AI-АГЕНТ ЯК ДЕТЕРМІНОВАНИЙ ІНТЕРФЕЙСНИЙ ШАР ЦИФРОВОЇ ПЛАТФОРМИ

Криволапов Г. Я.
Київський столичний університет імені Бориса Грінченка, м. Київ

ВСТУП

Стрімке поширення великих мовних моделей (LLM) у складі сучасних SaaS-продуктів сформувало стійкий патерн інтеграції штучного інтелекту у вигляді ізольованого діалогового вікна – чат-бота, що працює паралельно основному функціоналу платформи. Такий підхід забезпечує природномовний ввід, проте не змінює стану системи: користувач і далі змушений вручну виконувати послідовність кліків, налаштовувати фільтри й переходити між розділами. Робота присвячена обґрунтуванню альтернативної архітектурної ролі AI-агента – як детермінованого інтерфейсного шару, що оркеструє виклики функціональних модулів та мутує стан графічного інтерфейсу замість користувача.

Актуальність і постановка проблеми. Постійне розширення функціональних можливостей сучасних SaaS-екосистем закономірно ускладнює їхнє використання: інтерфейси перетворюються на лабіринт меню, а час пошуку потрібного інструменту зростає. Класичні графічні інтерфейси, побудовані на принципах прямої маніпуляції [1], вимагають від користувача точного знання архітектури системи та виконання багатокрокових операцій, що генерує критичне когнітивне навантаження. У відповідь на цей виклик фіксується перехід до взаємодії на основі намірів – нової парадигми інтерфейсу, яка зміщує функцію планування дій з користувача на систему [2]. Розробка архітектурних підходів, у яких штучний інтелект виступає інтелектуальним посередником між складним функціоналом і потребами користувача, має очевидну практичну значущість: вона дозволяє знизити операційне тертя в професійних середовищах, прискорити роботу з даними та забезпечити адаптивність інтерфейсів під унікальні запити, що формує наукове завдання формалізації відповідних архітектурних рішень.

Існуючі підходи до інтеграції штучного інтелекту в програмні продукти переважно зводяться до зовнішніх діалогових систем – чат-ботів, які генерують текстові підказки, але не керують внутрішньою логікою платформи [3]. Окремі частини рішення окреслено у дослідженнях мовних моделей-агентів, здатних до міркувань і виклику зовнішніх інструментів [4; 5], а також у концепції генеративного інтерфейсу [6], проте жодне з них не формалізує саме архітектурну роль AI як інтегрованого інтерфейсного шару платформи. Емпіричні роботи з автономних вебагентів демонструють здатність моделей виконувати дії в реальних середовищах [7], але не пропонують узагальненої моделі взаємодії, у якій природна мова, виклики API та мутація стану GUI були б замкнуті в єдиний

детермінований цикл. Саме відсутність такої комплексної моделі гібридної взаємодії в сучасному науковому просторі обумовила вибір теми дослідження..

Мета дослідження. Сформулювати концептуальну модель AI-агента як інтегрованого інтерфейсного шару цифрової платформи, що принципово відрізняється від класичного чат-бота характером результату взаємодії, а саме – здатністю безпосередньо змінювати стан GUI-компонентів та виконувати транзакції через Backend API на основі семантичної декомпозиції наміру користувача.

Аналіз останніх досліджень і публікацій. Класична інтеграція LLM у програмний продукт реалізується через діалогову систему, що приймає текстовий запит і повертає текстову відповідь. Такий механізм добре справляється з розпізнаванням наміру, але результат залишається обмеженим неструктурованим текстовим форматом [3]. Для професійних середовищ із високою щільністю даних та необхідністю точної візуалізації звичайна текстова відповідь не може замінити функціональний графічний інтерфейс: вона позбавляє користувача інструментів прямого контролю – миттєвого фільтрування, масштабування, редагування параметрів у формі [2; 3].

