Issue 15, 2024

https://doi.org/10.23939/sisn2024.15.238

УДК 004.42

СИСТЕМА КЕРУВАННЯ ПРОЦЕСОМ БЕЗПЕРЕРВНОЇ ДОСТАВКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Андрій Берко[1], Іван Коблик[2]

Національний університет "Львівська політехніка",
кафедра інформаційних систем та мереж, Львів, Україна
[1] E-mail: Andrii.Y.Berko@lpnu.ua, ORCID: 0000-0001-6756-5661
[2] E-mail: ivan.koblyk.mnsam.2022@lpnu.ua ORCID: 0009-0000-9413-4533

Сьогодні напрям DevOps є надзвичайно важливою складовою у виконанні IT-проектів різного масштабу. Це обумовлює потребу у виробленні новітніх прогресивних підходів, методів та технологій для забезпечення ефективної діяльності DevOps фахівців. Одним з таких популярних вирішень є неперервна інтеграція (Continuous Integration) і неперервна доставка (Continuous Delivery) програмного забезпечення. Такий підхід (CI/CD) дає змогу значно прискорити процеси розроблення, тестування та впровадження програмного забезпечення, підвищити якість результатів розроблення та забезпечити контрольованість та керованість цих процесів. Ключовим чинником у процесах CI/CD є використання засобів автоматизації роботи DevOps фахівців. У роботі описано систему керування процесом безперервної доставки програмного забезпечення розроблену на підставі сучасного досвіду застосування провідних компаній.

Ключові слова: DevOps; CI/CD; ChatOps; безперервна інтеграція; безперервна доставка.

Вступ

Для ефективного, успішного та конкурентноспроможного бізнесу в IT індустрії необхідно постійно пристосовуватися до нових вимог та трендів ринку, реагувати на потреби та бажання своїх клієнтів і мати змогу задовольняти їх у найкоротші терміни. У зазначених умовах гнучка розробка програмного забезпечення з якомога меншими фінансовими та часовими витратами, регулярні та стабільні випуски нових версій є ключовими складовими успішного продукту. Протягом останнього десятиліття спостерігається суттєве зростання ролі нової практики, яка дістала назву DevOps (акронім англ. development і operations). Це філософія, яка має на меті поєднання напрямів розробки (Development) та експлуатації (Operations) програмного забезпечення заради досягнення високої продуктивності, швидкості та надійності випусків програмного забезпечення. DevOps має велику кількість складових, інструментів та ключових принципів. Чільною засадою філософії DevOps є автоматизація, фундаментальним компонентом якої є CI/CD – поєднання безперервної інтеграції (англ. Continuous Integration) [18, 19] та безперервного розгортання або доставки (англ. Continuous Delivery) програмного забезпечення в процесі розробки [1–5]. Сьогодні на ринку присутня значна кількість програмних рішень, які надають можливості управління CI/CD процесами[6–8]. Усі вони відрізняються між собою функціоналом, принципом роботи, синтаксичними особливостями файлів конфігурації, та найголовніше – інтерфейсом користувача. З огляду на вищевказані особливості, актуальним є створення системи керування процесом безперервної доставки програмного забезпечення, яка дасть змогу фахів-
[2]цям DevOps легко та швидко управляти конкретним рішенням через зрозумілий інтерфейс користувача з різних пристроїв, включаючи смартфони.

Постановка проблеми

Поєднання технології чат-ботів та філософії DevOps вилилось у кардинально нове та інноваційне явище під назвою ChatOps. Сам термін вперше згадується 2013 року та був запропонований компанією GitHub на презентації їхнього нового продукту Hubot [9]. Відтоді ChatOps була практично і успішно прийнята багатьма організаціями як додатковий інструмент для фреймворку DevOps [10–13]. Головною ідеєю цієї практики є впровадження використання чат-ботів у процеси розробки та комунікації між членами команди. Беручи до уваги вже згадані можливості ботів та потреби в автоматизації процесів розробки, таких як CI/CD тощо, а також постійну робочу комунікацію між членами IT-команд, ця практика є надзвичайно перспективною. Із впровадженням ChatOps, автоматизація перетворюється на надзвичайно прозорий процес, оскільки всі члени команди залучені до комунікації та мають змогу в реальному часі керувати та відслідковувати усі процеси, які були виконані та запущені, за допомогою систем обміну миттєвими повідомленнями [14–15]. Простота використання також є однією із беззаперечних переваг ChatOps, адже чат-боти надають команди, які є інтуїтивно зрозумілими, оскільки люди можуть відправляти команди чат-ботам так само, як звичайні повідомлення будь-якому зі своїх колег.

Метою роботи є розроблення системи керування процесом безперервної доставки програмного забезпечення.

Аналіз останніх досліджень та публікацій

Тематика безперервної інтеграції/безперервної доставки програмного забезпечення, нажаль, є недостатньо висвітленою в наукових дослідженнях та наукових публікаціях. Незважаючи на важливість та актуальність ця проблема, значною мірою, залишається у практичній сфері. Сьогодні є значна кількість сервісів, які надають послуги безперервної інтеграції та доставки, та відповідних ChatOps сервісів, які дозволяють керувати ними за допомогою систем обміну миттєвими повідомленнями. У всіх цих рішень є свої особливості, такі як: архітектура, ціна, спектр послуг, інтерфейс користувача та месенджери, з якими вони інтегровані. Серед найпоширеніших CI/CD інструментів та відповідних ChatOps систем для керування ними можна виділити, зокрема, такі.

1. **Хмарний сервіс безперервної інтеграції та доставки Azure DevOps Services** та ChatOps система Azure DevOps керування ним. Azure DevOps Services є рішенням, розробленим корпорацією Microsoft. Сервіс покладається на хмарну платформу Microsoft Azure [16]. Користувач взаємодіє з системою через веб-сайт або за допомогою консольного доступу, вхід до системи надається за допомогою облікового запису Microsoft (рис. 1).

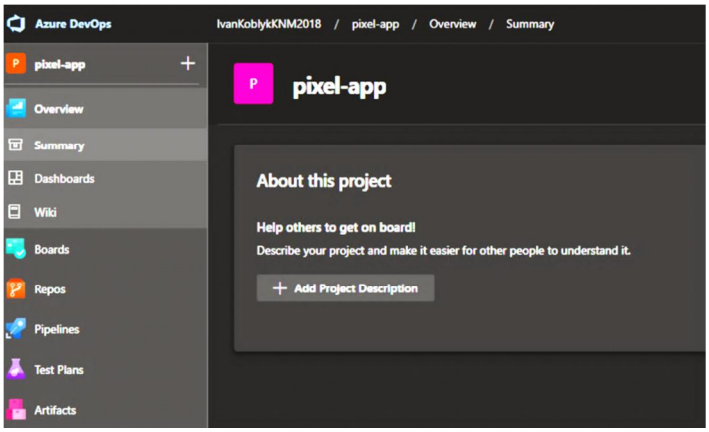

Рис. 1. Головна сторінка сервісу "Azure DevOps Services"

Сервіс з керування згаданим інструментом методами ChatOps має назву Azure DevOps. Він є розширенням платформи для командної роботи Microsoft Teams. Це розширення дуже легко підключається до командного чату та дозволяє отримувати сповіщення про події, які відбуваються в системі, такі як: завершення завдання конвеєру безперервної доставки чи інтеграції, додавання нових змін у репозиторій з кодом проєкту, створення нового завдання на спринт-дошках (рис. 2).

