https://doi.org/10.31891/2307-5732-2026-363-65
УДК 004.8:004.9:658.012.23

ПОЛУЕКТОВА НАТАЛІЯ
Запорізький інститут економіки та інформаційних технологій
https://orcid.org/0009-0005-9637-4916
e-mail: n.poluektova@econom.zp.ua
МАТВІЇШИНА НАДІЯ
Запорізький національний університет
https://orcid.org/0000-0001-7938-4622
e-mail: mnv2902@gmail.com

УПРАВЛІННЯ ЗАЛЕЖНОСТЯМИ У ПРОЄКТАХ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВИКОРИСТАННЯМ МУЛЬТИАГЕНТНОГО ПІДХОДУ ТА НАВЧАННЯ З ПІДКРІПЛЕННЯМ

У роботі розглянуто проблему управління залежностями у великих проєктах розробки програмного забезпечення, що характеризуються високою складністю, динамічністю вимог та невизначеністю. Запропоновано модель мультиагентної системи, яка інтегрує методи штучного інтелекту (навчання з підкріпленням) для адаптивного управління залежностями між завданнями. Архітектура моделі включає три типи спеціалізованих агентів: агенти завдань, що відстежують статуси та параметри робіт; агенти ресурсів, які оптимізують розподіл людських і технічних ресурсів; агенти ризиків, що прогнозують можливі затримки та конфлікти. Центральним елементом є координуючий агент, який на основі даних від інших агентів перебудовує граф залежностей, змінює пріоритети та формує нові сценарії виконання. Для його оптимізації використано алгоритм Proximal Policy Optimization (PPO), що дозволяє агенту навчатися ефективним стратегіям управління. Експериментальна оцінка моделі проведена на тестовому проєкті в Jira, де порівнювалися результати роботи базового алгоритму та RL-агента. Метрики включали кількість конфліктів ресурсів, середній ризик задач та довжину критичного шляху. Результати показали, що агент здатний усувати конфлікти, скорочувати критичний шлях та адаптивно вдосконалювати стратегії управління залежностями. Ключовою новизною є поєднання мультиагентної архітектури з навчанням з підкріпленням, що забезпечує не лише автоматизацію рутинних рішень, але й адаптивне навчання оптимальним діям. Водночас модель має обмеження, пов'язані з якістю даних Jira та складністю масштабування на дуже великі проєкти. Робота може слугувати експериментальною платформою для подальших досліджень мультиагентних систем у сфері управління проєктами.

Ключові слова: *агентна модель, штучний інтелект, підкріплювальне навчання, управління проєктами*

POLUEKTOVA NATALIA
Zaporizhzhya Institute of Economics and Information Technologies
MATVIYISHYNA NADIA
Zaporizhzhya National University

DEPENDENCY MANAGEMENT IN SOFTWARE DEVELOPMENT PROJECTS USING MULTI-AGENT APPROACH AND REINFORCEMENT LEARNING

This paper addresses the problem of dependency management in large-scale software development projects, which are characterized by high complexity, dynamic requirements, and uncertainty. A multi-agent system model is proposed that integrates artificial intelligence methods and reinforcement learning to enable adaptive management of task dependencies. The architecture consists of three types of specialized agents: task agents that monitor task status and parameters; resource agents that optimize the allocation of human and technical resources; and risk agents that predict potential delays and conflicts. The central element is the coordinator agent, which, based on information from other agents, rebuilds the dependency graph, adjusts task priorities, and generates new execution scenarios. Optimization is achieved using the Proximal Policy Optimization (PPO) algorithm, allowing the agent to learn effective management strategies. Experimental evaluation was conducted on a test project in Jira, comparing the performance of a rule-based algorithm with that of the RL-based coordinator agent. Evaluation metrics included the number of resource conflicts, average task risk, and critical path length. Results demonstrated that the agent can eliminate conflicts, shorten the critical path, and adaptively improve dependency management strategies. The key novelty lies in combining multi-agent architecture with reinforcement learning, which provides not only automation of routine decisions but also adaptive learning of optimal actions. At the same time, the model has limitations related to the quality of Jira data and the challenges of scaling to very large projects. The study contributes an experimental platform for further research on multi-agent systems in project management, highlighting their potential to enhance planning efficiency, reduce risks, and improve decision-making in dynamic software development environments.

