DOI: 10.32347/2412-9933.2025.64.153-160
УДК 005.4:004.413

Червяков Костянтин Віталійович
Здобувач PhD кафедри програмного забезпечення автоматизованих систем,
https://orcid.org/0009-0001-3640-8273
Черкаського національного університету імені Богдана Хмельницького, Черкаси
Катаєв Дмитро Сергійович
Кандидат технічних наук, старший викладач кафедри інформаційних технологій проєктування
https://orcid.org/0009-0001-3055-6451
Черкаський державний технологічний університет, Черкаси

МОДЕЛІ ГНУЧКОГО УПРАВЛІННЯ ІТ-ПРОЄКТАМИ В УМОВАХ ВИСОКОЇ ЧАСТОТИ РЕЛІЗІВ

Анотація. *У сучасних умовах цифровізації економіки та зростання вимог до швидкості й якості розроблення програмного забезпечення особливої актуальності набуває гнучке управління ІТ-проєктами в умовах високої частоти релізів. Динамічні зміни ринкового середовища, ускладнення архітектури програмних систем і постійна трансформація потреб користувачів зумовлюють необхідність переходу від жорстко регламентованих моделей управління до адаптивних і гібридних підходів, які поєднують структурованість планування з гнучкістю реалізації. У статті здійснено комплексний аналіз сучасних моделей і методів гнучкого управління ІТ-проєктами з акцентом на практики безперервної інтеграції та безперервного постачання програмного забезпечення (Continuous Integration / Continuous Delivery, CI/CD). Проаналізовано предиктивні, гібридні та адаптивні підходи до управління проєктами з урахуванням рівня визначеності вимог, складності програмного продукту та інтенсивності змін у зовнішньому середовищі. Доведено, що адаптивні та гібридні моделі управління забезпечують кращу відповідність умовам високої частоти релізів завдяки коротким ітераціям, регулярному зворотному зв'язку та поступовій валідації результатів. Окрему увагу приділено аналізу узгодженості практик CI/CD з міжнародними стандартами управління проєктами та програмної інженерії, зокрема PMBOK і ISO/IEC/IEEE 32675:2022 «DevOps — Building Reliable and Secure Systems». Розкрито роль DevOps-підходів як інтеграційного механізму між процесами розроблення та експлуатації, що дає змогу підвищити керованість, стабільність і передбачуваність життєвого циклу програмного забезпечення. Обґрунтовано, що автоматизація процесів інтеграції, тестування та розгортання сприяє зменшенню операційних ризиків, зниженню впливу людського фактора та забезпечує повторюваність результатів. Проаналізовано умови та особливості впровадження CI/CD у діяльність українських ІТ-компаній з урахуванням рівня їхньої технологічної зрілості, організаційної структури та ресурсних обмежень. Зазначено, що системний підхід до інтеграції гнучких моделей управління та практик CI/CD створює передумови для формування стійких процесів розроблення й постачання ПЗ в умовах високої динаміки цифрового середовища.*

Ключові слова: безперервна інтеграція; безперервна доставка; ІТ-проєкт; гнучке управління; частота релізів; Agile; Lean

Вступ

У сучасному ІТ-середовищі гнучке управління проєктами поєднує адаптивні практики з формалізованими процесами, щоб утримувати баланс між змінними вимогами та обмеженими ресурсами. Спираючись на власні тези [1] в яких досліджував підходи Agile/Lean і практики раннього та частого постачання інкрементів, команди вибудовують короткі цикли зворотного зв'язку із стейкхолдерами, що дає змогу оперативно коригувати цілі й дорожню карту продукту. У методологічній рамці PMBOK [2] це відображається через тріаду керованості змін – тимчасовість (чіткі часові межі та етапність), унікальність (орієнтація на специфічний результат у мінливому контексті) та інтерактивність (поступовий розвиток із пріоритезацією поточних потреб). Така інтеграція забезпечує структуроване управління часом, якістю, ризиками й ресурсами, водночас залишаючи простір для адаптації; у результаті прискорюються релізи, підвищується передбачуваність і стабільність постачання, а якість продукту підтверджується регулярною апробацією на кожній ітерації.

Натомість практика безперервного постачання (Continuous Delivery, CD) інтегруються з низкою міжнародних стандартів, які закріплюють принципи автоматизації, надійності та постійного вдосконалення процесів. Зокрема, стандарт ISO/IEC/IEEE 32675:2022 DevOps – Building Reliable and Secure Systems [3] визначає вимоги та рекомендації щодо впровадження DevOps-підходів для забезпечення безперервності постачання, узгоджуючи технічні аспекти автоматизації з організаційними практиками управління якістю та безпекою.

