DOI
УДК 004.4:004.8

О.В. Нестеренко, д-р техн. наук,
e-mail: oleksandr_nesterenko@feu.edu.ua; orcid.org/0000-0001-5329-889X
В.В. Федоров, канд. фіз.-мат. наук,
e-mail: volodymyr.fedorov@feu.edu.ua; orcid.org/0009-0004-2901-3646
П.П. Яцук, канд. техн. наук,
e-mail: petro_yatsuk @feu.edu.ua; orcid.org/ 0009-0002-7124-4849
Перший європейський університет
Україна, Київ, 03187, пр-т Академіка Глушкова, 42

Штучний інтелект у командній роботі в програмній інженерії

Штучний інтелект (ШІ) стає невід'ємною частиною програмної інженерії, розширюючи можливості автоматизації процесів розробки і супроводження програмного забезпечення упродовж усього його життєвого циклу. В сучасних умовах набирає актуальності питання щодо командної роботи людина-машина (Human-machine teaming, HMT), тобто взаємодія людей і ШІ як членів команди. Проаналізовано основні виклики, пов'язані зі співпрацею людини та ШІ, зокрема з урахуванням методології групової динаміки. Запропоновано трансформацію моделі Такмена, що дозволяє описати еволюцію команд програмної інженерії як процес співрозвитку людини та ШІ. Також запропоновано розширення моделі ролей Белбіна, що посилає до формування гібридної системи розподілу ролей у командах програмної інженерії інтегрованих з ШІ. В результаті побудовано інтегрований фреймворк, що поєднує модель стадій розвитку команд, рольову модель та можливості ШІ, який дозволяє описати команди програмної інженерії як гібридні соціо-технічні системи. Запропоновано концептуальну модель ШІ-підтриманої командної роботи та її деталізацію UML-діаграмами. Результати проведеного дослідження підтверджують, що ШІ стає інтелектуальним партнером команди. Це призводить до трансформації командної взаємодії та її перспективного розвитку.

К л ю ч о в і с л о в а: людино-машинна взаємодія, групова динаміка, модель Такмена, модель Белбіна, UML, інтелектуалізація, програмне забезпечення.

Використання штучного інтелекту (ШІ) в програмній інженерії є новим напрямком досліджень, який останнім часом отримує все більше уваги. ШІ стає невід'ємною частиною сучасної програмної інженерії, розширюючи

Стаття поширюється на умовах ліцензії відкритого доступу CC BY 4.0

можливості автоматизації й підтримки технічних рішень на усіх етапах життєвого циклу програмного забезпечення (ПЗ). Застосування таких технологій, як машинне навчання (ML), глибинне навчання (DL), великі мовні моделі (LLM) та обробка природної мови (NLP) змінює традиційні підходи до розробки ПЗ та його супроводження. Перспективними ділянками застосування ШІ є генерування коду, прогнозування помилок (дефектів) для покращення якості ПЗ, управління життєвим циклом розробки ПЗ та оцінка зусиль, обробка природної мови в інженерії вимог, а також пошук даних з репозитаріїв ПЗ [1]. Найпоширеніше використання ШІ охоплює такі фази програмної інженерії, як аналіз вимог, проектування, розробка, тестування, випуск та обслуговування [2].

В програмній інженерії як суттєвий чинник ефективності процесів розробки ПЗ розглядається вплив поточного стану команди розробників. Сучасні команди працюють у складних колаборативних середовищах, що формуються з систем керування кодом, управління задачами, CI/CD платформ, комунікаційних інструментів, тощо. Наразі застосування ШІ в цих складових є вже не окремими локальними інструментами підтримки діяльності, а невід'ємною частиною воркфлоу. В програмній інженерії визначається, що реалізація програмних проектів має базуватися не лише на їх параметрах та моделях предметної області як основи бази знань для підтримки управлінських рішень, а й на особистісних характеристиках програмістів відповідно до методології групової динаміки та комунікацій [3]. В сучасних умовах набирає актуальності питання щодо командної роботи людина-машина (Human-machine teaming, HMT), тобто взаємодія людей і ШІ як членів команди. Дослідники пропонують поняття «колаборативного інтелекту» і доводять, що такі системи мають широкий спектр переваг, включаючи ефективність, якість, креативність, безпеку та задоволення людини [4]. Роль комп'ютерів у підтримці співпраці людей більше не обмежується допоміжною роллю, а й виконує функцію кооперативного учасника, що створює нові виклики для вивчення «колаборативного інтелекту», починаючи від базових теорій співпраці та закінчуючи ключовими технологіями [5].

Варто зазначити, що з-за ефекту «чорної скриньки» моделі та технології ШІ все ще далекі від широкого практичного впровадження в програмній інженерії. Зокрема ШІ викликає багато занепокоєнь щодо його відповідального використання. Результати багатьох досліджень свідчать не лише про зростаючу інтеграцію ШІ в різні галузі людської діяльності, а й вказують, у першу чергу, на збільшення уваги до етичних та правових аспектів використання ШІ [6]. Відсутність пояснень у рішеннях, що пропонує ШІ, створює небажані ризики для їх застосування в критично важливих завданнях програмної інженерії, таких, наприклад, як виявлення

вразливостей, де прозорість прийняття рішень має першорядне значення [7]. Наприклад, серйозними задачами програмної інженерії є перевірки коду. У цьому за своєю суттю людино-центричному процесі важливим є питання як інженерам-програмістам орієнтуватися у впровадженні ШІ у спільні з ним робочі процеси, адже діяльність з перевірки коду є багатовимірною, охоплюючи когнітивні, емоційні та поведінкові виміри. Впровадження перевірок за допомогою ШІ впливає на деякі з цих атрибутів. Так, наприклад, потреба в емоційній регуляції та механізмах подолання може бути меншою, однак когнітивне навантаження стає іноді вище, зокрема при роботі зі зворотним зв'язком, згенерованим ШІ, через його надмірну деталізацію. При цьому сприйняття зворотного зв'язку від ШІ обмежене недовірою та відсутністю пояснювального контексту в перевірці [8].