Keywords: *agent-based model, artificial intelligence, reinforcement learning, project management*

Стаття надійшла до редакції / Received 18.02.2026
Прийнята до друку / Accepted 11.03.2026
Опубліковано / Published 26.03.2026

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Сучасні проєкти розробки програмного забезпечення характеризуються високим рівнем складності, динамічністю вимог та значною невизначеністю на всіх етапах життєвого циклу. Зростання масштабів програмних систем та команд розробки, активне впровадження технологій штучного інтелекту в процеси розробки, зумовлюють підвищені вимоги до процесів управління проєктами. В цих умовах ефективність діяльності менеджера проєкту значною мірою залежить від здатності оперативно аналізувати великі обсяги інформації, прогнозувати ризики та приймати обґрунтовані управлінські рішення.

Навіть сучасні гнучкі методології проєктного управління, такі як Agile, Scrum або Kanban, значною мірою покладаються на експертні знання та інтуїцію менеджера проєкту, що обмежує можливості масштабування управлінських рішень та підвищує ризик помилок. У цьому контексті зростає інтерес до використання методів штучного інтелекту (ШІ) в управлінні проєктами.

Наразі загальновідомі наступні можливості використання методів та моделей ШІ в управлінні проєктами розробки програмного забезпечення.

– Автоматизація рутинних завдань: ШІ може автоматизувати завдання, що повторюються, зменшуючи кількість людських помилок і звільняючи членів команди для більш стратегічної діяльності [1].

– Поліпшення прийняття рішень: такі інструменти. як агенти ШІ, можуть допомогти в аналізі ризиків та розробці стратегічних рекомендацій, оптимізуючи робочі процеси проекту [1].

– Планування на основі знань: системи ШІ аналізують великі набори даних для надання практичних рекомендацій, підтримуваний ШІ, може оптимізувати виконання проекту, що призводить до підвищення ефективності, підвищуючи якість рішень, які приймає менеджер проектів [2].

– Поліпшений розподіл ресурсів: ШІ може оптимізувати розподіл ресурсів, прогнозуючи потреби проекту та потенційні вузькі місця, забезпечуючи ефективне використання ресурсів [3].

– Підходи на основі нечіткої логіки допомагають кількісно оцінити ефективність управління проектами в умовах невизначеності, вирішуючи такі проблеми, як нечіткі вимоги та прогалини у комунікації [4].

Перспективним напрямом у розв'язуванні таких завдань є застосування агентного підходу. Інтелектуальні агенти, що представляють окремі функціональні ролі або процеси управління, здатні автономно аналізувати стан проєкту, взаємодіяти між собою та підтримувати менеджера проєкту у прийнятті рішень. Такий підхід забезпечує децентралізовану обробку інформації, підвищує адаптивність системи управління та створює передумови для автоматизації рутинних управлінських задач.

Реалізація повністю автономних інтелектуальних мультиагентних систем (MAS) та їх впровадження в прийняті на підприємстві моделі проєктного управління є надзвичайно складною задачею. М.Вулдрідж [5] підкреслює, що загальні мультиагентні системи потребують ретельної розробки архітектури, визначення протоколів взаємодії та механізмів координації, що значно ускладнює їхнє впровадження у реальні бізнес-середовища, Н.Дженінгс та ін.[6] зазначають, що практичне застосування MAS часто обмежується високою вартістю розробки та складністю інтеграції з існуючими системами управління. Тому більш реалістичним підходом є поступове впровадження окремих агентних систем, які виконують специфічні функції (наприклад, система управління ризиками чи система оптимізації ресурсів), а не одразу повна мультиагентна платформа.

Метою даної роботи є розробка моделі мультиагентної системи, яка дозволяє динамічно управляти залежностями між завданнями у великих проєктах розробки ПЗ. Для реалізації мети були розв'язані наступні завдання: зроблений огляд джерел з тематики дослідження, описана архітектура моделі, розбудований прототип MAS, проведена експериментальна оцінка ефективності запропонованої моделі.

Аналіз досліджень та публікацій

Автори роботи [7] вперше ввели поняття, наближене до поняття інтелектуального агента, визначили «раціонального агента» як систему, що сприймає середовище через сенсори та діє на нього через актуатори, прагнучи максимізувати функцію корисності. У 1980–90-х роках вже активно розроблялися мультиагентні системи (MAS) [8].