Мета дослідження

Мета даного дослідження полягає у здійсненні комплексного аналізу сучасних наукових моделей та методів гнучкого управління IT-проєктами з акцентом на практики безперервної інтеграції та постачання (continuous integration/continuous deployment, CI/CD), визначенні їх узгодженості як з міжнародними стандартами (зокрема ISO/IEC/IEEE 32675:2022), так і з відкритими практиками та фреймворками DevOps/Agile, а також на обґрунтуванні перспектив їхнього застосування в українських IT-компаніях для підвищення ефективності, адаптивності та конкуренто-спроможності у динамічних умовах цифрової економіки.

Аналіз публікацій

В умовах зростання частоти та непередбачуваності змін постала потреба в гнучкості управління IT-проєктами. У контексті управління IT-проєктами гнучкість організації полягає у здатності своєчасно трансформувати структуру, масштаби діяльності та функціональні процеси відповідно до змін зовнішнього середовища й внутрішніх потреб, зберігаючи при цьому системну цілісність і орієнтацію на розвиток, ефективність та результативність [4].

Основою моделей гнучкого управління є Agile, який охоплює такі підходи Scrum, Kanban та Lean, вони орієнтовані на підвищення адаптивності організацій, здатність швидко реагувати на зовнішні та внутрішні зміни, а також на забезпечення безперервного вдосконалення процесів [5]. У рамках Lean-менеджменту ключовою ідеєю є створення максимальної цінності для клієнта при мінімальних втратах ресурсів команди. В посібнику [6] про Lean підкреслюється важливість усунення всіх видів неефективності – простоїв, надлишкових дій, зайвої документації чи дублювання зусиль. У центрі цієї філософії – п’ять принципів: визначення цінності для клієнта, виявлення потоку створення цінності, забезпечення безперервного потоку, реалізація принципу витягування та постійне вдосконалення процесів.

У поєднанні з Agile-практиками Lean дозволяє підвищити операційну ефективність, скоротити час поставки продукту й забезпечити стале зростання якості результатів. Таким чином, Lean виступає не лише інструментом оптимізації процесів, а й філософією управління, орієнтованою на цінність для користувача та гнучке реагування на зміни.

Agile – це узагальнений термін для набору методологій та практик, що ґрунтується на цінностях та принципах, викладених у Маніфесті гнучкої розробки програмного забезпечення (Agile Manifesto) [7] та дванадцяти принципах, які його супроводжують. Його сутність полягає в створенні програмного забезпечення через ітеративні процеси, тісну співпрацю з користувачами та постійну адаптацію до змін [8].

Методологія Agile виникла як реакція на обмеження предиктивного підходу (waterfall), що бере свій початок від виробничих методів Генрі Форда та був згодом адаптований до розробки програмного забезпечення. Починаючи з 2001 р., Agile швидко набував популярності у сфері програмної інженерії та управління проєктами, набувши численних підходів та практик [9].

Імпульсом до його виникнення стала критика тривалих життєвих циклів та низької гнучкості каскадної моделі. Уже на початку 1990-х років стало очевидним, що між моментом виявлення бізнес-потреби та поставкою готового програмного продукту кінцевому споживачеві може минати кілька років. За цей час умови на ринку та стратегічні пріоритети організацій змінювались настільки, що значна частина продукту втрачала актуальність і скасовувалася до завершення. Така ситуація приводила до значних втрат часу та ресурсів що й зумовило пошук альтернативних підходів до організації процесу розробки [10].

DevOps – це методологія, яка містить набір практик, що поєднує розробку ПЗ (Dev) та IT-операції (Ops) [11]. Основною метою DevOps є скорочення циклу розробки ПЗ і подбати про забезпечення безперервного постачання програмних компонентів на кінцеве програмне середовище. Автоматизовані процеси (CI/CD) сприяють зменшенню витрат і дозволяють скоротити терміни впровадження нового ПЗ. Інструменти DevOps дають можливість оптимізувати робочий час і уникнути зайвого ручного втручання.

Виклад основного матеріалу

Сучасна практика управління IT-проєктами базується на використанні трьох основних підходів до розробки: предиктивного, гібридного та адаптивного. Вибір відповідного підходу зумовлюється рівнем визначеності вимог, швидкістю

змін середовища, складністю технічної реалізації та очікуваною частотою постачання цінності користувачам.

Підхід до розробки – це засіб, який використовується для створення та розвитку продукту, послуги або результату впродовж життєвого циклу проєкту. Існують різні підходи до розробки. Зазвичай використовують три підходи: предиктивний, гібридний та адаптивний.

