

Пузікова Анна Валентинівна

кандидат фізико-математичних наук, доцент, доцент кафедри інформаційних та цифрових технологій
Центральноукраїнський державний університет імені Володимира Винниченка, м. Кропивницький, Україна
ORCID 0000-0002-6843-5583
a.v.puzikova@cuspu.edu.ua

Резіна Ольга Василівна

кандидат педагогічних наук, доцент, доцент кафедри інформаційних та цифрових технологій
Центральноукраїнський державний університет імені Володимира Винниченка, м. Кропивницький, Україна
ORCID 0000-0001-6077-9413
o.v.riezina@cuspu.edu.ua

Баранюк Олександр Філімонович

кандидат технічних наук, доцент, доцент кафедри інформаційних та цифрових технологій
Центральноукраїнський державний університет імені Володимира Винниченка, м. Кропивницький, Україна
ORCID 0000-0003-1151-0092
o.f.baraniuk@cuspu.edu.ua

Мелешко Максим Костянтинович

розробник програмного забезпечення
компанія "Cargo LLC", м. Київ, Україна
ORCID 0009-0006-6880-308X
meleshko.maksym@ukr.net

МОДЕЛЮВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ОЦІНЮВАННЯ АРХІТЕКТУРИ ПРИВАТНОЇ БЛОКЧЕЙН-СИСТЕМИ

Анотація. У статті розглянуто проблему моделювання і реалізації архітектури приватної блокчейн-системи, що забезпечує належний рівень безпеки, конфіденційності та прозорості процесів. Метою дослідження є моделювання та експериментальне оцінювання архітектури приватної блокчейн-системи на основі технології WebAssembly, яка забезпечує високу продуктивність і масштабованість смартконтрактів на рівні сучасних корпоративних платформ, але при цьому суттєво знижує складність інфраструктурного розгортання, налаштування та подальшого адміністрування.

У роботі виконано аналіз сучасних концепцій технології блокчейн із погляду їх архітектури та оцінювання продуктивності. Запропоновано концептуальну архітектуру блокчейн-системи, яка реалізує повний процес функціонування: від формування структури блоку та обчислення його хешу до взаємодії з термінальним інтерфейсом, включно із вибором валідатора, встановленням WebSocket-з'єднань між вузлами, зберіганням стану акаунтів, блоків і транзакцій та криптографічним захистом.

Ядро блокчейн-системи реалізовано на платформі Node.js із використанням TypeScript. Для зберігання даних застосовано key-value сховище LevelDB. Мову контрактів реалізовано засобами AssemblyScript. Криптографічний захист транзакцій забезпечено алгоритмами цифрового підпису на основі еліптичної криптографії з використанням кривої secp256k1. Термінальний інтерфейс побудовано на основі бібліотек React та Ink з використанням ANSI-рендерингу. Як механізм консенсусу обрано Proof-of-Stake. Виконання смартконтрактів реалізовано в ізольованому середовищі WebAssembly runtime.

Для перевірки надійності розробленої системи у середовищі WasmExecutor проведено фаззинг-тестування з ін'єкцією низькорівневого байт-коду для визначення стійкості до виходу за межі лінійної пам'яті та виникнення нескінченних циклів виконання. Для кількісного оцінювання продуктивності проведено навантажувальне тестування засобами Hyperledger Caliper 0.6.0. Більшість тестових сценаріїв виконано успішно, а розроблена система продемонструвала високий рівень стабільності. Отримані результати підтверджують доцільність запропонованого архітектурного підходу для побудови приватних блокчейн-систем зі спрощеним розгортанням.

Ключові слова: архітектура блокчейн-системи, продуктивність, смартконтракт, WebAssembly, React, термінальний інтерфейс, конфіденційність, контрольований доступ.

Anna Puzikova

PhD, Associate Professor, Associate Professor of the Department of Information and Digital Technologies.
Work place: Volodymyr Vynnychenko Central Ukrainian State University, Kropyvnytskyi, Ukraine
ORCID 0000-0002-6843-5583

© 2026 Пузікова А.В., Резіна О.В., Баранюк О.Ф., Мелешко М.К. Цей матеріал ліцензовано за умовами **CC BY 4.0**.

<https://creativecommons.org/licenses/by/4.0/>

a.v.puzikova@cuspu.edu.ua

Olga Riezina

PhD, Associate Professor, Associate Professor of the Department of Information and Digital Technologies.

Work place: Volodymyr Vynnychenko Central Ukrainian State University, Kropyvnytskyi, Ukraine

ORCID 0000-0001-6077-9413

o.v.riezina@cuspu.edu.ua

Oleksandr Baraniuk

PhD, Associate Professor, Associate Professor of the Department of Information and Digital Technologies.

Work place: Volodymyr Vynnychenko Central Ukrainian State University, Kropyvnytskyi, Ukraine

ORCID 0000-0003-1151-0092

o.f.baraniuk@cuspu.edu.ua

Maksym Meleshko

Software Developer

Work place: Cargoo LLC, Kyiv, Ukraine

ORCID 0009-0006-6880-308X

meleshko.maksym@ukr.net

MODELLING AND EXPERIMENTAL EVALUATION OF A PRIVATE BLOCKCHAIN SYSTEM ARCHITECTURE

Abstract. *The article addresses the problem of modeling and implementing the architecture of a private blockchain system that ensures an appropriate level of security, confidentiality, and transparency of processes. The aim of this research is to model and experimentally evaluate the architecture of a private blockchain system based on WebAssembly technology, which provides high performance and smart contract scalability comparable to those of modern enterprise platforms while significantly reducing the complexity of infrastructure deployment, configuration, and administration.*

The paper presents an analysis of modern blockchain technology concepts from the perspective of the architecture of such systems and the evaluation of their performance. A conceptual architecture of a blockchain system is proposed, which implements the complete operational workflow: from block structure formation and hash computation to interaction with the terminal interface, including validator selection, establishment of WebSocket connections between nodes, storage of account, block, and transaction states, as well as cryptographic protection.

The core of the blockchain system was implemented using the Node.js platform with TypeScript. A LevelDB key-value storage was used to store data. The contract language was implemented using AssemblyScript. Cryptographic protection of transactions was provided using elliptic-curve digital signature algorithms based on the secp256k1 curve. The terminal interface was built using the React and Ink libraries with ANSI rendering. Proof-of-Stake was selected as the consensus mechanism. The WebAssembly runtime was used as the execution environment for smart contracts, ensuring isolated code execution. Smart contract execution is implemented in an isolated WebAssembly runtime environment.

To verify the reliability of the developed system, fuzz testing was performed in the WasmExecutor environment with the injection of low-level bytecode to evaluate resistance to linear memory out-of-bounds access and infinite execution loops. Quantitative performance evaluation was conducted through load testing using Hyperledger Caliper 0.6.0. The majority of test scenarios were completed successfully, and the developed system demonstrated a high level of stability. The obtained results confirm the feasibility of the proposed architectural approach for building private blockchain systems with simplified deployment.

