Валентина Бойко

(Біла Церква, Україна)

СУЧАСНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

(Програмне забезпечення)

ВИКОРИСТАННЯ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ОПТИМІЗАЦІЇ ПРОЦЕСУ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Штучний інтелект стрімко змінює процес розробки програмного забезпечення, автоматизуючи рутинні завдання, підвищуючи продуктивність розробників і покращуючи якість коду. Використання ШІ дозволяє скорочувати час на розробку, дає можливість оптимізувати тестування, знаходити помилки та навіть генерувати код.

Автоматизація процесів програмування важлива з ряду причин:

- *Прискорення розробки.* Автоматизація рутинних завдань, таких як написання шаблонного коду або форматування, дозволяє розробникам зосередитися на складних і творчих аспектах роботи. Інструменти на основі ШІ (GitHub Copilot, ChatGPT) допомагають генерувати код, скорочуючи час на розробку.
- *Підвищення якості коду.* Автоматичне тестування та аналіз коду знижують ризик появи помилок на ранніх етапах розробки. Статичний аналіз коду допомагає знаходити проблеми ще до його виконання.
- *Оптимізація ресурсів.* Автоматизація дозволяє ефективніше розподіляти ресурси, зменшуючи потребу у великих командах тестувальників і розробників. Інструменти CI/CD (Continuous Integration / Continuous Deployment) забезпечують швидке та стабільне розгортання додатків.
- *Мінімізація людського фактора.* Автоматизовані процеси виключають типові людські помилки, які можуть спричинити збої в системі. Використання AI для пошуку вразливостей підвищує безпеку ПЗ.

▪ *Гнучкість та адаптивність.* Автоматизовані процеси легко масштабуються та адаптуються під нові технології та зміни у вимогах проєкту. Можливе швидке оновлення та виправлення помилок без зупинки сервісу (hotfixing).

Завдяки автоматизації команди розробників можуть створювати більш складні, безпечні та продуктивні програмні рішення, витрачаючи менше часу та зусиль. Це дозволяє компаніям швидше адаптуватися до ринку та залишатися конкурентоспроможними.

Отже, автоматизація в розробці програмного забезпечення дозволяє значно підвищити ефективність, знизити кількість помилок і прискорити випуск продуктів. У сучасному світі, де конкуренція між компаніями зростає, а складність програмного забезпечення постійно збільшується, автоматизація стає ключовим фактором успіху.

Сьогодні існує багато інструментів, які використовують AI, щоб підвищити продуктивність програмістів. Розглянемо деякі з них.

Інструменти для генерації та автодоповнення коду:

✓ *GitHub Copilot* – це інструмент, створений GitHub та OpenAI, який пропонує цілі рядки коду та повноцінні функції. Цей інструмент використовує машинне навчання і величезний масив коду, доступний в репозиторіях GitHub;

✓ *Tabnine* – AI-асистент для розробників, який аналізує код і пропонує контекстно-залежні підказки, пропонує рекомендації щодо завершення коду на основі deep learning для різних мов програмування та платформ. Він також легко інтегрується з різними IDE;

✓ *CodeWhisperer (AWS)* – інструмент від Amazon, який аналізує код і допомагає розробникам писати чистий та ефективний код. Це інструмент для спільного програмування, який полегшує співпрацю розробників — він дозволяє програмувати в парі та відстежувати зміни коду в режимі реального часу.

Інструменти для автоматичного тестування [2, с.30]:

✓ *DeepCode* – AI-інструмент для аналізу коду, який знаходить помилки та вразливості;

✓ *Diffblue Cover* – використовує AI для написання тестів у Java, що допомагає скоротити час на тестування;

✓ *Testim* – платформа для автоматизації тестування UI, яка застосовує машинне навчання для пошуку помилок;

✓ *Codium AI* – помічник для написання тестів, який пропонує тести прямо під час кодування, щоб допомогти вам уникнути прикрих помилок та багів. Надає інтелектуальні ідеї тестів під час написання коду і навіть може написати їх за вас.

Інструменти для аналізу та безпеки коду:

✓ *SonarQube* – аналізує якість коду та виявляє вразливості;

✓ *Snyk* – платформа для пошуку та усунення вразливостей у залежностях та контейнерах;

✓ *Checkmarx* – інструмент для автоматичного сканування коду на наявність загроз безпеці.