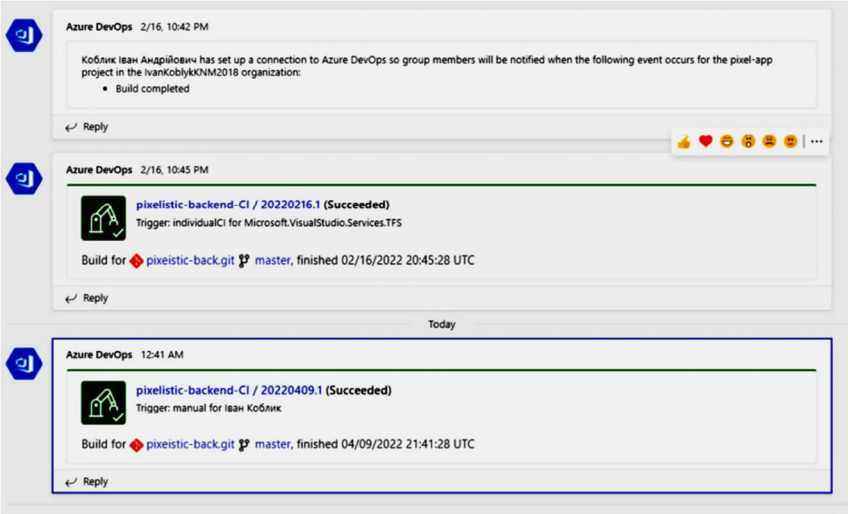

Рис. 2. Сповіщення від ChatOps системи "Azure DevOps" у каналі MS Teams

Перевагами даного сервісу є легкість підключення через облікові записи Microsoft, просте налаштування та користувацький інтерфейс, інтеграція з продуктами та сервісами Microsoft, можливості масштабування проєктів. Недоліками можна вважати доволі високі ціни на послуги обмежений функціонал, який дозволяє лише моніторити події, але не створювати їх безпосередньо з чату, наявність тільки хмарного варіанту платформи.

2. Сервіс безперервної доставки Heroku Pipelines. Цей сервіс є частиною хмарної платформи Heroku компанії Salesforce. Він надає послуги конвеєрів з безперервної доставки, які надзвичайно швидко та легко налаштовуються та не вимагають від користувачів глибоких технічних знань. Користувач сервісу має змогу створювати конвеєр безперервної доставки (CD pipeline), обрати репозиторій з кодом проекту на GitHub (рис. 3), вибрати додаткові налаштування, зокрема, послугу "eview apps", яка дозволяє автоматично запускати конвеєри безперервної доставки при створенні запиту на зміни у репозиторії проекту та розгортати версію програми на тимчасовому тестовому сервері, який самознищується за певний проміжок часу [17].

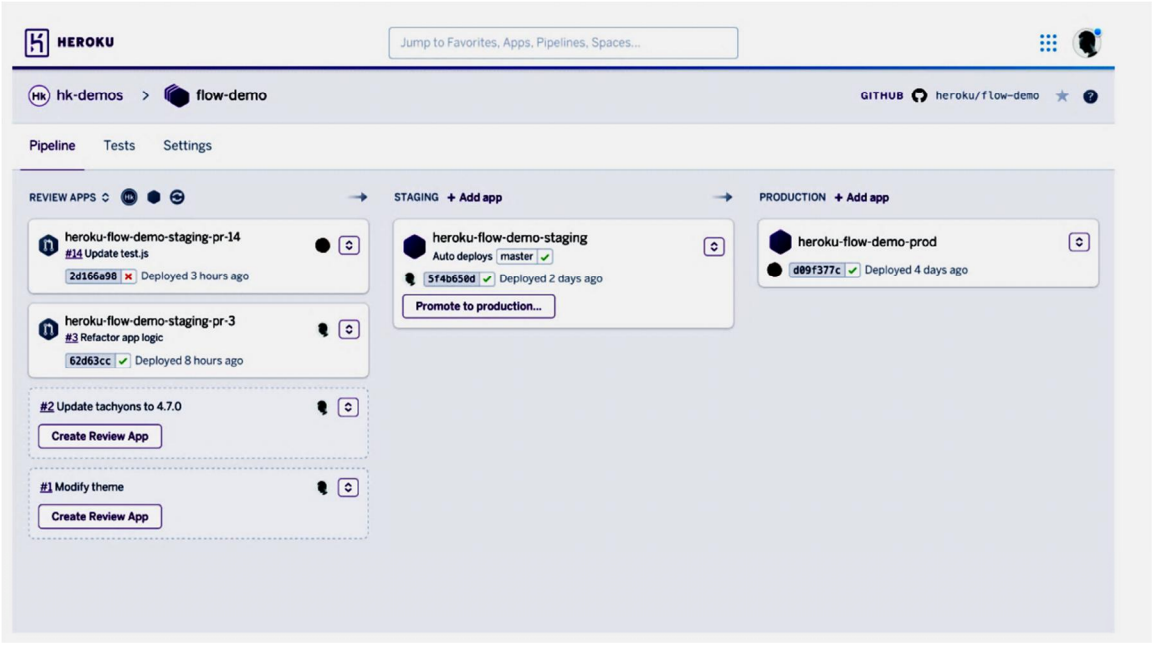

Рис. 3. Інтерфейс сервісу неперервної доставки ПЗ Heroku Pipelines

Heroku ChatOps – система керування конвеєрами Heroku pipelines, є додатком до корпоративного месенджеру Slack. Користувачі взаємодіють з системою через чат за допомогою текстових команд. Засіб дає можливості запуску конвеєра безперервної доставки безпосередньо з чату, вибравши певну гілку з репозиторію з кодом проекту та певне середовище для розгортання (рис. 4), перегляду списку конвеєрів, списку останніх випусків, а також можливість налаштування автоматичних сповіщень з Heroku напряму в чат.

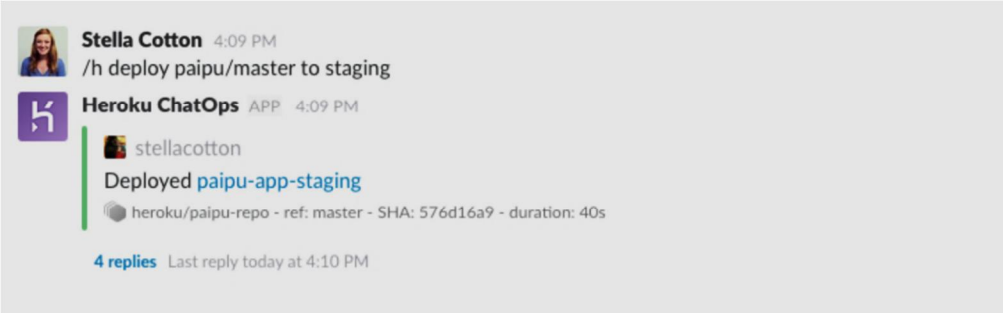

Рис. 4. Сповіщення від ChatOps системи Heroku ChatOps у каналі Slack

Перевагами даного сервісу є просте налаштування та користувацький інтерфейс, недоліками можна вважати доволі високі ціни на послуги та відсутність глибокого налаштування та контролю [17].