Keywords: *blockchain system architecture, performance, smart contracts, WebAssembly, React, terminal user interface, confidentiality, access control.*

1. Вступ

За останні п'ятнадцять років блокчейн-технології знайшли застосування в різних сферах діяльності, а світовий ринок блокчейну демонструє інтенсивний розвиток: у 2024 році його обсяг оцінювався у 31,28 млрд дол. США, а до 2030 року, за прогнозами, досягне 1 431,54 млрд дол. США при середньорічному темпі зростання 90,1 % [1]. Попри такі тенденції, у сфері блокчейн-технологій досі не сформовано єдиного стандарту щодо проектування та реалізації таких систем, а галузь характеризується відсутністю уніфікованої термінології, різноманіттям архітектурних підходів і неоднозначністю трактування базових понять [2]. Відсутність чітких стандартів і вимог створює додаткові виклики для розробників блокчейн-систем.

2. Постановка проблеми

Станом на 2026 рік зросла потреба у приватних блокчейн-системах, до яких висувається комплекс технічних і організаційних вимог. Для великих корпоративних рішень на даний час пропонуються системи: Hyperledger Fabric, Hyperledger Sawtooth, Corda, Quorum. Усі ці системи характеризуються модульною

архітектурою, забезпечують керований доступ користувачів мережі та конфіденційність транзакцій. В той же час ці системи можуть демонструвати обмеження продуктивності. Так, Ethereum через архітектуру стекової віртуальної машини Ethereum Virtual Machine (EVM), яка працює переважно з 256-бітними словами (тоді як сучасні процесори (x86-64, ARM64) оптимізовані під 64-бітні операції) для виконання однієї 256-бітної операції потребує десятків 64-бітних інструкцій сучасного процесора [3]. Такий підхід вимагає додаткових обчислювальних витрат та негативно впливає на продуктивність системи. Іншим недоліком відкритих блокчейн-систем є обмеженість середовища розробки смартконтрактів. Зокрема, для платформ на основі Ethereum основними мовами програмування є Solidity та Vyper, які суттєво відрізняються від традиційних мов розробки програмного забезпечення (Java, C#, Python, Go, Node.js), що не можуть напівпрямую використовуватись в EVM. Це зумовлює необхідність додаткової підготовки розробників та підвищує складність створення й супроводу блокчейн-застосунків. Крім того, середовище виконання EVM має обмежену підтримку стандартних інструментів програмної інженерії, налагодження та оптимізації програмного коду. У результаті ускладнюється тестування, профілювання та підвищення продуктивності смартконтрактів, що створює додаткові накладні витрати під час виконання транзакцій і негативно впливає на загальну продуктивність блокчейн-системи [4]. Для іншої блокчейн-системи Hyperledger Fabric, яка забезпечує досить високу продуктивність, гнучке управління доступом, конфіденційність транзакцій та інтеграцію з наявними корпоративними системами [5], в роботі [6] обговорюється складність розгортання, налаштування/оптимізації та адміністрування. Таким чином, використання зазначених вище блокчейн-систем для невеликих компаній не завжди є оптимальним і обґрунтованим рішенням.

З метою швидкого впровадження блокчейн-технології, що забезпечує достатній рівень безпеки, конфіденційності та прозорості процесів, для простих сценаріїв (наприклад, голосування з урахуванням ваги голосу), доцільним є створення блокчейн-системи, що не потребує складної інфраструктури та/або висококваліфікованого персоналу. Це також надає можливість значно знизити витрати на розгортання та підтримку системи. Успішне впровадження таких рішень вимагає ретельного проєктування.

Ключовими вимогами до приватних блокчейн-систем є: модульність архітектури, продуктивність, масштабованість, конфіденційність, відмовостійкість та контрольований доступ учасників мережі [7]. Продуктивність приватних блокчейн-систем оцінюється за показниками пропускну здатності, затримки підтвердження транзакцій, масштабованості та ефективності використання ресурсів [8]. Одним із рішень, які дозволяють істотно підвищити швидкість обробки коду є використання середовища виконання смартконтрактів WebAssembly runtime [9] – [11].

3. Аналіз останніх досліджень і публікацій

Концепція технології блокчейн широко представлена в науковій літературі. У цьому огляді розглянуті питання архітектури блокчейн-систем та оцінювання їхньої продуктивності.

У фундаментальній роботі [12] блокчейн визначається як послідовність блоків, яка містить повний список записів транзакцій, подібно до публічного реєстру. Автори окреслюють загальну архітектуру блокчейн-системи і наводять таксономію таких систем: публічний (Public), приватний (Private) та консорціумний (Consortium) блокчейни. Зазначається, що в публічному блокчейні всі записи є відкритими, а будь-хто може брати участь у процесі консенсусу. У консорціумному блокчейні до процесу консенсусу допускається лише група попередньо визначених вузлів. Приватний блокчейн розглядається як централізована мережа, яка повністю контролюється однією організацією.

Питання архітектури блокчейн-системи є ключовим у багатьох дослідженнях. Більшість дослідників [13] – [15], дотримуються таксономії, представленої у [12], доповнюючи її таким типом архітектури як гібридний блокчейн (Hybrid Blockchain). Зауважується, що значна частина організацій прагне одночасно використовувати переваги як публічної так і приватної технології блокчейн, тому вони впроваджують гібридні рішення, які поєднують елементи обох підходів.

Бхутта та ін. [16] розглядають наявні архітектури та компоненти блокчейну в контексті криптовалют, смартконтрактів та застосувань блокчейну взагалі. Автори простежують еволюцію архітектури від Blockchain 1.0 до Blockchain 3.0 з детальним розбором структури блоку, організації мережі та механізмів досягнення консенсусу.

У статті [17] представлено дворівневу блокчейн-орієнтовану структуру довіри, адаптовану до унікальних потреб як відкритих, так і приватних архітектур. Для відкритих систем ця структура підвищує рівень довіри та безпеки завдяки прозорому моделюванню довіри, верифікації на основі блокчейну та механізмів підзвітності, захищеним від несанкціонованого доступу. Для приватних систем вона посилює внутрішнє оцінювання довіри та знижує ризики шляхом захисту взаємодій у закритому середовищі з обмеженим доступом.

У ґрунтовному дослідженні [18] подано всебічний огляд архітектур у системах електронного голосування на основі технології блокчейн. Розглядаються чотири основні типи архітектур: архітектура приватного блокчейну з використанням Hyperledger Fabric для створення безпечного, контрольованого та ефективного середовища цифрового голосування; архітектура публічного блокчейна на базі платформи Ethereum, яка реалізує правила і логіку процесу голосування, забезпечує прозорість та захист від підробки голосів, а також надає виборцям можливість взаємодіяти з блокчейном через веб-інтерфейс; архітектура консорціумного блокчейну на основі Quorum; архітектура гібридного блокчейну, що поєднує елементи публічних і приватних

блокчейнів із використанням сильних сторін кожного з них для побудови універсального та ефективного рішення.

Низку робіт присвячено використанню технології WebAssembly при створенні блокчейн-застосунків. Автори одностайні щодо доцільності використання WebAssembly для оптимізації продуктивності, масштабованості та безпеки в різних технологічних середовищах [9] – [11], забезпечення безперебійної взаємодії між блокчейн та середовищами довіреного виконання [19], а також розробки надійних криптографічних бібліотек [11], [20].

