[3] Page Object model with Ruby and Selenium: https://www.selenium.dev/documentation/en/guidelines_and_recommendations/page_object_models/.

[4] Page Object model in Cucumber: https://automationpanda.com/2017/01/23/behavior-driven-python-getting-started-with-page-object-models/.

[5] Selenium official website: https://www.selenium.dev/.

[6] Selenium WebDriver with Ruby: https://www.selenium.dev/selenium/docs/api/rb/.

[7] WebDriver architecture and implementation details: https://www.selenium.dev/documentation/en/webdriver/driver_requirements/.

Олег Левченко, Ірина Борецька*

студент 6-го курсу, НЛТУ України, Львів, Україна. o.levtschen@gmail.com

* к.т.н., доцент кафедри КН, НЛТУ України, Львів, Україна.

https://orcid.org/0000-0002-6767-104X, boretska@nltu.edu.ua

Автоматизація процесів розгортання програмних сервісів за допомогою системи керування життєвим циклом Keptn

Анотація - Створено конвеєр, що дозволяє в автоматичному режимі виконати процес розгортання програмного сервісу послідовно у всіх запрограмованих оточеннях. Конвеєр використовує ворота контролю якості для оцінки якості розгортання у кожній фазі та приймає рішення про перехід до наступної, використовуючи цілі рівня обслуговування (SLO). Результати дослідження можуть бути використані для побудови систем управління процесами розгортання програмних сервісів та автоматизації операцій різних рівнів складності.

Ключові слова - конвеєр, розгортання сервісів, автоматизація операцій, ворота контролю якості, цілі рівня обслуговування, оркестрація.

Розвиток Інтернету та технологій сприяв удосконаленню програмних сервісів та способів їх розміщення на серверах. З розвитком віртуалізації ефективність використання фізичних серверів підвищилась завдяки запуску кількох віртуальних машин паралельно на одному сервері. Поява контейнеризації збі-

* науковий керівник

льшила ефективність, віртуалізуючи лише самі програмні сервіси. З розповсюдженням концепції мікросервісної архітектури кількість сервісів, які розгортаються, може налічувати тисячі та збільшуватись кратно кількості оточень, одночасно створюючи проблему контролю якості розгорнутих сервісів. Для максимізації ефективності та контролю якості постає необхідність в конвеєрах управління процесами розгортання сервісів та автоматизації операцій. Основні вимоги до таких конвеєрів включають: паралельне управління процесами багатьох сервісів у багатьох оточеннях, використання воріт контролю якості [1] відповідно до поставлених цілей рівня обслуговування (SLO) [2], автоматизація операцій пост-розгортання, незалежність від постачальників хмарних послуг та інструментів, декларативний спосіб написання коду.

Метою даного дослідження є вивчення сучасних підходів і методів для розгортання програмних сервісів, вибір необхідних технологій та інструментів, а також створення базового автоматизованого конвеєру розгортання програмного сервісу на базі системи керування життєвим циклом Keptn, що дозволить прискорити процес випуску якісного коду, виявляючи помилки та деградацію якості на ранніх етапах, та мінімізує негативний вплив на кінцевого користувача, не допустивши релізу неякісного коду, або ж виконавши автоматизовані операції для стабілізації сервісу.

Під час розроблення проекту використано широкий спектр сучасних технологій та інструментів: система керування життєвим циклом Keptn для виконання коду конвеєру та загальної оркестрації процесів, Docker, Kubernetes, Helm, Argo Rollouts для контейнеризації, Dynatrace для моніторингу та оцінки якості, Locust для тестування навантаження, YAML для написання коду конвеєру та конфігурацій SLI/SLO.

Для реалізації завдання обирано архітектуру розгортання DSP з такими середовищами: розробки (Dev), дзеркало оточення кінцевого користувача (Stage), оточення кінцевого користувача (Prod). Рішення про дозвіл на просування між фазами буде прийматись автоматизованими QG (рис. 1).

* науковий керівник

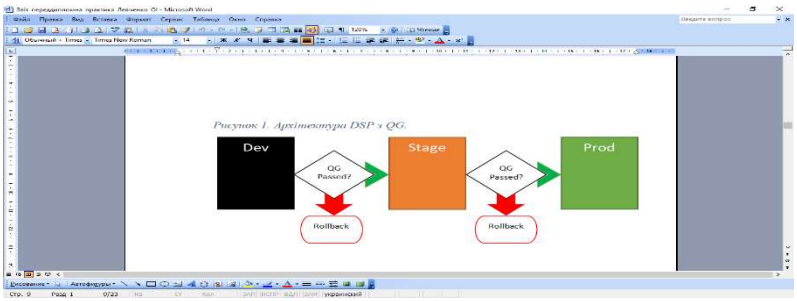

Рис. 1. Архітектура DSP з QG

Після розгортання в оточення останнього користувача необхідно впевнитись у якості роботи сервісу так якомога оперативніше відреагувати на можливі проблеми. Для цього використано функціонал автоматизованих операції та створено послідовність дій для самостійної стабілізації сервісу за допомогою масштабування (рис. 2).

