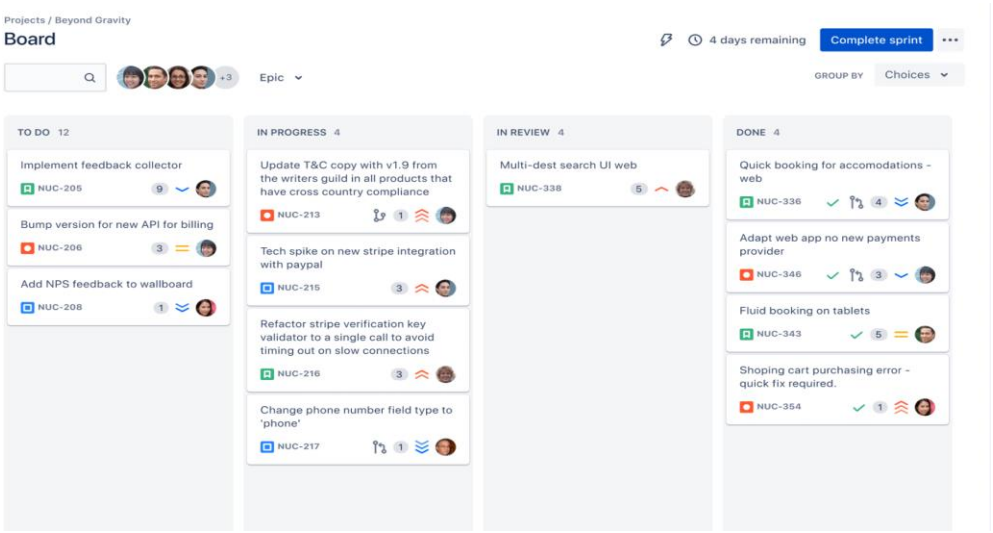

Рисунок 2.5 – Приклад налаштування Scrum-дошки в Jira для блокчейн-проєкту

Джерело: [45]

2. Структура Беклогу (Backlog Structure) Для уникнення хаосу, беклог структурується за ієрархією [19]:

– Epics (Епіки): Великі функціональні блоки (наприклад, "Модуль KYC", "Інтеграція гаманця"). Епіки можуть охоплювати кілька спринтів.

– User Stories (Історії користувача): Опис функціоналу з точки зору користувача. Формат: "Як [роль], я хочу [дія], щоб [цінність]".

– Tasks (Технічні завдання): Завдання, які не несуть прямої цінності для користувача, але необхідні для системи (наприклад, "Налаштування CI/CD").

– Bugs (Баги): Помилки, знайдені тестувальниками або користувачами.

3. Техніки пріоритезації Враховуючи специфіку DeFi, де нові вразливості виявляються щодня, критично важливою є правильна пріоритезація. Ми пропонуємо використовувати метод MoSCoW для категоризації завдань кожного спринту [33]:

– Must have (Критично): Функціонал ядра (Core Smart Contracts), без якого платформа не працює (наприклад, функція transfer() або approve());

– Should have (Важливо): Інтеграції з популярними гаманцями (Metamask, Ledger), які є важливими для успіху, але можуть зачекати один спринт;

– Could have (Бажано): Покращення UX/UI, темна тема інтерфейсу, додаткові мови;

– Won't have (Не зараз): Фічі, які ми свідомо відкладаємо (наприклад, підтримка NFT на першому етапі).

Для оцінки складності завдань (Estimation) у розподіленій команді використовується техніка Planning Poker. Це дозволяє уникнути когнітивного упередження "авторитету" (коли всі погоджуються з думкою тімліда) та отримати реалістичну оцінку в Story Points, а не в годинах, що є більш точним для інтелектуальної праці [22].

2.3.2. Інструменти контролю версій та якості коду (VCS & QA)

Для блокчейн-проєкту принцип "Code is Law" (Код є законом) є фундаментальним. Помилка в коді смарт-контракту може призвести до безповоротної втрати активів. Тому менеджер повинен забезпечити суворий контроль за тим, що потрапляє в репозиторій.

1. Стратегія розгалуження GitFlow Використання GitHub або GitLab організовується за суворими правилами [16]:

– гілка main (або master) містить виключно стабільний код, який пройшов аудит і готовий до деплою в Mainnet;

– гілка develop є основною робочою версією, куди зливаються нові фічі;

– кожна нова функція або виправлення розробляється в окремій короткоживучій гілці feature/xxx або bugfix/xxx;

– Branch Protection Rules: Менеджер налаштовує технічні обмеження, які забороняють пряме внесення змін (push) в гілки main та develop. Злиття (Merge) можливе лише через Pull Request (PR) після отримання схвалення (Approve) від мінімум двох Senior-розробників та успішного проходження всіх автотестів.

2. Автоматизований контроль якості (Quality Gates) Менеджер не повинен перевіряти код вручну, але він налаштовує інструменти (наприклад, SonarQube), які автоматично аналізують код при кожному коміті. Встановлюються "Ворота якості" (Quality Gates), без проходження яких задача не може рухатися далі по дошці Jira [17]:

– покриття коду тестами (Code Coverage) > 95% для смарт-контрактів;

– відсутність критичних вразливостей (за звітами сканерів Slither/MythX);

– відсутність "запахів коду" (Code Smells) та дублювання.

2.3.3. Метрики ефективності та моніторинг прогресу (KPIs)

Для прийняття обґрунтованих рішень (Data-Driven Management) менеджер використовує набір метрик, адаптованих для Agile-розробки (табл. 2.4).

Таблиця 2.4 - Ключові метрики ефективності розробки платформи «TransactChain»

Метрика	*Що вимірює*	*Значення для менеджера*	*Інструмент*
Velocity (Швидкість)	Кількість Story Points, завершених командою за один спринт [22].	Дозволяє прогнозувати, скільки роботи команда може виконати в майбутньому. Необхідна для планування дати релізу.	Jira Burndown Chart
Cycle Time (Час циклу)	Час, що проходить від моменту взяття задачі в роботу ("In Progress") до її завершення ("Done").	Індикатор ефективності процесів. Високий Cycle Time сигналізує про наявність "вузьких місць" (наприклад, довге очікування Code Review) [3].	Jira Control Chart
Lead Time	Час від створення запиту (ідеї) до його реалізації.	Показує, як швидко компанія реагує на потреби ринку (Time-to-Market).	Jira Control Chart
Defect Escape Rate	Кількість багів, знайдених користувачами після релізу, відносно загальної кількості багів.	Ключовий показник якості. Для фінансової системи має прагнути до нуля.	GitHub Issues
Gas Usage Efficiency	Вартість виконання транзакцій смарт-контракту.	Економічна ефективність продукту. Якщо контракт "спалює" забагато газу, він буде неконкурентним [8].	Hardhat Gas Reporter

2.3.4. Управління знаннями та документацією (Knowledge Management)

У розподілених командах знання часто губляться в приватних чатах. Завдання менеджера – перетворити неявні (tacit) знання в явні (explicit). Для цього використовується Confluence або Notion, організовані за принципом "Живої документації" [26].