Поряд з архітектурою одним із пріоритетних питань у дослідженнях блокчейн-систем є питання визначення та оптимізації продуктивності. Пеїлін Женг та ін. [21] представляють загальні та детальні показники продуктивності, що надають можливість користувачеві точно відстежувати цей показник на різних етапах блокчейну. Поєднуючи дані блокчейну та споживання ресурсів, дослідники визначають такі загальні показники продуктивності: транзакції за секунду (Transactions Per Second, TPS), Середня затримка відповіді (Average Response Delay), Транзакції на процесор (Transactions Per CPU), Транзакції за секунду пам'яті (Transactions Per Memory Second), Транзакції на дискове введення/виведення (Transactions Per Disk I/O), Транзакцій на мережеві дані (Transactions Per Network Data). Для розробників блокчейну виокремлюються показники, що відповідають кожному важливому етапу процесу: швидкість виявлення вузлів (Peer Discovery Rate), швидкість відповіді RPC (RPC Response Rate), швидкість поширення транзакцій (Transaction Propagating Rate), Час виконання контракту (Contract Execution Time), Час оновлення стану (State Updating Time), Час досягнення консенсусу (Consensus-Cost Time).

Саїхянг Фан та ін. [22] класифікують всі рішення щодо оцінювання продуктивності блокчейну на дві загальні категорії: емпіричний аналіз та аналітичне моделювання. В емпіричному аналізі вони розглядають сучасні методології емпіричної оцінки блокчейну: бенчмаркінг, моніторинг, експериментальний аналіз та моделювання. В аналітичному моделюванні досліджують стохастичні моделі для оцінювання продуктивності основних алгоритмів консенсусу блокчейну.

Поглиблене дослідження характеристик продуктивності Hyperledger Fabric здійснюється в роботі [5]. Автори надають аналіз системи із використанням 15 змінних, що стосуються архітектури, конфігурації налаштування, бізнес-логіки, мережі та надійності. Перевагою цього аналізу є рекомендації, які пропонуються системним архітекторам та інженерам інфраструктури для розробки застосунків на основі Fabric.

Огляд літератури вказує на тенденцію до поєднання переваг різних типів блокчейн-архітектур шляхом використання гібридних моделей, що дозволяють одночасно забезпечити прозорість, довіру та контрольований доступ до інформації. Розвиток блокчейн-систем сьогодні зумовлений не лише удосконаленням архітектурних рішень, а й необхідністю підвищення продуктивності, масштабованості та безпеки. Зокрема це досягається шляхом оптимізації механізмів консенсусу, використання технології WebAssembly, довірених середовищ виконання та вдосконалення методів оцінювання продуктивності. Таким чином, питання проєктування архітектури приватної блокчейн-системи, яка забезпечує оптимальний баланс між безпекою, масштабованістю та ефективністю роботи системи є актуальною задачею.

4. Мета і задачі дослідження

Метою статті є розробка та експериментальне оцінювання архітектури приватної блокчейн-системи на основі технології WebAssembly, яка забезпечує високу продуктивність і масштабованість смартконтрактів на рівні сучасних корпоративних платформ, але при цьому суттєво знижує складність інфраструктурного розгортання, налаштування та подальшого адміністрування.

Для досягнення поставленої мети потрібно вирішити такі завдання:

- проаналізувати обмеження наявних корпоративних платформ та обґрунтувати доцільність застосування технології WebAssembly;
- виконати проєктування архітектури спрощеної приватної блокчейн-системи;
- провести експериментальні випробування розробленого проєкту за допомогою інструменту Hyperledger Caliper під різним транзакційним навантаженням;
- здійснити порівняльний аналіз отриманих метрик із референсною системою Hyperledger Fabric.

5. Результати дослідження

У нашому дослідженні запропоновано архітектуру і розроблено модель блокчейн-системи для корпоративних рішень у вигляді повнофункціонального децентралізованого автономного блокчейн-вузла мережі, що поєднує в собі ролі клієнта, валідатора та сервера.

Система забезпечує ключові вимоги до приватних блокчейн-систем – побудована за модульним принципом, є масштабованою, відмовостійкою, гарантує контрольований доступ учасників мережі та конфіденційність транзакцій.

5.1. Архітектура блокчейн-системи

Кожен запущений примірник програми є самостійним вузлом мережі, який реалізовано на платформі Node.js із використанням мови програмування TypeScript. Для зберігання даних використовується сховище LevelDB типу ключ-значення. Криптографічний захист транзакцій реалізовано із застосуванням бібліотеки elliptic, що реалізує алгоритми цифрового підпису на основі еліптичної криптографії (крива secp256k1) та

забезпечує сумісність із сучасними блокчейн-стандартами. Клієнтський інтерфейс побудовано на основі компонентів React, а для консольного виводу використовується ANSI-рендеринг.

Архітектурно система складається з шести основних модулів (рис.1):

Модуль ядра блокчейну Blockchain Core містить чотири класи: Block, Blockchain, Mempool і Transaction. Клас Block відповідає за формування структури блоку та обчислення його хешу. Клас Blockchain є центральним, він відповідає за валідацію та додавання нових блоків до ланцюга, а також за ініціалізацію системи при першому запуску. Клас Mempool виконує роль тимчасового сховища для транзакцій, які очікують свого підтвердження в блокчейні. Клас Transaction визначає структуру транзакції та підтримує шість типів операцій: TRANSFER, STAKE, UNSTAKE, REWARD, DEPLOY та CALL.

Модуль консенсусу Consensus містить клас PoS, який реалізує алгоритм Proof-of-Stake (PoS) для вибору валідатора, що має право сформувати наступний блок.

Модуль мережі Net містить клас P2PNetwork, який відповідає за встановлення WebSocket-з'єднань між вузлами, синхронізацію ланцюга та трансляцію нових транзакцій і блоків.

Модуль зберігання стану Store містить клас StateDB, який є абстракцією над базою даних LevelDB і забезпечує зберігання стану всіх акаунтів, блоків та транзакцій.

Модуль криптографії Crypto містить клас Wallet, який забезпечує генерацію ключових пар на основі алгоритму Ed25519, формування цифрового підпису та перевірку цифрових підписів.

Модуль інтерфейсу UserInterface створює графічний інтерфейс на базі бібліотек React та Ink, щоб взаємодіяти із системою через термінал.

Модуль WasmExecutor реалізує ізольоване виконання смартконтрактів у форматі WebAssembly (WASM). Завдяки AssemblyScript як підмножині TypeScript смартконтракти компілюються безпосередньо у формат WASM без необхідності додаткових обчислювальних потужностей середовища виконання [9].

Ключовою архітектурною особливістю розробленої системи є інтеграція її модулів через механізм React Context, реалізований у вигляді об'єкта NodeContext. Такий підхід дозволяє уникнути використання глобальних змінних чи статичних синглтонів. Завдяки цьому забезпечується наскрізний доступ компонентів користувацького інтерфейсу до поточного стану системи без необхідності каскадної передачі параметрів через ієрархічну структуру батьківських елементів.

