

ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МЕТОДИКИ СТРУКТУРНОГО СИНТЕЗУ АДАПТИВНИХ ІНТЕРФЕЙСІВ НА ОСНОВІ АНАЛІЗУ НАМІРІВ КОРИСТУВАЧА

Криволапов Гліб Ярославович

студент групи МППМ-І-25-14д,

Київського столичного університету імені Бориса Грінченка,

науковий керівник – д.т.н., проф. Бондарчук А.П.

ВСТУП

Актуальність і постановка проблеми. Сучасні інформаційні системи класу SaaS містять сотні функцій, розподілених між десятками екранів, меню й панелей фільтрів. За законом Хіка–Хаймана час вибору зростає з логарифмом кількості альтернатив [1; 2], тож користувач витрачає зусилля на пошук елементів керування, а не на саму задачу. Парадигма прямої маніпуляції [3] покладає декомпозицію задачі на людину, тоді як великі мовні моделі (LLM) уможливили взаємодію на основі намірів: користувач описує бажаний результат, а система сама визначає шлях до нього [4]. Проте типовий чат-бот повертає текст і не дає візуального контролю над результатом [5]. У попередніх роботах [5 – 7] запропоновано гібридну модель, у якій AI-агент є інтерфейсним шаром-оркестратором між графічним інтерфейсом (GUI) і функціональним ядром. На її основі розроблено методику структурного синтезу адаптивних інтерфейсів.

Мета дослідження. Експериментально перевірити методику на програмному прототипі: оцінити точність інтерпретації намірів, латентність, вартість, надійність і безпеку шару-оркестратора та порівняти адаптивний режим із класичним інтерфейсом прямої маніпуляції.

Аналіз останніх досліджень і публікацій. Здатність LLM діяти через інструменти показано в парадигмі ReAct [8] і підході Toolformer [9], проте цикл «міркування – дія – спостереження» потребує кількох звернень до моделі на задачу, що збільшує латентність і вартість. GUI-агенти [10] імітують дії користувача без доступу до контракту API, а генеративний інтерфейс [11] синтезує нові елементи під кожен запит, але не керує наявним функціоналом платформи. Схильність моделей до галюцинацій [12] вимагає валідації виведення, а людино-центрований підхід [13] – поєднання автоматизації з контролем людини. У попередніх дослідженнях було обґрунтовано гібридну модель «природномовний ввід – графічний вивід» [5], трирівневу архітектуру з шаром-оркестратором [6] і роль AI-агента як детермінованого інтерфейсного шару [7], а емпіричну перевірку моделі визначено напрямом подальших досліджень [6].

МЕТОДИ ТА ЗАСОБИ ДОСЛІДЖЕННЯ

Задачу структурного синтезу формалізовано як відображення

$$F:(I,C) \rightarrow (P,\Delta S), \quad (1)$$

де I – намір; C – контекст GUI (подання, фільтри, вибрані об'єкти, історія сесії); P – план викликів інструментів серверного ядра; ΔS – мутація стану

клієнтського сховища. Модель лише формує план за одне звернення в строгому режимі виклику інструментів із JSON Schema, а валідацію, виконання, побудову патча й пояснення здійснює детермінований оркестратор; критичні дії потребують підтвердження, мутації оборотні. Методика охоплює шість етапів: інвентаризацію модулів, опис контрактів API, формалізацію простору намірів (дев'ять класів), відображення «намір – дії», правила мутації GUI та механізми контролю.

Прототип вебсистеми реалізації нерухомості побудовано на Vue 3 (Pinia), NestJS 11, PostgreSQL 16 і Redis 7; ядро публікує шість інструментів (пошук, фільтрація, порівняння, картка об'єкта, заявка, навігація) через Model Context Protocol. Планувальником слугували моделі DeepSeek V4 Pro (перша ітерація) і DeepSeek-V4.1-Flash (підсумкова система) з нульовою температурою генерації в режимі без розширених міркувань.

