УДК 004.896 **DOI:** 10.31673/2412-9070.2025.050756
**Я. І. КОРНАГА,** доктор техн. наук, професор;
ORCID: 0000-0001-9768-2615
**А. В. ОЛЕКСІЙ,** аспірант,
ORCID: 0009-0005-2872-9850
Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського"

# ВПЛИВ ЗАСОБІВ ШТУЧНОГО ІНТЕЛЕКТУ НА АРХІТЕКТУРУ ТА ТЕСТУВАННЯ ВИСОКОНАВАНТАЖЕНИХ ІНФОРМАЦІЙНИХ СИСТЕМ

***У сучасному світі інформаційні системи (ІС) відіграють ключову роль у функціонуванні майже всіх сфер діяльності: від фінансів і телекомунікацій до охорони здоров'я та державного управління. Із зростанням обсягів даних, кількості користувачів та швидкості обміну інформацією значно зростає навантаження на обчислювальні ресурси ІС. Це породжує нові виклики щодо забезпечення їхньої надійності, масштабованості та стабільності — особливо в умовах високонавантажених середовищ.***

***Забезпечення сталого функціонування таких систем потребує не лише класичних інженерних рішень, а й новітніх підходів до проєктування, аналізу та тестування. В останні роки особливу увагу привертають інструменти з підтримкою штучного інтелекту (ШІ), які дедалі активніше інтегруються у процеси розробки, моніторингу та експлуатації ІС.***

***ШІ-інструменти пропонують можливості автоматичної генерації архітектурних рішень, предиктивного тестування, виявлення аномалій у режимі реального часу та оптимізації використання ресурсів. Проте впровадження таких рішень у високонавантажених системах супроводжується низкою ризиків та обмежень, пов'язаних з валідацією, безпекою та пояснюваністю моделей.***

***Метою даної статті є аналіз впливу інструментів з підтримкою ШІ на архітектурні рішення та методи тестування високонавантажених інформаційних систем.***

***Особливу увагу приділено як перевагам, так і потенційним викликам, які виникають під час інтеграції таких інструментів у процеси проєктування та підтримки систем зі складними і динамічними навантаженнями.***

**Ключові слова:** інформаційна система; штучний інтелект; архітектура; обчислювальна система; високонавантажена система; тестування; програмне забезпечення; оптимізація продуктивності.

## *Постановка проблеми*

Високонавантажені інформаційні системи широко застосовуються у фінансових технологіях, телекомунікаціях, електронній комерції та високопродуктивних обчисленнях (HPC). Їм притаманні жорсткі вимоги до масштабованості, відмовостійкості, високої доступності та сталого функціонування. Відповідно, критичним стає питання відмовостійкості та стабільної роботи таких систем. Високонавантажені інформаційні системи часто обробляють значні обсяги транзакцій у режимі реального часу, що потребує ретельної оптимізації архітектури та ефективного використання ресурсів. Будь-які збої або затримки можуть призвести до фінансових втрат, зниження довіри користувачів та репутаційних ризиків для організації. Тому проєктування та експлуатація таких систем нерозривно пов'язані з впровадженням механізмів моніторингу, балансування навантаження та автоматизованого відновлення після збоїв.

При проєктування таких систем виникає складний вибір архітектурних підходів — від монолітних до мікросервісних або serverless-рішень — а також реалізацію механізмів балансува-

ння навантаження, кешування, асинхронної обробки та масштабування. Ці рішення нерозривно пов'язані з компромісами між узгодженістю даних і доступністю згідно з теоремою CAP [2].

Тестування високонавантажених ІС стикається з обмеженнями у відтворенні реального навантаження, нестачею автентичних даних і труднощами виявлення помилок синхронізації (race conditions) [3]. Ці фактори суттєво ускладнюють забезпечення надійності таких систем.

У контексті забезпечення сталого функціонування високонавантажених ІС актуальним є системне вивчення можливостей застосування інструментів штучного інтелекту для підтримки архітектурних рішень та тестування. Враховуючи швидкий розвиток технологій машинного навчання, існує потреба у критичному аналізі їхніх сильних і слабких сторін, а також в оцінці потенційних ризиків впровадження у високонавантажене середовище.