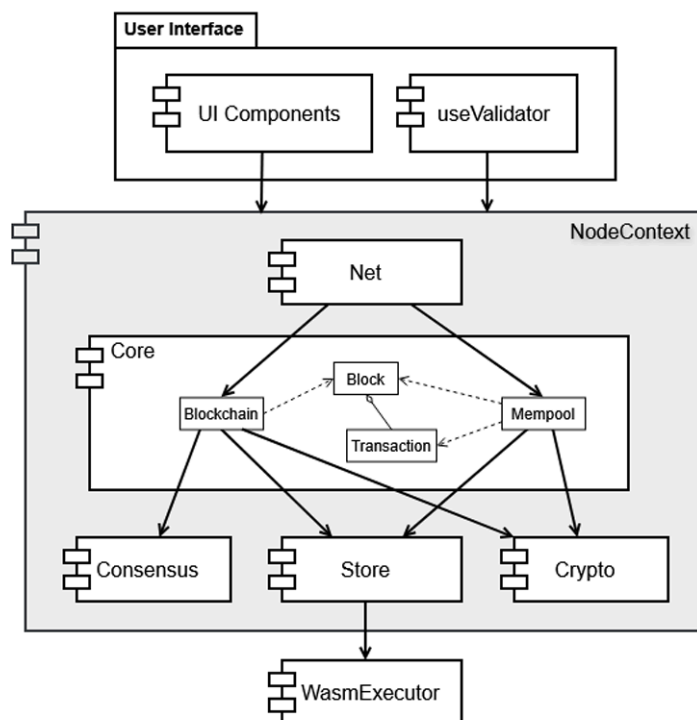


Рис. 1. Загальна структура системи

5.2. Принцип роботи блокчейн-системи

Після запуску застосунку виконується ініціалізація всіх основних компонентів блокчейн-вузла (рис. 2). Насамперед створюється екземпляр блокчейну, який перевіряє наявність локальної бази даних, відновлює поточний стан ланцюга блоків та, за необхідності, автоматично розгортає смарт-контракт у форматі WASM. Далі ініціалізується пул транзакцій для зберігання непідтверджених транзакцій, запускається мережевий модуль, який відкриває WebSocket-сервер для взаємодії з іншими вузлами, після чого всі створені компоненти об'єднуються в єдине середовище виконання та стають доступними іншим модулям системи.

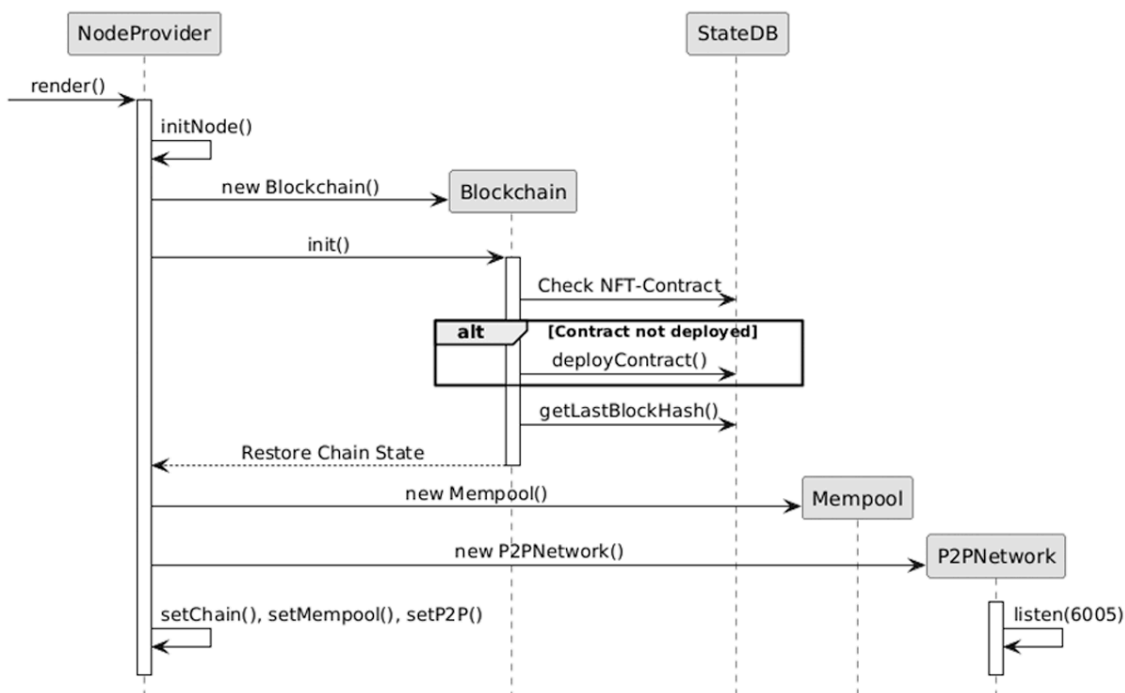


Рис. 2. Ініціалізація блокчейн-системи

Після завершення ініціалізації вузол переходить у робочий режим. Він підтримує з'єднання з іншими учасниками мережі, приймає нові транзакції та блоки, синхронізує локальний ланцюг і періодично виконує процедуру вибору валідатора відповідно до алгоритму PoS.

Процес обробки транзакції починається з її формування користувачем через графічний інтерфейс (рис. 3). На основі введених параметрів створюється транзакція відповідного типу, яка підписується приватним ключем відправника за алгоритмом Ed25519. Після цього транзакція передається до локального пулу транзакцій, де виконується перевірка коректності цифрового підпису та необхідної ваги величини stake. Лише після успішного проходження всіх перевірок транзакція вважається дійсною та додається до черги непідтверджених транзакцій.

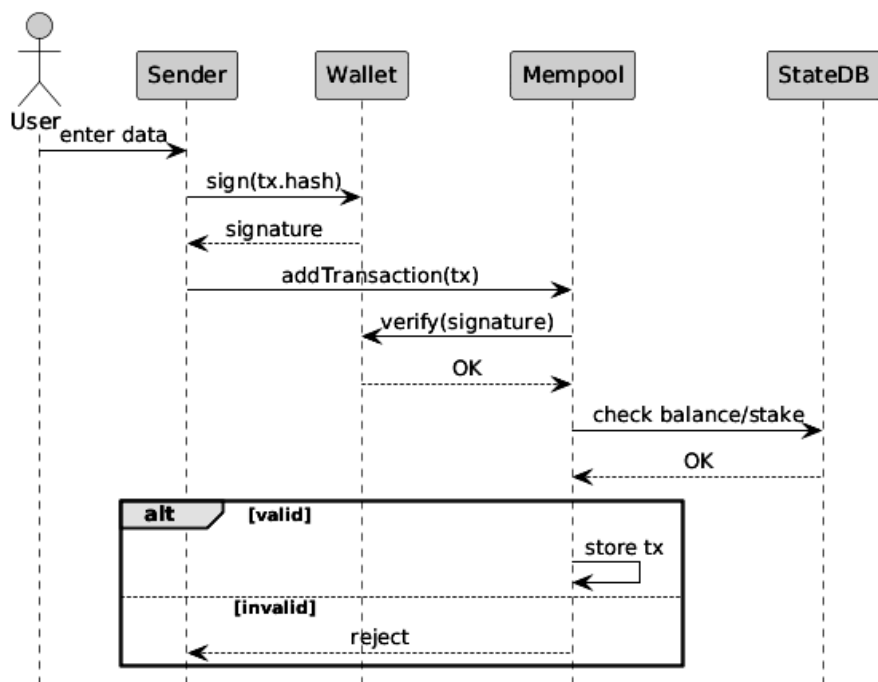


Рис. 3. Створення і валідація транзакції

Одночасно із збереженням у локальному пулі транзакція транслюється всім підключеним вузлам мережі. Кожен вузол після отримання повідомлення виконує аналогічні перевірки, додає нову транзакцію до власного

пулу та ретранслює її іншим учасникам мережі. Такий механізм забезпечує швидке поширення інформації про нові транзакції без використання центрального сервера.