Інструменти для управління проєктами та DevOps:

✓ *Jira + AI* – інтелектуальні рекомендації та автоматизація завдань у системі управління проєктами.

✓ *AIOps (Artificial Intelligence for IT Operations)* – AI-підхід до оптимізації DevOps-процесів.

✓ *CircleCI + AI* – автоматизує процеси CI/CD, забезпечуючи швидше та безпечніше розгортання.

Інструменти для генерації документації та підтримки:

✓ *Mintlify* – AI-інструмент для автоматичної генерації документації з коду.

✓ *ChatGPT* – допомагає розробникам отримувати швидкі відповіді на питання щодо коду, алгоритмів та концепцій.

Попри значні переваги, використання штучного інтелекту в розробці програмного забезпечення супроводжується певними викликами. Впровадження AI потребує не лише технічних змін, а й адаптації процесів, командної культури та управління ризиками.

Розглянемо деякі ризики і виклики [1, с.20].

Залежність від ШІ та якість згенерованого коду

- AI-асистенти, такі як GitHub Copilot чи ChatGPT, можуть генерувати код, але він не завжди є оптимальним або безпечним.
- Розробники повинні вручну перевіряти код, тестувати його та вдосконалювати, що додає додаткові кроки до робочого процесу.

Етичні та правові питання

- Код, створений AI, може містити уривки з ліцензованого або захищеного авторським правом програмного забезпечення.
- Виникають питання відповідальності: хто несе відповідальність за помилки або вразливості в коді, згенерованому ШІ?

Безпека та конфіденційність

- Використання AI-інструментів може спричинити витік конфіденційних даних, якщо код передається на зовнішні сервери для обробки.
- ШІ може ненавмисно інтегрувати у код уразливості, які складно виявити на ранніх етапах розробки.

Витрати на впровадження та інтеграцію

- Високоякісні AI-інструменти можуть бути дорогими, а їхнє впровадження потребує навчання персоналу.
- Не всі компанії готові інвестувати у дорогі рішення та перебудовувати свої процеси.

Опір з боку розробників

- Деякі програмісти можуть сприймати AI як загрозу своїй професії, що може викликати небажання адаптувати нові технології.
- AI змінює звичні підходи до написання коду, що вимагає зміни мислення та навичок.

Потреба в контролі та людському втручанні

- AI не може повністю замінити людську логіку та креативність. Він може допомогти, але все ще потребує коригування з боку людини.
- Важливо розуміти обмеження AI, щоб уникати сліпого використання його рішень без перевірки.

Попри всі виклики, впровадження ШІ у розробку ПЗ є неминучим. Важливо правильно балансувати між автоматизацією та контролем з боку розробників, враховуючи безпеку, етичні аспекти та якість роботи AI. Компанії, які адаптуються до нових реалій і навчаться ефективно використовувати AI, отримають значну конкурентну перевагу.

Розглянемо основні тенденції майбутнього ШІ в розробці ПЗ.

Штучний інтелект як співавтор коду

- AI буде не тільки підказувати кодові фрагменти, а й створювати цілі функціональні модулі та архітектурні рішення.
- Розвиток Low-Code та No-Code платформ дозволить навіть людям без глибоких знань у програмуванні створювати додатки за допомогою AI.

Автоматизація тестування та відлагодження

- AI зможе не лише знаходити баги, а й автоматично їх виправляти, аналізуючи контекст коду та пропонуючи оптимальні рішення.
- Очікується вдосконалення self-healing систем, які самостійно усуватимуть критичні помилки без втручання людини.

Оптимізація процесів DevOps та CI/CD

- AI допоможе прогнозувати проблеми з розгортанням програм, підбирати оптимальні конфігурації серверів та автоматизувати процеси оновлення програмного забезпечення.
- Використання машинного навчання для аналізу продуктивності та виявлення потенційних вузьких місць у додатках.

Покращення безпеки та кіберзахисту

- AI буде активно використовуватися для виявлення вразливостей, аналізу атак та захисту програмного коду від злому.
- Автоматичне шифрування, адаптивні системи безпеки та поведінковий аналіз користувачів допоможуть запобігати кібератакам.