Зростання потреби в засобах ШІ, які є не лише функціональними, але й етичними, прозорими та орієнтованими на користувача призвело до популярності людино-центричного ШІ (Human-Centred Artificial Intelligence, HCAI). Наголошується на важливості ітеративних, орієнтованих на користувача процесах інтеграції інтерпретованості, зменшення упередженості та етичних міркувань у системах HCAI [9]. На сучасному етапі застосування ШІ йдеться вже не про пересічний базовий промтинг, а здебільше про цифрові асистенти, інтелектуальні системи підтримки прийняття рішень, про системи Agentic AI, де агенти взаємодіють і координуються між собою. Перспективним напрямком є застосування в програмній інженерії багатоагентних автономних систем (Multi-agent autonomous systems, MAS), які краще справляються з завданнями, що охоплюють кілька областей, ніж окремі автономні агенти. Однак впровадження таких систем також створює безліч проблем, серед яких однією з основних є стратегічний розподіл завдань між людьми та MAS надійним чином [10].

Отже, ШІ розширює можливості в галузі програмної інженерії завдяки прискореному генеруванню ідей та автоматизації процесів, роблячи деякі традиційні практики застарілими. Однак людино-центрична перспектива все ще залишається важливою. Забезпечення людського нагляду та осмислені інтерпретації необхідне для підтримки технічної точності, сприяння етичній відповідальності та стимулювання прогресу в цій галузі [11].

Досягнення в сфері LLM призвели до широкого поширення генеративного ШІ (Generative AI), або чат-ботів великих мовних моделей. В галузі програмної інженерії використання генеративного ШІ також може бути корисним інструментом, адже він здатен підтримувати всі фази розробки ПЗ [12]. Оцінюючи інструменти генеративного ШІ на основі таких критеріїв, як зручність використання, ефективність та інтеграція в існуючі робочі процеси, дослідження підкреслюють важливість співпраці людини та ШІ. Припускається, що хоча генеративний ШІ може суттєво підтриму-

вати завдання розробки програмного забезпечення, залишаються важливими людський нагляд та критична оцінка результатів, створених ШІ [13].

У зв'язку із цим існує потреба розуміти принципи того, що передбачає відповідальна розробка ПЗ у практиці застосування ШІ, а також ретельно оцінювати результативність впровадження інструментів та методів ШІ в програмну інженерію [14]. Тому вважається, що майбутнє за гібридною моделлю, яка поєднує традиційні підходи з автоматизацією на основі ШІ [15, 16]. У цьому контексті важливим є врахування основних аспектів пояснюваного ШІ (explainable AI — XAI) та практичне використання інструментів та методів пояснюваного ШІ [17].

Інтеграція технологій ШІ дозволяє автоматизувати розробку ПЗ, оптимізувати процеси розробки, підтримувати прийняття технічних рішень та покращити командну координацію. Але незважаючи на це, питання впливу НМТ на ефективність команди отримали ще недостатньо уваги у спільноті програмної інженерії [18]. Потрібен новий ландшафт досліджень та практики, окреслюючи сучасні тенденції та трансформації, ключові виклики та можливості, та уявляючи майбутнє, де люди та ШІ працюють синергетично, а людина постає в основі екосистеми програмної інженерії [19]. Методологія досліджень має базуватися на концепціях, моделях, методах та інформаційних технологіях, враховуючи вимоги до складності, підтримки управлінських рішень на основі знаннєво-орієнтованих інструментів та їх комплексного інформаційного забезпечення [20].

Метою роботи є аналіз можливостей використання технологій ШІ для підтримки командної роботи в програмній інженерії та розробка концептуальної моделі ШІ-підтриманої командної взаємодії.

Трансформація моделей групової динаміки. Класичною моделлю для опису еволюції команд в теорії групової динаміки є модель Такмена (Tuckman), що передбачає основні фази (Forming—Storming—Norming—Performing), необхідні і неминучі для того, щоб група (команда) розвивалась і досягала результатів. Згодом модель була доповнена стадією Adjourning, яка у контексті моделі є фінальною стадією, що передбачає розформування команди, завершення її роботи або перенесення діяльності на інший час. Щоб краще зрозуміти, як розвивається співпраця у команді людини та ШІ, необхідні подальші дослідження, засновані на теоретичній моделі Такмена. В умовах інтеграції ШІ у командну роботу в програмній інженерії ця модель не втрачає актуальності, але трансформується у нову форму, яку можна охарактеризувати як соціо-технічна модель команди, доповненої штучним інтелектом (socio-technical AI-augmented team model), де ШІ виступає активним агентом у командній взаємодії [21].

У табл. 1 наведено системну інтерпретацію трансформації кожної стадії моделі Такмена в нових умовах.

Таблиця 1

Класично	З AI	Трансформація	Результат
Forming — формування команди, ознайомлення з ролями, невизначеність	AI-Augmented Onboarding — автоматичний онбординг, AI одразу включається в роботу	AI виступає як фасилітатор адаптації та брокер знань (пояснення codebase, документації, рекомендації задач	Зменшується час входження нових учасників у команду
Storming — конфлікти, боротьба за ролі, узгодження підходів	Human-AI Alignment Phase — з'являються нові типи конфліктів «людина vs AI», різні рівні довіри до AI, конфлікти через автоматизацію	Команда вирішує коли довіряти AI, як інтегрувати AI у workflow, визначає межі автоматизації	Формується протокол співпраці людини та штучного інтелекту. Спостерігаються нові явища, такі як алгоритмічна напруга, калібрування довіри
Norming — встановлення норм, стабілізація взаємодії	AI-Integrated Workflow Formation — формуються нові стандарти: використання AI в кодуванні, перевірки коду за допомогою ШІ, автоматизованих DevOps-процесів	Виникають правила управління ШІ, політики використання ШІ, норми взаємодії команди та ШІ	AI стає інституціоналізованою частиною процесу
Performing — ефективна робота, висока продуктивність	AI-Enhanced High-Performance Team — команда працює як гібридна когнітивна система	AI генерує код, AI прогнозує ризики, люди приймають стратегічні рішення (data-driven decision making, continuous optimization, predictive development)	Значне зростання продуктивності, скорочення часу виходу на ринок, підвищення якості ПЗ за рахунок прийняття рішень на основі даних, безперервної оптимізації, прогнозного розвитку
Adjourning — завершення проекту, розпуск команди	Knowledge Preservation & AI Memory — знання зберігаються завдяки базі знань ШІ, автоматичним навченим моделям	AI стає системою організаційної пам'яті	Знання та досвід команди не втрачаються, покращується наступний цикл відродження і розробки