Формування нового блоку здійснюється вузлом, який відповідно до алгоритму PoS отримав право виконувати роль валідатора в поточному часовому слоті. Через задані інтервали часу кожен вузол самостійно визначає результат вибору валідатора, використовуючи однаковий алгоритм і спільні параметри мережі. Якщо вузол обрано лідером, він формує новий блок, додаючи до нього всі транзакції з локального пулу, а також службову транзакцію винагороди валідатора. Після цього обчислюється хеш блоку, блок підписується цифровим підписом валідатора та передається до модуля блокчейну для перевірки.

Перед включенням нового блоку до локального ланцюга виконується комплексна валідація. Перевіряється правильність посилання на попередній блок, послідовність індексів, коректність часових міток, право валідатора на створення блоку, а також достовірність його цифрового підпису. Якщо всі перевірки завершуються успішно, транзакції послідовно застосовуються до поточного стану системи, оновлюються баланси акаунтів, змінюються значення величини stake, зберігається новий блок і оновлюється стан бази даних.

Після успішного підтвердження блоку всі транзакції, що увійшли до його складу, вилючаються з пулу, а сформований блок транслюється іншим вузлам мережі. Отримавши новий блок, кожен вузол виконує аналогічний цикл перевірки, застосування змін до локального стану та синхронізації власного ланцюга. Завдяки цьому всі вузли підтримують однаковий стан розподіленого реєстру без використання централізованого координатора.

Запропонована система підтримує виконання смарт-контрактів у форматі WASM в ізольованому середовищі, що виключає їхній безпосередній доступ до внутрішніх структур блокчейну. Взаємодія зі сховищем стану реалізована через обмежений набір функцій читання та запису, які надаються ядром системи. Такий підхід забезпечує контрольоване виконання смарт-контрактів, ізоляцію програмного коду та підвищує безпеку роботи блокчейн-платформи.

Відмітимо, що для розробленої приватної блокчейн-системи обрано механізм консенсусу PoS на відміну від рішення, реалізованого в Hyperledger Fabric, яка орієнтована на контрольоване блокчейн-середовище з ідентифікованими учасниками мережі. Для досягнення консенсусу в Hyperledger Fabric застосовуються механізми Raft (для централізованої мережі з довіреними учасниками) та/або BFT-based ordering services (для централізованої мережі з частково довіреними учасниками), які забезпечують високу продуктивність і незначні затримки операцій читання/запису.

У корпоративному середовищі, де учасники мережі є ідентифікованими та частково довіреними і є потреба в децентралізованому блокчейн-середовищі, застосування механізму PoS дає можливість організувати контрольовану модель валідації транзакцій із забезпеченням необхідного рівня безпеки та масштабованості. У запропонованій приватній блокчейн-системі частка участі (stake) може інтерпретуватись як рівень довіри, набір повноважень або вага участі вузла в мережі. Зокрема, у системах електронного голосування частка участі може визначатися роллю організації, кількістю делегованих голосів, часткою власності або рівнем авторизації валідатора.

5.3. Експеримент

Для перевірки працездатності системи було розгорнуто локальний кластер з кількох ізольованих нод, комунікація між якими відбувалася у локальній мережі.

Створення розривів WebSocket-з'єднань та відновлення дозволило верифікувати алгоритм консенсусу в умовах візантійських відмов. У середовищі WasmExecutor було проведено фазинг-тестування, що передбачало ін'єкцію низькорівневого байт-коду для перевірки стійкості до виходу за межі лінійної пам'яті та виникнення нескінченних циклів виконання [23]. При цьому інтеграційні тести підтвердили коректне перехоплення інструкції переривання без аварійного завершення основного процесу Node.js. Також було додатково реалізовано механізм фільтрації невалідних об'єктів на мережевому рівні до виконання етапу повної серіалізації. Це запобігає ситуації, коли валідація криптографічних підписів безпосередньо в Event Loop блокує рендеринг термінального інтерфейсу при спам-атаках на мемпул.

Для кількісної оцінки продуктивності розробленої блокчейн-системи було проведено навантажувальне тестування за допомогою інструменту Hyperledger Caliper 0.6.0, який вимірює такі показники продуктивності як пропускна здатність транзакцій/читання і затримка транзакцій/читання (мінімальна, максимальна, середня). Тести продуктивності передбачали фіксовану швидкість надсилання транзакцій, кількість процесів (workers) генерації навантаження – 5. Попередньо до реєстру було завантажено 20000 активів. Тривалість кожного тесту – 60 с. Отримані метрики продуктивності за чотирма сценаріями відповідно до тестів [24] наведено у таблиці 1.

Перший сценарій виконує сліпий запис одного ключа розміром 100 байт. Фіксована кількість транзакцій за секунду – 3000. Розроблена система стабільно обробила усі транзакції без втрат і помилок. Середня затримка становила лише 0,12 с, що свідчить про високу швидкість мережі.

Другий сценарій здійснює сліпий запис одного ключа розміром 1000 байт. Фіксована кількість транзакцій за секунду – 3000. При цьому спостерігається зростання середньої затримки (0,2 с) та поява поодиноких помилок.

Третій сценарій передбачає виконання операцій читання/запису одного ключа розміром 100 байт. У цьому тесті транзакція випадковим чином вибирає вже існуючий ключ з даними, зчитує його, а потім змінює цей ключ. Фіксована кількість транзакцій за секунду становить 2550. Поява помилок може свідчити про ймовірність

використання одного й того ж ключа у двох одночасних транзакціях, зважаючи на відносно невелику кількість активів – 20000. Зауважимо, що для цього ж тесту, проведеного для Hyperledger Fabric 2.5 на іншому апаратному забезпеченні з іншою топологією та за наявності 1 мільйону активів не було отримано помилок перевірки MVCC_READ_CONFLICT [24].

Таблиця 1.

Результати тестування блокчейн-системи

Назва тесту	Успішність	Помилки	Швидкість надсилення (TPS)	Максимальна затримка, с	Мінімальна затримка, с	Середня затримка, с	Пропускна здатність (TPS)
Створення активу-100	180005	0	3000.2	2.28	0.00	0.12	3000.2
Створення активу-1000	179483	522	3000.1	4.20	0.00	0.20	3000.1
Читання/запис активу-100	152058	951	2550.3	1.32	0.00	0.01	2550.3
Читання/запис активу-1000	91806	0	1530.2	1.30	0.00	0.00	1530.2

Четвертий сценарій моделює виконання операцій читання/запису одного ключа розміром 1000 байт. Фіксована кількість транзакцій за секунду становить 1530. При цьому загальні результати щодо кількості успішних транзакцій та середньої затримки можна вважати достатньо ефективними.