Предиктивний підхід (waterfall) застосовується в умовах високої стабільності вимог та процесів. Він передбачає детальне планування на початковому етапі проєкту, що дає змогу жорстко контролювати строки, обсяг і ресурси. Такий підхід менш придатний для середовищ з високою динамікою змін, характерних для IT-сектору.

Гібридний підхід поєднує елементи адаптивних та предиктивних моделей, забезпечуючи баланс між структурованістю та гнучкістю. Він доцільний у випадках, коли певні частини продукту мають чітко визначені характеристики, а інші – потребують ітеративного уточнення через зворотний зв'язок із користувачами.

Адаптивний підхід (Agile), натомість, базується на принципах гнучкого управління: короткі ітерації, часті поставки інкрементів продукту, тісна взаємодія з замовником і швидке реагування на зміну вимог. У контексті практик Continuous Delivery (CD) саме адаптивні та гібридні моделі вважаються найбільш ефективними, оскільки вони дозволяють забезпечити високу частоту релізів, швидке тестування змін та автоматизоване постачання.

У таблиці наведено порівняльний аналіз основних параметрів виконання типових IT-проєктів залежно від обраної методології – предиктивної (waterfall), гібридної та адаптивної (Agile). Як показано, у предиктивному підході рівень визначеності вимог є високим, проте частота релізів низька, а зворотний зв'язок формується лише наприкінці, що знижує гнучкість процесу. Гібридна модель характеризується частковою визначеністю вимог, помірною частотою релізів і періодичним зворотним зв'язком на окремих етапах, забезпечуючи баланс між контрольованістю та адаптивністю. Натомість адаптивний підхід (Agile) передбачає змінність вимог, високу частоту релізів і постійний зворотний зв'язок, що сприяє найвищому рівню актуальності процесів безперервного постачання (CD) та забезпечує максимальну гнучкість розробки.

В статті [12] авторка дійшла до висновку про еволюцію управління проєктами від жорстко стандартизованих підходів до контекстно-орієнтованих гібридних моделей, які враховують очікувань швидкості виходу версій та невизначеність вимог замовника. Автор підкреслює, що універсального методу управління не існує, а традиційні підходи, зокрема Waterfall, поступово втрачають ефективність у ситуаціях, коли потреби клієнта змінюються протягом життєвого циклу продукту.

Дослідження підтверджує, що понад 80% керівників проєктів у сучасних організаціях застосовують гібридні моделі, комбінуючи структурованість Waterfall із гнучкістю Agile, Scrum чи Kanban. Найбільш поширеною є інтегрована модель Water-Scrum-Fall, у якій етапи планування залишаються формалізованими, а розроблення та постачання виконуються ітеративно з постійним зворотним зв'язком. Такі моделі підсилюються практиками Design Thinking та DevOps, що сприяє створенню адаптивного управлінського середовища з орієнтацією на цінність для користувача.

Отримані результати підтверджують зміщення фокусу від концепції «єдиної найкращої практики» до умовного управління, яке поєднує різні стандарти, методології та інструменти відповідно до типу проєкту, рівня невизначеності та зрілості команди. Такий підхід узгоджується з сучасними тенденціями, відображеними у PMI Disciplined Agile (DA) [13], і може стати методологічною основою для побудови гібридних моделей управління IT-проєктами в умовах безперервного постачання, що є ключовим напрямом вашої наукової статті.

Сучасні дослідження підтверджують, що практики Continuous Integration (CI) та Continuous Delivery/Deployment (CD) стали ключовими для компаній, які прагнуть скоротити час виходу нових функціональних оновлень та підвищити якість програмного забезпечення. Вони впроваджують гібридні та адаптивні моделі управління, забезпечуючи швидкий фідбек та високу частоту релізів [14].

Таблиця – ***Порівняльна таблиця сфер виконань у типовому IT-проєкті***

Параметр / методологія	Предиктивний (waterfall)	Гібридний	Адаптивний (Agile)
Визначеність вимог	Висока	Часткова	Низька/змінна
Частота релізів	Рідко	Помірна	Часто
Зворотний зв'язок	На кінці	В обраних етапах	Постійно
Актуальність CD	Помірна	Висока	Найвища
Гнучкість	Низька	Середня	Висока

Критерії ефективності та механізми управління ризиками в умовах інтенсивних релізів

Перехід до моделей гнучкого управління з високою частотою релізів (Daily чи Hourly releases) вимагає не лише зміни методології, а й впровадження специфічної системи показників для оцінки результативності та раннього виявлення деструктивних відхилень. В умовах, коли часовий лаг між написанням коду та його потраплянням у продуктивне середовище мінімізується, традиційні методи контролю якості стають недостатніми.