Ключова інтерпретація цих зрушень полягає у тому, що модель Такмена трансформується з моделі розвитку людської команди у модель коеволюції людини та штучного інтелекту, який стає активним учасником командної взаємодії, що дозволяє описати еволюцію сучасних команд програмної

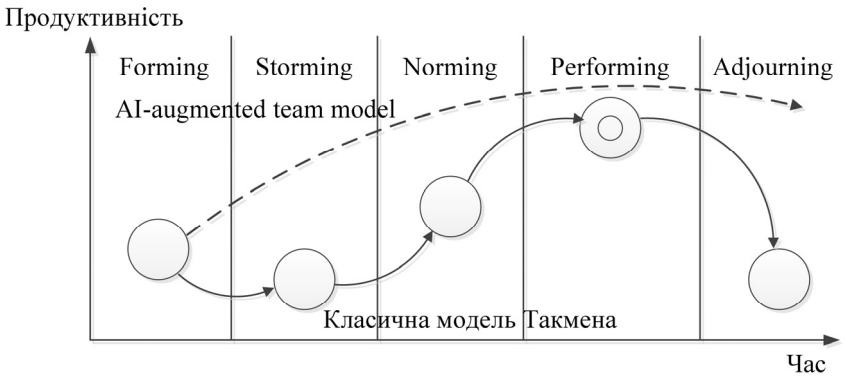


Рис. 1. Трансформація класичної моделі Такмена в модель команди, доповненої штучним інтелектом

інженерії як процес співрозвитку людини та ШІ (рис. 1). У цьому контексті виникають нові наукові категорії, такі як співпраця людини та ШІ, ШІ як партнер по команді, алгоритмічна довіра, команди гібридного інтелекту, метрики оцінки зрілості Human—AI команди на кожній стадії та ін.

Одним з найважливіших процесів у розвитку команди є розподіл ролей серед учасників. Узгодження співробітників з ролями, які відповідають їхнім сильним сторонам, зменшує суб'єктивне прийняття рішень у процесах управління персоналом.

Для створення збалансованих, високопродуктивних команд у сучасному бізнес-середовищі значний потенціал має аналітика аналізу ролей на основі штучного інтелекту [22]. Для ефективної роботи команди, зокрема в програмній інженерії, необхідними є дев'ять поведінкових ролей, які описує модель командних ролей Белбіна (Belbin). В умовах інтеграції ШІ у командну діяльність ця модель розширюється до моделі розподілу ролей між людьми та ШІ (Human—AI Role Distribution Model), де частина «людських» ролей підсилюється ШІ завдяки автоматизації, а частина ролей перерозподіляється між людьми та ШІ.

Загальна трансформація моделі Белбіна полягає у тому, що в умовах використання ШІ формується гібридна екологія ролей, де ролі розподіляються не лише між людьми та ШІ-інструментами, а й формуються гібридні ролі (human+AI). Узагальнено цю трансформацію можна представити у табл. 2.

Ключова трансформація моделі Белбіна полягає у переході від розподілу ролей між людьми до перерозподілу між людьми та системами ШІ. При цьому, зокрема у командах програмної інженерії, з'являються нові ролі, такі як AI Facilitator (інтегрує ШІ у workflow), Prompt Engineer

(забезпечення ефективної взаємодії з ШІ), AI Auditor (перевіряє результати відгуків ШІ), AI Trainer (налаштовує моделі ШІ), тощо.

Таким чином, команда стає гібридною когнітивною соці-технічною системою, де ШІ виконує когнітивні функції, а люди — соціальні та стратегічні функції. При цьому ШІ не замінює ролі, а перерозподіляє когнітивне навантаження.

На рис. 2 подано інтегрований теоретичний 3-вимірний фреймворк, що об'єднує модель стадій розвитку команди Такмена, ролі Белбіна та можливості ШІ у контексті програмної інженерії. Разом вони формують систему командної роботи на основі взаємодії між людиною та ШІ (Human—AI Co-evolving Team System).

Принцип інтеграції полягає у тому, що на кожній стадії Такмена ролі Белбіна перерозподіляються, а ШІ підсилює або трансформує ролі. Команда та ШІ розвиваються разом, тобто еволюція команди є функцією колаборації (люди + ШІ). В цьому процесі відбувається зсув когнітивного навантаження, коли ШІ бере на себе аналітику, генерацію, прогнозування, а за людиною залишається креативність, соціальна взаємодія, стратегічні рішення.

Таблиця 2

Роль Белбіна	ШІ-трансформація	Рівень автоматизації
Plant (Генератор ідей)	AI-Augmented Innovator (Новатор, доповнений штучним інтелектом)	Середній
Resource Investigator (Дослідник ресурсів)	AI-Knowledge Explorer (Дослідник знань за допомогою ШІ)	Високий
Coordinator (Координатор)	AI-Supported Orchestrator (Розподіл задач, аналіз навантаження команди за підтримки ШІ)	Середній
Shaper (Оформлювач ідей)	Human-driven AI (ШІ, керований людиною)	Низький
Monitor Evaluator (Контролер-оцінювач)	Monitor Evaluator AI-Augmented (Контролер-оцінювач, доповнений ШІ)	Високий
Teamworker (Командний працівник)	Human-centric (Людиноцентрична роль)	Низький
Implementer (Виконавець)	AI-Accelerated (Прискорення виконання за допомогою штучного інтелекту	Високий
Completer Finisher (Оздоблювач-завершувач)	AI Quality Control (Контроль якості на основі ШІ)	Високий
Specialist (Спеціаліст, фахівець)	Hybrid Expert (Гібридний експерт)	Середній