**Аналіз останніх досліджень і публікацій**

Значна кількість сучасних досліджень сфокусована на інтеграції інструментів штучного інтелекту у процеси розробки та підтримки високонавантажених ІС. Особливо перспективним виявилося застосування нейромереж, зокрема автоенкодерів, для виявлення аномалій у поведінці обчислювальних систем. Так, у роботі [4] описано методику виявлення відхилень у HPC-середовищі на основі навчання моделі на «нормальному» режимі роботи суперкомп'ютера. У результаті точність виявлення аномалій сягнула 88–96 %.

Ще одним напрямом досліджень є застосування підкріплювального навчання (reinforcement learning, RL) для автоматичної генерації навантажувальних сценаріїв. У роботі [5] запропоновано агент RL, який не потребує доступу до вихідного коду або внутрішньої моделі системи, однак перевищує за ефективністю класичні підходи. Подібні методи застосовано й до автоматизованого тестування відеоігор, де RL-агенти виявляють проблемні ділянки (наприклад, падіння FPS) з вищою ефективністю порівняно з випадковими або ручними стратегіями [6].

Загалом, аналіз наукової літератури демонструє активне поширення ШІ-методів у сфері архітектурного аналізу, спостережуваності та навантажувального тестування, що свідчить про їхню ефективність і практичну цінність.

У статті буде розглянуто наступні задачі:

- проаналізувати вплив ШІ-інструментів на архітектуру високонавантажених ІС;
- оцінити їх роль у тестуванні таких систем;
- виокремити основні переваги та обмеження ШІ-технологій у контексті забезпечення сталого функціонування обчислювальних ресурсів.

***Основна частина***

Штучний інтелект (ШІ) поступово трансформує практики розробки програмного забезпечення, проте особливе значення його застосування має у сфері тестування високонавантажених інформаційних систем. Одним із найперспективніших напрямів інтеграції ШІ є автоматична генерація тестів, що забезпечує більш повне покриття функціоналу, скорочує час перевірки системи та дозволяє виявляти дефекти ще на ранніх етапах. Для складних архітектур, де кількість можливих сценаріїв взаємодії користувачів і підсистем є надзвичайно великою, класичні методи тестування часто виявляються недостатніми. Саме тут ШІ відкриває нові можливості, формуючи тести на основі аналізу попередніх помилок, логів роботи системи або навіть описів вимог до програмного забезпечення.

Моделі машинного навчання здатні не лише відтворювати типові шаблони поведінки користувачів, а й прогнозувати нетипові або критичні сценарії, які важко було б змоделювати вручну. Це значно підвищує надійність і сталість функціонування високонавантажених систем, адже такі тести охоплюють не лише очікувані шляхи використання, але й потенційні «вузькі місця» архітектури. Використання мовних моделей, зокрема Claude або GPT-подібних систем, дозволяє на основі опису API або специфікацій автоматично генерувати комплексні набори тестів, які адаптуються до змін у коді.

Важливо зазначити, що ШІ не забезпечує стовідсоткової точності: частина згенерованих тестів може виявитися надмірною або малоефективною, а їхнє покриття все ще потребує перевірки розробниками. Водночас людський фактор також не гарантує повноти — ручне тесту-

вання часто пропускає рідкісні або комбіновані сценарії. Тому оптимальною практикою стає поєднання обох підходів, де ШІ відповідає за масове створення тестів, а людина — за валідацію та пріоретизацію.

Завдяки цьому організації отримують новий рівень стабільності у своїх інформаційних системах: час на регресійне тестування зменшується, кількість виявлених критичних дефектів зростає, а процес релізів стає більш передбачуваним. У перспективі розвиток технологій у цій сфері дозволить реалізувати сценарії повністю автоматизованого тестування, де роль інженера-тестувальника буде зосереджена на контролі якості та налаштуванні правил для моделей.

Інтеграція інструментів штучного інтелекту у процеси підтримки функціонування обчислювальних ресурсів істотно трансформує підходи до побудови та супроводу високонавантажених інформаційних систем. З одного боку, технології ШІ дозволяють підвищити ефективність моніторингу, передбачення відмов, а також оптимізації ресурсного навантаження. Завдяки здатності обробляти великі обсяги операційних даних у реальному часі, моделі на основі машинного навчання можуть виявляти приховані патерни у зміні навантаження, що відкриває можливість динамічного масштабування інфраструктури до початку критичних піків активності. Наприклад, в умовах хмарного середовища системи, що інтегрують LSTM-мережі для прогнозування навантаження, демонструють підвищення ефективності розподілу обчислювальних потужностей до 17% у порівнянні з традиційними евристичними методами [7].