Для оцінки ефективності впроваджених моделей управління доцільно використовувати групу метрик DORA (*DevOps Research and Assessment*), які дозволяють кількісно виміряти зрілість процесів CI/CD:

– *Deployment Frequency (DF)* – частота успішних розгортань, що безпосередньо корелює з адаптивністю проєкту;

– *Lead Time for Changes (LTTC)* – час від моменту фіксації змін у репозиторії до їх успішного розгортання;

– *Change Failure Rate (CFR)* – відсоток релізів, що призвели до критичних помилок або потребували відкату;

– *Failed Service Recovery Time (FSRT)* – середній час відновлення працездатності системи після збою.

Особливістю управління в умовах високої частоти релізів є трансформація ризик-менеджменту. Якщо у предиктивних моделях ризики оцінюються на етапі планування фаз, то в адаптивних системах управління ризиками інтегрується безпосередньо в конвеєр постачання. Ключовим механізмом тут виступає концепція «Shift-Left», яка передбачає перенесення процедур контролю безпеки та тестування на якомога ранні етапи життєвого циклу.

У межах розробки гнучких моделей управління нами виокремлено три рівні стабілізації процесу при інтенсивному постачанні:

1. *Технологічний рівень:* автоматизація «димових» тестів (smoke testing) та модульного покриття. Кожна ітерація розглядається як потенційно готовий до релізу артефакт, що мінімізує накопичення інтеграційного боргу.

2. *Процесний рівень:* впровадження тригераційних механізмів зупинки конвеєра при виявленні критичного дефекту (аналог принципу «Jidoka» у Lean). Це запобігає поширенню помилки на наступні етапи та змушує команду зосередитись на стабілізації поточного інкременту.

3. *Когнітивний рівень:* зміна організаційної культури в бік «No-Blame Culture». В умовах високої швидкості релізів помилки неминучі, тому управління фокусується не на пошуку винних, а на швидкості виявлення та виправлення дефекту.

Таким чином, ефективність моделі гнучкого управління в умовах високої частоти релізів визначається не лише швидкістю доставки коду, а й здатністю системи до самовідновлення та мінімізації впливу людського фактора через глибоку автоматизацію управлінських рішень. Це дозволяє українським ІТ-компаніям масштабувати процеси без втрати якості, забезпечуючи сталий розвиток програмних продуктів у висококонкурентному середовищі.

У розробці ПЗ, CI/CD – це синтез безперервної інтеграції (continuous integration) та безперервного розгортання (continuous delivery). Його головна мета полягає в оптимізації та прискоренні життєвого циклу розробки програмного забезпечення. Команди, які працюють за принципами Agile та впроваджують безперервне постачання, досягають більшої гнучкості, зниження кількості відмов, а також підвищеного контролю над процесом доставки продукту кінцевим користувачам [15].

Continuous Integration (CI) – це практика автоматизованого об’єднання змін у програмному коді, яка передбачає регулярне інтегрування внесених розробниками правок у спільну гілку. Кожне таке злиття супроводжується запуском автоматизованих тестів для перевірки коректності та надійності внесених змін.

Метою CI є усунення проблеми конфліктів між паралельними гілками розробки. У традиційному підході, коли інтеграція всіх змін відбувалася одночасно у визначений день, процес був трудомістким і супроводжувався великою кількістю помилок [16].

Безперервна інтеграція є важливою для розробки програмного забезпечення використовуючи методологію Agile, оскільки вона покращує процес розробки завдяки впровадженню невеликих ітераційних змін та забезпеченню стабільності коду. Використовуючи інструменти безперервної інтеграції, команди можуть автоматизувати збірки, тести та розгортання, покращуючи загальний робочий процес розробки.

Ця інтеграція не тільки сприяє ефективному перегляду коду, але й підтримує стабільність середовища розробки. Завдяки впровадженню CI в гнучкі методології організації досягають чудової якості програмного забезпечення та безперебійного безперервного процесу інтеграції, що зрештою призводить до кращої продуктивності DevOps і високоякісної доставки програмного забезпечення [17].

Основні елементи, з яких складається безперервне постачання:

– централізація коду для зручності злиття змін;

– тестування для перевірки якості коду;

– автоматизована збірка – компіляція вихідного коду у працюючий артефакт.

Безперервна доставка та розгортання (Continuous Delivery/Deployment, CD) є логічним продовженням практики безперервної інтеграції. Вона передбачає автоматичне розгортання змін коду у тестових або продуктивних середовищах після успішного проходження попередньо визначених перевірок. Такий підхід забезпечує постійну готовність програмного забезпечення до розгортання, що значно скорочує час постачання нових функціональних можливостей та виправлення помилок кінцевим користувачам.