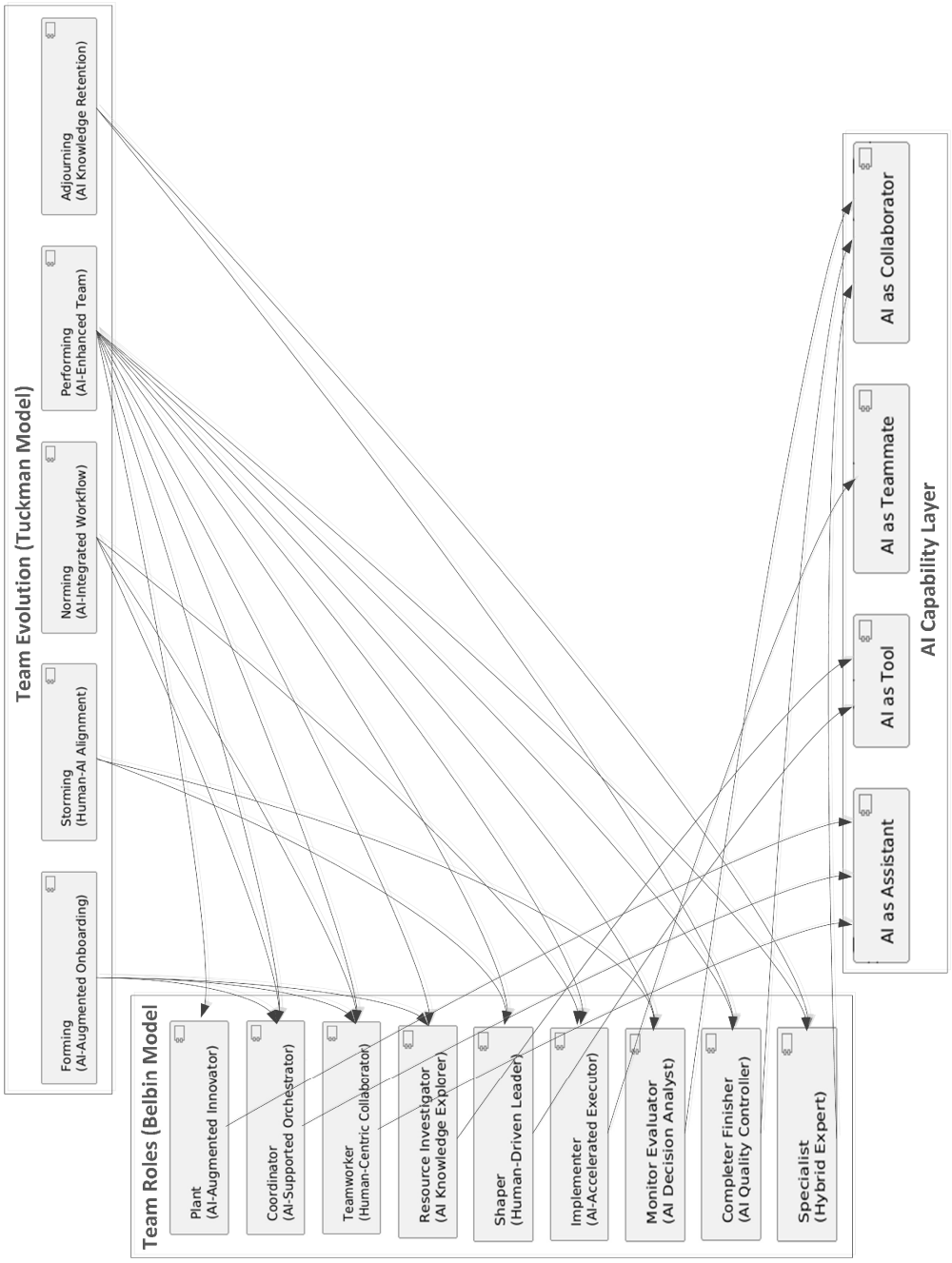


Рис. 2. 3-вимірний фреймворк, що об'єднує моделі Такмена і Белбіна та можливості ШІ у контексті програмної інженерії

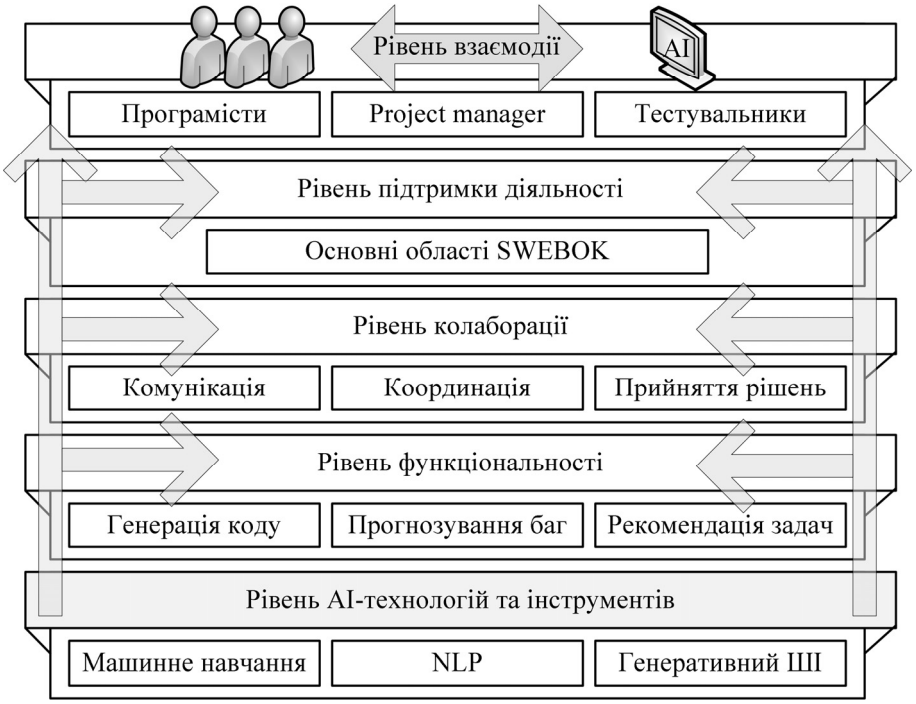


Рис. 3. Концептуальна модель ШІ-підтримуваної командної роботи

Концептуальна модель ШІ-підтриманої командної роботи. Системну модель AI-підтриманої командної роботи (S) можна формалізувати у вигляді множин $S = \langle T, P, L, C, A, I, R \rangle$, де T (Team) — множина учасників команди, P (Processes) — множина процесів розробки ПЗ, L (Role) — множина ролей розподілених в команді, C (Collaboration) — механізми взаємодії, A (AI Capabilities) — функції AI, I (Infrastructure) — інструменти та платформи, R (Relations) — множина зв'язків (відношень).