Структура бази знань для «TransactChain»:

1. Product Requirements Document (PRD): Опис бізнес-логіки, user flow та критеріїв приймання.

2. Technical Architecture: Діаграми архітектури, опис математичних моделей та алгоритмів консенсусу.

3. API Documentation: Автоматично генерована документація (наприклад, через Swagger) для інтеграції з банківськими системами.

4. Onboarding Guide: Покрокова інструкція для нових розробників (як налаштувати середовище, де взяти ключі доступу). Це дозволяє скоротити час адаптації нового співробітника з 2 тижнів до 2-3 днів.

5. Post-Mortems: Архів звітів про інциденти ("розбір польотів"). Це дозволяє команді вчитися на власному досвіді і системно усувати причини помилок [17].

2.3.5. Інструменти фасилітації та управління комунікаціями

Менеджер проєкту виступає фасилітатором командних зустрічей. Для розподілених команд важливо використовувати спеціалізовані інструменти, що замінюють фізичну дошку та стікери [31].

– Miro / FigJam: Для проведення мозкових штурмів, Sprint Planning та візуалізації архітектури;

– EasyRetro / Retrium: Для проведення ретроспектив. Використання технік на кшталт "Mad Sad Glad" або "Start Stop Continue" дозволяє структурувати зворотний зв'язок та покращувати процеси від спринту до спринту;

– Geekbot: Для проведення асинхронних стендапів у Slack, що економить час на дзвінках.

РОЗДІЛ 3
РОЗРОБКА АДАПТОВАНОЇ МОДЕЛІ ГНУЧКОГО УПРАВЛІННЯ (AGILE) ПРОЄКТОМ «TRANSACTCHAIN»

3.1. Побудова гібридної моделі управління процесом розробки платформи (адаптація Scrum/Kanban)

Аналіз, проведений у попередніх розділах, виявив фундаментальне протиріччя проєкту «TransactChain»: необхідність швидкої розробки та адаптації до змін ринку (властива стартапам) вступає в конфлікт з вимогою абсолютної надійності та відповідності регуляторним нормам (властивою банківським системам). Стандартні фреймворки, такі як Scrum, фокусуються на швидкості постачання цінності, часто нехтуючи вбудованими механізмами контролю фінансових ризиків. З іншого боку, каскадні моделі (Waterfall) є надто повільними для динамічного ринку DeFi [28].

Для вирішення цієї дилеми нами розроблено авторську Адаптивну модель управління "Compliance-Driven Agile" (CDA). Суть моделі полягає в інтеграції процедур безпеки (Security) та комплаєнсу (Compliance) безпосередньо в тіло ітеративного процесу розробки, а не винесення їх в окремий етап "після розробки". Це дозволяє виявляти регуляторні та технічні ризики на ранніх стадіях, коли вартість їх виправлення є мінімальною [15].

3.1.1. Концептуальна архітектура моделі

Основою запропонованої моделі є модифікований життєвий цикл спринту, в якому паралельно існують три потоки створення цінності:

1. Feature Stream (Потік фіч): Розробка бізнес-функціоналу (транзакції, гаманці, інтерфейси).
2. Enabler Stream (Потік забезпечення): Технічні завдання, що створюють фундамент для майбутніх фіч (архітектура, CI/CD, рефакторинг).
3. Regulatory Stream (Потік регулювання): Задачі з аудиту, перевірки відповідності нормам MiCA/GDPR та оновлення юридичної документації [9].

Нижче наведено схему розробленої моделі (рис. 3.1).