Залежно від рівня автоматизації процес може бути організований як безперервна доставка, коли рішення про випуск приймається вручну, або як безперервне розгортання, що передбачає повністю автоматизоване перенесення змін у продуктивне середовище [18].

Основні елементи з яких складається безперервна доставка:

– виконання збірки та запуск тестування після прийняття змін коду;

– процес розгортання виконується автоматично за допомогою скрипту;

– розділення програми та змінні чи параметри середовища;

– створення автоматичного пайплайну.

Стратегії архітектурної трансформації та інфраструктурного забезпечення інтенсивних циклів розробки

Реалізація моделей гнучкого управління в проєктах із високою частотою релізів вимагає докорінної архітектурної трансформації програмних систем. Монолітні структури стають перешкодою для частого розгортання через високу взаємозалежність компонентів. Найбільш ефективним підходом є перехід до мікросервісної архітектури, де кожен функціональний блок розглядається як незалежна одиниця розгортання. Це дозволяє командам працювати над сервісами ізольовано, мінімізуючи ризики регресійних помилок.

Технологічною основою такої гнучкості виступає контейнеризація (Docker) та оркестрація (Kubernetes), які забезпечують ідентичність середовищ розробки та експлуатації. Важливим інструментом є концепція Infrastructure as Code (IaC), що дозволяє автоматично розгортати необхідні потужності під кожен цикл тестування. Для керованості релізами особливого значення набуває паттерн Feature Flags (функціональні прапорці). Він дозволяє відокремити технічне розгортання коду від активації функцій для користувача. Менеджер проєкту може здійснювати «темні запуски», проводити canary-релізи та миттєво вимикати проблемний функціонал через панель керування без процедури повного відкату (rollback) версії, що значно підвищує структурну стійкість проєкту.

У дисертації [19] у результаті проведеного дослідження сформульовано рекомендації щодо вибору інструментів залежно від цілей, архітектури продукту, ресурсів і контексту організації. Передусім важливо чітко визначити бізнес-мету впровадження автоматизації та обрати рішення, яке найкраще відповідає конкретним завданням – від безперервної інтеграції (наприклад, Jenkins) до доставки чи розгортання (Spinnaker, GitLab CI/CD). Вибір технологій повинен враховувати рівень технічної підготовки команди, швидкість її адаптації до нових інструментів, а також обмеження безпеки й політику зберігання даних компанії (локальні чи хмарні рішення). Не менш важливо оцінювати бюджетні можливості та розмір проєкту: для стартапів ефективнішими можуть бути хмарні CI/CD-сервіси, тоді як великі організації частіше використовують гібридні або локальні інфраструктури.

Крім того, доцільно поєднувати кілька інструментів для різних сценаріїв використання, що підвищує надійність і безперервність бізнес-процесів. Вибір системи має враховувати архітектуру продукту, наявність механізмів моніторингу, автоматичного виявлення проблем і можливостей самовідновлення процесів. Практичні результати дослідження, доповнені фінансово-економічними розрахунками та аналізом ризиків, можуть бути використані як методичні рекомендації для впровадження CI/CD у IT-компаніях, що прагнуть підвищити ефективність, стабільність та керованість процесів безперервного постачання.

У ході дослідження [20] було запропоновано модель впровадження CI/CD для оптимізації управління IT-проєктами. Результати показали, що застосування підходів безперервної інтеграції та доставки суттєво підвищує ефективність управління, забезпечуючи автоматизацію процесів розробки, тестування та розгортання програмного забезпечення.

Було встановлено, що вибір відповідних інструментів та налаштування процесів CI/CD є критичними чинниками успішного впровадження методології. Окрему увагу приділено забезпеченню якості коду та безперебійній роботі системи, що є основою стабільності життєвого циклу програмного продукту.

Організаційні бар’єри та економічна доцільність впровадження гнучких моделей

Ефективність моделей управління критично залежить від трансформації організаційної культури. Ключовим бар’єром є наявність «силосів» – розрізнених відділів розробки та експлуатації з конфліктними KPI. Перехід до культури DevOps

передбачає колективну відповідальність за продукт. В українських реаліях основними стримуючими факторами залишаються: низький рівень автоматизації тестів, що змушує команди повертатися до ручного тестування, та ігнорування безпеки (Security Debt). Інтеграція практик DevSecOps дозволяє автоматизувати перевірку вразливостей без зупинки конвеєра постачання.

