Деордиця О.О.
аспірант,
Державний вищий навчальний заклад
«Приазовський державний технічний університет»

DOI: https://doi.org/10.36059/978-966-397-602-0-25

ТРАНСФОРМАЦІЯ ПРОЦЕСІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПІД ВПЛИВОМ ТЕХНОЛОГІЙ ШТУЧНОГО ІНТЕЛЕКТУ

Динаміка розвитку сучасних інформаційних технологій обумовлює необхідність переосмислення класичних підходів до інженерії програмного забезпечення. У цьому контексті штучний інтелект виступає не лише як технологічний інструмент, але і як системоутворюючий фактор, що визначає нову логіку побудови, функціонування та еволюції програмних систем. Трансформація процесів розробки ПЗ під впливом ШІ має комплексний характер і охоплює як технічні, так і організаційно-методологічні аспекти, що дозволяє говорити про формування нової інженерної парадигми – AI-driven software engineering [2, с. 238].

На рівні теоретико-методологічних засад відбувається зміщення від алгоритмічної детермінованості до стохастично-адаптивних моделей, у яких поведінка системи визначається не лише закладеними правилами, але й навчанням на даних. Це зумовлює появу гібридних систем, у яких традиційні програмні компоненти інтегруються з моделями машинного навчання, що мають ймовірнісну природу. Відповідно, змінюються підходи до верифікації, тестування та забезпечення якості, оскільки класичні формальні методи не завжди є релевантними для оцінювання поведінки таких систем.

Етап інженерії вимог зазнає трансформації у напрямі впровадження когнітивних технологій, які забезпечують автоматизовану інтерпретацію природної мови та формалізацію знань. Застосування методів обробки природної мови (NLP) дає змогу здійснювати не лише синтаксичний, але й глибинний семантичний аналіз вимог, виявляти приховані залежності,

суперечності та неповноту специфікацій. Важливим доповненням є реалізація механізмів трасування вимог, що забезпечують автоматизоване встановлення зв’язків між вимогами, архітектурними рішеннями та їх реалізацією в програмному коді. Це, своєю чергою, сприяє підвищенню узгодженості, прозорості та керованості складних програмних проєктів [4, с. 148–149].

У сфері архітектурного проєктування спостерігається перехід до концепції self-adaptive systems, де архітектура здатна змінюватися під впливом зовнішніх факторів. Інтелектуальні системи аналізують телеметричні дані, історію експлуатації та контекст використання системи, що дозволяє здійснювати динамічну реконфігурацію компонентів. Крім того, застосування ШІ сприяє автоматизації вибору архітектурних патернів, оцінюванню технічного боргу та прогнозуванню довгострокових наслідків прийнятих рішень.

Процес програмування зазнає найбільш виражених трансформацій. Генеративні моделі забезпечують синтез програмного коду, документації та тестів на основі високорівневих описів задач. Це призводить до переходу від ручного кодування до концепції «програмування намірами» (intent-based programming), де ключову роль відіграє формалізація задачі, а не її реалізація на низькому рівні. У результаті змінюється когнітивна структура діяльності розробника, що поступово переходить від виконавця до архітектора та куратора інтелектуальних систем [3].

Важливим напрямом трансформації є автоматизація управління знаннями у процесі розробки. Інтелектуальні системи здатні акумулювати, структурувати та повторно використовувати знання, отримані в ході реалізації попередніх проєктів. Це сприяє формуванню ефекту «організаційної пам’яті», що підвищує ефективність прийняття рішень та зменшує залежність від індивідуального досвіду окремих розробників [1, с. 198].

Тестування та забезпечення якості ПЗ переходять до моделі інтелектуального контролю якості. Використання ШІ дозволяє реалізувати адаптивні стратегії тестування, що базуються на аналізі ризиків, поведінкових патернів користувачів та структурних характеристик коду. Генерація тестів здійснюється

з урахуванням контексту використання системи, що підвищує їхню релевантність. Крім того, застосування методів машинного навчання для аналізу журналів подій дозволяє здійснювати ранню діагностику дефектів.