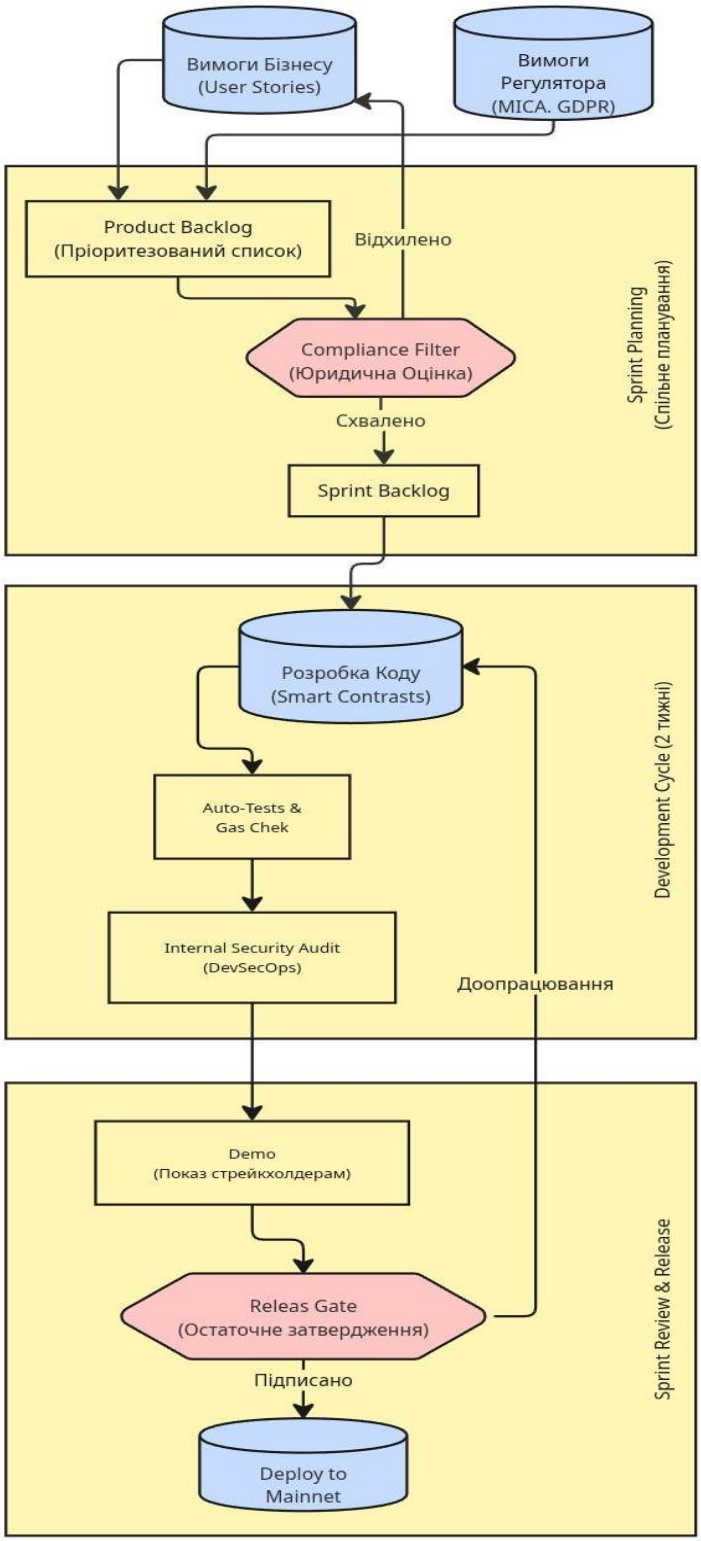

Рисунок 3.1 – Концептуальна схема адаптованої моделі управління "Compliance-Driven Agile"

Джерело: розроблено автором

3.1.2. Адаптація ролей та відповідальності (RACI Matrix)

У класичному Scrum існують лише три ролі: Product Owner, Scrum Master та Team [2]. Для проєкту «TransactChain», де помилка може коштувати мільйони доларів, ця структура є недостатньою через розмивання відповідальності за безпеку. Ми пропонуємо розширити матрицю ролей, інтегрувавши спеціалізовані функції (табл. 3.1).

Таблиця 3.1 Адаптація ролей Scrum-команди для проєкту «TransactChain»

Роль	*Стандартна функція (Scrum)*	*Адаптована функція в моделі «TransactChain»*	*Ключові обов'язки*
Product Owner (PO)	Максимізація цінності продукту, пріоритезація беклогу.	Відповідає за баланс між бізнес-цілями та регуляторними обмеженнями.	Узгодження вимог з банками-партнерами; управління Regulatory Log.
Scrum Master	Фасилітатор, усуває перешкоди.	"Process Guardian": Слідкує за тим, щоб команда не пропускала етапи перевірки безпеки заради швидкості.	Захист Definition of Done; організація Security Freeze.
Development Team	Крос-функціональна команда розробників.	Включає спеціалізованих Blockchain Engineers та Cryptographers.	Розробка смарт-контрактів; Gas Optimization; написання Unit-тестів.
Compliance Officer (Нова роль)	*Відсутня*	Інтегрований учасник команди (не зовнішній аудитор). Має право "вето".	Участь у Sprint Planning; перевірка відповідності User Stories нормам AML/KYC [10].
Security Lead (Нова роль)	*Відсутня*	Відповідає за DevSecOps пайплайн та безпеку архітектури.	Проведення внутрішнього аудиту смарт-контрактів; управління Bug Bounty програмою.

Для чіткого розподілу повноважень розроблено матрицю RACI (Responsible, Accountable, Consulted, Informed) для ключових процесів [20]:

– затвердження релізу: Accountable – Product Owner; Responsible – Security Lead; Consulted – Compliance Officer;

– зміна смарт-контракту: Accountable – Tech Lead; Responsible – Developer; Consulted – Security Lead.

3.1.3. Модифікація артефактів управління

Для забезпечення прозорості, аудируємості та відповідності стандартам ISO 20022 [18], стандартний набір артефактів Scrum (Backlog, Increment) доповнюється специфічними документами:

1. Integration Backlog (Інтеграційний беклог) Оскільки «TransactChain» має взаємодіяти із застарілими (legacy) банківськими системами, ми виділяємо технічні завдання з інтеграції API в окремий потік. Це дозволяє команді бекенду працювати у своєму ритмі, не блокуючи розробку смарт-контрактів, яка вимагає більше часу на тестування [27].

2. Regulatory Log (Журнал відповідності) Це "живий" документ, у якому кожна User Story (користувацька історія) лінкується з конкретною статтею регуляторного акту.

Приклад: "User Story: Реєстрація користувача" <-> "Regulatory Requirement: MiCA Art. 14 (KYC procedures)". Це забезпечує Traceability (простежуваність) вимог: ми завжди можемо довести аудитору, чому система працює саме так [1].

3. Розширений Definition of Done (DoD) Критерій готовності – це головний запобіжник якості. У нашій моделі задача не може вважатися зробленою ("Done"), якщо вона не пройшла спеціальні перевірки [22].

Приклад DoD для смарт-контракту:

– код написано та прокоментовано (NatSpec формат для документації);

– Unit-тести покривають >95% коду (Code Coverage);

– пройдено автоматичний аналіз вразливостей (Slither/MythX);

– Gas Report: Вартість виконання транзакції не перевищує встановлений ліміт;

– Legal Sign-off: Compliance Officer підтвердив відповідність логіки контракту нормам AML;

– Security Sign-off: Security Lead провів Code Review і не знайшов критичних вразливостей.

3.1.4. Процесна інновація: "Security Freeze" та "Hardening Sprint"

Для мітигації ризиків, описаних у Розділі 2, ми впроваджуємо два специфічних етапи в життєвий цикл розробки:

1. Security Freeze (Заморожування безпеки) Впроваджується в кінці кожного стандартного спринту (за 2 дні до завершення).

Суть: Розробка нового коду повністю зупиняється.

Діяльність: Команда займається виключно спробами "зламати" власний код (Internal Hackathon), написанням документації та виправленням знайдених багів [5].

Результат: Це знижує вірогідність потрапляння критичних багів у Mainnet, оскільки увага переключається з "написання" (creative mode) на "перевірку" (critical mode).

2. Hardening Sprint (Спринт стабілізації) Проводиться перед кожним великим релізом (Major Release), зазвичай раз на 3-4 звичайних спринти.

Суть: Протягом цього спринту нові фічі не розробляються.

Діяльність: Проведення зовнішнього аудиту смарт-контрактів (залучення фірм типу Certik/Hacken), масштабне навантажувальне тестування, навчання персоналу підтримки.