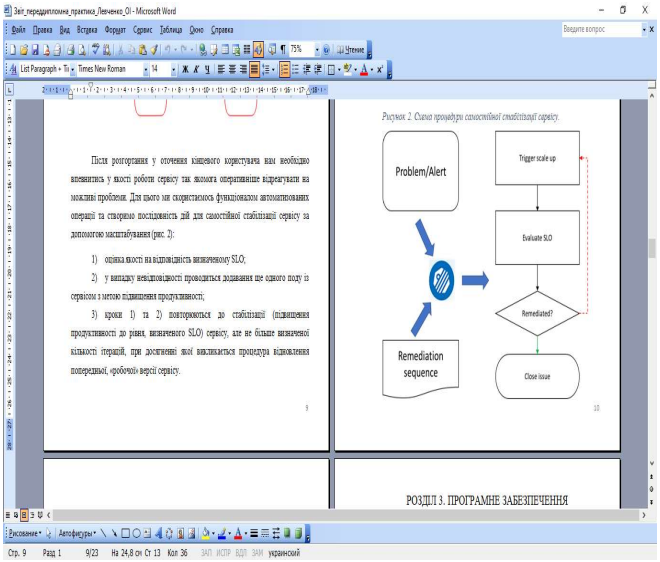

Рис. 2. Схема процедури самостійної стабілізації сервісу

Для створення проєкту в Keptn необхідно описати конфігурацію конвеєру у shipyard.yaml файлі. Конфігурація потребує перелічення оточень (stages), кожне з яких має включати щонайменше один ланцюжок задач (sequences), що, в свою чергу, має містити мінімум одну задачу (tasks).

Оскільки операції в різних оточеннях можуть відбуватись паралельно та відрізнятись, Keptn для збереження конфігурацій оточень використовує гілки репозиторію. Основна гілка master слугує для збереження глобальної конфігурації проєкту, тобто конвеєру та налаштувань Dynatrace, включно з SLI. Файли

* науковий керівник

конфігурацій решти оточень зберігаються у гілках “dev”, “stage”, “prod” відповідно.

Базова структура файлів для оточення виглядає так, як зображено на рис. 3. Директорія “helm” використовується для збереження конфігураційних файлів розгортання сервісу за допомогою helm. Директорія “job” призначена для зберігання конфігурації задач, які мають виконуватись в межах ланцюжків задач, а також автоматизованих задач пост-розгортання, наприклад, самостабілізації або відновлення попередньої версії. Файл визначення SLO має бути збережений у директорії «<service>». Додатково можуть бути присутні інші директорії, якщо вони необхідні для виконання задач конвеєру.

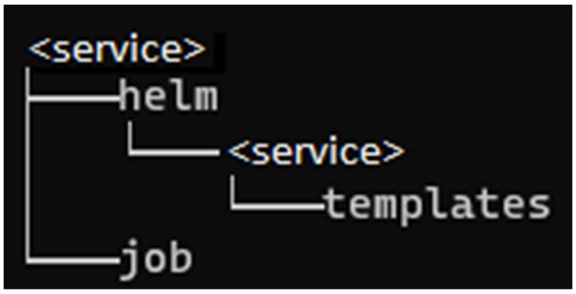

Рис. 3. Базова структура директорій оточення

Висновок. Результатом дослідження є реалізований конвеєр, що дозволяє в автоматичному режимі виконати процес розгортання програмного сервісу послідовно у всіх запрограмованих оточеннях, виконати оцінку якості розгортання у кожній фазі та приймаючи рішення про перехід до наступної. Завдяки незалежності від конкретного вендора у конвеєрі на базі системи керування життєвим циклом Keptn можуть бути використані різноманітні інструменти, що дозволяє будувати системи управління процесами розгортання програмних сервісів та автоматизації операцій різних рівнів складності.

Перелік посилань

[1] Quality Gates: What They Are and How to Use Them. [Електронний ресурс] – режим доступу до ресурсу: https://linearb.io/blog/quality-gates

[2] SRE Book. [Електронний ресурс] – режим доступу до ресурсу: https://sre.google/sre-book/service-level-objectives/

* науковий керівник