Крім того, великі мовні моделі, такі як GPT або Claude, дедалі частіше застосовуються для генерації, рефакторингу та верифікації коду на етапі розробки та тестування, що зменшує навантаження на DevOps-команди та дозволяє оперативно усувати помилки до моменту їхнього впливу на функціонування системи. Наприклад, середовище Cursor, яке реалізує підтримку контексту повного проєкту, дає змогу значно швидше виявляти порушення контрактів API або невідповідності у конфігураційних файлах, що у типових випадках унеможливлює падіння продуктивного середовища через людський фактор.

Однак одночасно із зазначеними перевагами впровадження ШІ-підходів породжує низку обмежень. По-перше, виникає проблема "чорної скриньки": більшість моделей, які приймають рішення на основі нейромережевої архітектури, залишаються непрозорими для кінцевих користувачів та інженерів підтримки. Це ускладнює трасування причин збоїв та їхнє локалізоване відтворення. По-друге, багато сучасних інструментів на основі LLM мають високе енергоспоживання та потребують значних обчислювальних ресурсів для інференсу, що суперечить вимогам сталості в інфраструктурах із обмеженою пропускною здатністю або бюджетом. По-третє, надмірна автоматизація процесів тестування або генерації коду може призводити до зниження якості через втрату контролю та зменшення глибини аналізу архітектурних рішень.

Ще одним важливим напрямком є застосування інструментів ШІ на етапах експлуатації, зокрема для виявлення аномалій у поведінці системи. Використання автоенкодерів або рекурентних нейронних мереж дозволяє у реальному часі ідентифікувати відхилення від типової поведінки вузлів HPC-кластерів. Це особливо цінне у великих інформаційних системах, де класичне моніторинг-засноване реагування має затримки і не дає змоги швидко локалізувати джерело деградації продуктивності.

Для систематизації переваг і обмежень ШІ-технологій у контексті сталого функціонування обчислювальних ресурсів доцільно навести наступну порівняльну таблицю.

**Переваги та обмеження ШІ-технологій**

| Характеристика | Переваги | Обмеження |
|---|---|---|
| Прогнозування навантаження | Висока точність, адаптивність, попередження піків у реальному часі | Необхідність навчання на великих обсягах специфічних даних |

| | | |
|---|---|---|
| Автоматизоване тестування та генерація коду | Зменшення витрат часу на супровід, вища швидкість релізів | Можливе створення нестійкого коду, обмеження якості покриття |
| Масштабованість і автономне керування ресурсами | Оптимізація витрат, зниження ризику перевантажень | Високі обчислювальні вимоги, складність у валідації ухвалених рішень |
| Інтерпретованість рішень | Часткова пояснюваність через логування та протоколи запитів до моделей | Неможливість відстежити причинно-наслідкові зв’язки у глибоких моделях (наприклад, трансформерах) |
| Сталість інфраструктури | Зниження ручного втручання, підтримка самоадаптивних середовищ | Підвищене енергоспоживання та вуглецевий слід при інференсі великих моделей |

Для оцінки практичного впливу інструментів з підтримкою ШІ на якість роботи високонавантажених інформаційних систем було проведено локальне дослідження у вигляді контрольованого експерименту. У ньому взяли участь три команди розробників, кожна з яких працювала у даному сценарії: без підтримки ШІ, з частковою підтримкою ШІ та з повним циклом роботи з ШІ.