Тоді концептуально AI-підтриманую командну роботу можна запропонувати у вигляді п'яти-рівневої загальної моделі, що репрезентує множину R (рис. 3).

Рівень взаємодії (рівень програмної команди) представляє множини T і L, де учасники (програмісти, тестувальники, DevOps, Project manager та ін.) взаємодіють не лише між собою, а й з інструментарієм ШІ. Рівень підтримки діяльності відповідає множині P і пов'язаний з областями знань (Knowledge Area), визначених міжнародним посібником з програмної інженерії SWEBOK (Software Engineering Body of Knowledge), який в межах спільноти розробників ПЗ відіграє важливу роль як флагманський основоположний та структурний документ для створення програмних продуктів. З 18-ти областей знань можна виділити шість основних, а саме Вимоги до

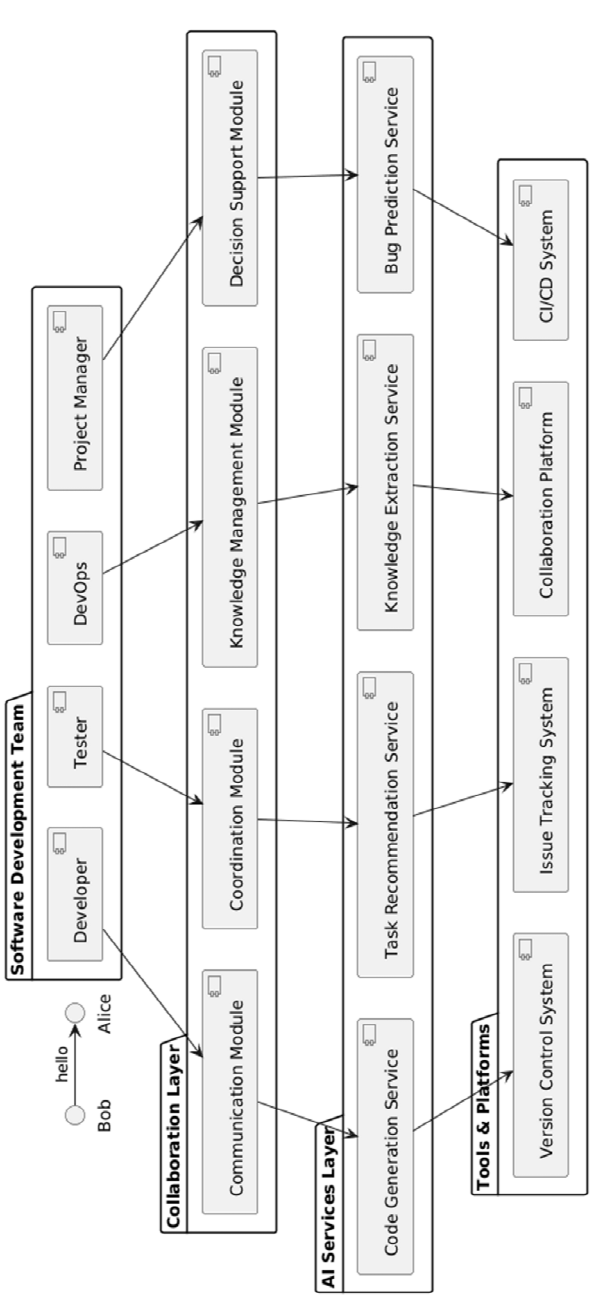


Рис. 4. UML Component Diagram ШІ-підтримуваної командної роботи

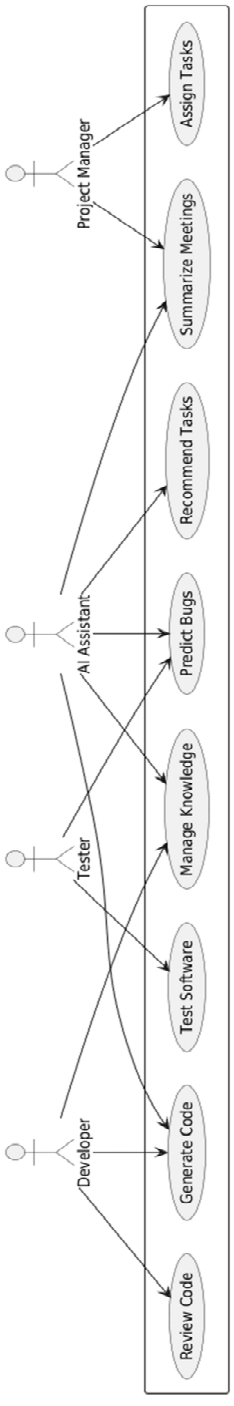


Рис. 5. UML Use Case Diagram в умовах ШІ-підтримуваної командної роботи

ПЗ (Software Requirements), Архітектура ПЗ (Software Architecture), Проектування ПЗ (Software Design), Конструювання ПЗ (Software Construction), Тестування ПЗ (Software Testing) та Супровід ПЗ (Software Maintenance). ШІ знаходить застосування у першу чергу для підтримки процесів, які відносяться до вказаних областей знань.

Рівень колаборації (множина *C*) пов'язаний із комунікаціями, координацією, спільним прийняттям рішень та управлінням знаннями, де ШІ стає невід'ємним інструментом підтримки. Рівень функціональності (множина *A*) деталізує конкретні функції, що цілком можуть виконуватись штучним інтелектом, такі як генерація коду, прогнозування помилок, рекомендації щодо завдань, видобування нових знань та ін. Нарешті, Рівень ШІ-технологій та інструментів (множина *I*) визначає моделі і алгоритми, що знаходять застосування в програмній інженерії (машинне навчання, обробка природньої мови, великі мовні моделі) та відповідний інструментарій на їх основі (генеративний ШІ, агентні системи, автоматизовані фреймворкі, такі, наприклад, як GitHub Copilot, OpenAI Codex, GitLab, Jira, Microsoft Teams).