Експеримент 1 оцінював точність на корпусі з 96 формулювань для дев'яти класів намірів і на відкладеному наборі з 18 нових формулювань (кожне – тричі: 288 і 54 прогони). Строгим успіхом задачі вважали повний збіг підсумкового стану GUI (фільтрів, подання, вибірки, маршруту, форми підтвердження) з еталоном. Експеримент 2 вимірював латентність і вартість (по 30 прогонів на клас) та зіставляв час відповіді з оцінкою

$$T_{total} = T_{ctx} + n_{LLM} \cdot T_{LLM} + \sum T_{tool,i} + T_{render}, \quad (2)$$

де T_{ctx} – час збирання контексту; n_{LLM} – кількість звернень до моделі тривалістю T_{LLM} ; $T_{tool,i}$ – тривалість i -го виклику інструмента; T_{render} – час застосування патча. Експеримент 3 перевіряв підтвердження критичних дій, обмеження частоти, валідацію запитів, змагальні запити й деградацію за відмови моделі. Порівняння з класичним інтерфейсом виконано за моделлю KLM-GOMS [15]:

$$T = n_K \cdot t_K + n_P \cdot t_P + n_H \cdot t_H + n_M \cdot t_M + \sum_j R_j, \quad (3)$$

де n_K , n_P , n_H , n_M – кількість натискань клавіш, вказувань мишею, перенесень руки та ментальних підготовок із нормативними тривалостями $t_K = 0,28$ с, $t_P = 1,1$ с, $t_H = 0,4$ с, $t_M = 1,35$ с; R_j – час очікування відповіді системи. Задачі: T1 – будинки у Львові; T2 – двокімнатні квартири в Києві до 100 000 \$; T3 – T2 із сортуванням і порівнянням двох найдешевших; T4 – звуження T2 до новобудов; T5 – картка найдешевшого об'єкта у Львові; T6 – заявка на найдешевшу квартиру з T2; T7 – список власних заявок.

Перевірялися гіпотези: Н1–Н4 – адаптивний режим зменшує час виконання складених задач, кількість операцій, можливостей помилки та когнітивне навантаження; Н5 – план формується за одне звернення, а час відповіді не виходить за межі оцінки (2); Н6 – нижня межа 95 % довірчого інтервалу (ДІ) частки успішних задач не нижча за 90 %. Вимірювання виконувала спеціальна тестова програма, що працює з прототипом як із чорною скринькою; для часток обчислено ДІ Вілсона, для середнього часу – ДІ за t -розподілом.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

У першій ітерації (DeepSeek V4 Pro) строгий успіх задачі становив 88,2 %. Аналіз 34 невдалих прогонів показав, що більшість невдач спричиняв код системи, а не модель: нормалізація перетворювала «від 40 до 70 тисяч» на «від 40 до 70000», англійські назви міст не зіставлялися з каталогом, а схема фільтра не дозволяла зняти умову. Далі виправлено десять дефектів і замінено модель на DeepSeek-V4.1-Flash (змінюю однієї змінної середовища); щоб розділити вплив цих змін, нову модель прогнано на базовій і виправленій версіях (табл. 1, у дужках – 95 % ДІ).

Таблиця 1

Точність інтерпретації наміру для різних конфігурацій планувальника, %

Конфігурація планувальника	Точність класу наміру	Успіх задачі (строго), корпус	Успіх задачі (строго), відкладений набір
DeepSeek V4 Pro, базова версія	97,9 (95,5–99,0)	88,2 (84,0–91,4)	29,6 (19,1–42,8)
DeepSeek-V4.1-Flash, базова версія	99,0 (97,0–99,6)	92,4 (88,7–94,9)	57,4 (44,2–69,7)
DeepSeek-V4.1-Flash, виправлена версія	100,0 (98,7–100,0)	96,9 (94,2–98,3)	98,1 (90,2–99,7)
Евристичний планувальник, виправлена версія	86,5 (78,2–91,9)	63,5 (53,6–72,5)	—

На відкладеному наборі успіх зріс з 29,6 до 98,1 % (53 задачі з 54), а інтервали трьох конфігурацій не перетинаються, тобто виправлення узагальнюються на нові формулювання. Підсумкова система прийняла план з першого звернення в усіх 288 прогонах і повторила однаковий план у трьох прогонах для 96,9 % формулювань; її перевага над детермінованим евристичним планувальником (96,9 проти 63,5 %) статистично значуща, а нижня межа ДІ (94,2 %) перевищує 90 %, що підтверджує гіпотезу Н6. В одному з трьох невдалих формулювань модель замінила невідому системі умову (ціну за квадратний метр) найближчою відомою замість уточнювального запитання.