Вибір моделі Claude 4 для проведення експерименту з інтеграції інструментів штучного інтелекту у процеси тестування був зумовлений поєднанням кількох факторів. По-перше, ця модель демонструє високу здатність до обробки великих текстових масивів, що є критичним у випадках, коли необхідно аналізувати журнали роботи системи, звіти тестувальників або складні конфігураційні файли. По-друге, Claude виявляє стійкість до контекстних зсувів і здатна коректно підтримувати довгі ланцюжки інструкцій, що забезпечує точність при багатокроковому тестуванні складних обчислювальних сценаріїв. По-третє, у порівнянні з іншими моделями, такими як GPT або LLaMA, Claude продемонструвала нижчу кількість хибнопозитивних результатів у завданнях пошуку дефектів, що безпосередньо впливає на ефективність всієї системи. У підсумку, саме ця модель забезпечила баланс між якістю аналізу та швидкістю обробки даних, що зробило її оптимальним вибором для дослідження.

Сценарій дослідження полягав у покроковому порівнянні різних підходів до покриття тестами REST API. На початковому етапі використовувалися виключно ручні методи: команда тестувальників створювала тест-кейси для ключових endpoint'ів API на основі документації. Це дозволило досягти базового рівня покриття, але вимагало значних витрат часу. На другому етапі було впроваджено часткову інтеграцію Claude: модель автоматично генерувала додаткові тест-кейси для менш критичних endpoint'ів, а також пропонувала варіанти негативних сценаріїв. На третьому етапі відбулася повна інтеграція ШІ, коли генерація усіх тестів — як позитивних, так і негативних — виконувалася Claude, а команда лише здійснювала валідацію та мінімальні правки. Це дало можливість суттєво збільшити кількість перевірених endpoint'ів та скоротити час, необхідний на створення повного набору тестів.

Основним критерієм оцінки було середнє значення часу, витраченого на вирішення типових завдань у межах моделювання навантаження та супутньої оптимізації сервісів. Результати наведено нижче.