Ці обмеження стають особливо критичними на тлі вичерпання можливостей класичної парадигми прямої маніпуляції, на якій будувалися графічні інтерфейси з 1980-х років [1]. Прямою маніпуляцією забезпечувався принцип «бачу і керую»: користувач безпосередньо взаємодіяв з об’єктами на екрані, а кожна дія мала миттєвий візуальний відгук. Однак для сучасних SaaS-екосистем кількість одночасно представлених функціональних об’єктів настільки велика, що сама ідея прямого керування кожним з них генерує надлишкове операційне тертя. Природномовний інтерфейс мав би розв’язати цю проблему, проте через зовнішню позицію чат-бота щодо платформи його потенціал реалізується лише частково. Принциповою ознакою чат-бота як інтеграційного патерна є його зовнішня позиція щодо платформи. Бот існує паралельно функціональному ядру системи: він відповідає на запит, але не торкається конфігурації застосунку – не активує приховані фільтри, не заповнює форми, не здійснює навігацію. Користувач отримує підказку природною мовою та повертається до традиційного імперативного керування – серії кліків, що відтворюють описаний агентом сценарій. У результаті природномовний ввід не усуває операційного тертя, а лише додається до нього як додатковий комунікаційний шар.

Альтернативним напрямком, спрямованим на подолання обмежень суто діалогового виходу, є дослідження мовних моделей, здатних до міркувань разом із виконанням дій. Парадигма ReAct формалізувала чергування внутрішніх кроків міркування з викликами зовнішніх інструментів, що дозволило агентам адаптивно реагувати на середовище

[4]. Підхід Toolformer довів, що мовні моделі здатні самостійно навчатися викликати API, передаючи структуровані запити до програмних модулів та опрацьовуючи їхні відповіді [5]. Ці роботи створили технічний фундамент для виходу за межі текстового виводу, проте вони зосереджені на завданнях у відносно ізольованих середовищах і не формалізують саме архітектурну роль агента в межах цифрової платформи з графічним інтерфейсом.

Спробою частково подолати цей розрив є концепція Generative UI, що передбачає динамічне рендерування адаптивних візуальних віджетів безпосередньо під контекст запиту [6]. Однак Generative UI вирішує лише задачу візуальної форми: модель навчається малювати інтерфейс під запит, але не керує наявним функціоналом платформи – її пошуком, фільтрами, базою даних. Як показують емпіричні дослідження автономних агентів у реальних вебсередовищах, без глибокої синхронізації з бекенд-інфраструктурою та функціональним станом платформи модель не здатна забезпечити цілісного управління системою [7]. Тобто проблема залишається відкритою: природна мова досі не може виступати повноцінним механізмом керування платформою.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Запропонована концептуальна модель позиціонує AI-агента не як зовнішнього співрозмовника, а як інтегрований архітектурний компонент, вбудований між користувачем і функціональними модулями платформи. Природна мова використовується виключно як механізм введення високорівневого наміру; далі агент здійснює семантичну декомпозицію цього наміру на формалізовані команди, ініціює виклики API та змінює стан GUI-компонентів для відображення результату. Принципова відмінність такої моделі – це характер виходу: не текстове повідомлення, а мутація стану самої системи.

На прикладному рівні це означає, що агент автоматично заповнює форми, активує приховані фільтри, встановлює значення повзунків, формує таблиці порівняння та здійснює навігацію між розділами. Користувач не натискає кнопок і не обирає пунктів меню – він формулює намір, а спостерігає за результатом у звичних візуальних компонентах. Відповідно змінюється і його роль: з оператора кожної дії він перетворюється на куратора процесу, який верифікує виконання та за потреби коригує параметри ручним редагуванням або уточненням мовою.

Така архітектурна роль детермінована за двома вимірами. По-перше, агент завжди працює у замкненому контурі «намір–план–виклик API–мутація GUI» [4; 5], тобто кожен крок має передбачуваний технічний результат, а не ймовірнісний текстовий вивід. По-друге, кінцевий стан системи завжди відображається у графічному інтерфейсі – користувач бачить, які фільтри встановлено, які параметри застосовано і яка саме вибірка побудована, що забезпечує прозорість виконання та можливість

миттєвого ручного коригування. Загальна архітектурна композиція системи, у якій AI-агент виступає інтерфейсним шаром між клієнтським середовищем та функціональним ядром, представлена на рисунку 1, а розгортання замкненого контуру за стадіями циклу взаємодії – у таблиці 1.