Одночасно розвивалися теорія та практика проєктного менеджменту, зокрема в галузі розробки програмного забезпечення. В 1996 році з'явилися зведення стандартів з управління проєктами розробки програмного забезпечення PMBOK Guide [Project Management Institute. A Guide to the Project Management Body of Knowledge [9], в якому були стандартизовані процеси управління проєктами, введені поняття знань, процесних груп і областей та PRINCE [11] - методологія управління проєктами в якої робиться акцент на процесах контролю, з виділенням ролей та відповідальності.

В результаті поняття раціонального, а далі, інтелектуального агента стало теоретичною рамкою для побудови автономних систем прийняття рішень та управління ризиками, а класичні моделі MAS стали базою для розподіленого управління завданнями, що напряму співвідноситься з управлінням проєктами.

Найбільш ґрунтовний аналіз досліджень в галузі використання ШІ для управління проєктами виконаний І. Табода та ін. [3]. Вони проаналізували біля 100 робот, в яких показується, як AI інтегрується у планування, ризик-менеджмент та управління ресурсами. Аналіз сучасних трендів, нові можливості та ризики використання ШІ у проєктному управлінні наведений в роботі Й. Лі та ін.[11].

Далі представлені останні дослідження, релевантні до теми даної роботи.

В роботі О. Чередніченко та ін. [12] описано побудову мультиагентної моделі для управління проєктами у розподілених IT командах. Модель враховує адаптивні методології, невизначеність та автоматизацію комунікацій між учасниками.

Т.Нгуєн, та ін. [13] пропонують систему автоматичної рекомендації залежностей між завданнями у великих проєктах ПЗ. Вона використовує історичні дані та машинне навчання для динамічного оновлення зв'язків, що зменшує ручну роботу менеджерів.

Робота А. Алмаклі теж показує, як AI-агенти можуть автоматично перебудовувати залежності між завданнями у відповідь на зміни вимог чи пріоритетів та дозволяють уникати перевантаження команд і

зменшувати ризики. Однак, запропонована модель має певні обмеження – орієнтована на невеликі проєкти, які швидко адаптуються до змін, згідно Agile-методології, не виконує перерозподіл ресурсів згідно оновлених залежностей, не передбачає прямої інтеграції в одну з існуючих на підприємстві систем управління проєктами.

Проведений огляд джерел з теми дослідження свідчить про актуальність теми та необхідність розбудови нових методів та інструментів впровадження інтелектуальних агентних систем в практику управління проєктами розробки програмного забезпечення.

Виклад основного матеріалу

В якості цільової системи управління проєктами розглянемо Jira, як одну з найпоширеніших. У спрощеному вигляді, реалізація великих проєктів може бути розбита на етапи, за допомогою виділення окремих задач. Дуже важливим при цьому є завдання певних залежностей між окремими завданнями. Завдання можуть бути пов'язані через статуси *"blocks"*, *"is blocked by"*, *"relates to"*. Ці залежності задаються вручну користувачем і залишаються статичними.

Актуальним завданням є розробка моделі управління залежностями, яка здатна адаптивно реагувати на зміни у проєкті, автоматично перебудовувати структуру зв'язків між завданнями та враховувати ризики й доступність ресурсів. Така модель має забезпечити підвищення ефективності планування, зменшити кількість конфліктів та покращити якість управлінських рішень.

Важливою вимогою для цього дослідження є також можливість отримання даних з системи Jira та обробка інформації в стандарті цієї системи.

Запропонована модель управління залежностями базується на використанні мультиагентного підходу, який дозволяє забезпечити адаптивність та гнучкість у процесі планування й виконання завдань. На відміну від традиційних методик, де залежності задаються статично і потребують ручного коригування, модель передбачає автоматичне оновлення структури зв'язків між завданнями у відповідь на зміни вимог, пріоритетів чи появу нових задач.

У центрі моделі знаходиться координуючий агент, який здійснює моніторинг усіх завдань та їхніх взаємозалежностей. Він отримує інформацію від спеціалізованих агентів, що відповідають за різні аспекти управління: агентів завдань, агентів ресурсів та агентів ризиків. Агенти завдань відстежують статуси окремих робіт і сигналізують про зміни, агенти ресурсів оптимізують розподіл людських та технічних ресурсів, а агенти ризиків прогнозують можливі затримки чи конфлікти. На основі отриманих даних координуючий агент перебудовує графік залежностей, формує нові сценарії виконання та пропонує менеджеру оптимальні рішення.