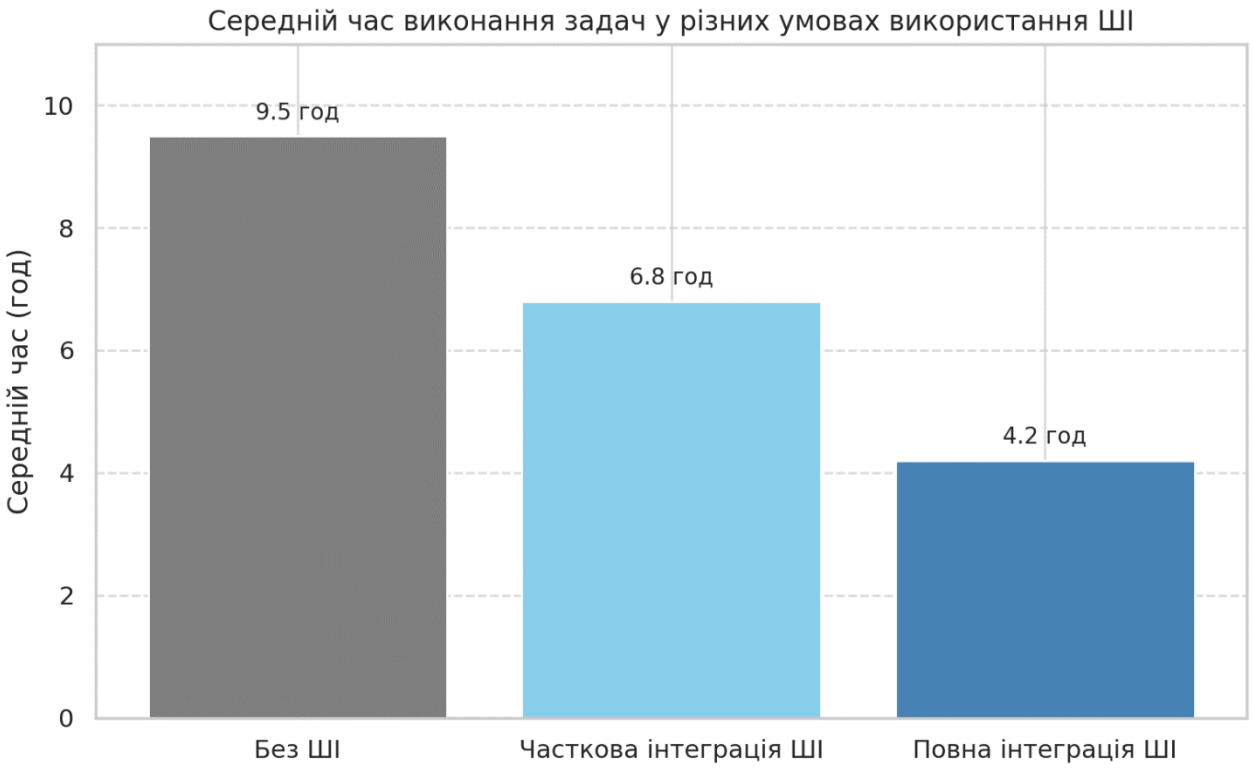


**Рис.1. Результати дослідження**

Як видно з графіка, використання інструментів з підтримкою ШІ дозволило значно скоротити час, необхідний для виконання завдань, пов'язаних із підтримкою продуктивності систем. Найбільший ефект спостерігався у групі з повною інтеграцією - час виконання знизився більш ніж вдвічі порівняно з базовим середовищем без ШІ. Це підтверджує практичну цінність AI-рішень навіть у короткострокових сценаріях впровадження. У сценарії, де застосовувалися виключно ручні методи, середній час завершення повного циклу тестування склав близько 9,5 годин. При цьому рівень виявлення дефектів становив лише 72 %. Такий показник пояснюється обмеженою здатністю тестувальників охопити всі можливі сценарії та необхідністю значного ручного аналізу отриманих результатів. Перехід до часткової інтеграції ШІ, коли інтелектуальні моделі використовувалися для автоматичної генерації тестових сценаріїв та попереднього аналізу логів, дозволив знизити середній час тестування до 5.6 годин, що майже удвічі швидше за ручний підхід. Водночас рівень виявлення дефектів зріс до 86 %. Це пов'язано з тим, що алгоритми машинного навчання здатні моделювати нетипові сценарії використання системи та виявляти приховані залежності, які залишаються поза увагою під час традиційного тестування. Повна інтеграція інструментів з підтримкою ШІ забезпечила ще більш відчутний ефект: середній час завершення тестування скоротився до 3,4 годин, а рівень виявлення дефектів піднявся до 91 %. Водночас важливо зазначити, що навіть у цьому сценарії залишилася похибка, оскільки жодна модель не забезпечує абсолютної точності. Наприклад, моделі на основі трансформерів, використані для аналізу логів та генерації тестових випадків, у деяких випадках генерували надлишкові або непридатні сценарії. Це знижувало ефективність і призводило до хибнопозитивних результатів. Проте подібні похибки можна порівняти із помилками, яких допускається людина під час традиційного аналізу, і в сумі інтеграція ШІ все ж демонструє значну перевагу.