3. Інструмент безперервної доставки GitLab CI/CD. Цей інструмент є однією з додаткових послуг сервісу GitLab української компанії GitLab Inc, основним призначенням якого є надання розміщення віддалених репозиторіїв контролю версій. Оскільки програмний код проектів зберігається на одному й тому самому сервісі, що й конвеєри безперервної доставки, то інтеграція з кодом і підключення CI/CD є надзвичайно легким, швидким і безболісним процесом (рис. 5). Структура конвеєрів описується файлами з розширенням yaml, які є легкими для розуміння та експлуатації. GitLab CI/CD дає можливість користувачам використовувати для виконання процесів складання і розгортання артефактів як агентів, наданих сервісом безпосередньо, або створених та підтримуваних самими користувачами [16].

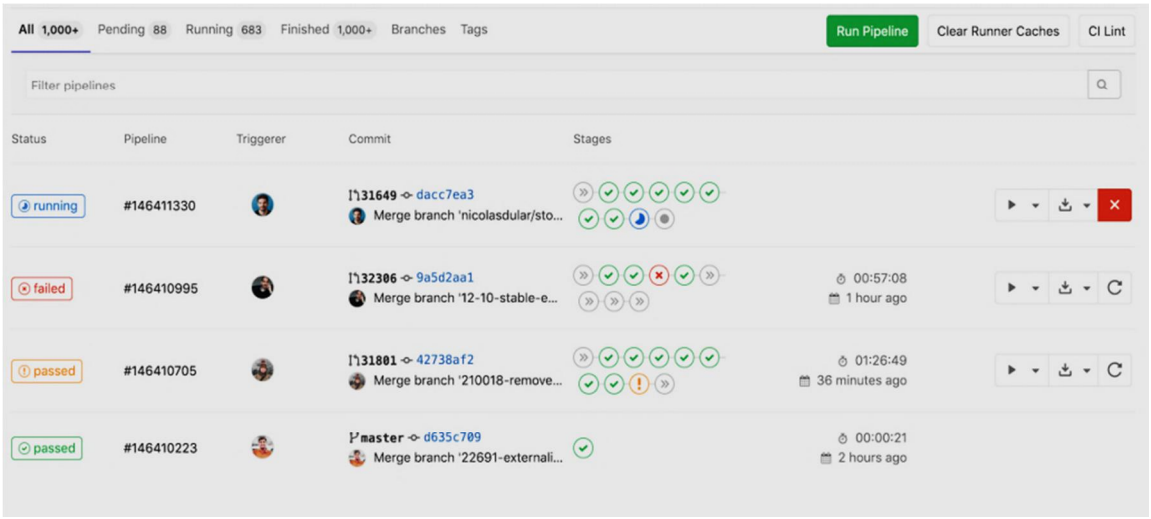

Рис. 5. Інтерфейс сервісу безперервної доставки та інтеграції GitLab CI/CD

GitLab ChatOps – система керування конвеєрами *GitlLab CI/CD,* реалізовану у вигляді додатку до месенджеру Slack (рис. 6). Взаємодія з ботом здійснюється через так звані слеш-команди [16].

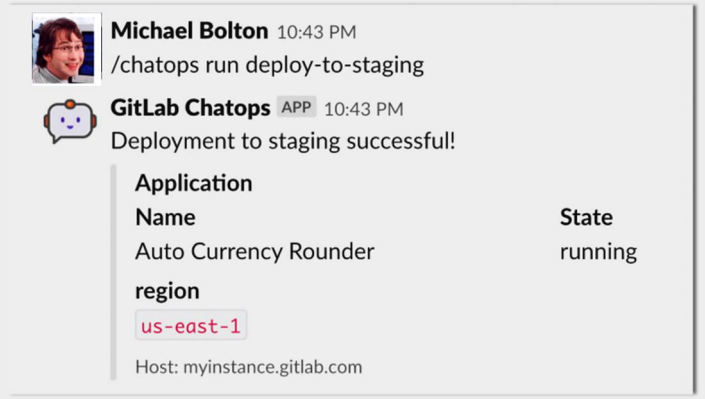

Рис. 6. Сповіщення від ChatOps системи GitlLab CI/CD у каналі Slack

До переваг цього сервісу слід віднести гнучкість, яку надає можливість передачі аргументів для конвеєрів безперервної інтеграції та доставки, що урізноманітнює спектр можливих завдань, які вони можуть виконувати, та керування ними з чату. До недоліків же варто включити той факт, що сам сервіс GitLab є двох версій: хмарна і власного хостингу. Відповідно інтеграція ChatOps відбувається по-різному для кожної з версій; також сповіщення щодо виконання запуску конвеєрів не використовують файлів журналів, які корисно переглядати у разі невдалих складань чи розгортань.

Результати аналізу і оцінювання платформ *CI/CD* та засобів керування *ChatOps* подано у таблиці 1.

Таблиця 1.

Результати аналізу платформ CI/CD

Назва CI/CD платформи	Сильні сторони	Недоліки
Хмарний сервіс безперервної інтеграції та доставки Azure DevOps Services	– легкість підключення через облікові записи Microsoft, – просте налаштування та користувацький інтерфейс, інтеграція з продуктами та сервісами Microsoft, – можливості масштабування проєктів.	– високі ціни на послуги – обмежений функціонал, – наявність тільки хмарного варіанту платформи
Сервіс безперервної доставки Heroku Pipelines	– просте налаштування, – зручний користувацький інтерфейс,	– високі ціни на послуги, – відсутність глибокого налаштування – слабкі засоби контролю процесів
Інструмент безперервної доставки GitLab CI/CD	– гнучкість налаштувань, – можливість передачі аргументів для конвеєрів CI/CD, – широкий спектр завдань CI/CD, – можливість керування з чату	– інтеграція ChatOps і GitLab CI/CD виконується по-різному у хмарній версії і власному хостингу, – сповіщення щодо запуску конвеєрів не застосовують файлів журналів

Аналіз функціональних можливостей, переваг та недоліків відомих систем керування процесом безперервної доставки програмного забезпечення, дає змогу зробити висновок про доцільність розроблення платформи власної платформи, яка забезпечить виконання таких завдань:

– вибір і ініціювання конвеєра безперервної інтеграції /доставки програмного забезпечення;

– вибір гілки системи контролю версій, з якої слід запустити конвеєр;
– вибір проєкту;
– перегляд результатів складання чи розгортання програмного продукту;
– можливість входу в сервіс за допомогою облікового запису;
– можливість перегляду файлів журналів складання чи розгортання програмного продукту.