Подальшу формалізацію концептуальної моделі можна представити у вигляді UML-моделей. Так, структурна модель у вигляді Component Diagram відображає архітектуру AI-підтриманої командної роботи, її основні компоненти та взаємозв'язки (рис. 4). На рис. 5 представлено поведінкову модель UML (Use Case Diagram) на прикладі таких акторів, як Developer, Tester, Project Manager та AI-Assistant. Особливість полягає у тому, що AI-Assistant виступає як окремий актор, що підкреслює його роль як віртуального учасника команди.

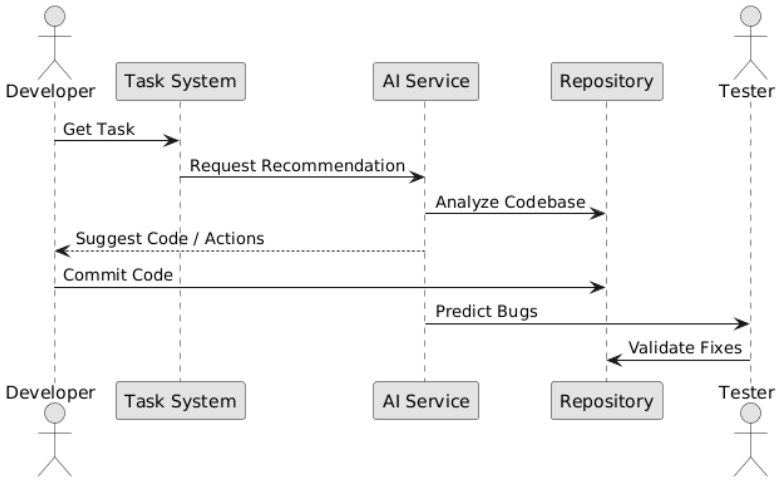


Рис. 6. UML Sequence Diagram в умовах ШІ-підтримуваної командної роботи

На рис. 6 представлено діаграму динамічної взаємодії (UML Sequence Diagram), де AI-assisted реалізує сценарій виконавця завдвнь (task execution).

Інтерпретація наведених моделей полягає у тому, що виробництво програмної продукції функціонує як соціо-технічна система, де люди (*T*) виконують ролі (*L*) та утілюють процеси (*P*), взаємодіючі через механізми колаборації (*C*) за підтримки і підсилення інструментарію ШІ (*A*), що реалізується через інфраструктуру (*I*).

Наукова та практична цінність цієї формалізації дозволяє перейти від концептуальної моделі AI-Driven Team Collaboration до подальшого математичного моделювання системи, симуляції та оцінки ефективності діяльності ШІ-команди, водночас забезпечуючи основу для побудови метрик ШІ-зрілості команди та розробки архітектур ШІ-систем [23].

Висновки

Розглянуто проблему колаборації технологій ШІ у командній роботі в програмній інженерії. Проаналізовано основні виклики, пов'язані зі співпрацею людини та ШІ, питання відповідальної розробки ПЗ у практиці застосування ШІ, врахування основних аспектів пояснюваного ШІ, стратегічний розподіл завдань між людьми та агентними системами.

У межах дослідження було запропоновано розширення моделі Такмена з урахуванням інтеграції ШІ як активного учасника командної взаємодії, що дозволяє описати еволюцію команд програмної інженерії як процес співрозвитку людини та ШІ. Запропоновано розширення моделі ролей Белбіна з урахуванням інтеграції ШІ як функціонального агента, що призводить до формування гібридної системи розподілу ролей у командах програмної інженерії. Запропоновано інтегрований фрейм, що поєднує модель стадій розвитку команд, рольову модель та можливості ШІ, що дозволяє описати команди програмної інженерії як гібридні соціо-технічні системи зі співрозвитком людини та ШІ систематизацію

Запропоновано концептуальну модель ШІ-підтриманої командної роботи та її деталізацію UML-діаграмами. Наукова та практична цінність цієї формалізації дозволяє у подальшому перейти до математичного моделювання системи, оцінки ефективності діяльності ШІ-команди та її зрілості.

Результати проведеного аналізу підтверджують, що ШІ стає інтелектуальним партнером команди, що призводить до трансформації командної взаємодії. В цьому процесі відбувається зсув когнітивного навантаження, коли ШІ бере на себе аналітику, генерацію, прогнозування, а за людиною залишається креативність, соціальна взаємодія, стратегічні рішення.