3.2. Планування життєвого циклу реалізації проєкту: дорожня карта (Roadmap), команда та комунікації

Після визначення методології та архітектури процесів (Розділ 3.1), наступним кроком є детальне планування реалізації проєкту «TransactChain». Враховуючи обрану гібридну модель "Compliance-Driven Agile", ми відмовляємося від жорсткого річного планування на користь стратегії "хвильового планування" (Rolling Wave Planning) [28]. Це означає, що детально плануються лише найближчі ітерації (спринти), тоді як довгострокові цілі окреслюються на рівні етапів дорожньої карти (Roadmap).

3.2.1. Дорожня карта проєкту (Product Roadmap)

Життєвий цикл створення платформи розбито на чотири ключові фази, загальною тривалістю 12 місяців до виходу на ринок (Go-to-Market). Для зручності сприйняття та деталізації план реалізації розділено на два півріччя.

Перше півріччя: Розробка MVP (рис 3.2а)

Фаза 1: Inception (Ініціація) – Місяці 1-2. Мета: формування ядра команди, вибір технологічного стеку, розробка Proof of Concept (PoC). Ключові події: підписання угод з партнерами, затвердження юридичної концепції (Legal Opinion) [12].

Фаза 2: Core Development (Розробка ядра) – Місяці 3-6. Мета: створення MVP (Minimum Viable Product), що включає базові смарт-контракти та API шлюз. Робота ведеться у закритому контурі (DevNet).

Нижче наведено графік робіт для першого етапу (рис. 3.2а).

Друге півріччя: Безпека та Запуск (рис 3.2б)

Фаза 3: Security & Compliance Hardening (Стабілізація) – Місяці 7-9. Мета: підготовка до запуску в Mainnet. Активності: зовнішні аудити безпеки, Bug Bounty програма, інтеграційне тестування з банками-партнерами.

Фаза 4: Launch & Scale (Запуск) – Місяці 10-12. Мета: розгортання в основній мережі, онбординг перших клієнтів.

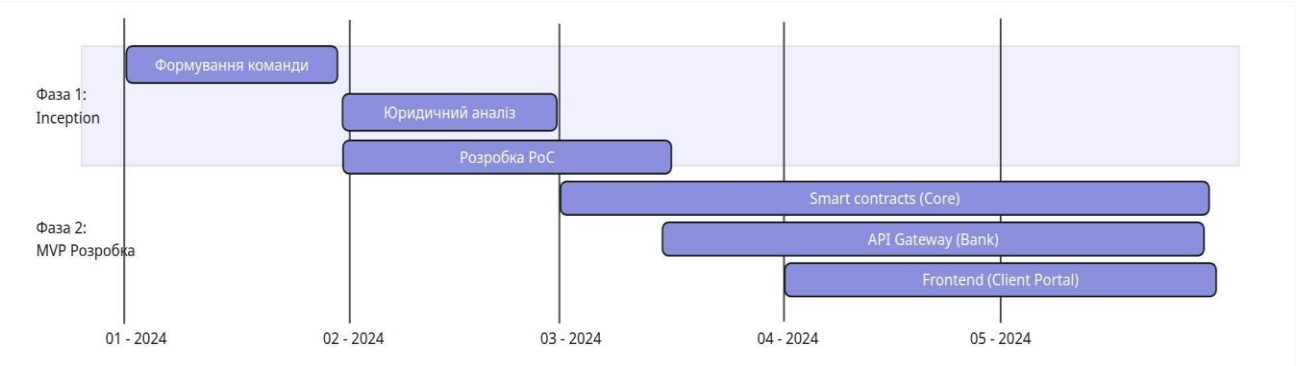

Рисунок 3.2а – Календарний план-графік: Старт та MVP

Джерело: розроблено автором

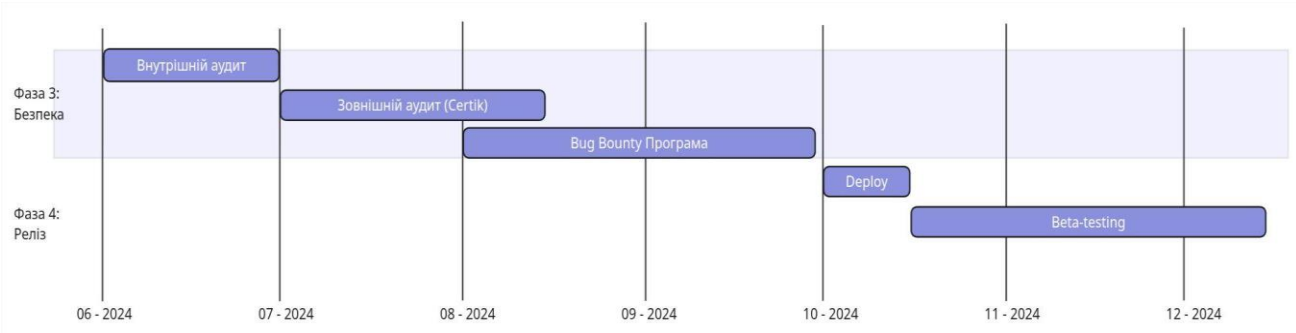

Рисунок 3.2б – Календарний план-графік: Аудит та Реліз

Джерело: розроблено автором

3.2.2. Формування та розвиток команди

Для реалізації проєкту необхідна крос-функціональна команда. Відповідно до моделі, описаної в п. 3.1, штатний розпис включає наступні ролі (табл. 3.2).

Таблиця 3.2 – Ресурсний план команди проєкту

Роль	*Кількість*	*Seniority Level*	*Ключові компетенції*
Product Owner	1	Senior	Fintech domain, Agile CSPO, Business Analysis [2].
Scrum Master	1	Middle/Senior	Фасилітація, JIRA Admin, конфлікт-менеджмент.
Blockchain Devs	3	Senior	Solidity/Rust, EVM, Cryptography standards [8].

Роль	*Кількість*	*Seniority Level*	*Ключові компетенції*
Product Owner	1	Senior	Fintech domain, Agile CSPO, Business Analysis [2].
Backend Devs	2	Middle	Node.js/Go, Microservices, API Security.
Compliance Officer	1	Senior	AML/KYC regulations, GDPR, Corporate Law [9].
Security Lead	1	Senior	White-hat hacking, Smart contract audit tools [17].
QA Engineer	2	Middle	Automated testing, Testnet interactions.

Процес онбордингу (Onboarding Strategy) Враховуючи складність домену, період адаптації нового співробітника може сягати 2 місяців. Для скорочення цього часу до 2 тижнів ми впроваджуємо систему "Buddy System": за кожним новачком закріплюється ментор, а доступ до знань забезпечується через структуровану базу в Confluence (як описано в п. 2.3).