З економічного погляду, впровадження високочастотних релізів суттєво знижує вартість володіння продуктом (TCO). Економія досягається за рахунок зменшення витрат на виправлення помилок, виявлених на ранніх етапах. Крім того, це дозволяє швидше реалізувати концепцію MVP (Minimum Viable Product) та отримувати прибуток раніше, знижуючи фінансові ризики розробки незатребуваного функціоналу. Оптимізація ресурсів через автоматизацію рутинних операцій дозволяє фокусувати інтелектуальний капітал команди на створенні унікальної цінності, що забезпечує стратегічну перевагу вітчизняних IT-компаній на глобальному ринку.

Висновки

Аналіз джерел та отриманих результатів щодо моделі гнучкого управління IT-проєктами в умовах високої частоти релізів дозволяє зробити низку важливих висновків і практичних рекомендацій.

Гнучкі підходи, зокрема Agile, CI/CD, виявилися найбільш ефективними в динамічному середовищі, оскільки забезпечують швидке реагування на зміни, зменшення ризиків та підвищення якості програмних продуктів.

Міжнародні стандарти, серед яких PMBOK та ISO/IEC/IEEE 32675:2022, формують методологічну основу для поєднання адаптивних практик із формалізованими процесами управління, що сприяє підвищенню надійності, передбачуваності та безпеки проєктів. Часті ітерації та короткі цикли постачання дозволяють досягти більшої відповідності програмного забезпечення очікуванням користувачів, хоча водночас вимагають зрілої інфраструктури автоматизації тестування й розгортання.

Для українських IT-компаній перспективним є впровадження гібридних та адаптивних моделей управління, які поєднують структурованість із високим рівнем гнучкості та формують підґрунтя для зростання конкурентоспроможності на глобальному ринку.

Додатково проведений аналіз літератури та міжнародних стандартів підтвердив, що:

– Гнучкість управління є ключовою характеристикою сучасних організацій, які прагнуть забезпечити стійкість до змін середовища та технологічної еволюції.

– Agile-підходи, зокрема Scrum, Kanban та Lean, виступають базовими моделями, орієнтованими на постійне вдосконалення процесів і створення максимальної цінності для клієнта при мінімальних втратах ресурсів.

– Lean-менеджмент як частина гнучких методологій підсилює ефективність управління IT-проєктами завдяки усуненню неефективності, оптимізації потоків створення цінності та філософії постійного вдосконалення.

– Еволюція управлінських моделей демонструє відхід від універсального підходу до контекстно-орієнтованих гібридних систем управління, які поєднують елементи Waterfall, Agile, DevOps та Design Thinking.

– Впровадження CI/CD підтвердило свою ключову роль у підвищенні стабільності, якості коду та швидкості постачання функціональності, особливо в умовах високої частоти релізів.

Отже, в подальшому дослідженні варто зосередитись на формуванні методичних рекомендацій з впровадження гібридних підходів, адаптованих до українського IT-ринку, а також на розробці моделей оцінки ефективності CI/CD і Lean-практик для підвищення зрілості організаційних процесів у галузі програмної інженерії.

Список літератури

1. Червяков К. В., Супруненко О. О. Інформаційні моделюючі технології, системи та комплекси. Проблеми управління програмними проєктами в умовах безперервного постачання. 2024. 268 с.

2. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). Seventh Edition and The Standard for Project Management. Newtown Square, Pennsylvania, USA : Project Management Institute, Inc., 2021. 250 p.

3. ISO/IEC 20582:2025(en). Software and systems engineering – Capabilities of build and deployment tools. URL: https://www.iso.org/obp/ui/ru/#iso:std:iso-iec:20582:ed-1:v1:en (дата звернення: 22.01.2026).

4. Яворський Р. Т., Самуляк В. Ю. Теоретичні основі та моделі гнучкого управління розвитком підприємства. *Проблеми сучасних трансформацій. Серія: економіка та управління*. 2025. 19. URL: https://doi.org/10.54929/2786-5738-2025-19-04-03 (дата звернення: 22.01.2026).

5. Scrum, Agile чи Kanban: Як обрати ідеальну методологію для вашого проєкту? URL: https://business-broker.com.ua/blog/scrum-agile-kanban-skhozhosti-vidminnosti-ta-porady-dlia-vyboru/ (дата звернення: 22.01.2026).

6. Методології Agile та Lean: Повний посібник. URL: https://worksection.com/ua/blog/agile-vs-lean.html (дата звернення: 22.01.2026).

