УДК 004.41:004.8

ШІ-АСИСТЕНТИ ТА АГЕНТИ У РОЗРОБЦІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ: МОЖЛИВОСТІ, РИЗИКИ ТА ПЕРСПЕКТИВИ

Данілов Є. В.
Національний університет «Одеська юридична академія»
Україна, Одеса
jre4b4enok@gmail.com

Анотація. *Проаналізовано можливості, ризики та перспективи використання ШІ-асистентів та агентів у розробці програмного забезпечення на прикладі сучасних моделей GPT-6 Astra, Claude Opus 5.5, DeepSeek V4.1 та Kimi K3. Зіставлено практику vibe coding без фахової підготовки та ШІ-асистовану інженерію, яку виконує підготовлений програміст. Виокремлено основні групи ризиків (вразливості коду, галюцинації пакетів, технічний борг, втрата навичок, автономність агентів) і узагальнено заходи їх зниження. Обґрунтовано, що перспективи програмістів пов'язані зі зміщенням ролі від написання коду до архітектури та верифікації результатів.*

Ключові слова: *штучний інтелект; великі мовні моделі; ШІ-агенти; vibe coding; розробка програмного забезпечення.*

Вступна частина. Інтеграція великих мовних моделей (LLM) у розробку програмного забезпечення (ПЗ) спричинила найрадикальніший зсув у підходах до створення ПЗ з часів появи компіляторів. Нова практика, відома як vibe coding, стосується створення продукту через ітеративний діалог зі ШІ природною мовою, часто без глибокого аудиту коду людиною [1]. Інструменти генеративного ШІ змінюють життєвий цикл розробки ПЗ (SDLC) і саму роль програміста [2], а питання, чи замінить ШІ розробників, залишається предметом дискусій. Одні дослідження фіксують зростання продуктивності, інші вказують на ризики безпеки та деградацію навичок, тому актуальним є зіставлення сучасних моделей, пов'язаних із ними ризиків і перспектив фахівців.

Метою роботи є систематизація можливостей, ризиків і перспектив використання ШІ-асистентів та агентів (GPT-6 Astra, Claude Opus 5.5, DeepSeek V4.1, Kimi K3) у розробці ПЗ та порівняння підходів vibe coding і ШІ-асистованої інженерії.

Основна частина. Перші масові ШІ-асистенти (GitHub Copilot, Tabnine, CodeWhisperer) надають функціонал для генерації коду, автоматичного пошуку помилок, створення технічної документації та тестування, скорочують час на рутинні завдання й спрощують освоєння професії початківцями. У межах SDLC ШІ-інструменти застосовуються на всіх етапах: від збору вимог до розгортання та підтримки, зокрема в парадигмі AIOps [2]. Систематичні огляди фіксують зростання продуктивності на 26-30% і швидшу зміну ітерацій, найбільше помітні серед розробників початкового рівня [1]. ШІ ефективно бере на себе шаблонний код, а швидке створення прототипів і мінімально життєздатних продуктів (MVP) стає доступним навіть особам без профільної освіти. Водночас ШІ зміщує роль програміста від написання коду до архітектурної діяльності та контролю якості [3].

Нове покоління моделей перетворило асистентів на агентів, здатних самостійно виконувати довгі багатокрокові завдання. Для огляду було вибрано чотири найновіші моделі (табл. 1).

Таблиця 1 – Сучасні моделі для розробки ПЗ

Модель	Розробник	Доступ	Особливості для розробки
GPT-6 Astra	OpenAI	Платні тарифи ChatGPT, API	Автономна робота з кодом, браузером і професійними програмами; контекст 1,05 млн токенів [4]
Claude Opus 5.5	Anthropic	API, AWS, Google Cloud, Microsoft Azure	Довготривалі агентні завдання, аудит коду; Terminal-Bench 4.0 – 66,4 % [5]
DeepSeek V4.1-Flash-1017	DeepSeek	Відкриті ваги (MIT)	Агентне кодування: планування проєкту, написання коду, виклик інструментів [6, 7]
Kimi K3	Moonshot AI	Відкриті ваги	2,8 трлн параметрів; контекст 1 млн токенів; довготривале кодування [8]

Складено автором на основі [4-8].

OpenAI позиціонує GPT-6 Astra (вересень 2026) як модель для автономної роботи з браузером, кодом і професійними програмами; вона першою в компанії отримала критичний рівень кіберможливостей. Контекстне вікно моделі становить 1,05 млн токенів, а вартість API – $10 за мільйон вхідних і $50 за мільйон вихідних токенів [4].

Claude Opus 5.5 (вересень 2026) за даними Anthropic набрала 66,4% у тесті Terminal-Bench 4.0 проти 55,8% у Fable 5.1 та 52,3% в Opus 5, працює швидше більш ніж на 30% і коштує приблизно на 40% менше за Opus 5; у внутрішніх випробуваннях модель оптимізувала швидкість завантаження вебзастосунків у 39 випадках із 40 [5].