Виклад основного матеріалу

Основними результатами даної роботи є проектування та розроблення системи управління безперервною доставкою програмного забезпечення. У процесі проектування системи було виконано завдання:

– аналіз та моделювання процесів та функцій системи управління безперервною доставкою програмного забезпечення;
– моделювання об’єктів системи;
– розроблення загального алгоритму функціонування системи;
– вибір та обґрунтування методів і засобів розроблення;
– розроблення та тестування програмного забезпечення системи.

Для аналізу та моделювання процесів і функцій системи було обрано принцип структурного проектування із застосуванням методології IDEF. На початковому етапі аналізу та моделювання розроблено IDEF0 контекстну діаграму інформаційної системи керуванням процесу безперервної доставки програмного забезпечення. Така діаграма (діаграма верхнього рівня), будучи вершиною деревоподібної структури діаграм, показує призначення системи (основну функцію) та її взаємодію із зовнішнім середовищем. Створена контекстна діаграма проектованої інформаційної системи складається з головного процесу “Здійснити керування процесом безперервної доставки програмного забезпечення”, входом є “Нова версія ПЗ”, виходом – “Нова версія ПЗ, розгорнута на сервері”, механізмом виконання дії є “Система контролю версій” та “Інструмент для безперервної інтеграції”, управлінням та обмеженням є “Стандарти розробки ПЗ” (рис.7).

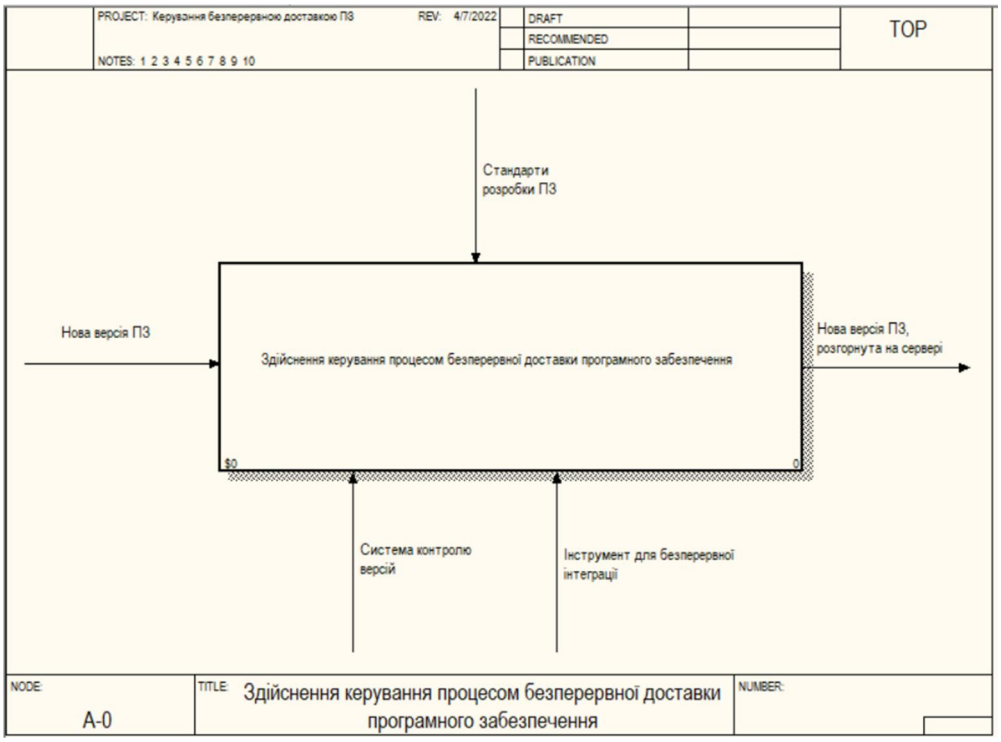

Рис. 7. Контекстна діаграма керування процесом безперервної доставки програмного забезпечення

На наступному етапі моделювання шляхом декомпозиції основного процесу було виділено 4 основні бізнес-процеси, такі як:

- розгорнути нову версію ПЗ конкретного проекту;
- розгорнути нову версію ПЗ з певної гілки;
- розгорнути нову версію ПЗ з конвеєру безперервної доставки;
- переглянути результат розгортання.

Результати такої декомпозиції показано у вигляді IDEF0 діаграми першого рівня (рис. 8).

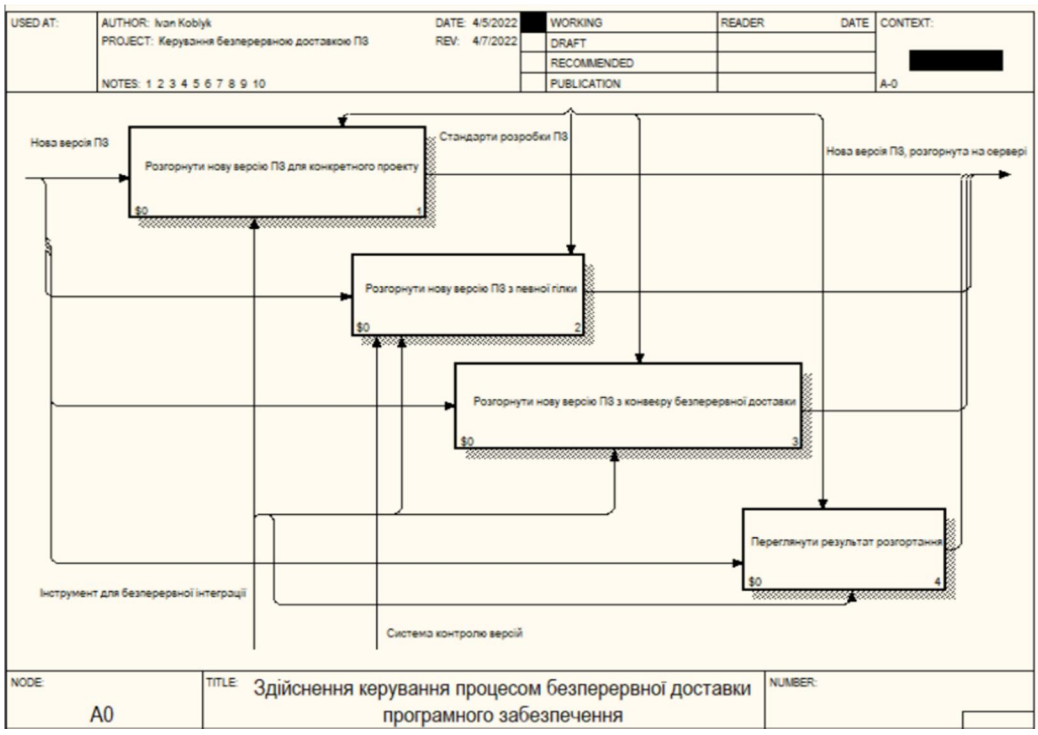

Рис. 8. Діаграма декомпозиції "Здійснення керування процесом безперервної доставки програмного забезпечення"

Подальшими кроками аналізу та моделювання системи управління безперервною доставкою програмного забезпечення є декомпозиція визначених підпроцесів до рівня функціональних блоків. Результати декомпозиції подано в табл. 2.

