УДК 004.4.056

СУЧАСНІ ПРАКТИКИ CI/CD ТА ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ЇХ РЕАЛІЗАЦІЇ З УРАХУВАННЯМ ВИМОГ БЕЗПЕКИ

Саблєв А. І.
e-mail: artem.sabliev@nure.ua
Науковий керівник– к.т.н., ст. викл., Обрізан В. І.
Харківський національний університет радіоелектроніки, каф. АПОТ
м. Харків, Україна

This work is devoted to modern CI/CD practices and tools used in software development. Continuous integration, delivery and deployment have become important parts of the software lifecycle. Their use helps accelerate release processes, improve code quality and reduce the amount of routine manual work. Special attention is paid to the automation of building, testing and deployment. Security is also an essential part of modern CI/CD pipelines, especially in projects with high requirements for reliability and software quality. For this reason, pipelines often include code analysis and vulnerability scanning tools. The combination of these practices and tools makes software development and delivery more efficient.

У сучасних умовах розроблення програмного забезпечення підходи CI/CD посідають важливе місце в організації процесів інтеграції, тестування, доставки та розгортання змін. Їх використання дає змогу скоротити тривалість циклу розроблення, зменшити кількість ручних операцій, підвищити повторюваність процесів і забезпечити більш своєчасне виявлення помилок.

Безперервна інтеграція передбачає регулярне внесення змін до спільного репозиторію з подальшим автоматичним запуском процедур складання та тестування. Такий підхід сприяє ранньому виявленню проблем сумісності, знижує ризик накопичення дефектів і спрощує командну роботу над програмним продуктом. Безперервна доставка забезпечує підтримання програмного забезпечення у стані готовності до випуску, а безперервне розгортання передбачає автоматизоване впровадження змін після проходження визначених контрольних етапів перевірки. У сукупності ці практики формують основу сучасного підходу до автоматизації життєвого циклу програмного забезпечення.

Реалізація CI/CD передбачає використання спеціалізованих інструментів, що забезпечують виконання окремих етапів конвеєра. До найбільш поширених платформ належать Jenkins, GitLab CI/CD, GitHub Actions та Azure DevOps. Jenkins залишається одним із найбільш

функціонально гнучких рішень завдяки розвинутим можливостям налаштування та великій кількості плагінів. Це робить його доцільним для складних середовищ, у яких необхідна інтеграція з великою кількістю зовнішніх сервісів та гнучке налаштування сценаріїв автоматизації.

GitLab CI/CD є прикладом інтегрованої платформи, у межах якої механізми керування репозиторієм, запуску конвеєрів, зберігання артефактів і частина додаткових сервісів поєднані в єдиному середовищі. Такий підхід спрощує забезпечує цілісність процесів. GitHub Actions, у свою чергу, орієнтований на тісну взаємодію з репозиторіями GitHub і використовується для автоматизації типових сценаріїв контейнеризації, тестування, публікації пакетів і розгортання.

Суттєве значення в сучасних CI/CD-процесах мають інструменти контейнеризації та оркестрації. Docker забезпечує стандартизацію середовища виконання застосунків і дозволяє ізолювати програмне забезпечення разом із його залежностями, що підвищує переносимість між різними етапами розробки та експлуатації. Kubernetes використовується для автоматизованого керування контейнеризованими застосунками, масштабування сервісів і підтримання їхньої працездатності в кластерному середовищі. У поєднанні з Helm такі засоби дають змогу формалізувати та повторно використовувати сценарії розгортання.

Окрему роль у структурі CI/CD-конвеєра відіграють засоби зберігання артефактів та залежностей. Для цього широко застосовуються Nexus Repository та JFrog Artifactory. Їх використання дозволяє централізовано зберігати пакети, контейнери та інші результати складання, що є важливим для контролю версій, повторного використання артефактів і забезпечення відтворюваності процесів.

Поряд із засобами автоматизації складання та розгортання важливого значення набувають інструменти контролю якості та безпеки. Одним із найпоширеніших рішень для статичного аналізу коду є SonarQube. Цей інструмент дозволяє виявляти дефекти, дублювання, порушення правил кодування, а також окремі конструкції, що можуть створювати потенційні вразливості. Його інтеграція в CI/CD-конвеєр дає змогу використовувати результати аналізу як критерій допуску збірки до наступних етапів.

У сучасній практиці все більшого поширення набуває підхід DevSecOps, відповідно до якого механізми безпеки інтегруються безпосередньо в конвеєр розроблення. У межах такого підходу перевіряється не лише працездатність програмного забезпечення, а й

ризики, пов’язані з його кодом, залежностями, контейнерними образами та конфігураціями інфраструктури. Для аналізу сторонніх бібліотек часто застосовується OWASP Dependency-Check, який дозволяє виявляти відомі вразливості у використовуваних залежностях. Це є особливо важливим з огляду на значну кількість зовнішніх компонентів, що використовуються в сучасних програмних проєктах.

Для сканування контейнерних образів, файлових систем і репозиторіїв на наявність відомих уразливостей та помилкових конфігурацій доцільно використовувати Trivy. Його інтеграція до конвеєра дозволяє здійснювати перевірку образів до їх публікації або розгортання. Для виявлення секретів у репозиторіях застосовуються Gitleaks, який дозволяють знаходити випадково додані паролі, токени доступу та ключі.

Таким чином, сучасний CI/CD-конвеєр істотно відрізняється від підходу, що використовувався 10-15 років тому. Якщо раніше автоматизація здебільшого обмежувалася збиранням проєкту та запуском базових тестів, то сьогодні конвеєр охоплює повний цикл доставки програмного забезпечення: від перевірки коду й залежностей до контейнеризації, аналізу конфігурацій, контролю безпеки та автоматизованого розгортання. Сучасний CI/CD орієнтований не лише на швидкість випуску змін, а й на відтворюваність процесів, контроль якості, трасованість і раннє виявлення ризиків, що втілює найкращі практики розробки програмного забезпечення.

Список використаних джерел:

1. CI/CD Pipelines Evolution and Restructuring: A Qualitative and Quantitative Study / F. Zampetti та ін. 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), м. Luxembourg, 27 верес. – 1 жовт. 2021 р. URL: https://doi.org/10.1109/icsme52107.2021.00048 (дата звернення: 11.03.2026).
2. Sharma M. Review of the benefits of DAST (dynamic application security testing) versus SAST // International Journal of Management and Engineering Research 2021. № 1. С. 05-08.
3. Pratama M. R., Sulistiyo Kusumo D. Implementation of Continuous Integration and Continuous Delivery (CI/CD) on Automatic Performance Testing. 9th International Conference on Information and Communication Technology (ICoICT), м. Yogyakarta, Indonesia, 3–5 серп. 2021 р. URL: https://doi.org/10.1109/icoict52021.2021.9527496 (дата звернення: 11.03.2026).