7. Manifesto for Agile Software Development. URL: https://agilemanifesto.org/ (дата звернення: 22.01.2026).

8. Agile Alliance. What is Agile? Agile 101. URL: https://agilealliance.org/agile101/ (дата звернення: 22.01.2026).

9. Основи agile-методології: як впровадити в проєкти. URL: https://proit.com.ua/news/rozuminnya-agile-metodologiyi-osnovy-ta-zastosuvannya/ (дата звернення: 22.01.2026).

10. What is agile methodology? URL: https://www.redhat.com/en/topics/devops/what-is-agile-methodology (дата звернення: 22.01.2026).

11. IBM. What Is DevOps? URL: https://www.ibm.com/think/topics/devops (дата звернення: 22.01.2026).

12. Кордунова Ю., Смотр О., Кокотко І., Малець Р. Аналіз традиційного та гнучкого підходів до створення програмного забезпечення в динамічних умовах. *Управління розвитком складних систем*. 2021. 47. С. 71–77. URL: https://doi.org/10.32347/2412-9933.2021.47.71-77 (дата звернення: 22.01.2026).

13. Introduction to Disciplined Agile. URL: https://www.pmi.org/disciplined-agile/introduction-to-disciplined-agile (дата звернення: 22.01.2026).

14. Chava A. CI/CD and Automation in DevOps Engineering. *Asian Journal of Research in Computer Science*. 2024. Vol. 17(11). 520. URL: https://doi.org/10.9734/ajrcos/2024/v17i11520 (дата звернення: 22.01.2026).

15. Кузьміних В. О., Коваль О. В., Тараненко Р. А. Моделі та засоби управління ІТ-проєктами. Електронне мережеве навчальне видання. 2024. [Уточніть видавництво/місто].

16. What is CI/CD? URL: https://www.redhat.com/en/topics/devops/what-is-ci-cd (дата звернення: 22.01.2026).

17. Continuous integration in Agile development. URL: https://about.gitlab.com/topics/ci-cd/continuous-integration-agile/ (дата звернення: 22.01.2026).

18. CI/CD 101: Understanding Continuous Integration and Delivery. URL: https://www.armorcode.com/blog/ci-cd-101-understanding-continuous-integration-and-delivery (дата звернення: 22.01.2026).

19. Дашкова Є. А. Дослідження сучасних засобів та методів CI/CD з використанням Kubernetes : магістерська дис. Київ : НТУУ «КПІ ім. Ігоря Сікорського», 2020. 94 с.

20. Кравчук О. Модель впровадження CI/CD для оптимізації управління ІТ-проєктами. *Measuring and computing devices in technological processes*. 2023. 3. С. 73–82. URL: https://doi.org/10.31891/2219-9365-2023-75-8 (дата звернення: 22.01.2026).

Стаття надійшла до редколегії 06.12.2025

Cherviakov Kostiantyn
PhD student of the Department of Software of Automated Systems,
https://orcid.org/0009-0001-3640-8273
Bohdan Khmelnytsky National University of Cherkasy, Cherkasy
Kataiev Dmytro
PhD (Engineering), Associate Professor of the Department of Information Technology Design,
https://orcid.org/0009-0001-3055-6451
Cherkasy State Technological University, Cherkasy

MODELS OF AGILE MANAGEMENT OF IT PROJECTS IN CONDITIONS OF HIGH RELEASE FREQUENCY

Abstract. *Under the current conditions of economic digitalization and increasing demands for software development speed and quality, agile IT project management in high-release-frequency environments is becoming particularly relevant. Dynamic market changes, the increasing complexity of software architectures, and the continuous transformation of user needs necessitate a transition from rigid, highly regulated management models to adaptive and hybrid approaches that combine structured planning with flexible implementation. The article provides a comprehensive analysis of modern models and methods for agile IT project management, with a focus on Continuous Integration and Continuous Delivery (CI/CD) practices. Predictive, hybrid, and adaptive project management approaches are analyzed, considering the level of requirement certainty, software product complexity, and the intensity of external environmental changes. It is demonstrated that adaptive and hybrid management models ensure better alignment with high-release-frequency conditions through short iterations, regular feedback, and incremental result validation. Particular attention is paid to analyzing the consistency of CI/CD practices with international project management and software engineering standards, specifically PMBOK and ISO/IEC/IEEE 32675:2022 "DevOps — Building Reliable and Secure Systems." The role of DevOps approaches as an integration mechanism between development and operations processes is revealed, enabling improved controllability, stability, and predictability of the software lifecycle. It is substantiated that the automation of integration, testing, and deployment processes contributes to the mitigation of operational risks, reduction of human factor dependency, and increased repeatability of results. The conditions and peculiarities of CI/CD implementation within Ukrainian IT companies are analyzed, taking into account their technological maturity levels, organizational structures, and resource constraints. It is noted that a systemic approach to integrating agile management models and CI/CD practices creates the prerequisites for establishing sustainable software development and delivery processes in a highly dynamic digital environment.*