Усі 270 запитів експерименту 2 опрацьовано за одне звернення до моделі ($n_{LLM} = 1$): медіана часу відповіді за класами намірів становить 0,99–1,56 с, 95-й процентиль – не більше 1,75 с, середні значення лежать у межах оцінки (2), отже, гіпотезу Н5 підтверджено. Звернення до моделі займає 98,9–99,5 % часу відповіді, виклики інструментів – 2–9 мс на намір, тобто власні витрати оркестратора не перевищують 1 %. Завдяки незмінному префіксу промπτу 95–97 % вхідних токенів оплачуються за тарифом кешу, і вартість у непікові години становить 0,10–0,25 \$ за тисячу намірів – у 3,6 раза менше, ніж з V4 Pro, і щонайменше у 25 разів менше, ніж із закритими моделями. Епізод уповільнення постачальника в першій ітерації (до 6–9 с на звернення)

обґрунтовує граничний час звернення й деградацію до детермінованого планувальника.

В експерименті 3 механізм підтвердження критичної дії відпрацював в усіх шести сценаріях (повторне, чуже й одночасне підтвердження, відмова, повторна заявка), обмеження частоти відхилило 5 із 35 одночасних запитів точно за лімітом, а за тайм-ауту моделі евристичний планувальник дав коректні відповіді в 45 випадках із 45. Усі 10 змагальних запитів (ін'єкція інструкцій, видалення даних, перехід на системний шлях, заявка від чужого імені тощо) заблоковано.

Для адаптивного режиму за моделлю (3) розраховано канонічну оцінку (один ментальний оператор M на фразу наміру) і консервативну (один M на кожну умову). Порівняння (рис. 1) показало, що жоден режим не домінує.

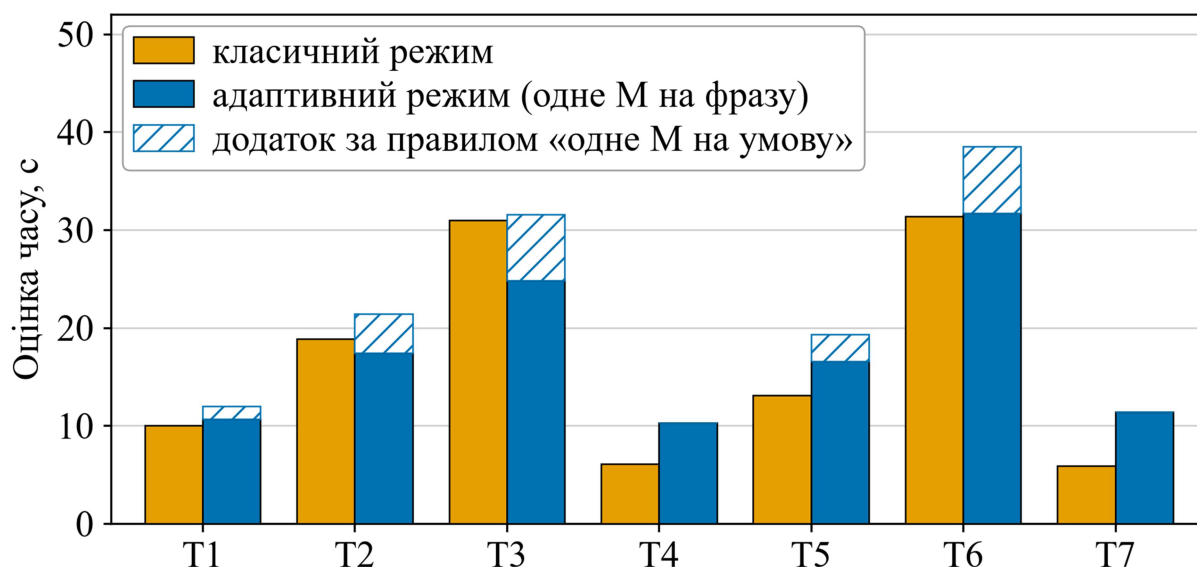


Рис. 1. Оцінка часу виконання задач T1–T7 за моделлю KLM-GOMS

Для простих задач (T1, T4, T5, T7) класичний режим швидший на 7–94 %, бо набір фрази й очікування відповіді моделі довші за вибір зі списку. У складеній задачі T3 адаптивний режим швидший на 20 % за канонічною оцінкою і на 11–34 % для досвідченого користувача клавіатури ($t_k = 0,20$ с); для задач пошуку він вигідніший, починаючи приблизно з чотирьох умов відбору. Адаптивний режим зменшує кількість вказувань мишею (1–3 проти 2–10) і точок вибору (1–2 проти 1–7), усуває вибір зі списків (до 11,9 біта на задачу), але потребує більше натискань клавіш і переносить навантаження з розпізнавання на пригадування, тому гіпотези Н1–Н4 підтверджено частково. Додаткові інвестиції в шар-оркестратор (88,1 тис. грн) для агентства з 20 операторами окуповуються за 1,4–2,2 року, хоча за консервативною оцінкою прямого виграшу в часі немає.