Пропонована модель управління залежностями може бути формально описана через представлення проєкту як орієнтованого графа завдань, де вершини відповідають окремим задачам, а дуги — залежностям між ними.

$$\mathrm{G} = (\mathrm{V}, \mathrm{E}), \tag{1}$$

де $V=\{v_1,v_2,\ldots,v_n\}$ — множина завдань, а $E \subseteq V \times V$ — множина залежностей між ними.

Кожне завдання v_i характеризується множиною атрибутів: статусом виконання $s_i \in S$, пріоритетом $p_i \in P$, необхідними ресурсами $r_i \in R$ та ризиком виконання $q_i \in Q$..

У системі функціонують агенти трьох типів.

Агенти завдань (TaskAgent) $AT=\{at_1,at_2,\ldots,at_n\}$ пов'язані із завданнями v_i та відповідають за моніторинг їхнього стану. Формально, агент завдання реалізує функцію

$$f_t(\mathrm{v}_i) = (\mathrm{s}_i, \mathrm{p}_i, \mathrm{r}_i, \mathrm{q}_i), \tag{2}$$

що повертає актуальний набір параметрів задачі.

Агенти ресурсів (ResourceAgent) $AR=\{ar_1,ar_2,\ldots,ar_m\}$ оптимізують розподіл ресурсів. Кожен агент $ar_i \in AR$ відповідає за аналіз та оптимізацію використання певного типу ресурсу з $R=\{r_1, r_2, \ldots r_m\}$(наприклад, людські ресурси, обладнання, час).

Вони реалізують функцію

$$\mathrm{f}_r(\mathrm{R}) \to \hat{R}, \tag{3}$$

де R — вхідна множина доступних ресурсів, а вихід — матриця призначень ресурсів до завдань.

Агенти ризиків (RiskAgent) $AQ=\{a_{q1},a_{q2},\ldots,a_{qk}\}$, що прогнозують можливі затримки та конфлікти. Їхня функція

$$\mathrm{f}_q(\mathrm{v}_i, \mathrm{E}) \to \Pr(\text{затримка чи конфлікт}) \tag{4}$$

оцінює ймовірність проблеми для конкретного завдання з урахуванням його залежностей.

Центральним елементом є координуючий агент (CoordinatorAgent) AC, який інтегрує інформацію від усіх інших агентів. Він реалізує функцію перебудови графа залежностей

$$\mathrm{f}_c(\mathrm{G}, \mathrm{A}_T, \mathrm{A}_R, \mathrm{A}_Q) \to \mathrm{G}' \tag{5}$$

де G′ — оновлений граф завдань із модифікованими залежностями та розподілом ресурсів.

CoordinatorAgent буде розглядатися як оптимізаційний агент**,** що приймає рішення про те, які залежності перебудувати, які задачі підняти у пріоритеті, як перерозподілити ресурси. Це класична задача reinforcement learning (RL)**,** коли агент отримує стан системи (граф задач, ресурси, ризики) і вибирає дії, щоб максимізувати винагороду.

Стан системи (ProjectEnv) при цьому визначається матрицею залежностей (1), приоритетами завдань, розподілом ресурсів, які отримуються від ResourceAgent та ймовірностями ризиків, які отримуються від RiskAgent.

Агент може обирати дії, такі, як перебудова залежності (видалити/додати ребро), зміна пріоритету задачі, перерозподіл ресурсу.

За виконання дій агент може отримувати винагороду відповідно до цілі оптимізації - мінімізація кількості конфліктів ресурсів, зменшення ймовірності затримок, скорочення часу виконання проєкту. Винагорода визначається рівнянням

$$R = -(\alpha C + \beta D + \gamma T) \quad [6]$$

де C — кількість конфліктів, D — кількість затримок, T — час виконання, а α,β,γ — вагові коефіцієнти.
Повна схема архітектури мультиагентної системи з елементами штучного інтелекту наведена на рис. 1.