Таблиця 2.

Результати декомпозиції підпроцесів безперервної доставки програмного забезпечення

Назва підпроцесу	Функціональний блок
Розгорнути нову версію ПЗ для проекту	1. Переглянути список проектів 2. Обрати необхідний проект 3. Здійснити розгортання для проекту
Розгорнути нову версію ПЗ з певної гілки	1. Переглянути список гілок репозиторію 2. Обрати потрібну гілку 3. Здійснити розгортання для гілки
Розгорнути нову версію ПЗ з конвеєра безперервної доставки	1. Відкрити список конвеєрів безперервної доставки 2. Обрати необхідний конвеєр 3. Виконати розгортання з вибраного конвеєра
Переглянути результат розгортання	1. Переглянути список виконаних доставок 2. Вибрати потрібний випуск 3. Переглянути статус та журнал розгортання

Моделювання об’єктів системи управління безперервною доставкою програмного забезпечення виконано за методологією “Сутність-зв’язок”. В результаті моделювання визначено основні категорії об’єктів (сутності), їх властивості (атрибути) та зв’язки, які подано у вигляді E-R діаграми в нотації Чена (рис. 9).

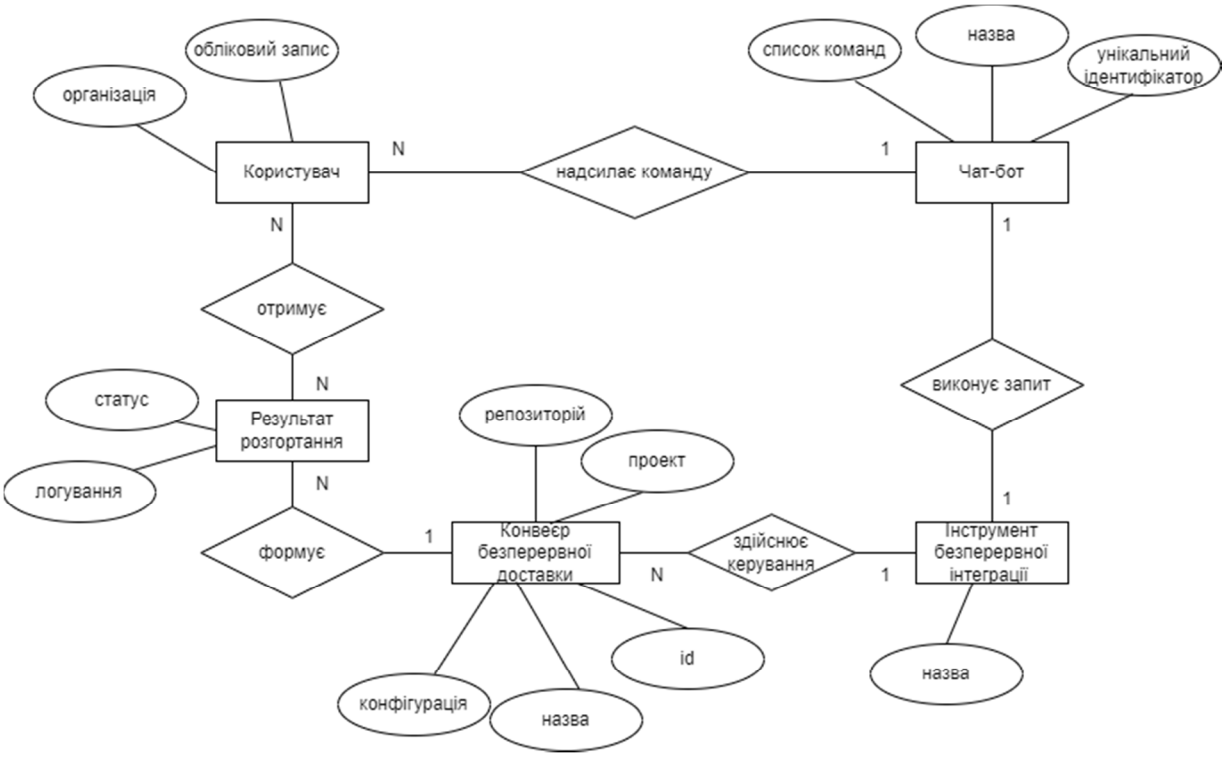

Рис. 9. E-R діаграма об’єктів системи управління безперервною доставкою програмного забезпечення

На діаграмі показано сутності: “Користувач”, “Чат-бот”, “Інструмент безперервної інтеграції”, “Конвеєр безперервної доставки”, “Результат розгортання”. У сутності “Користувач” визначено атрибути: обліковий запис і організація. Зазначена сутність має зв’язок типу “багато-до-багатьох” із сутністю “Результат розгортання” (користувачі отримують результати розгортань), а також зв’язок “агато-до-одного” з сутністю “Чат-бот” (користувачі надсилають команди чат-боту). Сутність “Чат-бот” має атрибути: список команд, назва, унікальний ідентифікатор. У неї є зв’язок “один-до-багатьох” із сутністю “Чат-бот” (користувачі надсилають команди чат-боту) та зв’язок “один-до-одного” з сутністю “Інструмент безперервної інтеграції” (чат-бот виконує запит до інструменту безперервної інтеграції). Сутність “Інструмент безперервної інтеграції” має один атрибут – назва. Зв’язок “один-до-багатьох” із сутністю “Конвеєр безперервної доставки” (інструмент може керувати багатьма конвеєрами доставки) та зв’язок “один-до-одного” з сутністю “Чат-бот” (чат-бот виконує запит до інструменту безперервної інтеграції). Сутність “Конвеєр безперервної доставки” має такі атрибути: репозиторій, назва, проект, конфігурація та id. Зв’язок “один-до-багатьох” із сутністю “Результат розгортання” (конвеєр безперервної доставки формує багато результатів розгортань), зв’язок “багато-до-одного” з сутністю “Інструмент безперервної інтеграції” (інструмент здійснює керування багатьма конвеєрами). Сутність “Результат розгортання” має наступні атрибути: статус та логування. Зв’язок “багато-до-багатьох” з сутністю “Користувач” (користувачі отримують результати розгортань), зв’язок “один-до-багатьох” із сутністю “Конвеєр безперервної доставки” (конвеєр безперервної доставки формує результати розгортань). Така модель визначає загальну архітектуру сервісу управління безперервною доставкою програмного забезпечення, розробленої у даній роботі.

На підставі результатів аналізу та моделювання процесів та об'єктів процесу безперервної доставки програмного забезпечення, розроблено систему управління такими процесами. Систему розроблено у форматі сервісу ChatOps. Такий формат обґрунтовано проведеним аналізом та позитивним досвідом розроблення аналогічних засобів компаніями Microsoft, Heroku, GitLab та іншими. Для розроблення програмних модулів сервісу використано технології та засоби: PyCharm, Telegram, Python, Flask, Azure DevOps, Azure DevOps Python Api, JSON, Bitbucket, Jira.