3.2.3. План управління комунікаціями

У розподіленій команді комунікація є "кровоносною системою" проєкту. Хаотичне спілкування призводить до десинхронізації, тоді як надмірна бюрократія вбиває ініціативу. Мною розроблено Матрицю комунікацій, яка регламентує інформаційні потоки (табл. 3.3).

Таблиця 3.3 – Матриця комунікацій проєкту «TransactChain»

Подія / Артефакт	*Учасники*	*Частота*	*Канал*	*Мета*
Sprint Planning	Вся команда	Раз на 2 тижні	Zoom/Teams	Визначення цілей спринту та оцінка задач [2].
Daily Stand-up	Розробники, SM	Щодня (до 11:00)	Slack Bot (Async)	Синхронізація статусу, виявлення блокерів [29].
Compliance Sync	PO, Compliance Officer	Щотижня	Офлайн/Zoom	Перегляд змін у регуляторному полі.

Security Review	Security Lead, Tech Lead	За запитом (Merge)	GitHub	Перевірка коду перед злиттям.
Sprint Review (Demo)	Вся команда + Стейкхолдери	Раз на 2 тижні	Zoom	Демонстрація працюючого інкременту [22].
Stakeholder Report	PO, Інвестори	Щомісяця	Email / PDF	Звіт про прогрес, бюджет та ризики.

3.2.4. Управління стейкхолдерами (Stakeholder Management)

Успіх проєкту залежить не лише від команди, а й від задоволення потреб зовнішніх зацікавлених сторін. Для «TransactChain» ключовими стейкхолдерами є:

1. Банки-партнери: Очікують надійності та інтеграції. Стратегія: залучення їх технічних спеціалістів до демо-показів API.
2. Регулятори: Очікують прозорості. Стратегія: проактивна підготовка звітів та відкритість архітектури для аудиту [10].
3. Інвестори: Очікують дотримання Roadmap. Стратегія: чесна комунікація про затримки та управління очікуваннями.

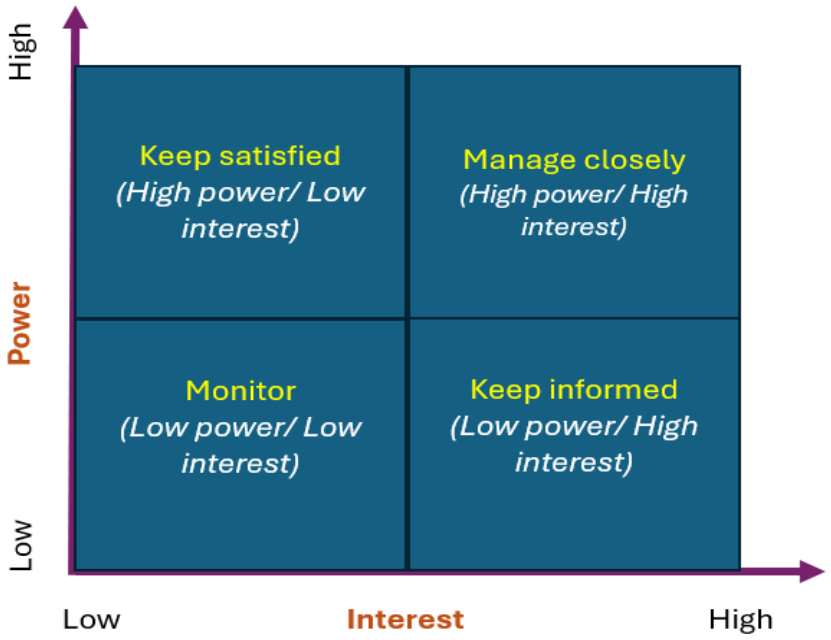

Рисунок 3.3 – Матриця стейкхолдерів (Вплив / Інтерес)

Джерело: [46]

3.3. Оцінка ефективності запропонованої моделі та управління ризиками проєкту

Впровадження адаптованої моделі "Compliance-Driven Agile" для створення платформи «TransactChain» спрямоване на досягнення балансу між швидкістю розробки та надійністю продукту. Для обґрунтування доцільності запропонованих управлінських рішень нами проведено комплексну оцінку ефективності, що включає стратегічний, операційний та економічний аспекти [32].

3.3.1. SWOT-аналіз впровадження моделі управління

Для стратегічної оцінки запропонованої методології використано метод SWOT-аналізу, який дозволяє виявити внутрішні сильні та слабкі сторони моделі, а також зовнішні можливості та загрози (табл. 3.4).

Таблиця 3.4 – SWOT-аналіз моделі управління "Compliance-Driven Agile"

Сильні сторони (Strengths)	*Слабкі сторони (Weaknesses)*
1. Early Risk Detection: Інтеграція аудиту безпеки в спринт дозволяє виявляти вразливості до деплою. 2. Transparency: Повна прозорість процесу для стейкхолдерів завдяки Regulatory Log. 3. Speed: Збереження ритму розробки (2-тижневі спринти) попри складність домену. 4. Compliance: Гарантія відповідності нормам MiCA "з коробки" [9].	1. Висока вартість команди: Потреба в дорогих спеціалістах (Security Lead, Compliance Officer). 2. Складність онбордингу: Високий поріг входу для нових розробників через специфічні вимоги до безпеки. 3. Уповільнення через бюрократію: Ризик затримок на етапі Compliance Filter.
Можливості (Opportunities)	*Загрози (Threats)*
1. Scalability: Модель легко масштабується на інші FinTech-продукти компанії. 2. Market Leadership: Швидший вихід на ринок (Time-to-Market) порівняно з конкурентами, що використовують Waterfall. 3. Trust: Підвищення довіри інституційних клієнтів завдяки прозорому аудиту [24].	1. Regulatory Changes: Раптові зміни законодавства можуть вимагати перебудови процесів. 2. Talent Shortage: Дефіцит кваліфікованих блокчейн-розробників та аудиторів на ринку. 3. Team Burnout: Високий темп спринтів у поєднанні з жорсткими вимогами якості може призвести до вигорання.

3.3.2. Прогнозна оцінка ключових показників ефективності (KPI)

Ефективність нової моделі управління вимірюється через порівняння з традиційними підходами (Waterfall або хаотичний Agile). Нами визначено ключові метрики (KPI), які демонструють переваги моделі (табл. 3.5).