Подальші дослідження варто зосередити на розробленні та оптимізації адаптивних моделей перспективного розвитку ШІ-асистентів для колаборативної роботи.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Kokol P. The use of AI in software engineering: A synthetic knowledge synthesis of the recent research literature. *Information*. 2024. Vol. 15, No. 6, 354.
2. Pai H.A., Sharath K.R., Babalad K. Integrating AI into software engineering: A critical review and future directions. *Proceedings of the 2025 International Conference on Intelligent Control, Computing and Communications (IC3)*. 2025. P. 20—25.
3. Nesterenko O., Selin Y. The teams information model for software engineering management. *Proceedings of the 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)*. 2021, 1. P. 341—344.
4. Schleiger E., Mason C., Naughtin C., et al. Collaborative Intelligence: A Scoping Review of Current Applications. *Applied Artificial Intelligence*. 2024. Vol. 38, No. 1, e2327890.
5. Shi M., Wang G., Zhang S., et al. AISCW: Artificial Intelligence Supported Cooperative Work the New Frontier of CSCW. *Computer Supported Cooperative Work and Social Computing. ChineseCSCW 2024. Communications in Computer and Information Science* / ed. by Sun H., et al. 2025, 2343. P. 3—17. Springer, Singapore.
6. Нестеренко О.В., Яцук П.П. ШІ (не) проти ШІ. *Екологічна безпека та природокористування*. 2025. Т. 56, № 4. С. 134—153.
7. Cao S., Sun X., Widyasari R., et al. A systematic literature review on explainability for ML/DL-based software engineering. *ACM Computing Surveys*. 2025. Vol. 58, No. 4, 95. P. 1—34.
8. Alami A., Ernst, N. Human and machine: How software engineers perceive and engage with AI-assisted code reviews compared to their peers. *Proceedings of the 2025 IEEE/ACM 18th International Conference on Cooperative and Human Aspects of Software Engineering (CHASE)*. 2025. P. 63—74.
9. Desolda G., Esposito A., Lanzilotti R., Piccinno A. Designing and Evaluating Human-Centred AI Systems: Best-Practices from a Multidisciplinary View. *CEUR Workshop Proceedings*. 2025, 3957. P. 170—178.
10. Ronanki K. Facilitating trustworthy human-agent collaboration in LLM-based multi-agent system oriented software engineering. *Companion proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE 2025)*. 2025. arXiv.
11. Trinkenreich B., Calefato F., Hanssen G., et al. Get on the train or be left on the station: Using LLMs for software engineering research. *Companion proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion 2025)*. 2025. P. 1503—1507.
12. Stutz K., Sandkuhl K., Möhring M. Empirical insights into the usage of generative AI in software engineering. *Lecture Notes in Business Information Processing*. 2026, 562. P. 257—271.
13. Fischer M., Lanquillon C. Evaluation of generative AI-assisted software design and engineering: A user-centered approach. *Lecture Notes in Computer Science*. 2024, 14734. P. 31—47.
14. Ulfsnes R., Moe N.B., Emmerhoff J., et al. Responsible AI in agile software engineering: An industry perspective. *Lecture Notes in Business Information Processing*. 2025, 524. P. 33—41.
15. Ain Q.U., Haq M., Jilani A.A.A., Sohail K.A comparison between traditional software engineering practices and AI-driven methodologies. *Generative AI in Software Engineering*. 2025. P. 57—92.
16. Fan, A., Gokkaya B., Harman M., et al. Large language models for software enginering: Survey and open problems. *Proceedings of the 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FOSE)*. 2023. P. 31—53.

17. Tantithamthavorn C., Jiarpakdee J. Explainable AI for software engineering. *Proceedings of the 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE 2021)*. 2021. P. 1—2.
18. Rauf I., Sharp H., Lopez T., Wermelinger M. Human-machine teaming and team effectiveness in AI tools for software engineering. *Proceedings of the 2025 Conference on Cooperative and Human Aspects of Software Engineering (CHASE)*. 2025. P. 75—80.
19. Abrahão S., Grundy J., Pezzè M., et al. Software Engineering by and for Humans in an AI Era. *ACM Transactions on Software Engineering and Methodology*. 2025. Vol. 34, No. 5, 129.
20. Трофимчук О., Нестеренко О., Нетесін І. Методологія проєктування інформаційно-аналітичних систем адміністративного управління. *Наука і інновації*. 2022. Т. 18, № 4. С. 25—40.
21. Kaelin V. C., Tewari M., Benouar S., Lindgren H. Developing teamwork: Transitioning between stages in human-agent collaboration. *Frontiers in Computer Science*. 2024, 6.
22. Brajković D.Ć., Brajković E., Volarić T. Application of the Random Forest Algorithm for Team Role Classification Based on Belbin's Self-perception Inventory. *Proceedings of the 3rd International Conference on Digital Transformation in Education and Artificial Intelligence Applications (MoStart 2025)*. 2026, 2609 CCIS. P. 283—294.
23. Lhazmir H., Samhale S., Louzaoui K., Benlhachmi K. The uneven journey of AI in software testing: A maturity model for industry adoption. *Proceedings of the 3rd International Conference on Digital Transformation in Education and Artificial Intelligence Applications (MoStart 2025)*. 2025, 2609 CCIS. P. 132—137.

Стаття надійшла 16.04.2026;
Прорецензована 30.04.2026

REFERENCES

1. Kokol, P. (2024). The use of AI in software engineering: A synthetic knowledge synthesis of the recent research literature. *Information*, 15(6), 354. https://doi.org/10.3390/info15060354
2. Pai, H. A., Sharath, K. R., & Babalad, K. (2025). Integrating AI into software engineering: A critical review and future directions. *In Proceedings of the 2025 International Conference on Intelligent Control, Computing and Communications (IC3)* (p. 20-25). https://doi.org/10.1109/IC363308.2025.10957449
3. Nesterenko, O., & Selin, Y. (2021). The teams information model for software engineering management. *In Proceedings of the 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)* (1, p. 341-344). https://doi.org/10.1109/CSIT52700.2021.9648737
4. Schleiger, E., Mason, C., Naughtin, C., et al. (2024). Collaborative Intelligence: A Scoping Review of Current Applications. *Applied Artificial Intelligence*, 38(1), e2327890. https://doi.org/10.1080/08839514.2024.2327890
5. Shi, M., Wang, G., Zhang, S., et al. (2025). AISCW: Artificial Intelligence Supported Cooperative Work the New Frontier of CSCW. In Sun, H., et al. (ed.). *Computer Supported Cooperative Work and Social Computing. ChineseCSCW 2024. Communications in Computer and Information Science, 2343* (p. 3-17). Springer, Singapore. https://doi.org/10.1007/978-981-96-2373-0_1
6. Nesterenko, O., & Yatsuk, P. (2025). AI (not) against AI. *Environmental Safety and Natural Resources*, 56(4), 134-153. https://doi.org/10.32347/2411-4049.2025.4.134-153. (in Ukrainian).