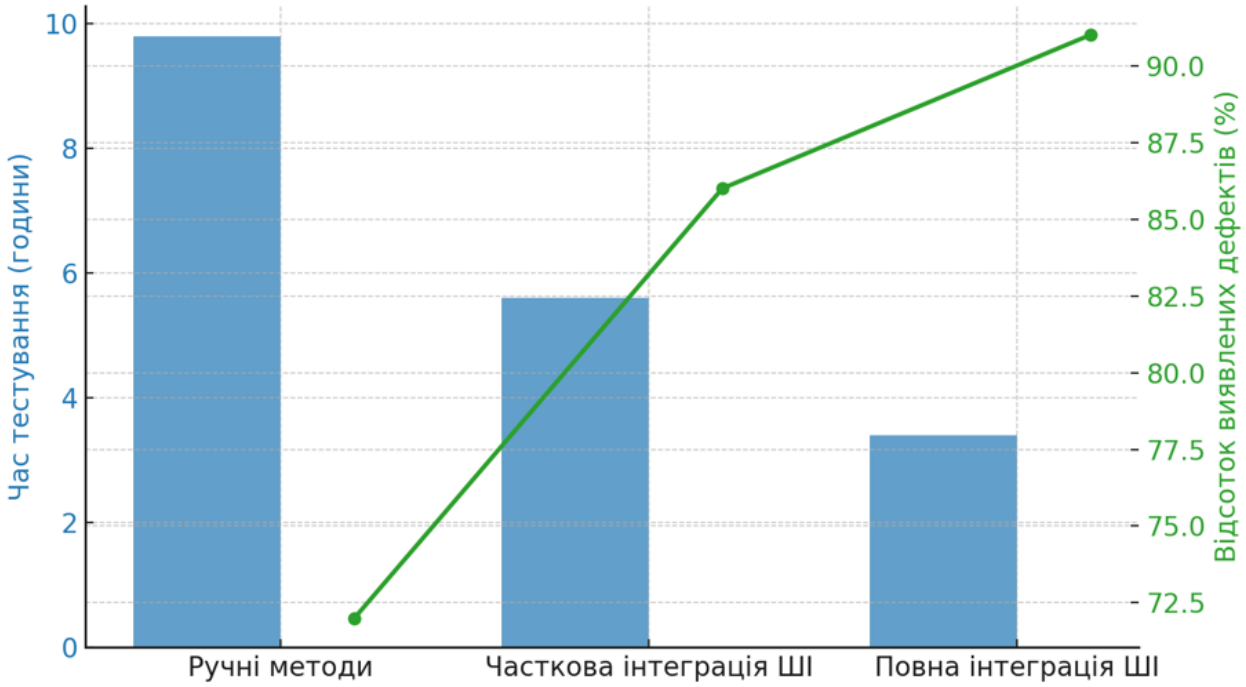


**Рис. 2. Порівняння підходів до тестування: ручні методи та інтеграція ШІ**

Варто також відзначити, що попри високу ефективність третього сценарію, частина роботи команди все ще полягала у вирішенні та переписуванні «галюцинацій» — тестів, які модель генерувала некоректно або з логічними неточностями. Ці похибки могли проявлятися у вигляді надлишкових або непотрібних сценаріїв, некоректно сформованих очікуваних результатів, а інколи — навіть у суперечності з документацією API. Розробникам доводилося уважно перевіряти кожен згенерований тест, аналізувати його відповідність реальній поведінці ендпоінтів і, при необхідності, переписувати або видаляти некоректні кейси.

Цей процес вимагав від команди не лише технічних навичок у роботі з API та тестовими фреймворками, а й глибокого розуміння логіки бізнес-процесів. Перевірка та корекція «галюцинацій» передбачає визначення чи є тест результатом реальної взаємодії системи, чи він виник через статистичну неточність моделі. Іноді навіть незначна похибка у згенерованому сценарії могла призвести до хибнопозитивних результатів або до пропуску критичних дефектів, тому роль розробника залишалася важливою, навіть при повній інтеграції AI.

Водночас варто підкреслити, що саме завдяки такій комбінованій роботі — автоматичній генерації та ручній корекції — вдалося досягти високого рівня покриття та значного скорочення часу тестування. Процес виправлення «галюцинацій» став важливим етапом навчання команди, оскільки дозволив виявити типові помилки моделей, зрозуміти їхні обмеження та оптимізувати стратегії тестування. Це демонструє, що ефективна інтеграція AI у процеси QA потребує не лише технологічного підходу, а й постійної експертної участі, яка забезпечує надійність та достовірність отриманих результатів.

### *Висновки*

Отримані результати демонструють чітку тенденцію: із зростанням рівня інтеграції інструментів з підтримкою ШІ середній час виконання типових задач суттєво зменшується. експериментальні дані підтверджують, що впровадження інструментів з підтримкою штучного інтелекту у процеси тестування високонавантажених інформаційних систем забезпечує істотне зменшення витрат часу та підвищення рівня надійності. Однак результати одночасно свідчать, що ШІ має розглядатися не як повна заміна людської експертизи, а як потужний інструмент для її підсилення. Саме комбінація автоматизованих моделей та професійної оцінки тестувальників здатна забезпечити найбільш стале та надійне функціонування обчислювальних ресурсів у динамічних умовах високих навантажень.

Експеримент підтвердив, що інтеграція інструментів з підтримкою ШІ суттєво підвищує ефективність тестування високонавантажених систем. Використання моделей для генерації сценаріїв та аналізу логів дозволило скоротити час перевірки більш, ніж удвічі та підвищити рівень виявлення дефектів з 72 % до 91 %. Тестування стало не лише точнішим, але й значно швидшим, що особливо важливо для підтримки продуктивності в умовах динамічних змін навантаження. Попри наявність окремих хибнопозитивних результатів, загальна продуктивність і точність процесу значно перевищили показники ручного тестування, що підтверджує практичну доцільність впровадження таких рішень у промислових умовах.