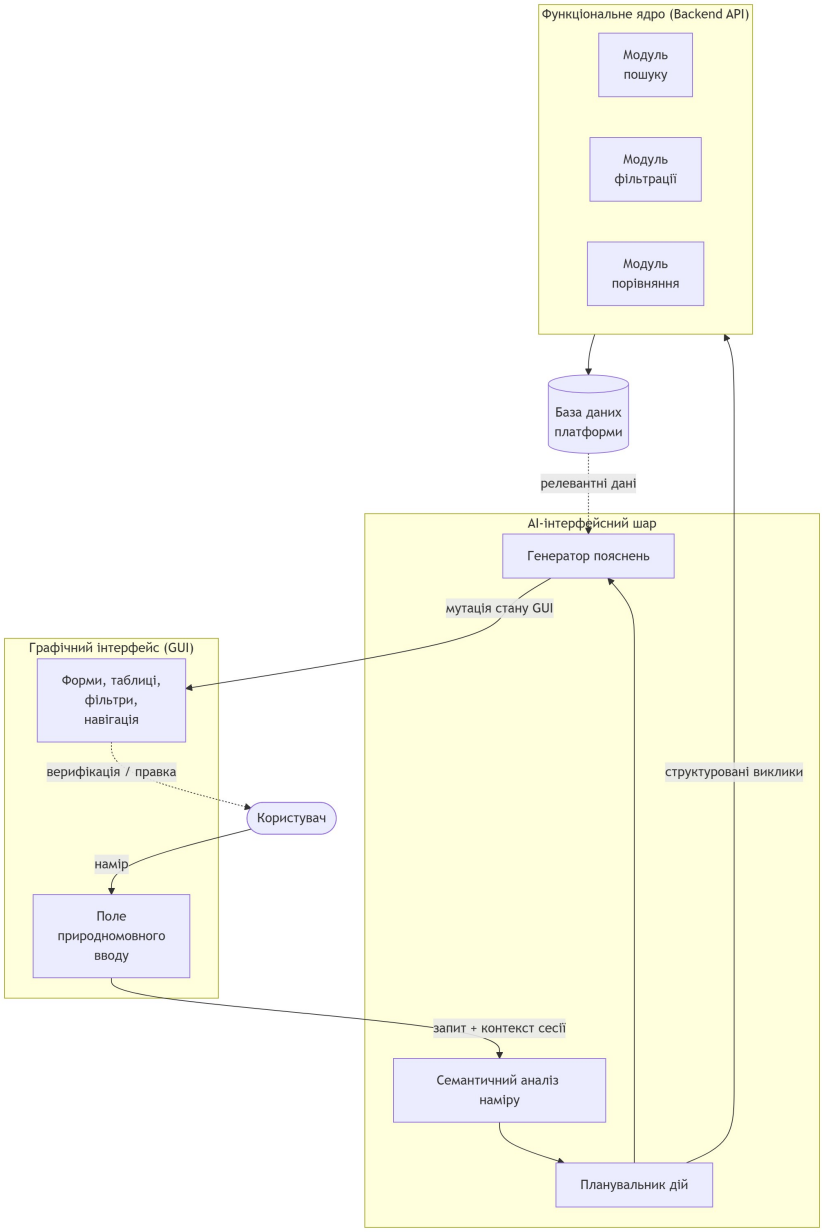

Рис. 1. Архітектурна композиція AI-інтерфейсного шару платформи

Таблиця 1

Стадії циклу взаємодії з функціями AI-інтерфейсного шару

Стадія циклу	**Дії AI-шару**	**Результат стадії**
Сприйняття наміру	Семантичний аналіз природномовного запиту, врахування поточного стану системи та історії сесії	Формалізована структура цілі та обмежень

Продовження таблиці 1

Планування дій	Декомпозиція цілі на послідовність викликів функціональних модулів платформи	Черга структурованих звернень до Backend API
Виконання у платформі	Виклик Backend API; автоматична мутація стану GUI – форми, фільтри, таблиці, навігація	Зміна стану цифрового середовища
Верифікація та корекція	Генерація пояснень логіки відбору; обробка ітеративних уточнень; збереження ручного контролю	Узгоджений із наміром стан системи

Запропонована роль AI-агента розташовується між двома усталеними парадигмами: класичним графічним інтерфейсом, що вимагає імперативної послідовності кліків, та ізольованим чат-ботом, що відповідає текстом без впливу на стан системи. Принципові відмінності цих трьох моделей за ключовими критеріями систематизовано у таблиці 2.