Основною компонентою сервісу є чат-бот ADO Orchestrator, який забезпечує керування процесами безперервної інтеграції та доставки, здійснюючи запуск конвеєрів безперервної інтеграції сервісу Azure DevOps Service. Архітектура розробленої системи передбачає, що і репозиторій із кодом проекту, для якого користувач здійснює інтеграцію та доставку, також розміщено в цьому сервісі.

Функціональним призначенням системи керуванням процесу безперервної доставки програмного забезпечення є надання розробникам, системним інженерам та іншим користувачам

a) можливості керувати конвеєрами безперервної інтеграції та доставки сервісу Azure DevOps Services;

b) отримувати та аналізувати повідомлення від Azure DevOps Services щодо запуску та виконання процесів безперервної доставки програмного забезпечення;

c) здійснювати моніторинг роботи конвеєрів безперервної інтеграції та доставки програмного забезпечення;

d) створювати, архівувати та переглядати журнали життєвих циклів процесів безперервної доставки програмного забезпечення.

Для доступу до сервісу користувачі використовують

- обліковий запис Microsoft з підпискою організації та проектів у сервісі Azure DevOps Services;
- обліковий запис для месенджера Telegram, на базі якого сформовано середовищем виконання програмних модулів чат-боту.

Застосування чат-боту ADO Orchestrator для управління процесами неперервної доставки програмного забезпечення побудовано за наступною схемою.

1. Клієнт здійснює вхід у систему, надсилаючи чат-боту повідомлення з даними, необхідними для інтеграції з сервісом Azure DevOps.

2. Після ініціювання сеансу, клієнт має можливість за допомогою інтерактивного меню надсилати команди для запуску потрібних операції та отримувати повідомлення про результати їх виконання у відповідь.

3. Повідомлення, що надсилаються користувачем чат-боту, постійно фільтруються, для отримання необхідної інформації та передачі її сервісу Azure DevOps, який через чат-бот надсилає її безпосередньо самому клієнтові.

Відповідно до такої схеми роботу розробленого чат-боту побудовано наступним чином.

1. Користувачеві слід знайти бот за його унікальним ім'ям в месенджері Telegram. Створений чат-бот має назву "ADO Orchestrator", а його унікальний ідентифікатор – "azuredevopstg_bot". Відкривши додаток чи веб-версію Telegram та перейшовши до поля пошуку, користувачеві слід ввести ідентифікатор бота. Варто обов'язково зазначити, що самому ідентифікатору має передувати символ "@". Перейшовши до сторінки самого бота, користувачі мають змогу побачити привітання з коротким описом його призначення та можливостей, а також логотип та назву бота (рис. 10).

2. Натиснувши кнопку "/start", користувач дає сигнал боту про початок роботи. Чат-бот у відповідь надсилає повідомлення, в якому просить надати персональний ключ доступу до акаунту в

системі Azure DevOps Services. Після того як користувачем надано ключ, бот надсилає повідомлення з проханням надати також і посилання на організацію в системі Azure DevOps Services. Отримавши необхідні дані, чат-бот намагатиметься здійснити вхід до системи. У випадку успішної авторизації в сервісі, бот надсилає повідомлення з проханням вибрати проект та список проектів у вигляді інтерактивних кнопок, натиснувши одну з яких, користувач може обрати потрібний йому проект. Натомість у разі помилки під час входу в обліковий запис чат-бот надсилає повідомлення про помилку авторизації та просить користувача повторно надіслати необхідні дані.

3. Вибравши потрібний проект із списку проектів, надісланих чат-ботом, користувач отримує у відповідь список конвеєрів безперервної інтеграції та доставки вибраного проекту. Наступним кроком після обрання конвеєру є обрання гілки репозиторію для системи контролю версій, для якої слід здійснити запуск відповідного конвеєру.

4. Обравши гілку зі списку, користувач отримує повідомлення про початок запуску відповідного складання та випуску. Він має змогу бачити унікальний номер випуску, а також можливість переглянути його статус та завантажити файли журналів, натиснувши відповідні інтерактивні кнопки, що були надіслані у цьому ж повідомленні.

5. Для здійснення моніторингу виконання конвеєру користувач має змогу аналізувати файли журналів процесів для конкретного випуску програмного забезпечення. При натисканні інтерактивної кнопки logs для відповідного випуску, чат-бот надсилає користувачеві у відповідь повідомлення, що містить інформацію про вибраний випуск та архів, у якому зберігаються його файли журналу (рис. 11).

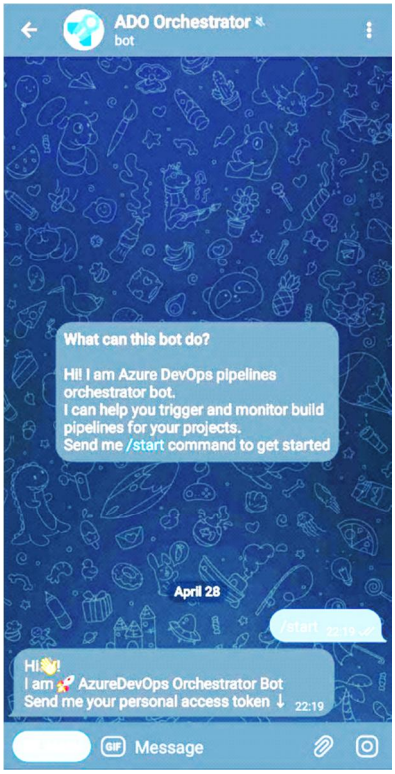

Рис. 10. Стартова сторінка чат-боту ADO Orchestrator

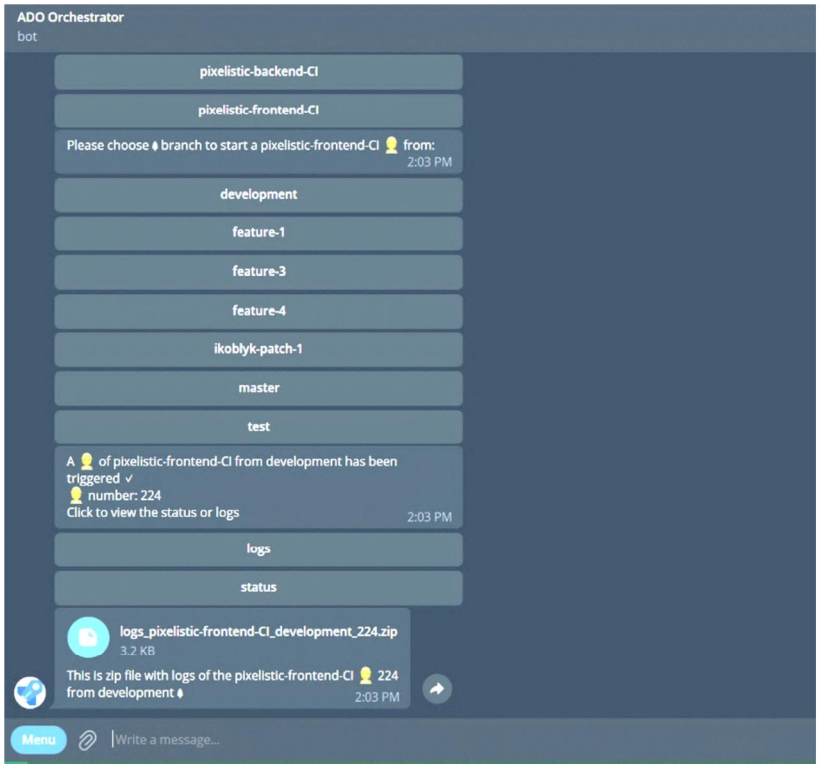

Рис. 11. Сторінка управління процесами чат-боту ADO Orchestrator

Тестування, да дослідне використання розробленого чат-бота для управління процесами неперервної доставки програмного забезпечення показали його коректну роботу, відповідність завданням та вимогам, здатність ефективно підтримувати роботу DevOps-інженера (рис. 12).

