УДК 004.41

В. Семенюк, В. Сенківський, В. Чичук, Б. Хоміцький, О. Кучма
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ОГЛЯД ІНСТРУМЕНТІВ БЕЗПЕРЕРВНОЇ ІНТЕГРАЦІЇ В СУЧАСНИХ ПРОЄКТАХ З РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

V. Semeniuk, V. Senkivskyi, V. Chychuk, B. Khomitskyi, O. Kuchma

OVERVIEW OF CONTINUOUS INTEGRATION TOOLS IN MODERN SOFTWARE DEVELOPMENT PROJECTS

Безперервна інтеграція (Continuous Integration (CI)) – це концепція написання коду, яка змушує розробників програмного забезпечення вносити зміни та постійно переглядати код у сховищах контролю версій. Однією з головних переваг безперервної інтеграції є те, що можна легко та швидко виявляти помилки. Оскільки кожна внесена зміна зазвичай невелика, ви можете швидко визначити конкретну зміну, яка спричинила дефект. Останнім часом CI стала таким стандартним протоколом і набором базових елементів для розробки програмного забезпечення [1].

Гнучкі методології створили постійний безперервний цикл між клієнтами та командами розробників програмного забезпечення в режимі реального часу. Дотримуючись цієї ідеї, DevOps будується на принципі циклу зворотного зв'язку в реальному часі у процесі розробки SDLC, зменшуючи ризики зупинки роботи розробників через велику кількість дефектів, забезпечення якості (QA).

У минулому команда розробників могла працювати самостійно протягом тривалого часу та об'єднувати свої зміни в програмний код лише після завершення основної гілки. Це ускладнює злиття коду, а також дозволяє накопичувати помилки без виправлення протягом тривалого часу [2]. Такі фактори не сприяли швидкому наданню клієнтам оновлень.

Розглянемо основні інструменти неперервної інтеграції.

Jenkins – одне з найпоширеніших програмних безкоштовних рішень CI з відкритим кодом. Це хмарний сервіс CI, написаний на Java, який працює на веб-сервері. Тисячі користувачів у всьому світі використовують Jenkins, оскільки він дає змогу швидко створювати та виконувати автоматизовані тести. Основні властивості:

- Безкоштовне.
- Налаштування робочого процесу.
- Велика кількість плагінів.
- Легка інсталяція для основних операційних.
- Орієнтовано на розробників.
- Добре відома та авторитетна компанія.

TeamCity – це універсальне бізнес-рішення CI, яке можна використовувати безкоштовно при кількості проєктів до 100. Можна запускати паралельні конструкції за допомогою TeamCity одночасно, використовувати мітки тощо. TeamCity легко встановити завдяки зручному інтерфейсу [3]. Основні властивості:

- Безкоштовно до 100 проєктів.
- Три потоки з трьома агентами збирання коду одночасно.
- Можна імпортувати вихідний код з двох різних VCS в одній компіляції.
- Можливість замінити тестувальників програмними агентами.
- Дозволяє перевіряти зміни без фіксації VCS.

Bamboo є продуктом від Atlassian і має швидкий і ефективний графічний інтерфейс користувача. Цей інструмент популярний серед розробників, які використовують інші інструменти Atlass [4]. Bamboo дозволяє створювати та об'єднувати нові гілки після автоматичного тестування. Основні властивості:

- Ефективна інтеграція з іншими інструментами Atlassian.
- Хороший спосіб надання сповіщень.
- Просте управління масштабуванням CI компанії.
- Автоматизація тестування.
- Автоматичне розпізнавання окремих збірок.

Buddy – це інструмент автоматизації DevOps для постійної інтеграції та розгортання. Цей інструмент був розроблений для роботи з проєктами на основі коду репозиторію Bitbucket і GitHub [5]. Buddy – це бізнес-інструмент із простим і легким у використанні інтерфейсом і оптимізованим дизайном. Служба, орієнтована на клієнта, підтримується 24 години на добу без вихідних і може бути встановлена на пристрій у версії клієнта. Основні властивості:

- Інтуїтивний інтерфейс користувача.
- Інтуїтивно зрозумілий дизайн процесу розгортання.
- Підтримка докерів.
- Доступні попередні налаштування та підказки.
- Забезпечує складну автоматизацію та потребує фундаментальних знань.
- Можливість модифікувати розроблений код.
- Клонування, змінна та універсальна автоматизація приміток.

GitLab CI – це продукт із відкритим кодом і безкоштовний інструмент постійної інтеграції. API GitLab має високий ступінь масштабування, його легко встановити та налаштувати для проєктів, розміщених на GitLab. Окрім тестування та створення проєктів, GitLab CI може використовуватись, де процес розробки потребує вдосконалення. Розробники GitLab вибирають індивідуальний GitLab CI, не замислюючись, оскільки безперервна інтеграція проєкту досягається автоматично [6].