Для чистоти експерименту з метою аналізу отриманих результатів було проведено тестування Hyperledger Fabric 2.5 за тими ж сценаріями та за тих самих умов, результати якого наведені в таблиці 2. Зауважимо, що отримані результати суттєво відрізняються від наведених у роботі [24] внаслідок зміни технічних умов тестування.

Таблиця 2.

Результати тестування Hyperledger Fabric 2.5

Назва тесту	Успішність	Помилки	Швидкість надсилення (TPS)	Максимальна затримка, с	Мінімальна затримка, с	Середня затримка, с	Пропускна здатність (TPS)
Створення активу-100	62889	1	1048.0	2.19	0.17	0.50	1014.8
Створення активу-1000	61182	0	1019.4	0.73	0.15	0.42	1016.2
Читання/запис активу-100	61054	1484	1042.3	2.13	0.21	0.54	1013.5
Читання/запис активу-1000	59404	1319	1012.0	1.03	0.18	0.50	1008.6

5.4. Результати

Загальний аналіз продуктивності засвідчує, що за пропускну здатності системи до 3000 TPS для простих операцій створення активів та приблизно 1500–2500 TPS для складніших сценаріїв читання і запису середня затримка перебувала в межах від 0,00 с до 0,20 с (табл. 1). Це вказує на швидке підтвердження транзакцій та високу швидкість реагування системи, що є важливим для приватної блокчейн-системи. Розроблена система продемонструвала високий рівень стабільності, оскільки більшість тестових сценаріїв було виконано без помилок або з мінімальною кількістю неуспішних транзакцій. Найбільшу кількість помилок зафіксовано під час створення активів великого розміру та виконання одночасних операцій читання і запису. Ймовірними причинами цього є конфлікти конкурентного доступу до даних.

Виявлені в процесі тестування обмеження продуктивності було усунуто шляхом впровадження черги мутацій стану, кешування попередньо скомпільованих WASM-модулів та механізму попередньої фільтрації мережевого трафіку. Створений прототип демонструє алгоритмічну життєздатність обраного стеку та підтверджує ефективність використання технології WebAssembly для розробленої блокчейн-системи.

6. Висновки та перспективи подальших досліджень

У статті запропоновано, обґрунтовано, практично перевірено і протестовано модель приватної блокчейн-системи для корпоративних рішень, що поєднує функції клієнта, валідатора та сервера в єдиному децентралізованому вузлі й забезпечує вимоги до подібних систем щодо модульності, масштабованості, відмовостійкості та захисту даних.

Основні результати дослідження:

1. Розроблено архітектуру приватної блокчейн-системи та реалізовано її програмну модель на основі WebAssembly. Модель забезпечує ефективну обробку транзакцій та підтримку високонавантажених сценаріїв та

за характеристиками продуктивності і масштабованості співвідносна із приватною блокчейн-системою Hyperledger Fabric. При цьому система характеризується простотою у розгортанні та налаштуванні;

2. Розроблено термінальний інтерфейс на основі бібліотек Ink та React, що не потребує DOM-дерева та браузерних механізмів рендерингу;

3. Проведено серію експериментальних випробувань із використанням інструменту Hyperledger Caliper для оцінювання пропускну здатності, затримок та рівня відмов транзакцій;

4. Виконано порівняльний аналіз продуктивності системи в умовах різної складності операцій (створення активів, а також читання та запис) із визначенням впливу навантаження на ключові метрики.

Результати тестування розробленої блокчейн-системи є порівняними з типовими показниками навантажувального тестування блокчейн-платформи Hyperledger Fabric, де пропускну здатність у виробничому середовищі становить близько 3000-3500 транзакцій за секунду при середній затримці менше 2 с. Також у деяких випадках отримані показники перевищують типову продуктивність публічних EVM-сумісних мереж та підтверджують ефективність розробленого підходу до побудови блокчейн-системи, а отже, її придатність для використання в корпоративних інформаційних системах. Перевагою запропонованої моделі блокчейн-системи над наявними аналогами є її простота розгортання та адміністрування в порівнянні з Hyperledger Fabric [25].

Зауважимо, що результати низки досліджень свідчать про те, що для блокчейн-систем із контрольованим доступом механізми консенсусу Raft і BFT-орієнтовані служби впорядкування транзакцій зазвичай демонструють вищі показники продуктивності та менші затримки (у випадку Raft) порівняно з механізмом PoS [26]. Водночас результати тестування розробленої системи засвідчили досягнення характеристик продуктивності, збіжних із відповідними показниками Hyperledger Fabric, що підтверджує обґрунтованість запропонованої архітектури та ефективність реалізованої моделі блокчейн-системи.

Подальші дослідження можуть бути спрямовані на пошук альтернативних (модифікованих) механізмів консенсусу та експериментальне оцінювання їх ефективності за показниками продуктивності, масштабованості та відмовостійкості системи.

Внесок авторів. Анна Пузікова – підготовка теоретичних основ дослідження, узагальнення результатів; Ольга Резіна – аналіз джерел, підготовка огляду літератури; Олександр Баранюк – архітектурні моделі; Максим Мелешко – емпіричне дослідження, збір емпіричних даних.

Декларація про штучний інтелект

При написанні статті штучний інтелект не використовувався.

Конфлікт інтересів

Автори заявляють про відсутність конфлікту інтересів та підтверджує, що під час підготовки цієї роботи не існувало жодних комерційних, фінансових чи інших взаємовідносин, які могли б бути розцінені як такі, що здатні вплинути на результати дослідження або їх інтерпретацію. Робота виконана відповідно до принципів академічної доброчесності, етичних норм проведення наукових досліджень та вимог редакційної політики щодо запобігання конфлікту інтересів.