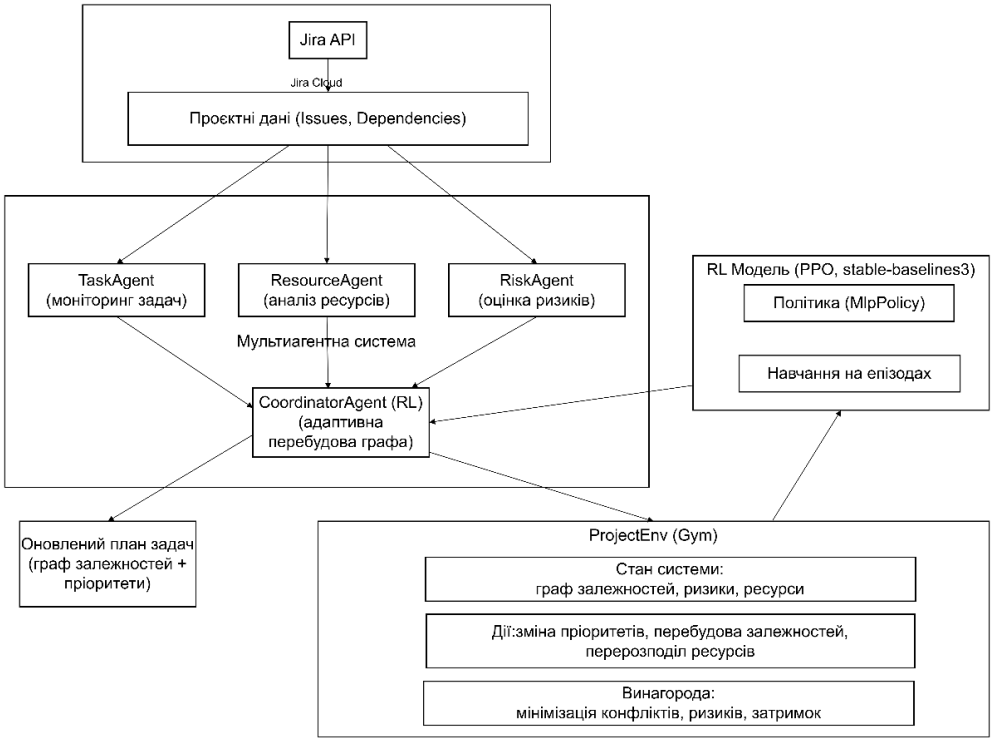

Рис. 1. Архітектура мультиагентної системи для динамічного управління завданнями в Jira

Таким чином, модель функціонує як адаптивна система, що у режимі реального часу перебудовує структуру проєкту, враховуючи зміни у вимогах, пріоритетах, ресурсах та ризиках. Це дозволяє зменшити кількість конфліктів, оптимізувати використання ресурсів та підвищити якість управлінських рішень.

Основні результати

Розроблений прототип моделі являє собою спрощену мультиагентну систему для адаптивного управління залежностями у проєктному середовищі Jira, інтегровану з методами штучного інтелекту. Архітектура додатку побудована за принципом розподіленої взаємодії спеціалізованих агентів, які виконують окремі функції моніторингу, аналізу та оптимізації, а також координуючого агента, що реалізує механізм навчання з підкріпленням.

Прототип був реалізований мовою програмування Python, з використанням бібліотек jira для інтеграції з Jira API; pandas для обробки вхідних даних; networkx для побудови графів залежностей; matplotlib для візуалізації; gym для опису середовища дії інтелектуального агента та stable-baselines3, який реалізує алгоритм Proximal Policy Optimization (PPO) для оптимізації політики координуючого агента на основі винагороди, що враховує ризики та конфлікти.

Була проведена низка експериментів з прототипом моделі, в яких виконувалось порівняння двох варіантів перебудови залежностей між задачами в системі Jira: за допомогою базового алгоритму на основі правил (зв'язки змінюються лише за простими умовами) та алгоритму з RL-CoordinatorAgent — адаптивною моделлю, яка навчається перебудовувати граф залежностей, мінімізуючи ризики та конфлікти за допомогою підкріплювального машинного навчання.

Для порівняння були використані наступні метрики:

- кількість конфліктів ресурсів (під конфліктом ресурсів маються на увазі дві задачі з однаковим виконавцем і перехресними залежностями);
- середній ризик задач (кількість залежностей × коефіцієнт ризику);
- довжина критичного шляху (кількість задач у найдовшому ланцюгу).