Китайські розробники зробили ставку на відкриті ваги. DeepSeek у квітні 2026 року представила V4 у відкритому вигляді та заявила про найкращу агентну здатність кодування серед відкритих моделей [6], а 17 жовтня випустила оновлення V4.1-Flash-1017 із вагами під ліцензією MIT, орієнтоване на сценарії, де агент планує проєкт, пише код, викликає інструменти та ітерує до робочого результату [7].

Kimi K3 від Moonshot AI (липень 2026) має 2,8 трлн параметрів і контекстне вікно 1 млн токенів та призначена для довготривалого кодування. За рейтингом Vals AI вона посіла друге місце загалом, поступившись Fable 5 і випередивши GPT-5.6 Sol, а в Arena очолила рейтинг із веброзробки [8]. Наведені показники отримані за різними тестами та порівняно з різними поколіннями моделей, тому пряме зіставлення моделей за ними некоректне.

За оцінкою ризиків у звіті Veracode, в якому проаналізовано 18 моделей, середня частка безпечного коду зросла з 56 до 62%, а частка завдань із вразливостями зменшилась із 44 до 38% [9]. Лідерами стали Claude Opus 5.5 та GPT-6-Sol із 73%, тобто навіть найкращі моделі не проходять понад чверть перевірок безпеки [10]. Для агентів ситуація гостріша: у тесті LSCBench шість провідних агентів виконали функціонально 73,6-91,3% завдань, проте код, що є одночасно робочим і безпечним, отримано лише у 16,4-25,1% випадків [11]. Щодо продуктивності, то METR визнала власний результат 2025 року (на 19% триваліше з ШІ) застарілим; оновлення лютого 2026 року вказує на ймовірне прискорення, але через зміни в поведінці розробників нові дані визнано ненадійними [12].

Автономність агентів залишається окремим ризиком: OpenAI повідомила, що під час окремих випробувань Astra могла приховувати свої дії [4], а у вересні 2026 року скасувала випуск GPT-6.1 Astra, бо модель могла діяти без дозволу користувача, звертатися до зовнішніх сервісів і приховувати частину дій [13]. Щодо DeepSeek Anthropic та OpenAI висувають звинувачення в неправомірній дистиляції власних моделей [6]. Раніше виявлені проблеми зберігають актуальність як тенденції, хоча отримані на попередніх поколіннях

моделей: 2025 року 45% зразків коду мали вразливості з переліку OWASP Top 10 [1]. Для SDLC характерні галюцинації неіснуючих бібліотек (slopsquatting), проблеми інтелектуальної власності та Shadow AI, виявлені в порівнянні GPT-4 і Gemini 2.5 Pro [2]. Масштабування vibe coding призводить до «темного боргу» [1]. У дослідженні Anthropic (2025) група з ШІ-асистентом набрала на 17% менше балів із нових концепцій [1], що змушує переглядати й підготовку інженерів. Ризики та заходи їх зниження зведено в табл. 2.

Таблиця 2 – Ризики використання ШІ-агентів і заходи їх зниження

Ризик	Прояв	Захід зниження
Кібербезпека	SQL-ін'єкції, XSS, слабка криптографія; 38 % завдань із вразливістю	Обов'язковий SAST/DAST та рев'ю перед коммітом
Галюцинації пакетів	Підключення неіснуючих бібліотек, slopsquatting	Перевірка залежностей та їх репутації
Технічний борг	«Темний борг», втрата контексту в довгих сесіях	Структуру задає людина; дрібні pull request
Втрата навичок	Некритичне прийняття згенерованого коду	ШІ як асистент; самостійне розв'язання базових задач
Автономність агентів	Дії без дозволу, приховування дій	Обмеження прав, ізольоване середовище, моніторинг дій

Складено автором на основі [1, 2, 4, 9-11, 13].

Порівняння підходів показує принципову різницю між vibe coding і ШІ-асистованою інженерією (табл. 3). Якщо кожен рядок згенерованого коду перевірено, протестовано й зрозуміло, це вже не vibe coding, а використання моделі як асистента [1]. Людина без фахової підготовки не здатна оцінити результат ШІ. У таких проєктах накопичуються помилки, вразливості та «темний борг», тому для виробничих систем чистий vibe coding визнається деструктивним, а його сенс обмежений прототипами, автоматизацією рутини й шаблонним кодом [1]. Для досвідченого програміста ШІ є потужним підсилювачем: рутинні завдання делегуються моделі, а увага зосереджується на архітектурі та перевірці коду. Економія часу ймовірна, але її розмір поки не встановлено надійно [12]. При цьому частину виграшу поглинає аудит результатів, оскільки безпечний код агенти створюють лише в меншості випадків [11]. Цінність фахівця зміщується від знання синтаксису до системного мислення, бачення продукту та оркестрації ШІ-агентів [1]. Про заміну програмістів поки взагалі не йдеться.

Таблиця 3 – Vibe coding і ШІ-асистована інженерія

Критерій	Vibe coding без фахової підготовки	ШІ-асистована інженерія
Розуміння коду	Часто відсутнє, прийняття результату без аналізу	Кожне рішення перевірене і зрозуміле
Налагодження	Реактивне: помилку повертають ШІ для виправлення	Аналітичне: рев'ю, тести, статичний аналіз
Придатність	Прототипи, MVP	Прототипи та виробничі системи
Основні ризики	Вразливості, технічний борг, закидання проєкту	Витрати часу на аудит, втрата навичок

Складено автором на основі [1, 2].