Список використаної літератури

1. Budzanivskiy, M. (2025, October 10). Blockchain Architecture Explained: Comprehensive Guide. Codebridge | Architecture-First Software & AI Systems Engineering. <https://www.codebridge.tech/articles/blockchain-architecture-explained-comprehensive-guide>
2. Xu, M., Guo, Y., Liu, C., Hu, Q., Yu, D., Xiong, Z., Niyato, D., & Cheng, X. (2024). Exploring Blockchain Technology through a Modular Lens: A Survey. ACM Computing Surveys. <https://doi.org/10.1145/3657288>
3. Wood, G. (2025). Ethereum: A secure decentralised generalised transaction ledger (Shanghai version efc5f9a). <https://ethereum.github.io/yellowpaper/paper.pdf>
4. di Angelo, M., Durieux, T., Ferreira, J. F., & Salzer, G. (2023). SmartBugs 2.0: An Execution Framework for Weakness Detection in Ethereum Smart Contracts. In J. Gurrola (Ed.), Proceedings of the 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 2102-2105). IEEE. <https://doi.org/10.1109/ASE56229.2023.00060>
5. Guggenberger, T., Sedlmeir, J., Fridgen, G., & Luckow, A. (2022). An in-depth investigation of the performance characteristics of Hyperledger Fabric. Computers & Industrial Engineering, 108716. <https://doi.org/10.1016/j.cie.2022.108716>
6. Nakaike, T., Zhang, Q., Ueda, Y., Inagaki, T., & Ohara, M. (2020). Performance characterization of Hyperledger Fabric. In 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) (pp. 1–9). IEEE. <https://doi.org/10.1109/ICBC48266.2020.9169454>
7. Welcome to Hyperledger Fabric — hyperledger-fabricdocs master documentation. (n.d.). Welcome to Hyperledger Fabric – hyperledger-fabricdocs master documentation. <https://fab.readthedocs.io/en/latest/>
8. Honar Pajooh, H., Rashid, M. A., Alam, F., & Demidenko, S. (2022). Experimental Performance Analysis of a Scalable Distributed Hyperledger Fabric for a Large-Scale IoT Testbed. Sensors, 22(13), 4868. <https://doi.org/10.3390/s22134868>
9. Țălu, M. (2025). A Comparative Study of WebAssembly Runtimes: Performance Metrics, Integration Challenges, Application Domains, and Security Features. Archives of Advanced Engineering Science, 1–13. <https://doi.org/10.47852/bonviewaaes52024965>
10. Stryzhenko, Y. (2025). Optimization of web application performance through the integration of Web Assembly, blockchain technologies, and modern DevOps practices. Futurity Proceedings, 3. <https://doi.org/10.5281/zenodo.15662748>

11. Park, B., Song, J., & Seo, S. C. (2020). Efficient Implementation of a Crypto Library Using Web Assembly. *Electronics*, 9(11), 1839. <https://doi.org/10.3390/electronics9111839>
12. Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In 2017 IEEE 6th International Congress on Big Data (BigData Congress) (pp. 557–564). IEEE. <https://doi.org/10.1109/BigDataCongress.2017.85>
13. Ismail, L., & Materwala, H. (2019). A Review of Blockchain Architecture and Consensus Protocols: Use Cases, Challenges, and Solutions. *Symmetry*, 11(10), 1198. <https://doi.org/10.3390/sym11101198>
14. Kesoju, M. (2023). Review on blockchain technology: Architecture, characteristics, benefits, algorithms, challenges and applications. *Mesopotamian Journal of CyberSecurity*, 2023, 73–84. <https://doi.org/10.58496/MJCS/2023/012>
15. Staley, I. (2026). Use-Case-Driven Blockchain Architectures in Financial Enterprises: Permissioned, Consortium, Hybrid, and Public Networks. *International Journal of Blockchain Technologies and Applications*, 4(1), 20–33. <https://www.ijbta.com/vol4/IJBTA-V4N1-38.pdf>
16. Bhutta, M. N. M., Khwaja, A. A., Nadeem, A., Ahmad, H. F., Khan, M. K., Hanif, M., Song, H., Alshamari, M., & Cao, Y. (2021). A survey on blockchain technology: Evolution, architecture and security. *IEEE Access*, 9, 61048–61073. <https://doi.org/10.1109/ACCESS.2021.3072849>
17. Bennaceur, J., Zouaghi, W., Thabet, K., Bennour, A., Ben Djemaa, R., & Mabrouk, A. (2025). Trust-centric blockchain framework: A three-layered architecture for securing open and private systems. In *Proceedings of the 27th International Conference on Enterprise Information Systems (ICEIS 2025)* (Vol. 1, pp. 1097–1109). SCITEPRESS. <https://doi.org/10.5220/0013446900003929>
18. Ohize, H. O., Onumanyi, A. J., Umar, B. U., Ajao, L. A., Isah, R. O., Dogo, E. M., Nuhu, B. K., Olaniyi, O. M., Ambafi, J. G., Sheidu, V. B., & Ibrahim, M. M. (2024). Blockchain for securing electronic voting systems: a survey of architectures, trends, solutions, and challenges. *Cluster Computing*, 28(2). <https://doi.org/10.1007/s10586-024-04709-8>
19. Zheng, X., Lu, J., Gu, Z., Chen, J. (2026). TrustZone-Based WebAssembly Smart Contract Execution for Privacy-Preserving Blockchain. In: Zhang, X.S., Cao, S. (eds) 2025 International Conference on Blockchain and Web3.0 Technology Innovation and Application Exchange. BWTAC 2025. Communications in Computer and Information Science, vol 2715. Springer, Singapore. https://doi.org/10.1007/978-981-95-4142-3_13
20. Protzenko, J., Beurdouche, B., Merigoux, D., & Bhargavan, K. (2019, May). Formally verified cryptographic web applications in WebAssembly. In 2019 IEEE Symposium on Security and Privacy (SP) (pp. 52–67). IEEE. <https://doi.org/10.1109/SP.2019.00064>
21. Zheng, P., Zheng, Z., Luo, X., Chen, X., & Liu, X. (2018, May). A detailed and real-time performance monitoring framework for blockchain systems. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice* (pp. 134–143). ACM. <https://doi.org/10.1145/3183519.3183546>
22. Fan, C., Ghaemi, S., Khazaei, H., & Musilek, P. (2020). Performance Evaluation of Blockchain Systems: A Systematic Survey. *IEEE Access*, 8, 126927–126950. <https://doi.org/10.1109/access.2020.3006078>
23. Wang, W., Tu, C., Meng, Z., Huang, W., & Xiong, Y. (2025). WACANA: A Concolic Analyzer for Detecting On-chain Data Vulnerabilities in WASM Smart Contracts. *ACM Transactions on Software Engineering and Methodology*. <https://doi.org/10.1145/3757914>
24. Singh, S. (2023, February 16). Benchmarking Hyperledger Fabric 2.5 performance [Blog post]. Hyperledger Foundation. <https://www.lfdecentralizedtrust.org/blog/2023/02/16/benchmarking-hyperledger-fabric-2-5-performance>
25. Nakaike, T., Zhang, Q., Ueda, Y., Inagaki, T., & Ohara, M. (2020). Hyperledger Fabric performance characterization and optimization using GoLevelDB benchmark. In 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) (pp. 1–9). IEEE. <https://doi.org/10.1109/ICBC48266.2020.9169454>
26. Sousa, J., Bessani, A., & Vukolic, M. (2017). A Byzantine fault-tolerant ordering service for the Hyperledger Fabric blockchain platform. *arXiv*. <https://arxiv.org/abs/1709.06921>