Еволюція ролі програміста

- AI поступово візьме на себе рутинні завдання, а програмісти більше зосередяться на проєктуванні архітектури, взаємодії з бізнесом та вирішенні творчих задач.
- Очікується зростання попиту на спеціалістів із AI-розробки та інженерів, які працюватимуть над вдосконаленням алгоритмів машинного навчання.

Майбутнє ШІ у розробці ПЗ відкриває величезні можливості. Проте, щоб ефективно використовувати ці інновації, розробникам доведеться адаптуватися до нової реальності, навчатися взаємодіяти з AI та використовувати його потенціал для створення ще більш інноваційних рішень.

У майбутньому роль програміста зміниться, оскільки AI-інструменти стануть ще потужнішими та більш інтегрованими в робочі процеси [3, с.96]. Програмісти більше зосередяться на творчих та стратегічних аспектах розробки, а рутинні операції поступово перейдуть під контроль AI.

Розглянемо деякі прогнози про роль програміста в майбутньому:

- ШІ автоматизує написання стандартного коду, і розробники більше займатимуться проєктуванням архітектури та оптимізацією алгоритмів.
- Замість ручного написання коду програмісти будуть працювати з AI-асистентами, які допомагатимуть створювати програмні рішення швидше.
- Завдяки таким платформам, як Bubble, OutSystems, Mendix, навіть люди без технічної освіти зможуть створювати додатки, використовуючи візуальні інтерфейси.
- Програмісти стануть більше консультантами та архітекторами програмних рішень, допомагаючи бізнесу інтегрувати технології.
- Важливими стануть знання AI та машинного навчання, оскільки програмісти повинні будуть не тільки використовувати AI-інструменти, а й розуміти, як вони працюють.
- Навички аналітичного мислення, креативності та системного підходу стануть більш затребуваними, ніж механічне написання коду.

- Колаборація з AI – програмісти працюватимуть у парі з AI-асистентами, які будуть допомагати знаходити помилки, пропонувати оптимізації та навіть прогнозувати можливі проблеми.
- Автоматизація тестування – AI повністю візьме на себе контроль за тестами, CI/CD та моніторингом продуктивності.

З’являться і нові ролі у сфері розробки:

- *Prompt Engineer* (Інженер запитів) – фахівець, який знає, як ефективно працювати з AI для отримання найкращих результатів.
- *AI Developer/Trainer* – розробник, який не просто пише код, а навчає та вдосконалює AI-моделі.
- *AI Product Manager* – спеціаліст, який займається інтеграцією AI-рішень у бізнес-процеси.

Програмування зміниться, але не зникне. AI не замінить програмістів повністю, а зробить їх роботу ефективнішою. У майбутньому програмісти більше зосередяться на проєктуванні, аналітиці та інтеграції AI, а рутинні завдання перейдуть під контроль розумних алгоритмів. Ті, хто адаптується до нових реалій, матимуть великі можливості для кар'єрного зростання.

Штучний інтелект відкриває значні можливості для розвитку програмного забезпечення. Проте, щоб його використання було успішним та безпечним, необхідно ретельно планувати, аналізувати та мінімізувати ризики. Головне – знайти баланс між автоматизацією та людським фактором, а також відповідально підходити до використання ШІ-технологій. Це запорука сталого розвитку та конкурентоспроможності ІТ-індустрії в майбутньому.

Література:

1. Бойко В. Програмування за допомогою систем генеративного штучного інтелекту: ризики та виклики / В. Бойко, М. Василенко, В. Слатвінська. – Київ: МАУП, 2023. – Інформаційні технології та суспільство, №2 (8). – С. 18-26.

2. Савчук Т., Приймак Н. Використання методів штучного інтелекту для прогнозування часу для виправлення дефектів розробки програмного забезпечення. – Вінниця: Інтернет-Освіта-Наука, 2016. – С. 30-31.
3. Сівіцкий В. BITO – штучний інтелект для допомоги розробникам програмного коду. – Збірник наукових праць Харківського національного економічного університету імені Семена Кузнеця, 2024. – Т. 15. – С. 95-97.

Бойко Валентина Миколаївна
09108, м. Біла Церква
вул. Зенітного полку, буд. 99, кв. 115
тел.: 0976261869
e-mail: vmboiko18@gmail.com
ВСП «Технолого-економічний фаховий коледж
Білоцерківського НАУ», викладач