Дідович О.М., студент,
Лєтошко А.І., аспірант,
Житомирський державний технологічний університет

РОЛЬ НЕПЕРЕРВНОЇ ІНТЕГРАЦІЇ (CONTINUOUS INTEGRATION) В ПРОЦЕСІ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Спеціалісти у сфері розробки програмного забезпечення (ПЗ) намагаються уникати великої кількості ручної роботи, такої як тестування, підготовка до розповсюдження продукту, розповсюдження продукту, інформування розробників, менеджерів та інженерів з забезпечення якості про наявність нової версії додатку.

Для досягнення даної мети використовується підхід до розробки, в основі якого лежить виконання регулярного автоматичного збирання проекту з використанням раніше створених сценаріїв.

Це дозволяє запобігати утворенню великої кількості помилок на різних етапах розробки програмного засобу та автоматизувати все те, що раніше виконувалося вручну.

Даний процес називається неперервною інтеграцією (англ. Continuous Integration, CI) та застосовується для контролювання ризиків, зменшення часових, грошових та інших витрат у процесі розробки ПЗ.

Однією з ключових переваг застосування такого підходу є можливість перевірки вихідного коду продукту за допомогою тестів після кожної фіксації змін, що дозволяє швидко виявляти помилки на будь-якому кроці розробки та зменшити витрати на їх усунення.

Для того, щоб використовувати неперервну інтеграцію, проект має задовольняти певним вимогам:

- необхідно використовувати систему контролю версій (СКВ), наприклад: Git, Mercurial, SVN, Perforce або іншу;
- потрібно забезпечити захищений канал доступу до вихідного коду проекту ззовні;
- необхідний віртуальний або реальний сервер зі встановленою системою автоматизації.

Схема роботи інтеграції може відрізнятися в залежності від вимог до проекту, але зазичай процес типової інтеграції складається з наступних кроків:

1. Запуск процесу інтеграції. Даний процес може виконуватися за розкладом, на вимогу користувача, або після певного тригеру (наприклад, після завантаження змін до СКВ).

2. Завантаження вихідного коду. Якщо дана операція є успішною, то виконується перехід до наступного кроку, а саме, виконання скриптів. У разі помилки усім користувачам, що містяться у списку сповіщення, буде надіслано відповідний лист. Даний лист містить у собі детальний опис (stack trace), що включає код помилки та дії, які необхідно виконати для її відтворення. Stack trace має вигляд, що зображений на рис. 1. Даний звіт може відправлятися після помилки на будь-якому етапі.

3. Виконання скриптів. Наприклад, підготування коду до розповсюдження: збирання коду, його оптимізація, перевірка на працездатність, виконання тестів, автоматичне завантаження даних на віддалені сервери та інше.

```
Coverage report generated for RSpec to /builds/gitlab-org/gitlab-
ce/coverage. 105370 / 201707 LOC (52.24%) covered.
/usr/local/bin/ruby -I/builds/gitlab-org/gitlab-ce/vendor/ruby/2.
1.0/gems/rspec-core-3.3.2/lib:/builds/gitlab-org/gitlab-ce/vendor
/ruby/2.1.0/gems/rspec-support-3.3.0/lib /builds/gitlab-org/gitla
b-ce/vendor/ruby/2.1.0/gems/rspec-core-3.3.2/exe/rspec --pattern
./spec/services/\*\*/\*_spec.rb failed

ERROR: Build failed: exit code 1
```

Рис. 1. *Частина stack trace*

4. Після успішного виконання усіх етапів користувачам відправляється лист, що містить попередньо сформований шаблон інформування про успішну інтеграцію.

На рис. 2 зображено блок-схему вищеописаного процесу інтеграції.

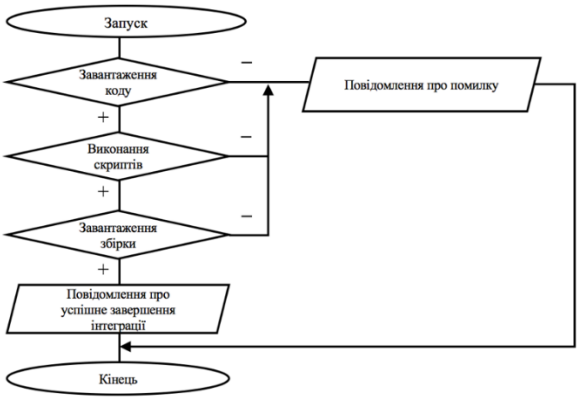

Рис. 2. *Блок-схема процесу інтеграції*

Неперервна інтеграція – це підхід до розробки програмного забезпечення при якому продукт може бути випущений у виробництво у будь-який момент часу. Для його впровадження використовується спеціальне програмне

забезпечення, базовим принципом роботи якого є моніторинг змін у репозиторії і запуск відповідних сценаріїв і задач.

На сьогоднішній день найпопулярнішими програмними системами у цій сфері є Travis CI та Jenkins. Обидва продукти безкоштовні, мають відкриту кодову базу, що дозволяє розширювати функціональність за допомогою власноруч розроблених модулів. За час їх існування набула розвитку спільнота користувачів, завдяки якій створена значна кількість плагінів та розширень, що дозволяють вирішити майже будь-яку проблему.

Перевагами Jenkins є гнучкість у конфігуруванні та підтримка більшості сучасних СКВ, але водночас широкі можливості даної системи ускладнюють процес налаштування.

На відміну від Jenkins, Travis тісно інтегрується лише з СКВ Git та має меншу кількість плагінів, але відзначається простотою настроювання.

При правильному використанні вищезазначених систем процес розповсюдження автоматизується, що повністю звільняє користувача від ручної роботи, зменшує кількість помилок на усіх етапах розробки, прискорює процес їх усунення та значно економить ресурси.