Загалом, результати підтверджують доцільність поступового впровадження ШІ в інженерні практики — особливо на етапах, де важливими є швидкість, повторюваність та економія часу фахівців. Водночас, при повній автоматизації зростає потреба у валідації та відповідному тестуванні вихідних рішень, що формує нові виклики у сфері архітектури та контролю якості.

***Список літератури***

*1. A. Kwan, J. Wong, H.-A. Jacobsen and V. Muthusamy, "HyScale: Hybrid and Network Scaling of Dockerized Microservices in Cloud Data Centres," 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 2019, pp. 80-90, doi: 10.1109/ICDCS.2019.00017*

*2. Bhagwan, R., Savage, S., Voelker, G.M. (2003). Understanding Availability. In: Kaashoek, M.F., Stoica, I. (eds) Peer-to-Peer Systems II. IPTPS 2003. Lecture Notes in Computer Science, vol 2735. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-45172-3_24*

*3. What are race conditions?: Some issues and formalizations. ACM, 1992. URL: https://dl.acm.org/doi/abs/10.1145/130616.130623*

*4. Andrea Borghesi, Andrea Bartolini, Michele Lombardi, Michela Milano ma Luca Benini, "Anomaly Detection using Autoencoders in High Performance Computing Systems," arXiv preprint, 2018. arXiv:1811.05269. https://arxiv.org/abs/1811.05269*

*5. Golrokh Hamidi, "Reinforcement Learning Assisted Load Test Generation for E-Commerce Applications", Master's thesis, Mälardalen University, 2020. https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1431385*

*6. R. Tufano, P. Tonella, M. Busjaeger, L. Pollock, "Using Reinforcement Learning for Load Testing of Video Games," Proceedings of the 44th International Conference on Software Engineering, ICSE 2022. arXiv:2201.06865. https://arxiv.org/abs/2201.06865*

*7. Chen, X., & Li, Y. (2023). AI-based workload prediction in cloud computing: A survey. Future Generation Computer Systems, 144, 143–157.*

*Y. Kornaga, A. Oleksii*

**THE IMPACT OF AI-ENABLED TOOLS ON THE ARCHITECTURE AND TESTING OF HIGH-LOAD INFORMATION SYSTEMS**

*In the modern world, information systems (IS) play a key role in the functioning of nearly all areas of activity — from finance and telecommunications to healthcare and public administration. As data volumes, user numbers, and information exchange speeds continue to grow, the load on IS computing resources increases significantly. This generates new challenges in ensuring their reliability, scalability, and stability — particularly in high-load environments. Ensuring the sustainable operation of such systems requires not only classical engineering solutions but also modern approaches to design, analysis, and testing. In recent years, AI-enabled tools have drawn increasing attention, as they are becoming more deeply integrated into the development, monitoring, and operation of IS. AI tools offer capabilities such as automated architectural decision generation, predictive testing, real-time anomaly detection, and resource usage optimization. However, the implementation of such solutions in high-load systems comes with a range of risks and limitations related to model validation, security, and explainability. The goal of this article is to analyze the impact of AI-enabled tools on architectural decisions and testing methods for high-load information systems. Special attention is given to both the advantages and potential challenges that arise during the integration of such tools into the design and maintenance processes of systems with complex and dynamic workloads. In this*

*context, AI-enabled tools are increasingly viewed as promising instruments for enhancing the efficiency and reliability of high-load information systems. These tools can support automated generation of test scenarios, predictive modeling of system behavior under varying workloads, and early detection of potential bottlenecks or failures. Moreover, AI-driven monitoring can adaptively allocate resources, helping to maintain optimal performance even under sudden spikes in demand. Despite these advantages, the adoption of AI in critical IS environments raises important concerns. Model accuracy, interpretability, and robustness under unforeseen conditions remain significant challenges, while security and compliance considerations impose additional constraints on deployment. Therefore, the integration of AI tools requires careful validation, iterative testing, and alignment with existing engineering practices to ensure that they genuinely contribute to the stability and resilience of high-load systems.*

***Keywords***: information system; artificial intelligence; architecture; computing system; high-load system; testing; software; performance optimization.