7. Cao, S., Sun, X., Widyasari, R., et al. (2025). A systematic literature review on explainability for ML/DL-based software engineering. *ACM Computing Surveys*, 58(4), Article 95, 1-34. https://doi.org/10.1145/3763230
8. Alami, A., & Ernst, N. (2025). Human and machine: How software engineers perceive and engage with AI-assisted code reviews compared to their peers. *In Proceedings of the 2025 IEEE/ACM 18th International Conference on Cooperative and Human Aspects of Software Engineering (CHASE)* (p. 63-74). https://doi.org/10.1109/CHASE66643.2025.00016
9. Desolda, G., Esposito, A., Lanzilotti, R., Piccinno, A. (2025). Designing and Evaluating Human-Centred AI Systems: Best-Practices from a Multidisciplinary View. CEUR Workshop Proceedings, 3957, 170-178.
10. Ronanki, K. (2025). Facilitating trustworthy human-agent collaboration in LLM-based multi-agent system oriented software engineering. *In Companion proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE 2025).* arXiv. https://doi.org/10.48550/arXiv.2505.04251
11. Trinkenreich, B., Calefato, F., Hanssen, G., et al. (2025). Get on the train or be left on the station: Using LLMs for software engineering research. *In Companion proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion 2025)* (p. 1503-1507). https://doi.org/10.1145/3696630.3731666
12. Stutz, K., Sandkuhl, K., & Möhring, M. (2026). Empirical insights into the usage of generative AI in software engineering. *Lecture Notes in Business Information Processing, 562* (p. 257-271). https://doi.org/10.1007/978-3-032-04375-7_16
13. Fischer, M., & Lanquillon, C. (2024). Evaluation of generative AI-assisted software design and engineering: A user-centered approach. *Lecture Notes in Computer Science, 14734* (p. 31-47). https://doi.org/10.1007/978-3-031-60606-9_3
14. Ulfsnes, R., Moe, N. B., Emmerhoff, J., et al. (2025). Responsible AI in agile software engineering: An industry perspective. *Lecture Notes in Business Information Processing, 524* (p. 33-41). https://doi.org/10.1007/978-3-031-72781-8_4
15. Ain, Q. U., Haq, M., Jilani, A. A. A., & Sohail, K. (2025). A comparison between traditional software engineering practices and AI-driven methodologies. *Generative AI in Software Engineering* (p. 57-92). https://doi.org/10.4018/979-8-3373-0370-3.ch002.
16. Fan, A., Gokkaya, B., Harman, M., et al. (2023). Large language models for software engineering: Survey and open problems. *In Proceedings of the 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FOSE)* (p. 31-53). https://doi.org/10.1109/ICSE-FoSE59343.2023.00008
17. Tantithamthavorn, C., & Jiarpakdee, J. (2021). Explainable AI for software engineering. *In Proceedings of the 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE 2021)* (p. 1-2). https://doi.org/10.1109/ASE51524.2021.9678580
18. Rauf, I., Sharp, H., Lopez, T., & Wermelinger, M. (2025). Human-machine teaming and team effectiveness in AI tools for software engineering. *In Proceedings of the 2025 Conference on Cooperative and Human Aspects of Software Engineering (CHASE)* (p. 75-80). https://doi.org/10.1109/CHASE66643.2025.00017
19. Abrahão, S., Grundy, J., Pezzè, M., et al. (2025). Software Engineering by and for Humans in an AI Era. *ACM Transactions on Software Engineering and Methodology*, 34 (5), art. no. 129. https://doi.org/10.1145/3715111
20. Trofymchuk, O., Nesterenko, O., & Netesin, I. (2022). Methodology for Designing Analytical Information Systems for Administrative Management. *Science and Innovation*, 18(4), 25-40. https://doi.org/10.15407/scine18.04.025. [in Ukrainian].
21. Kaelin, V. C., Tewari, M., Benouar, S., & Lindgren, H. (2024). Developing teamwork: Transitioning between stages in human-agent collaboration. *Frontiers in Computer Science*, 6. https://doi.org/10.3389/fcomp.2024.1455903

22. Brajković, D.Ć., Brajković, E., Volarić, T. (2026). Application of the Random Forest Algorithm for Team Role Classification Based on Belbin's Self-perception Inventory. In *Proceedings of the 3rd International Conference on Digital Transformation in Education and Artificial Intelligence Applications, MoStart 2025* (2609 CCIS, p. 283-294). https://doi.org/10.1007/978-3-032-02801-3_19
23. Lhazmir, H., Samhale, S., Louzaoui, K., & Benlhachmi, K. (2025). The uneven journey of AI in software testing: A maturity model for industry adoption. In *Proceedings of the 3rd International Conference on Digital Transformation in Education and Artificial Intelligence Applications, MoStart 2025* (2609 CCIS, p. 132-137). https://doi.org/10.1109/CiSt65886.2025.11224099.

Article received 16.04.2026;
Reviewed 30.04.2026

O.V. Nesterenko, V.V. Fedorov, P. P. Yatsuk

ARTIFICIAL INTELLIGENCE IN TEAMWORK IN SOFTWARE ENGINEERING

Artificial intelligence (AI) is becoming an integral part of software engineering, expanding the possibilities for automating software development and maintenance processes throughout its entire life cycle. In today's environment, the issue of human-machine teaming (HMT)-that is, the interaction between humans and AI as team members-is gaining relevance. This paper analyzes the main challenges associated with human-AI collaboration, particularly with regard to group dynamics methodology. A transformation of Tuckman's model is proposed, which allows for describing the evolution of software engineering teams as a process of co-development between humans and AI. An extension of Belbin's role model is also proposed, leading to the formation of a hybrid role-distribution system in software engineering teams integrated with AI. As a result, an integrated framework has been developed that combines the team development stage model, the role model, and the capabilities of AI, allowing software engineering teams to be described as hybrid socio-technical systems. A conceptual model of AI-supported teamwork and its detailed representation using UML diagrams are proposed. The results of this study confirm that AI is becoming an intellectual partner to the team. This leads to a transformation of team interaction and its future development.

K e y w o r d s: human-machine interaction, group dynamics, Tuckman model, Belbin model, UML, intellectualization, software.

НЕСТЕРЕНКО Олександр Васильович, д-р техн. наук, професор, зав. кафедри нформаційних технологій Першого європейського університету. У 1975 р. закінчив Київський політехнічний інститут. Область наукових досліджень — інтелектуальні системи підтримки прийняття рішень.

ФЕДОРОВ Володимир Володимирович, канд. фіз.-мат. наук, доцент кафедри інформаційних технологій Першого європейського університету. У 2009 р. закінчив Київський національний університет ім. Тараса Шевченка. Область наукових досліджень — розробка програмних систем та застосування штучного інтелекту.

ЯЦУК Петро Петрович, канд. техн. наук, доцент кафедри інформаційних технологій Першого європейського університету. У 1985 р. закінчив Одеську академію зв'язку. Область наукових досліджень — системи телекомунікацій, збереження даних та хмарні сервіси.