У межах DevOps-практик інтеграція ШІ формує концепцію AIOps, яка передбачає автоматизацію процесів моніторингу, аналізу інцидентів та управління інфраструктурою. Інтелектуальні системи здатні здійснювати кореляцію подій, виявляти причинно-наслідкові зв'язки між інцидентами та пропонувати оптимальні сценарії реагування. Це дозволяє підвищити стійкість систем та зменшити час простою [3].

Разом з тим, трансформація процесів розробки супроводжується низкою складних викликів. Одним із ключових є проблема верифікації та довіри до результатів, згенерованих ШІ. Стохастичний характер моделей ускладнює забезпечення детермінованості та відтворюваності результатів. Крім того, актуалізуються питання безпеки, оскільки згенерований код може містити приховані вразливості або небажані залежності.

Суттєвим є також етичний та правовий аспект використання ШІ у розробці ПЗ. Питання авторського права на згенерований код, відповідальності за помилки та прозорості алгоритмів набувають особливої актуальності. У цьому контексті виникає потреба у розробці нормативно-правових рамок, що регулюють використання ШІ в інженерній діяльності.

Соціотехнічна трансформація проявляється у зміні структури компетенцій фахівців. Виникає потреба у формуванні міждисциплінарних знань, що поєднують інженерію програмного забезпечення, аналіз даних та теорію машинного навчання. Крім того, важливим стає розвиток критичного мислення та здатності до інтерпретації результатів, отриманих за допомогою ШІ [2, с. 241].

Перспективним напрямом подальших досліджень є розробка формальних моделей інтеграції ШІ у життєвий цикл ПЗ, а також створення стандартів оцінювання якості інтелектуалізованих систем розробки. Особливої уваги потребує проблема пояснюваності (explainability) моделей, що є критично важливою для забезпечення довіри до систем, які використовуються у розробці ПЗ.

Отже, трансформація процесів розробки програмного забезпечення під впливом технологій штучного інтелекту має системний характер і охоплює всі етапи життєвого циклу ПЗ. Відбувається перехід від детермінованих підходів до адаптивних, дано-орієнтованих моделей, у яких ключову роль відіграють механізми машинного навчання та автоматизованого прийняття рішень.

Інтеграція когнітивних і генеративних технологій сприяє підвищенню ефективності розробки, оптимізації процесів проєктування, програмування та тестування, а також зміні ролі інженера, який дедалі більше виконує функції контролю, аналізу та валідації результатів, згенерованих ШІ.

Водночас актуалізуються проблеми забезпечення якості, безпеки, інтерпретованості та відтворюваності результатів, що зумовлює необхідність розробки нових методів верифікації та нормативних підходів до використання ШІ.

Таким чином, подальший розвиток інженерії програмного забезпечення пов’язаний із формуванням парадигми AI-driven software engineering, що передбачає інтеграцію людського та штучного інтелекту в межах єдиного інженерного процесу.

Література:

1. Ковалишин О., Чухрай Л., Заплатинський Н. Вплив використання генеративного штучного інтелекту на продуктивність розробників програмних продуктів. *Вісник Львівського національного університету природокористування. Сер.: Агроінженерні дослідження*. 2025. №28. С. 196–201. DOI: https://doi.org/10.31734/agroengineering2024.28.196

2. Кравчук О. Штучний інтелект у програмуванні: як ШІ змінює підхід до розробки та автоматизації коду. *Вісник Хмельницького національного університету*. 2024. №6. Т. 2. С. 238–242. DOI: https://doi.org/10.31891/2307–5732–2024–345–6–36

3. Alenezi M., Akour M. AI-Driven Innovations in Software Engineering: A Review of Current Practices and Future Directions. *Applied Sciences*. 2025. №15(3). P. 1344. DOI: https://doi.org/10.3390/app15031344

4. Cheng H., Husen J. H., Lu Y. et al. Generative AI for Requirements Engineering: A Systematic Literature Review. Software: *Practice and Experience.* 2026. Vol. 56. No. 2. Pp. 141–170. DOI: https://doi.org/10.1002/spe.70029.