Таблиця 3.5 Прогнозні показники ефективності впровадження моделі

Показник	*Традиційна модель (Benchmark)*	*Адаптована модель (Прогноз)*	*Ефект*
Time-to-Market (MVP)	9-12 місяців	6 місяців	Скорочення на 30-50% завдяки паралельним процесам [28].
Defect Escape Rate	15% (багів на проді)	< 2%	Зниження на 85% завдяки Security Freeze та CI/CD.
Cost of Change	Висока (переписування коду після аудиту в кінці)	Низька (виправлення під час спринту)	Економія до 40% бюджету розробки [22].
Regulatory Compliance	Перевірка перед релізом (ризик переробки)	Continuous Compliance	Усунення ризику повної переробки архітектури.

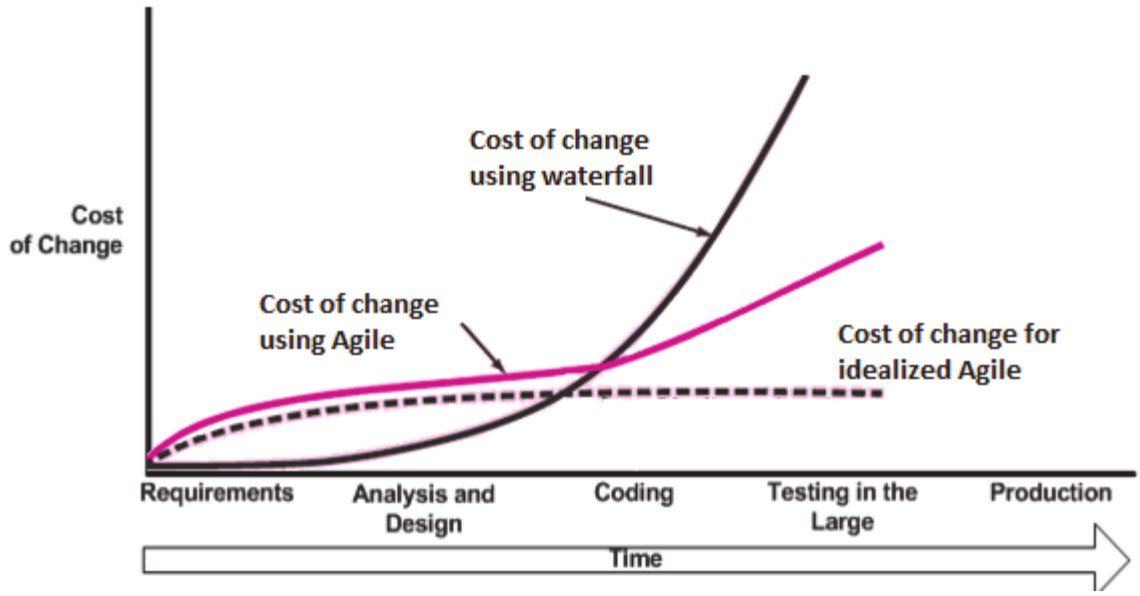

Рисунок 3.4 – Залежність вартості виправлення помилки від етапу її виявлення (Cost of Change Curve)

Джерело: [47]

Як видно з графіка (рис. 3.4), вартість виправлення помилки зростає експоненціально. Наша модель, яка зміщує тестування "вліво" (Shift-Left Testing), дозволяє виправляти помилки на етапі дизайну або кодування, що є в 10-100 разів дешевше, ніж виправлення після релізу.

3.3.3. Реєстр ризиків та план реагування

Управління ризиками є невід'ємною частиною запропонованої моделі. Нами розроблено реєстр топ-5 ризиків проєкту з оцінкою їх ймовірності та стратегією мітигації (табл. 3.6).

Таблиця 3.6 Реєстр ризиків проєкту «TransactChain»

ID	*Опис ризику*	*Ймовірність (1-5)*	*Вплив (1-5)*	*RPN (Risk Priority Number)*	*Стратегія реагування*
R1	Вразливість смарт-контракту: Злом протоколу хакерами.	3	5	15 (Critical)	Внутрішній аудит + Зовнішній аудит (Certik) + Bug Bounty програма [17].
R2	Зміна регуляцій: Нові вимоги MiCA роблять бізнес-модель нелегальною.	4	4	16 (Critical)	Залучення Compliance Officer на етапі планування. Гнучка архітектура [9].
R3	Технічний борг: Накопичення "швидких рішень" гальмує розвиток.	4	3	12 (High)	Виділення 20% ємності кожного спринту на рефакторинг (Enablers).
R4	Втрата ключових кадрів: Lead Developer йде з проєкту.	3	4	12 (High)	Детальна документація в Confluence, парне програмування (Knowledge sharing) [5].
R5	Затримка інтеграції: Банк-партнер не надає доступ до API вчасно.	5	2	10 (Medium)	Використання Mock-серверів для емуляції API, паралельна розробка.

3.3.4. Економічне обґрунтування

Впровадження моделі вимагає додаткових інвестицій у команду (Compliance Officer, Security Lead) та інструменти (аудити). Проте, розрахунок ROI (Return on Investment) показує, що ці витрати окупаються за рахунок уникнення катастрофічних збитків від зламів та штрафів.

Якщо середня вартість злому DeFi-протоколу становить $10-50 млн, а штраф за порушення GDPR/AML – до €20 млн або 4% обігу, то інвестиція в $200-300 тис. на рік у додаткові заходи безпеки та управління є економічно виправданою "страховкою" для бізнесу.

ВИСНОВКИ

У магістерській роботі вирішено актуальне науково-прикладне завдання підвищення ефективності процесу створення платформи «TransactChain» для фінансових установ шляхом розробки та обґрунтування адаптованої моделі гнучкого управління (Agile). Результати проведеного дослідження дозволяють зробити наступні узагальнюючі висновки та пропозиції:

1. Аналіз ринкового середовища та проблематики. Проведений у першому розділі аналіз сучасного стану ринку фінансових технологій засвідчив, що глобальна фінансова система перебуває на етапі фундаментальної трансформації. Ключовим трендом є перехід від традиційних централізованих банківських систем (Legacy Systems) до децентралізованих рішень на базі технології розподіленого реєстру (DLT). Встановлено, що існуюча інфраструктура (зокрема SWIFT) не відповідає вимогам цифрової економіки через низьку швидкість транскордонних платежів (3-5 днів) та високу вартість транзакцій. Водночас, фінансовий сектор стикається з безпрецедентним зростанням ризиків шахрайства, зокрема махінацій з фінансовою звітністю, що вимагає впровадження інструментів незмінного аудиторського сліду (Audit Trail), який забезпечує блокчейн [1].