Таблиця 2

Структурне порівняння трьох парадигм взаємодії

Парадигма	**Принцип взаємодії**	**Тип результату**	**Ціна для користувача**
Класичний GUI	Імперативна послідовність кліків та налаштувань	Статична сторінка або список	Висока: знання «карти» системи, ручна декомпозиція задачі
Ізольований чат-бот	Текстовий діалог поза функціоналом платформи	Текстова відповідь у діалоговому вікні	Середня: втрата прямого контролю, ручне виконання плану
AI-інтерфейсний шар	Намір → план → API → мутація стану GUI	Динамічна зміна стану цифрового середовища	Низька: делегування виконання, контроль через правку результату

Класичний GUI забезпечує максимальну прозорість і контроль ціною високого когнітивного навантаження: користувач сам має знати «карту» системи [1]. Чат-бот знижує бар’єр входу через природну мову, але втрачає прозорість і прямий контроль. AI-інтерфейсний шар поєднує переваги обох парадигм: природномовний ввід зберігає низький поріг входу, тоді як обов’язкова візуалізація дій у GUI повертає прозорість та можливість ручного коригування.

ВИСНОВКИ

У результаті проведеного дослідження було сформульовано концептуальну модель AI-агента як детермінованого інтерфейсного шару цифрової платформи, що принципово відмінна від ролі ізольованого чат-бота. Якщо чат-бот працює паралельно функціоналу системи й повертає текстову відповідь, то інтерфейсний шар вбудовується між користувачем і функціональними модулями та змінює стан платформи: заповнює форми, активує фільтри, здійснює навігацію тощо. Природна мова виступає як механізм введення наміру, а GUI – як середовище верифікації результату й ручного коригування. Така архітектурна роль усуває паралелізм AI та платформи і забезпечує цілісне управління складними SaaS-середовищами без втрати прозорості виконання та контролю користувача.

ДЖЕРЕЛА

1. Hutchins E. L., Hollan J. D., Norman D. A. Direct manipulation interfaces. *Human-Computer Interaction*. 1985. Vol. 1, № 4. P. 311–338. DOI: 10.1207/s15327051hci0104_2.

2. Nielsen J. AI: First New UI Paradigm in 60 Years. *UX Tigers*. 2023. URL: https://www.uxtigers.com/post/ai-new-ui-paradigm.

3. Ordoumpozanis K., Konstantakis M., Zoi S., Caridakis G. Generative AI: A Systematic Review of Related Interfaces and Interactions. *Proceedings of the 3rd International Conference of the ACM Greek SIGCHI Chapter*. 2025. P. 39–47. DOI: 10.1145/3749012.3749052.

4. ReAct: Synergizing reasoning and acting in language models / S. Yao, J. Zhao, D. Yu [et al.]. 2022. DOI: 10.48550/arXiv.2210.03629.

5. Toolformer: Language Models Can Teach Themselves to Use Tools / T. Schick, J. Dwivedi-Yu, R. Dessì [et al.]. 2023. DOI: 10.48550/arXiv.2302.04761.

6. Generative UI: A rich, custom, visual interactive user experience for any prompt. *Google Research Blog*. 2025. URL: https://research.google/blog/generative-ui-a-rich-custom-visual-interactive-user-experience-for-any-prompt/.

7. Dähling S., Razik L., Monti A. Enabling scalable and fault-tolerant multi-agent systems by utilizing cloud-native computing. *Autonomous Agents and Multi-Agent Systems*. 2021. Vol. 35, № 1. Art. 10. DOI: 10.1007/s10458-020-09489-0.

8. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., Hlushak, O., Zhyltsov, O., Ilich, L., Kobets, N., Kovaliuk, T., Kuchakovska, H., Lytvyn, O., Lytvyn, P., Mashkina, I., Morze, N., Nosenko, T., Proshkin, V., Radchenko, S., & Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science. https://doi.org/10.28925/9720213284km