References

1. Budzanivskyi, M. (2025, October 10). Blockchain Architecture Explained: Comprehensive Guide. Codebridge | Architecture-First Software & AI Systems Engineering. <https://www.codebridge.tech/articles/blockchain-architecture-explained-comprehensive-guide>
2. Xu, M., Guo, Y., Liu, C., Hu, Q., Yu, D., Xiong, Z., Niyato, D., & Cheng, X. (2024). Exploring Blockchain Technology through a Modular Lens: A Survey. *ACM Computing Surveys*. <https://doi.org/10.1145/3657288>
3. Wood, G. (2025). Ethereum: A secure decentralised generalised transaction ledger (Shanghai version efc5f9a). <https://ethereum.github.io/yellowpaper/paper.pdf>
4. di Angelo, M., Durieux, T., Ferreira, J. F., & Salzer, G. (2023). SmartBugs 2.0: An Execution Framework for Weakness Detection in Ethereum Smart Contracts. In J. Gurrola (Ed.), *Proceedings of the 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (pp. 2102–2105). IEEE. <https://doi.org/10.1109/ASE56229.2023.00060>
5. Guggenberger, T., Sedlmeir, J., Fridgen, G., & Luckow, A. (2022). An in-depth investigation of the performance characteristics of Hyperledger Fabric. *Computers & Industrial Engineering*, 108716. <https://doi.org/10.1016/j.cie.2022.108716>
6. Nakaike, T., Zhang, Q., Ueda, Y., Inagaki, T., & Ohara, M. (2020). Performance characterization of Hyperledger Fabric. In 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) (pp. 1–9). IEEE. <https://doi.org/10.1109/ICBC48266.2020.9169454>
7. Welcome to Hyperledger Fabric — hyperledger-fabricdocs master documentation. (n.d.). Welcome to Hyperledger Fabric – hyperledger-fabricdocs master documentation. <https://fab.readthedocs.io/en/latest/>
8. Honar Pajooh, H., Rashid, M. A., Alam, F., & Demidenko, S. (2022). Experimental Performance Analysis of a Scalable Distributed Hyperledger Fabric for a Large-Scale IoT Testbed. *Sensors*, 22(13), 4868. <https://doi.org/10.3390/s22134868>
9. Țălu, M. (2025). A Comparative Study of WebAssembly Runtimes: Performance Metrics, Integration Challenges, Application Domains, and Security Features. *Archives of Advanced Engineering Science*, 1–13. <https://doi.org/10.47852/bonviewaaes52024965>
10. Stryzhenko, Y. (2025). Optimization of web application performance through the integration of Web Assembly, blockchain technologies, and modern DevOps practices. *Futurity Proceedings*, 3. <https://doi.org/10.5281/zenodo.15662748>

11. Park, B., Song, J., & Seo, S. C. (2020). Efficient Implementation of a Crypto Library Using Web Assembly. *Electronics*, 9(11), 1839. <https://doi.org/10.3390/electronics9111839>
12. Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In 2017 IEEE 6th International Congress on Big Data (BigData Congress) (pp. 557–564). IEEE. <https://doi.org/10.1109/BigDataCongress.2017.85>
13. Ismail, L., & Materwala, H. (2019). A Review of Blockchain Architecture and Consensus Protocols: Use Cases, Challenges, and Solutions. *Symmetry*, 11(10), 1198. <https://doi.org/10.3390/sym11101198>
14. Kesoju, M. (2023). Review on blockchain technology: Architecture, characteristics, benefits, algorithms, challenges and applications. *Mesopotamian Journal of CyberSecurity*, 2023, 73–84. <https://doi.org/10.58496/MJCS/2023/012>
15. Staley, I. (2026). Use-Case-Driven Blockchain Architectures in Financial Enterprises: Permissioned, Consortium, Hybrid, and Public Networks. *International Journal of Blockchain Technologies and Applications*, 4(1), 20–33. <https://www.ijbta.com/vol4/IJBTA-V4N1-38.pdf>
16. Bhutta, M. N. M., Khwaja, A. A., Nadeem, A., Ahmad, H. F., Khan, M. K., Hanif, M., Song, H., Alshamari, M., & Cao, Y. (2021). A survey on blockchain technology: Evolution, architecture and security. *IEEE Access*, 9, 61048–61073. <https://doi.org/10.1109/ACCESS.2021.3072849>
17. Bennaceur, J., Zouaghi, W., Thabet, K., Bennour, A., Ben Djemaa, R., & Mabrouk, A. (2025). Trust-centric blockchain framework: A three-layered architecture for securing open and private systems. In *Proceedings of the 27th International Conference on Enterprise Information Systems (ICEIS 2025)* (Vol. 1, pp. 1097–1109). SCITEPRESS. <https://doi.org/10.5220/0013446900003929>
18. Ohize, H. O., Onumanyi, A. J., Umar, B. U., Ajao, L. A., Isah, R. O., Dogo, E. M., Nuhu, B. K., Olaniyi, O. M., Ambafi, J. G., Sheidu, V. B., & Ibrahim, M. M. (2024). Blockchain for securing electronic voting systems: a survey of architectures, trends, solutions, and challenges. *Cluster Computing*, 28(2). <https://doi.org/10.1007/s10586-024-04709-8>
19. Zheng, X., Lu, J., Gu, Z., Chen, J. (2026). TrustZone-Based WebAssembly Smart Contract Execution for Privacy-Preserving Blockchain. In: Zhang, X.S., Cao, S. (eds) 2025 International Conference on Blockchain and Web3.0 Technology Innovation and Application Exchange. BWTAC 2025. Communications in Computer and Information Science, vol 2715. Springer, Singapore. https://doi.org/10.1007/978-981-95-4142-3_13
20. Protzenko, J., Beurdouche, B., Merigoux, D., & Bhargavan, K. (2019, May). Formally verified cryptographic web applications in WebAssembly. In 2019 IEEE Symposium on Security and Privacy (SP) (pp. 52–67). IEEE. <https://doi.org/10.1109/SP.2019.00064>
21. Zheng, P., Zheng, Z., Luo, X., Chen, X., & Liu, X. (2018, May). A detailed and real-time performance monitoring framework for blockchain systems. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice* (pp. 134–143). ACM. <https://doi.org/10.1145/3183519.3183546>
22. Fan, C., Ghaemi, S., Khazaei, H., & Musilek, P. (2020). Performance Evaluation of Blockchain Systems: A Systematic Survey. *IEEE Access*, 8, 126927–126950. <https://doi.org/10.1109/access.2020.3006078>
23. Wang, W., Tu, C., Meng, Z., Huang, W., & Xiong, Y. (2025). WACANA: A Concolic Analyzer for Detecting On-chain Data Vulnerabilities in WASM Smart Contracts. *ACM Transactions on Software Engineering and Methodology*. <https://doi.org/10.1145/3757914>
24. Singh, S. (2023, February 16). Benchmarking Hyperledger Fabric 2.5 performance [Blog post]. Hyperledger Foundation. <https://www.lfdecentralizedtrust.org/blog/2023/02/16/benchmarking-hyperledger-fabric-2-5-performance>
25. Nakaike, T., Zhang, Q., Ueda, Y., Inagaki, T., & Ohara, M. (2020). Hyperledger Fabric performance characterization and optimization using GoLevelDB benchmark. In 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) (pp. 1–9). IEEE. <https://doi.org/10.1109/ICBC48266.2020.9169454>
26. Sousa, J., Bessani, A., & Vukolic, M. (2017). A Byzantine fault-tolerant ordering service for the Hyperledger Fabric blockchain platform. *arXiv*. <https://arxiv.org/abs/1709.06921>

Надійшла до редакції: 05.06.2026

Прийнята до друку: 15.09.2026

Опубліковано: 30.09.2026