2. Результати конкурентного аналізу. Здійснено глибокий порівняльний аналіз існуючих аналогів – мережі RippleNet та протоколу Aave Arc. Виявлено їхні системні обмеження:

– RippleNet, будучи лідером корпоративного сегменту, страждає від надмірної централізації управління та повільного циклу впровадження змін, що є характерним для каскадних моделей розробки.

– Aave Arc, як представник інституційного DeFi, несе високі технічні ризики через залежність від публічного блокчейну Ethereum та складність інтеграції з банківськими стандартами. На основі цього аналізу обґрунтовано необхідність створення платформи «TransactChain» як

гібридного рішення, що поєднує швидкість та інноваційність DeFi з надійністю, керованістю та комплаєнсом традиційних банківських систем.

3. Формування вимог до продукту. Визначено, що критичним фактором успіху платформи є реалізація принципу Compliance-by-Design. Сформульовано функціональні вимоги, які включають інтеграцію автоматизованих процедур AML/KYC безпосередньо у смарт-контракти та забезпечення відповідності нормам MiCA (Markets in Crypto-Assets) та GDPR. Запропоновано архітектурне рішення, яке передбачає роздільне зберігання даних: персональні дані клієнтів зберігаються "офф-чейн" у захищених базах даних, а в блокчейн записуються лише їхні криптографічні хеші, що вирішує конфлікт між вимогою "права на забуття" та незмінністю блокчейну.

4. Обґрунтування управлінської методології. Доведено, що використання традиційної каскадної моделі (Waterfall) для розробки інноваційного FinTech-продукту в умовах високої невизначеності є неефективним та ризикованим. Жорстке планування не дозволяє оперативно реагувати на зміни регуляторного середовища та технологічні інновації. Як оптимальний підхід обрано методологію Agile, яка забезпечує ітеративність розробки, постійний зворотний зв'язок від стейкхолдерів та можливість швидкої адаптації продукту. Проте, виявлено, що "чистий" Scrum [2] не містить достатніх механізмів контролю якості для фінансових систем критичної важливості.

5. Розробка адаптованої моделі управління. Головним науково-практичним результатом роботи є розробка авторської адаптованої моделі управління "Compliance-Driven Agile" (CDA). Запропонована модель є гібридом, що поєднує:

– Scrum [2] – для ритмічної розробки ядра продукту (Core Development);

– Kanban – для управління потоком задач з безпеки та підтримки (Security & Support);

– XP (Extreme Programming) – для забезпечення якості коду через парне програмування та TDD. Ключовою інновацією моделі є інтеграція етапів Compliance Filter (на вході у спринт) та Security Freeze (перед релізом) безпосередньо у життєвий цикл розробки. Також введено нові ролі у Scrum-команду: Compliance Officer та Security Lead, які мають право блокувати реліз у разі невідповідності стандартам.

6. Планування реалізації та комунікацій. Розроблено детальну дорожню карту (Roadmap) реалізації проєкту тривалістю 12 місяців, яка включає чотири фази: Ініціація, Розробка MVP, Стабілізація (Hardening) та Запуск. Запропоновано план управління комунікаціями для розподіленої команди, що базується на принципах асинхронної взаємодії та використанні сучасних інструментів (Jira, Slack, Confluence). Це дозволяє ефективно координувати роботу фахівців, що знаходяться в різних часових поясах, мінімізуючи комунікаційні затримки.

7. Оцінка ефективності та ризиків. Проведено комплексну оцінку ефективності запропонованої моделі:

– Стратегічна ефективність: SWOT-аналіз підтвердив, що модель дозволяє мінімізувати регуляторні ризики та підвищити довіру інституційних клієнтів.

– операційна ефективність. Розрахунок прогнозних KPI показує можливість скорочення Time-to-Market на 30-50% порівняно з традиційними підходами.

– якість. Очікується зниження кількості критичних дефектів у релізі на 85% завдяки зміщенню тестування на ранні етапи (Shift-Left Testing) та впровадженню практики Security Freeze. Розроблено реєстр ризиків та стратегії їх мітигації, що включають технічні (вразливості смарт-контрактів), регуляторні (зміни законодавства) та операційні аспекти.

На основі отриманих результатів, для успішного впровадження платформи «TransactChain» та аналогічних FinTech-проєктів рекомендується:

1. Впровадити роль Compliance Officer як інтегрованого учасника команди розробки, а не зовнішнього контролера, для забезпечення принципу "Compliance-as-Code".

2. Використовувати гібридну архітектуру зберігання даних для забезпечення відповідності GDPR без втрати переваг блокчейну.

3. Застосовувати практику "Security Freeze" в кінці кожного спринту для проведення внутрішніх хакатонів та аудиту коду перед деплоєм.

4. Інвестувати в автоматизацію тестування (CI/CD) та інструменти статичного аналізу коду смарт-контрактів (Slither, MythX) як обов'язкову частину Definition of Done.

Таким чином, розроблена адаптована модель управління є дієвим інструментом, який дозволяє вирішити головне протиріччя створення сучасних фінансових платформ – конфлікт між необхідністю швидких інновацій та вимогою абсолютної надійності. Її застосування сприятиме успішному виведенню платформи «TransactChain» на ринок та підвищенню конкурентоспроможності українського FinTech-сектору.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мігус І. П. Можливості використання технологій блокчейну для захисту від шахрайства. *Вчені записки Університету «КРОК»*. 2022. № 1(65). С. 84–94. DOI: 10.31732/2663-2209-2022-65-84-94.

2. Швабер К., Сазерленд Д. Посібник зі Скраму. Повний путівник гри у Скрам: правила гри. *Scrum.org*. 2020. URL: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Ukrainian.pdf (дата звернення: 24.11.2025).

3. Андерсон Д. Kanban: альтернативний шлях до Agile. Київ : Фабула, 2019. 304 с.

4. Cohn M. Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional, 2009. 504 p.

5. Beck K. Extreme Programming Explained: Embrace Change. 2nd ed. Addison-Wesley Professional, 2004. 224 p.

6. Ries E. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Currency, 2011. 336 p.

7. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. URL: https://bitcoin.org/bitcoin.pdf (дата звернення: 24.11.2025).

8. Antonopoulos A. M., Wood G. Mastering Ethereum: Building Smart Contracts and DApps. O'Reilly Media, 2018. 424 p.

9. Regulation (EU) 2023/1114 of the European Parliament and of the Council of 31 May 2023 on markets in crypto-assets (MiCA). *Official Journal of the European Union*. 2023. L 150. P. 40–205.