Порівняння з аналогами, які розглянуто у роботі, показує, що чат-бот ADO Orchestrator має певні переваги, зокрема такі:

1) поєднання в одному середовищі процесів управління та моніторингу процесів неперервної доставки програмного забезпечення;
2) можливість доступу та аналізу файлів журналів виконаних процесів;
3) простий та гнучкий інтерфейс користувача;
4) чат-бот використовує безкоштовне середовище функціонування – месенджер Telegram, на відміну від пропрієтарних засобів типу MS Teams або Slack;
5) сервіс може бути масштабовано та адаптовано до інших платформ, окрім месенджера Telegram, зокрема з використанням електронної пошти.

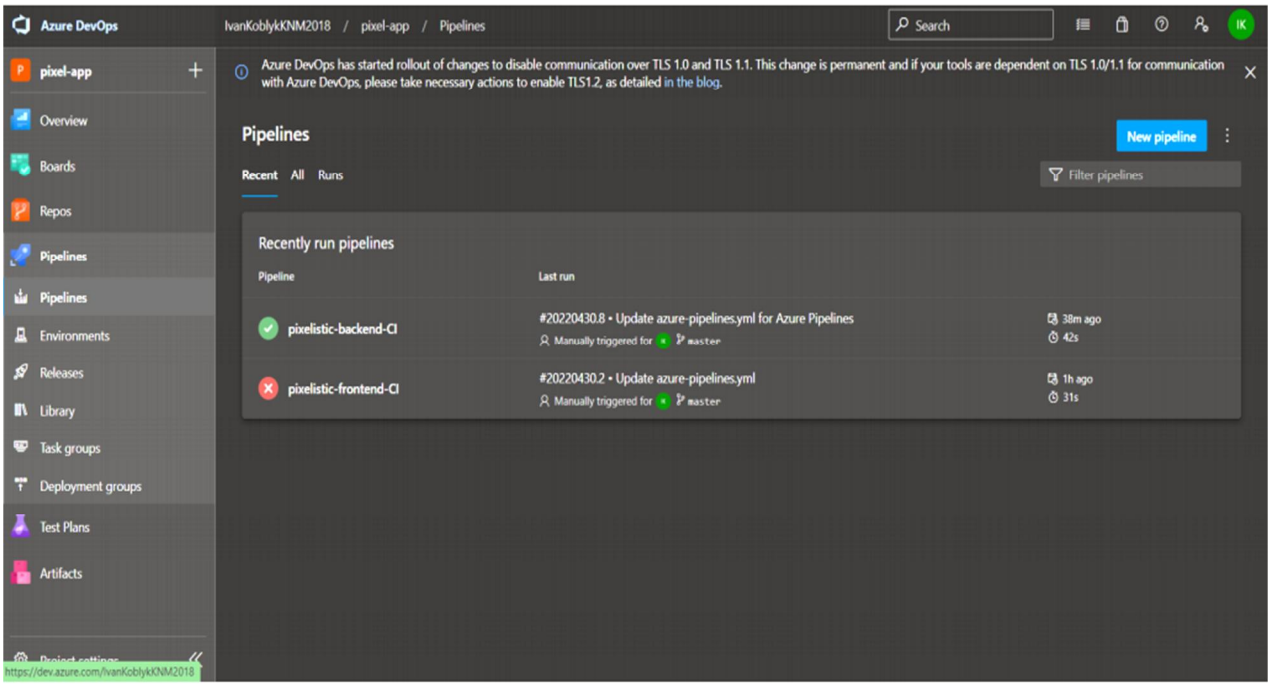

Рис. 12. Вікно повідомлень про виконання конвеєру в системі Azure DevOps

Висновки

У ході поданої роботи виконано актуальне завдання з проектування та розроблення системи управління процесами неперервної доставки програмного забезпечення. Дослідження предметної області діяльності DevOps фахівців показує важливість застосування в цій галузі новітніх підходів, що ґрунтуються, насамперед, на автоматизації процесів інтеграції та розгортання програмного забезпечення (CI/CD), яке розробляється в ході виконання IT-проектів. При цьому, сучасних ефективних засобів управління процесами CI/CD на ринку є недостатньо, а багато з них мають низку обмежень, недоліків, труднощів та незручностей у налаштуванні та застосуванні.

Результатом роботи є проект системи управління процесами неперервної доставки програмного забезпечення у форматі чат-бота як складової загального сервісу ChatOps. Такий формат використовують провідні компанії, такі як Microsoft, Salesforce, GitLab Inc. та інші у розробленні програмного забезпечення. Досвід таких лідерів підтвердив доцільність та ефективність побудови системи управління неперервною доставкою програмного забезпечення саме в такому виді.

У роботі застосовано інноваційне вирішення, яке поєднує в складі одного сервісу функції управління, моніторингу та аналізу виконання процесів неперервної доставки програмного забезпечення. Розроблений у роботі чат-бот ADO Orchestrator є повністю готовим до застосування програмним продуктом. Тестування та дослідне використання показало його придатність для автоматизації процесів неперервної доставки програмного забезпечення, а також ширшу, порівняно з аналогами, функціональність, гнучкість налаштувань, здатність до інтеграції з іншими сервісами DevOps та зручність користувацького інтерфейсу. Важливим аспектом є використання безкоштовної платформи месенджера Telegram, що робить продукт доступним для використання як у великих, так і малих бюджетних проектах.

Список літератури

1. Hall, J. A brief history of CI/CD (2021). Jonathan Hall. https://jhall.io/archive/2021/09/26/a-brief-history-of-ci/cd

2. Booch, G. (1998). Object-oriented analysis and design with applications. Addison-Wesley.

3. Fowler, M. (2024). Continuous Integration. Martinfowler.com. https://martinfowler.com/articles/continuousIntegration.html

4. Cooney, C. (2021). The Evolution of CI/CD From Bash to Buddy. Medium. https://medium.com/the-devops-corner/the-evolution-of-ci-cd-763df723f05b

5. Iyengar, R. (2021). Continuous Evolution: The CI/CD Story. Cloud Native Now. https:// https://cloudnativenow.com/topics/continuous-evolution-the-ci-cd-story/

6. Humble, G., & Farley, D. (2011). Continuous Delivery: Reliable Software Releases Build through, Test, and Deployment Automation. Addison-Wesley.