Варто звернути увагу на такі властивості цього продукту:

- Підтримка Docker.
- Конфігурація сервера швидкої збірки.
- Працює на кількох машинах одночасно.
- Сильна інтеграція продукту можлива за допомогою API.
- Опція захисту конфіденційних даних проєкту.

Коли компанія практикує CI, уся її робота регулярно інтегрується в основну вітку коду (розпізнається як магістральна або головна). Дослідження показали, що робота компанії покращується, коли розробники мають можливість об'єднувати свій програмний код з основною віткою. Перед фактичним злиттям проводиться серія автоматизованих перевірок, щоб переконатися, чи не виникають помилки регресії [7]. Якщо ці програмні продукти містять дефекти, команда зазвичай зупиняється, щоб виправити помилки.

Усі подальші процеси повинні використовувати пакети, створені збіркою CI. Ці побудови мають бути чисельними та відтворюваними. Принаймні раз на день потрібно успішно запускати процес складання проєкту з виконанням набору автоматичного тестування. Починають із написання багатьох тестів, які охоплюють пріоритетну з точки зору якості функціональність системи. Після цього перевіряють усі нові функції. Ці тести повинні бути проведені швидко для отримання оперативного зворотного зв'язку від розробників. Принаймні один раз на день тести повинні пройти успішно. Зрештою, розробники отримуватимуть інформацію щоденно, якщо тести пройдуть

успішно та код буде злитий з основною віткою. Система CI, яка виконує автоматичне тестування, також повинна візуалізувати статус команди. Не рекомендується використовувати повідомлення електронною поштою; багато людей ігнорують сповіщення електронною поштою або створюють фільтр, який приховує повідомлення. Системні сповіщення чату є кращим і популярнішим способом досягнути цього. Безперервна інтеграція часто включає додаткові дії, які також передбачають вищу продуктивність розробки програмного забезпечення.

Стиль побудови, зосереджений на основній вітці коду, де розробники будують невеликі ділянки та об'єднують свої завдання в окрему вітку принаймні щодня, а не на довготривалих вітках додатків. Для CI потрібне автоматизоване модульне тестування. Ці тести мають бути достатньо всебічними, щоб гарантувати належне функціонування програмного забезпечення. Тести також мають тривати кілька хвилин або менше. Якщо автоматизоване модульне тестування триває довше, розробники не хочуть запускати його часто. Якщо тести виконуються рідко, результати багатьох різних змін можуть ускладнити локалізацію помилок та відладку. Тести, які проводяться рідко, важко підтримувати.

Складно створити супроводжувані пакети модульних тестів. Хорошим вирішенням цієї проблеми є практика розробки, керованої тестуванням (TDD). TDD надає багато переваг: одна полягає в тому, що розробники пишуть гнучкий, зручний для тестування код, який, як наслідок, зменшує витрати на обслуговування автоматизованих наборів тестів. Багато компаній не мають пакетів модульних тестів, які можна підтримувати, і все ще не практикують TDD.

Таким чином, CI гарантує безперервну роботу команди проєкту. Впровадження CI дає більшу швидкість розгортання, надійніші системи та якісніші програми. Перевага CI є значною. Останні дослідження підтверджують це твердження, допомагаючи підкреслити зв'язки між створенням, проєктуванням і впровадженням програмного забезпечення. Постійна інтеграція в проєкти допомагає знизити ризики організації роботи команди.

Література

1. Fowler, Martin, and Matthew Foemmel. "Continuous integration." (2006).

2. Hüttermann, Michael. "Introducing DevOps." DevOps for Developers. Berkeley, CA: Apress, 2012. 15-31.

3. Melymuka, Volodymyr. TeamCity 7 continuous integration essentials. Packt Publishing, 2012.

4. Brechner, Eric. Agile project management with Kanban. Pearson Education, 2015.

5. Vanbrabant, Bart, Thomas Delaet, and Wouter Joosen. "Authorizing and directing configuration updates in contemporary IT infrastructures." Proceedings of the 3rd ACM workshop on Assurable and usable security configuration. 2010.

6. Arefeen, Mohammed Shamsul, and Michael Schiller. "Continuous Integration Using Gitlab." Undergraduate Research in Natural and Clinical Science and Technology Journal 3 (2019): 1-6.

7. Senapathi, Mali, Jim Buchan, and Hady Osman. "DevOps capabilities, practices, and challenges: Insights from a case study." Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018. 2018.