Keywords: continuous integration; continuous delivery; IT project; methods; models; analysis

References

1. Cherviakov, K. V., & Suprunenko, O. O. (2024). *Informatsiini modeliuiuchi tekhnolohii, systemy ta kompleksy. Problemy upravlinnia prohramnymy proektamy v umovakh bezperervnoho postachannia* [Information modeling technologies, systems and complexes. Problems of software project management in conditions of continuous delivery]. 268 p.
2. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*. (2021). Seventh Edition and The Standard for Project Management. Newtown Square, Pennsylvania, USA: Project Management Institute, Inc. 250 p.
3. ISO/IEC 20582:2025(en). Software and systems engineering — Capabilities of build and deployment tools. URL: https://www.iso.org/obp/ui/ru/#iso:std:iso-iec:20582:ed-1:v1:en
4. Yavorskyi, R. T., & Samuliak, V. Yu. (2025). Theoretical bases and models of agile management of enterprise development. *Problems of Modern Transformations. Series: Economics and Management*, 19. https://doi.org/10.54929/2786-5738-2025-19-04-03
5. Scrum, Agile or Kanban: How to choose the ideal methodology for your project? URL: https://business-broker.com.ua/blog/scrum-agile-kanban-skhozhosti-vidminnosti-ta-porady-dlia-vyboru/
6. Agile and Lean Methodologies: A Complete Guide. URL: https://worksection.com/ua/blog/agile-vs-lean.html
7. Manifesto for Agile Software Development. URL: https://agilemanifesto.org/
8. Agile Alliance. What is Agile? Agile 101. URL: https://agilealliance.org/agile101/
9. Basics of agile methodology: how to implement it in projects. URL: https://proit.com.ua/news/rozuminnya-agile-metodologiyi-osnovy-ta-zastosuvannya/
10. What is agile methodology? URL: https://www.redhat.com/en/topics/devops/what-is-agile-methodology
11. IBM. What Is DevOps? URL: https://www.ibm.com/think/topics/devops
12. Kordunova, Yu., Smotr, O., Kokotko, I., & Malets, R. (2021). Analysis of traditional and agile approaches to software creation in dynamic conditions. *Upravlinnia rozvytkom skladnykh system*, 47, 71–77. https://doi.org/10.32347/2412-9933.2021.47.71-77
13. Introduction to Disciplined Agile. URL: https://www.pmi.org/disciplined-agile/introduction-to-disciplined-agile
14. Chava, A. (2024). CI/CD and Automation in DevOps Engineering. *Asian Journal of Research in Computer Science*, 17(11), 520. https://doi.org/10.9734/ajrcos/2024/v17i11520
15. Kuzminykh, V. O., Koval, O. V., & Taranenko, R. A. (2024). *Modeli ta zasoby upravlinnia IT-proektamy* [Models and tools for IT project management]. Electronic network educational edition.
16. What is CI/CD? URL: https://www.redhat.com/en/topics/devops/what-is-ci-cd
17. Continuous integration in Agile development. URL: https://about.gitlab.com/topics/ci-cd/continuous-integration-agile/
18. CI/CD 101: Understanding Continuous Integration and Delivery. URL: https://www.armorcode.com/blog/ci-cd-101-understanding-continuous-integration-and-delivery
19. Dashkova, Ye. A. (2020). *Doslidzhennia suchasnykh zasobiv ta metodiv CI/CD z vykorystanniam Kubernetes* [Research of modern CI/CD tools and methods using Kubernetes] (Master's thesis). Kyiv: NTUU "Igor Sikorsky Kyiv Polytechnic Institute". 94 p.
20. Kravchuk, O. (2023). CI/CD implementation model for optimization of IT project management. *Measuring and computing devices in technological processes*, 3, 73–82. https://doi.org/10.31891/2219-9365-2023-75-8

Посилання на публікацію

APA Cherviakov, K., & Kataiev, D. (2025). Models of agile management of IT projects in conditions of high release frequency. *Management of Development of Complex Systems*, 64, 153–160, dx.doi.org\10.32347/2412-9933.2025.64.153-160.

ДСТУ Червяков К. В., Катаєв Д. С. Моделі гнучкого управління ІТ-проєктами в умовах високої частоти релізів. *Управління розвитком складних систем.* Київ, 2025. № 64. С. 153 – 160, dx.doi.org\10.32347/2412-9933.2025.64.153-160.