Винагорода формується як негативна функція цих метрик (чим менше — тим краще).

На першому етапі був створений тестовий проєкт в Jira, в якому створені 7 завдань, задані виконавці (ресурси), пріоритети та залежності (рисунок 2). Дані з Jira передаються до моделі через API Jira.

Отримані задачі з Jira:

	task_id	status	priority	assignee	dependencies
0	KAN-1	In Progress	High	Виконавець1	[]
1	KAN-2	To Do	Medium	Виконавець1	[KAN-1]
2	KAN-3	To Do	Medium	Виконавець1	[KAN-1]
3	KAN-4	In Progress	Low	Виконавець2	[KAN-2]
4	KAN-5	To Do	Low	Виконавець2	[KAN-2]
5	KAN-6	To Do	High	Виконавець3	[KAN-4, KAN-5]
6	KAN-7	To Do	Medium	Виконавець4	[KAN-6]

Рис.2. Дані, отримані з проєкту Jira

В першому експерименті пропонувалося зменшити кількість конфліктів з врахуванням ризиків, це реалізовувалося завданням функції винагороди виду:

$$reward = -(5C + 2D + 0.5T). \quad (7)$$

Результат представлений на рис. 2

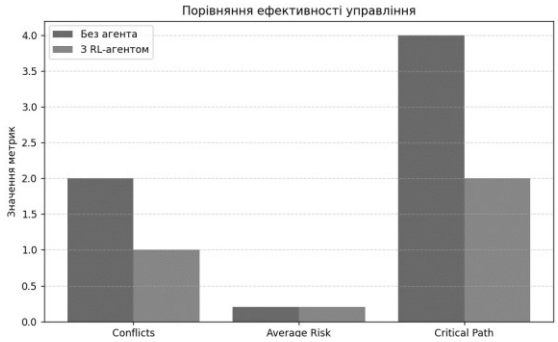

Рис. 3. Результати роботи моделі з функцією винагороди (7)

Можна бачити, що агент зменшив кількість конфліктів (з 2 до 1), перебудував залежності так, щоб ланцюг став коротшим (критичний шлях скоротився з 4 до 2), ризик залишився незмінним, бо він прямо пропорційний кількості залежностей, а агент лише змінив їхню структуру, не кількість.

В другому експерименті була спроба перебудувати функцію винагороди так, щоб агент повністю прибрав конфлікти. Для цього, окрім винагороди, був введений штраф за наявність конфліктів:

$$rewards = \begin{cases} 100 - (2D + 0{,}5T), \text{якщо } C = 0 \\ -(10C + 2D + 0{,}5T, \text{якщо } C \neq 0 \end{cases} \quad (8)$$

Результати представлені на рис 4,5.

Бачимо, що мета експерименту досягнута - агент усунув усі конфлікти, при цьому критичний шлях все ж знизився, отже, він знайшов спосіб прибрати конфлікти так, щоб ще й зменшити довжину основного ланцюга задач. Це означає, що агент оптимізує одночасно дві метрики.

На рис. 6 представлений алгоритм дії інтелектуального агента для останнього експерименту. Він показує і загальні принципи використання методів навчання з підкріпленням і спосіб зміни пріоритетів управління через зміну функції винагороди.

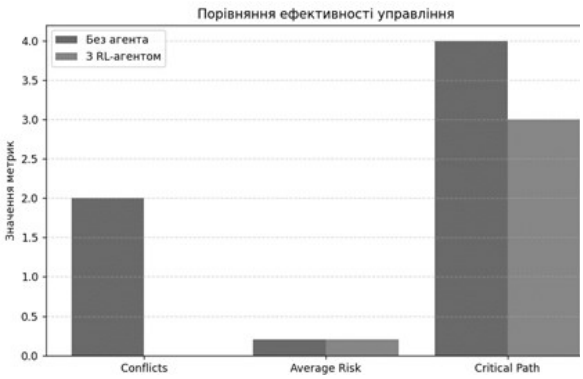

Рис. 4. Результати роботи моделі з функцією винагороди (8)