ВИСНОВКИ

Експериментальне дослідження підтвердило працездатність методики структурного синтезу адаптивних інтерфейсів на основі аналізу намірів користувача. Підсумкова система виконала 96,9 % задач корпусу (детермінований планувальник – 63,5 %) і 98,1 % задач відкладеного набору; більшість помилок першої ітерації спричиняв код системи, а не модель. Кожен намір опрацьовується за одне звернення до моделі з медіаною відповіді 0,99–

1,56 с і вартістю 0,10–0,25 \$ за тисячу намірів, а механізми підтвердження, валідації й деградації відпрацювали в усіх сценаріях. Порівняння за моделлю KLM-GOMS показало, що адаптивний режим вигідний для складених задач, а класичний – для простих, що обґрунтовує гібридну архітектуру зі збереженням прямої маніпуляції. Подальші дослідження передбачають перевірку оцінок KLM за участю користувачів і перенесення методики на інші предметні області.

ДЖЕРЕЛА

1. Hick W. E. On the rate of gain of information. Quarterly Journal of Experimental Psychology. 1952. Vol. 4, No. 1. P. 11–26. DOI: 10.1080/17470215208416600.
2. Hyman R. Stimulus information as a determinant of reaction time. Journal of Experimental Psychology. 1953. Vol. 45, No. 3. P. 188–196. DOI: 10.1037/h0056940.
3. Hutchins E. L., Hollan J. D., Norman D. A. Direct manipulation interfaces. Human-Computer Interaction. 1985. Vol. 1, No. 4. P. 311–338. DOI: 10.1207/s15327051hci0104_2.
4. Nielsen J. AI: First New UI Paradigm in 60 Years. UX Tigers. 2023. URL: <https://www.uxtigers.com/post/ai-new-ui-paradigm>
5. Криволапов Г. Трансформація парадигми взаємодії: перехід від GUI до гібридних інтерфейсів на основі намірів. Сучасні комп'ютерні системи та технології: матеріали VIII Всеукр. наук.-практ. інтернет-конф. студентів, аспірантів та молодих вчених за тематикою «Сучасні комп'ютерні системи та мережі в управлінні» (24 листопада 2025 р., м. Херсон, м. Хмельницький) / за ред. А. А. Григорової. Херсон: Книжкове видавництво ФОП Вишемирський В. С., 2025. С. 31–33. URL: <https://doi.org/10.5281/zenodo.17711825>
6. Бондарчук А., Криволапов Г. Модель гібридної взаємодії з платформою на основі AI-агента. Кібербезпека: освіта, наука, техніка. 2026. Т. 1, № 33. С. 647–656. DOI: 10.28925/2663-4023.2026.33.1246.
7. Криволапов Г. AI-агент як детермінований інтерфейсний шар цифрової платформи. Інформаційні технології – 2026: зб. тез XIII Всеукраїнської науково-практичної конференції молодих учених. Київ: Київський столичний університет імені Бориса Грінченка, 2026. С. 130–135.
8. ReAct: Synergizing Reasoning and Acting in Language Models / S. Yao et al. arXiv preprint. 2022. DOI: 10.48550/arXiv.2210.03629.
9. Toolformer: Language Models Can Teach Themselves to Use Tools / T. Schick et al. arXiv preprint. 2023. DOI: 10.48550/arXiv.2302.04761.
10. Large Language Model-Brained GUI Agents: A Survey / C. Zhang et al. arXiv preprint. 2024. DOI: 10.48550/arXiv.2411.18279.
11. 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/>
12. Survey of Hallucination in Natural Language Generation / Z. Ji et al. ACM Computing Surveys. 2023. Vol. 55, No. 12. Art. 248. P. 1–38. DOI: 10.1145/3571730.
13. Shneiderman B. Human-centered AI: ensuring human control while increasing automation. Proceedings of the 5th Workshop on Human Factors in Hypertext. 2022. P. 1–2. DOI: 10.1145/3538882.3542790.
14. Model Context Protocol Specification. Model Context Protocol. URL: <https://modelcontextprotocol.io/specification/latest>