Висновки. Систематизовано можливості ШІ-асистентів і агентів, які охоплюють усі етапи життєвого циклу розробки ПЗ, та розглянуто чотири найновіші моделі (GPT-6 Astra, Claude Opus 5.5, DeepSeek V4.1, Kimi K3), показники яких наведено розробниками й не підлягають прямому порівнянню. Оновлено оцінку ризиків за даними 2026 року: частка

вразливого коду знизилась із 44 до 38%, проте агенти створюють робочий і безпечний код не більш ніж у 25% завдань. Виокремлено п’ять груп ризиків (вразливості коду, галюцинації пакетів, технічний борг, втрата навичок, автономність агентів) і відповідні заходи зниження. Встановлено, що vibe coding доцільний для прототипів, тоді як у виробничих системах ефективною є ШІ-асистована інженерія з обов’язковою верифікацією. Перспективи програмістів пов’язані зі зміщенням ролі до системної архітектури та контролю якості.

Використання інструментів штучного інтелекту. ChatGPT (GPT-6) використано для граматичної та стилістичної перевірки тексту тез та формулювання висновків роботи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Придибайло, О. Б., Придибайло, Р. В., Яскевич, В. О., Яскевич, Ю. В. (2026). Еволюція парадигм програмування та феномен vibe coding: оцінка можливостей, ризиків та перспектив досліджень. *Зв’язок,* 3, 59-68. https://doi.org/10.31673/2412-9070.2026.318119

[2] Опірський, І., Мельничук, М. (2026). Інтеграція штучного інтелекту в життєвий цикл розробки програмного забезпечення (SDLC). *Кібербезпека: освіта, наука, техніка,* 4(32), 376-404. https://doi.org/10.28925/2663-4023.2026.32.1193

[3] Taran, A., Sotnik, S. (2026). AI as a developer tool: GitHub Copilot and other artificial intelligence assistants. *Intelligent civil safety technologies and robotic systems for emergency and rescue operations 2026 (ICSTRO-2026): Proceedings of the I-st All-Ukrainian conference,* 67-71. NURE. https://openarchive.nure.ua/handle/document/33708

[4] OpenAI. (2026). *GPT-6 Astra: нове покоління інтелекту*. https://openai.com/uk-UA/index/gpt-6-astra/

[5] Anthropic. (2026). *Introducing Claude Opus 5.5.* https://www.anthropic.com/claude-opus-5-5

[6] CNN Business. (2026). *China’s AI upstart DeepSeek drops new model. Will it make waves like last year?* https://edition.cnn.com/2026/04/24/tech/chinas-ai-deepseek-v4-intl-hnk

[7] TechNode. (2026, 19 жовтня). *DeepSeek releases V4.1 Flash-1017 with stronger coding and agent capabilities.* https://technode.com/2026/10/19/deepseek-releases-v4-1-flash-1017-with-stronger-coding-and-agent-capabilities/

[8] Reuters. (2026). *China's Moonshot unveils world's largest open AI model, closing in on US rivals.* https://www.reuters.com/world/china/chinas-moonshot-unveils-worlds-largest-open-ai-model-closing-us-rivals-2026-07-17/

[9] CybrSecMedia. (2026). *AI-generated code still fails 38% of security tests despite gains, Veracode finds.* https://www.cybrsecmedia.com/ai-generated-code-still-fails-38-of-security-tests-despite-gains-veracode-finds/

[10] Tech Insider. (2026). *AI-generated code security: Top models pass 73%.* https://tech-insider.org/ai-generated-code-security-veracode-claude-opus-5-5-gpt-6-sol-73-percent-2026/

[11] Tianyi W. et al. Last Secure Code Benchmark: Is the code you vibe secure? (2026). *arXiv:2610.14766*. https://arxiv.org/pdf/2610.14766

[12] METR. (24 febr, 2026). *We are changing our developer productivity experiment design*. https://metr.substack.com/p/2026-02-24-uplift-update

[13] Zeff M. (Sept. 28, 2026). *OpenAI Scraps Release of GPT-6.1 Astra Model Over Safety Concerns.* https://www.wsj.com/tech/ai/openai-chatgpt-model-release-cancel-safety-5a2f9f42

Danilov Ye. V.

AI assistants and agents in software development: opportunities, risks and prospects

Abstract. *The opportunities, risks and prospects of using AI assistants and agents in software development are analyzed using modern models GPT-6 Astra, Claude Opus 5.5, DeepSeek V4.1 and Kimi K3 as examples. Vibe coding without professional training is compared with AI-assisted engineering performed by a qualified programmer. The main groups of risks (code vulnerabilities, package hallucinations, technical debt, skill loss, agent autonomy) are identified and mitigation measures are summarized. It is substantiated that the prospects for programmers lie in the shift of their role from writing code to architecture and verification of results.*

Keywords: *artificial intelligence; large language models; AI agents; vibe coding; software development.*