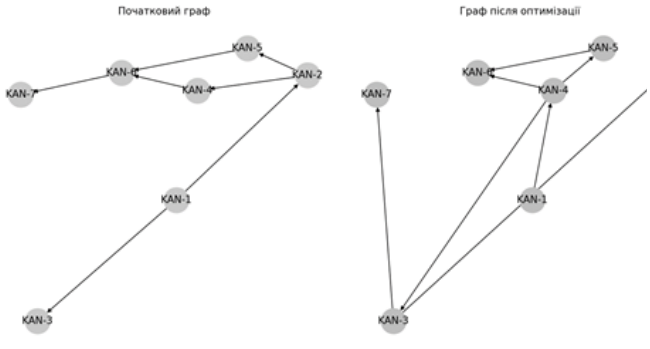

Рис. 5. Граф залежностей між задачами проєкту в результаті роботи моделі з функцією винагороди (8)

Таким чином було доведено, що інтелектуальний координуючий агент у моделі здатний отримувати інформацію про задачі, їхні залежності, ресурси та ризики та формувати уявлення про поточний стан системи, аналізувати можливі наслідки різних дій і вибирати ті, що ведуть до зменшення конфліктів, ризиків або скорочення критичного шляху. Агент діє не випадково, а керується функцією винагороди, яка визначає наскільки корисним є його вибір у контексті цілей проєкту. Завдяки навчанню він поступово адаптується, відкидає неефективні стратегії і закріплює ті, що дають кращий результат.

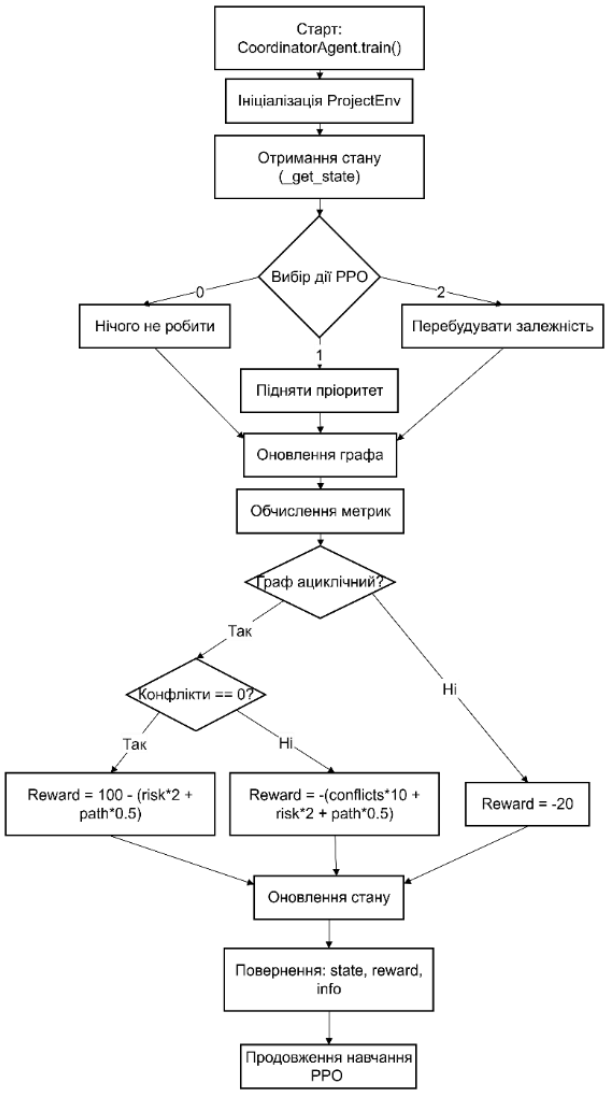

Рис. 6. Алгоритм дії інтелектуального агента, адаптований до цілей експерименту 2

Висновки

Отриманий додаток поєднує класичну архітектуру мультиагентних систем із сучасними методами штучного інтелекту. Використання методів підкріплювального навчання у координуючому агенті дозволяє системі не лише реагувати на зміни у проєкті, але й адаптивно навчатися оптимальним стратегіям управління залежностями, що забезпечує підвищення ефективності планування та зниження ризиків у великих програмних проєктах. Такий підхід дозволяє отримувати зменшення кількості конфліктів ресурсів, скорочення часу реагування на зміни у проєкті, зниження ймовірності затримок задач, підвищення узгодженості та якості планування у Jira.