10. FATF. Updated Guidance for a Risk-Based Approach to Virtual Assets and Virtual Asset Service Providers. Paris : FATF, 2021. 109 p.

11. General Data Protection Regulation (GDPR). Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016. *Official Journal of the European Union*. 2016. L 119. P. 1–88.

12. Report to the Nations: 2024 Global Study on Occupational Fraud and Abuse. *Association of Certified Fraud Examiners (ACFE)*. 2024. URL: https://www.acfe.com/report-to-the-nations/2024/ (дата звернення: 26.11.2025).

13. Ripple. The Future of Cross-Border Payments. *Ripple Insights*. 2023. URL: https://ripple.com/insights/future-cross-border-payments/ (дата звернення: 26.11.2025).

14. Aave. Aave Arc: Permissioned DeFi for Institutions. *Aave Governance Forum*. 2022. URL: https://governance.aave.com/t/arc-permissioned-defi/ (дата звернення: 28.11.2025).

15. Kniberg H. Scrum and XP from the Trenches. InfoQ, 2015. 140 p.

16. Humble J., Farley D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley Professional, 2010. 512 p.

17. Kim G., Debois P., Willis J., Humble J. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution Press, 2016. 480 p.

18. ISO/IEC/IEEE 12207:2017. Systems and software engineering – Software life cycle processes. *International Organization for Standardization*. 2017.

19. Project Management Institute. A Guide to the Project Management Body of Knowledge (PMBOK Guide). 7th ed. PMI, 2021. 250 p.

20. Верба В. А., Гребешкова О. М. Управління розвитком компанії : навч. посіб. Київ : КНЕУ, 2021. 450 с.

21. Poppendieck M., Poppendieck T. Lean Software Development: An Agile Toolkit. Addison-Wesley Professional, 2003. 240 p.

22. Sutherland J. Scrum: The Art of Doing Twice the Work in Half the Time. Crown Business, 2014. 256 p.

23. Buterin V. Ethereum Whitepaper. 2013. URL: https://ethereum.org/en/whitepaper/ (дата звернення: 28.11.2025).

24. Swan M. Blockchain: Blueprint for a New Economy. O'Reilly Media, 2015. 152 p.

25. Tapscott D., Tapscott A. Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World. Portfolio, 2016. 368 p.

26. Алькема В. Г., Кириченко О. С. Менеджмент організацій: навч. посіб. Кн.1. Київ : Університет «КРОК», 2023. 276 с.

27. Сумець О. М. Проєктно-орієнтоване управління організацією: навч. посіб. Київ : Університет "КРОК", 2022. 167 с.

28. Highsmith J. Agile Project Management: Creating Innovative Products. Boston : Addison-Wesley, 2009. 432 p.

29. Larman C., Vodde B. Scaling Agile: Experience Report from a Large Project in a Multinational Telecom Company. *Agile Conference*. 2008. P. 23–31.

30. Schwaber K. Agile Software Development with Scrum. *Agile Journal*. 2004. Vol. 3, No. 5. P. 12–18.

31. Рокоча В. В., Одягайло Б. М., Терехов В. І. Міжнародний менеджмент: навч. посіб. Київ : Університет економіки та права «КРОК», 2016. 172 с.

32. Данченко О. Б., Дзюба Т. В. Маркетингові дослідження у проєктах: навч. посіб. Київ : Університет "КРОК", 2021. 224 с.

33. Батенко Л. П., Загородніх О. А., Ліщинська В. В. Управління проектами: навч. посіб. Київ : КНЕУ, 2003. 231 с.

34. What is the difference between blockchain and cryptocurrency? *MNBC*. URL: https://mnbc.info/ua/u-chomu-riznytsja-mizh-blokchejnom-i-kryptovaljutoju/ (дата звернення: 24.11.2025).

35. The fraud triangle theory for opportunity rationalization pressure. *Vecteezy*. URL: https://www.vecteezy.com/vector-art/46968414-the-fraud-triangle-theory-for-opportunity-rationalization-pressure (дата звернення: 24.11.2025).

36. How to Interpret Total Value Locked (TVL) in DeFi. *Messari*. URL: https://messari.io/report/how-to-interpret-total-value-locked-tvl-in-defi (дата звернення: 24.11.2025).

37. Flow diagram of blockchain and smart contracts. *ResearchGate*. URL: https://www.researchgate.net/figure/Flow-diagram-of-blockchain-and-smart-contracts_fig7_366028984 (дата звернення: 26.11.2025).

38. TransferGo to use Ripple's On-Demand Liquidity (ODL). *Reddit*. URL: https://www.reddit.com/r/Ripple/comments/e1wdby/transfergo_to_use_ripples_on demand_liquidity_odl/ (дата звернення: 26.11.2025).

39. DeFi 3.0: The Institutional DeFi. *Medium*. URL: https://medium.com/@wonderful_lilac_beetle_302/defi-3-0-the-institutional-defi-505b1036b81f (дата звернення: 26.11.2025).

40. What is off-chain storage? *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/what-is-off-chain-storage/ (дата звернення: 26.11.2025).

41. Методології розробки програмного забезпечення: огляд та порівняння. *Evergreens*. URL: https://evergreens.com.ua/ua/articles/software-development-metodologies.html (дата звернення: 28.11.2025).

42. Agile терміни та головні методології (Частина 2). *PM Tips*. URL: https://pmtips.com.ua/post/agile-termini-golovni-metodologiyi-castina-2 (дата звернення: 28.11.2025).

43. Best Practices for Managing Remote Teams. *Apploye*. URL: https://apploye.com/blog/best-practices-for-managing-remote-teams/ (дата звернення: 28.11.2025).

44. Building a security conscious CI/CD pipeline. *Snyk*. URL: https://snyk.io/fr/blog/building-security-conscious-ci-cd-pipeline/ (дата звернення: 02.12.2025).

45. Jira Scrum Board. *Atlassian*. URL: https://www.atlassian.com/agile/project-management/scrum-board (дата звернення: 02.12.2025).

46. Power-Interest grid. *RIGCERT*. URL: https://rigcert.education/resources/power-interest-grid-for-stakeholder-identification-and-management (дата звернення: 02.12.2025).

47. Cost change curve of traditional and agile methodology. *ResearchGate*. URL: https://www.researchgate.net/figure/Cost-change-curve-of-traditional-and-agile-methodology-23_fig9_312564218 (дата звернення: 02.12.2025).