7. Sharma, A. (2023). A Brief History of DevOps, Part IV: Continuous Delivery vs. Continuous Deployment. Circleci. https://circleci.com/blog/a-brief- history-of-devops-part-iv-continuous-delivery-and-continuous-deployment

8. Heusser, M. (2021). Continuous delivery vs. continuous deployment: Which to choose? TechTarget. https://www.techtarget.com/searchitoperations/tip/Continuous -delivery-vs continuous-deployment-Which-to-choose

9. Sigler, E. (2014). What Is ChatOps? PagerDuty.https://www.pagerduty.com/blog/what-is-chatops Waits, T. (2015). ChatOps in the DevOps Team. Carnegie Mellon University. https://insights.sei.cmu.edu/blog/chatops-in-the-devops-team

10. IBM Cloud Education (2021). What Are the Benefits of ChatOps? IBM. https://www.ibm.com/cloud/blog/benefits-of-chatops

11. Gursimran, S, (2020). A Complete Guide to ChatOps. XenonStack. https://www.xenonstack.com/blog/a-complete-guide-to-chatops

12. Cron, N. (2021). ChatOps: Join The Conversation. Forbes. https://www.forbes.com/sites/forbestechcouncil/2021/10/26/chatops-join-theconversation/?sh=5e6cd7707717

13. Ceci, L. (2023). Number of mobile phone messaging app users worldwide from 2018 to 2025. Statista. https://www.statista.com/statistics/483255/number-of-mobile-messaging-users-worldwide

14. Miguel, F. B. (2021). The rise of messaging platforms. Medium. https://medium.com/chatbot-news-daily/the-rise-of-messenger-platforms-and-its-legal-implications-62fe73355122

15. Silverthorne, V. (2020). A surprising benefit of CI/CD: Changing development roles. Gitlab. https://about.gitlab.com/blog/2020/07/16/ci-cd-changing- roles

16. Siig, K.(2021). Heroku Pipelines Success Guide. Judoscale. https://judoscale.com/guides/heroku-pipelines

17. Duvall, P., Matyas S. & Glover A. (2007). Continuous Integration: Improving Software Quality and Reducing Risk. Addison-Wesley.

18. Дубленич, Р. & Струк, Є. (2017). Побудова CI/CD процесу розроблення програмного забезпечення з використанням TeamCity та Go CD. SCSIT, Volume 864(1), 250–256.

References

1. Hall, J. A brief history of CI/CD (2021). Jonathan Hall. https://jhall.io/archive/2021/09/26/a-brief-history-of-ci/cd

2. Booch, G. (1998). Object-oriented analysis and design with applications. Addison-Wesley.

3. Fowler, M. (2024). Continuous Integration. Martinfowler.com. https://martinfowler.com/articles/continuousIntegration.html

4. Cooney, C. (2021). The Evolution of CI/CD From Bash to Buddy. Medium. https://medium.com/the-devops-corner/the-evolution-of-ci-cd-763df723f05b

5. Iyengar, R. (2021). Continuous Evolution: The CI/CD Story. Cloud Native Now. https:// https://cloudnativenow.com/topics/continuous-evolution-the-ci-cd-story/

6. Humble, G., & Farley, D. (2011). Continuous Delivery: Reliable Software Releases Build through, Test, and Deployment Automation. Addison-Wesley.

7. Sharma, A. (2023). A Brief History of DevOps, Part IV: Continuous Delivery vs. Continuous Deployment. Circleci. https://circleci.com/blog/a-brief- history-of-devops-part-iv-continuous-delivery-and-continuous-deployment

8. Heusser, M. (2021). Continuous delivery vs. continuous deployment: Which to choose? TechTarget. https://www.techtarget.com/searchitoperations/tip/Continuous -delivery-vs continuous-deployment-Which-to-choose

9. Sigler, E. (2014). What Is ChatOps? PagerDuty.https://www.pagerduty.com/blog/what-is-chatops

10. Waits, T. (2015). ChatOps in the DevOps Team. Carnegie Mellon University. https://insights.sei.cmu.edu/blog/chatops-in-the-devops-team

11. IBM Cloud Education (2021). What Are the Benefits of ChatOps? IBM. https://www.ibm.com/cloud/blog/benefits-of-chatops

12. Gursimran, S, (2020). A Complete Guide to ChatOps. XenonStack. https://www.xenonstack.com/blog/a-complete-guide-to-chatops

13. Cron, N. (2021). ChatOps: Join The Conversation. Forbes. https://www.forbes.com/sites/forbestechcouncil/2021/10/26/chatops-join-theconversation/?sh=5e6cd7707717

14. Ceci, L. (2023). Number of mobile phone messaging app users worldwide from 2018 to 2025. Statista. https://www.statista.com/statistics/483255/number-of-mobile-messaging-users-worldwide

15. Miguel, F. B. (2021). The rise of messaging platforms. Medium. https://medium.com/chatbot-news-daily/the-rise-of-messenger-platforms-and-its-legal-implications-62fe73355122

16. Silverthorne, V. (2020). A surprising benefit of CI/CD: Changing development roles. Gitlab. https://about.gitlab.com/blog/2020/07/16/ci-cd-changing- roles

17. Siig, K.(2021). Heroku Pipelines Success Guide. Judoscale. https://judoscale.com/guides/heroku-pipelines

18. Duvall, P., Matyas S. & Glover A. (2007). Continuous Integration: Improving Software Quality and Reducing Risk. Addison-Wesley.

19. Dublenych, R. & Struk, Y. (2017). Pobudova CI/CD procesu rozroblennia programnogo zabezpechennia z vykorystanniam TeamCity ta Go CD. SCSIT, Volume 864(1), 250–256.

INFORMATION SYSTEM FOR MANAGING THE PROCESS OF CONTINUOUS SOFTWARE DELIVERY

Andriy Berko[1], Ivan Koblyk[2]

Lviv Polytechnic National University,
Information Systems and Networks Department, Lviv, Ukraine
[1] E-mail: Andrii.Y.Berko@lpnu.ua, ORCID: 0000-0001-6756-5661
[2] E-mail: ivan.koblyk.mnsam.2022@lpnu.ua ORCID: 0009-0000-9413-4533

Today, the direction of DevOps is an essential component in the execution of IT projects of various scales. This determines the need to develop the latest progressive approaches, methods, and technologies to ensure the effective activity of DevOps specialists. One of such popular solutions is continuous integration (CI) and continuous delivery (CD) of software. This approach (CI/CD) makes it possible to significantly speed up the processes of software development, testing, and implementation, improve the quality of development results, and ensure the controllability and manageability of these processes. A key factor in CI/CD processes is the use of tools to automate the work of DevOps specialists. The work describes a system for managing the continuous delivery of software, developed based on the modern application experience of leading companies.

Keywords: DevOps; CI/CD; ChatOps; Continuous Integration; Continuous Delivery.