Разом із тим, модель має певні обмеження. Її ефективність значною мірою залежить від якості даних, що надходять із системи Jira, адже неповні або некоректні дані можуть знижувати точність прогнозів і рішень агентів. Крім того, масштабування моделі на дуже великі проєкти може бути ускладненим через зростання кількості залежностей та ресурсних конфліктів, що потребує додаткових оптимізаційних механізмів і обчислювальних ресурсів.

Розроблений прототип може розглядатися як експериментальна платформа для дослідження мультиагентних систем у сфері управління проєктами, що поєднує формальні моделі залежностей із адаптивними алгоритмами штучного інтелекту.

Література

1. Islam Md Khyrul, Ahmed Hasib, Al Bashar Mahboob, Taher Md Abu. Role of Artificial Intelligence and Machine Learning in Optimizing Inventory Management Across Global Industrial Manufacturing & Supply Chain: A Multi-Country Review // *International Journal of Management Information Systems and Data Science*. – 2024. – Vol. 1, № 2. – P. 1–14. DOI:10.62304/ijmisds.v1i2.105

2. Adamantiadou Eleni, Tsironis Loukas. Leveraging Artificial Intelligence in Project Management: A Systematic Review of Applications, Challenges, and Future Directions // *Applied Sciences*. – 2024. – Vol. 14, № 3. – Article 1125. – DOI: 10.3390/app14031125.

3. Taboada Ianire, Daneshpajouh Abouzar, Toledo Nerea, de Vass Tharaka. Artificial Intelligence Enabled Project Management: A Systematic Literature Review // *Applied Sciences*. – 2023. – Vol. 13, № 8. – Article 5014. – DOI: 10.3390/app13085014.

4. Anthony Robert N., Govindarajan Vijay, Hartmann Frank, Kraus K., Nilsson G. *Management Control Systems*. – 12th ed. – New York: McGraw-Hill Education, 2014. – ISBN 978-0073100890.

5. Wooldridge Michael. *An Introduction to MultiAgent Systems*. – 2nd ed. – Chichester, UK: John Wiley & Sons, 2009. – ISBN 978-0470519462.

6. Jennings Nicholas R., Bussmann Stefan. Agent-Based Control Systems: Why Are They Suited to Engineering Complex Systems? // *IEEE Control Systems Magazine*. – 2003. – Vol. 23, № 3. – P. 61–73. – DOI: 10.1109/MCS.2003.1200249.

7. Russell Stuart J., Norvig Peter. *Artificial Intelligence: A Modern Approach*. – 4th ed. – Pearson, 2020. – ISBN 978-0134610993.

8. Jennings Nicholas R., Wooldridge Michael (eds.). *Foundations of Distributed Artificial Intelligence*. – Wiley-Interscience, 1996. – ISBN 978-0471536840.

9. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*. – 7th ed. – Newtown Square, PA: Project Management Institute, 2021. – ISBN 978-1628256642.

10. Office of Government Commerce (OGC). *Managing Successful Projects with PRINCE2*. – 6th ed. – London: The Stationery Office, 2017. – ISBN 978-0113315338.

11. Li Yongkui, Jing Sheng, Ding Ronggui, Josyula Hari Prasad, Todorović Marija. AI for Project Management: Revolutions, Trends, and Challenges // *Frontiers of Engineering Management*. – 2025. – Vol. 12. – P. 1236–1241. – DOI: 10.1007/s42524-025-00345-7.

12. Cherednichenko O., Matveiev O., Yanholenko O., Maneva R. Multi-Agent Modeling of Project Management Processes in Distributed Teams. – Kharkiv: National Technical University "Kharkiv Polytechnic Institute", 2025. https://ceur-ws.org/Vol-2851/paper12.pdf

13. Nguyen T., Nguyen H., Nguyen T., Nguyen H. TaDeR: Task Dependency Recommendation in Software Projects // *Proceedings of the 44th International Conference on Software Engineering (ICSE 2022)*. – 2022. https://doras.dcu.ie/27946/1/sensors-22-09492.pdf

14. Almalki A. AI-Driven Decision Support Systems for Agile Project Management // *Journal of Systems and Software Almalki, Sultan. (2025)* https://www.researchgate.net/publication/389971338_AI-Driven_Decision_Support_Systems_in_Agile_Software_Project_Management_Enhancing_Risk_Mitigation_and_Resource_Allocation