№ 6(34) 2024
НАУКА і ТЕХНІКА СЬОГОДНІ
серії: право, економіка, педагогіка, техніка, фізико-математичні науки

УДК: 004.4:004.5:004.6

https://doi.org/10.52058/2786-6025-2024-6(34)-1027-1042

Постова Світлана Анатоліївна кандидат педагогічних наук, доцент, Декан фізико-математичного факультету, Доцент кафедри комп'ютерних наук та інформаційних технологій, Житомирський державний університет імені Івана Франка, вул. В. Бердичівська, 40, м. Житомир, 10008, тел.: (093) 129-51-31, https://orcid.org/0000-0002-0864-6290

Мельник Анна Віталіївна кандидат педагогічних наук, Старший викладач кафедри комп'ютерних наук та інформаційних технологій, Житомирський державний університет імені Івана Франка, вул. В. Бердичівська, 40, м. Житомир, 10008, тел.: (097) 750-23-39, https://orcid.org/0000-0001-7983-3598

ДОСВІД ІНТЕГРАЦІЇ DEVOPS: КЕЙСИ ВЕЛИКИХ IT ПРОЄКТІВ

Анотація. Стаття присвячена вивченню основних аспектів інтеграції для концепції DevOps, яка широко використовується провідними IT-компаніями, які розробляють та надають послуги з випуску програмного забезпечення. В статті охарактеризовано основні компоненти інтеграції DevOps, культура, автоматизація, вимірювання та обмін, що поєднує в собі комплекс безперервних процесів – безперервна інтеграція, безперервне розгортання та безперервна доставка.

Метою статті є вивчення основних складових компонентів, які можуть використовуватися в концепції DevOps для великих IT-проєктів, а також проведення дослідження робочих процесів, таких як безперервна доставка, безперервна інтеграція та безперервне розгортання програмного забезпечення. Розглянуто та проаналізовано основні аспекти безперервної інтеграції, що складається з автоматизованої збірки, модульного тестування, тестування забезпечення якості, генерації артефактів, циклу зворотного зв'язку, знань управління конфігурацією. На основі цих аспектів розглянуто характеристику основних процесів, які містить безперервна інтеграція, яка складається з побудови вихідного коду, самоперевірки, підготовки артефакту, забезпечення якості, перевірки безпеки та публікації артефакту.

Проаналізовано використання концепції DevOps великими IT-компаніями, такими як Netflix та Microsoft. Проаналізувавши обидва великі IT-проєкти можна зробити висновок, що концепція DevOps може використовуватися як для стрімінгових сервісів (Netflix), так і для хмарних платформ (Microsoft Azure), де було проаналізовано ідентичні параметри: архітектура, інтеграція, інфраструктура, автоматизація та оркестрація,

моніторинг та логування, безпека. Після чого було проаналізовано потенційні виклики та проблеми, де ефективним рішенням може слугувати автоматизація та безперервні процеси з використанням мікросервісної архітектури, що дозволить скоротити час розгортання з місяців до хвилин, а також сприятиме підвищенню надійності та швидкості релізів.

Ключові слова: інтеграція, DevOps, безперервна інтеграція, безперервна доставка, безперервне розгортання, IT-проєкти.

Postova Svitlana Anatolyivna Candidate of Pedagogical Sciences, Associate Professor, Dean of the Faculty of Physics and Mathematics, Associate Professor of the Department of Computer Sciences and Information Technologies, Zhytomyr Ivan Franko State University, st. V. Berdychivska, 40, Zhytomyr, 10008, phone: (093) 129-51-31, https://orcid.org/0000-0002-0864-6290

Melnyk Anna Vitalyivna, Candidate of Pedagogical Sciences, Senior Lecturer of the Department of Computer Sciences and Information Technologies, Zhytomyr Ivan Franko State University, st. V. Berdychivska, 40, Zhytomyr, 10008, phone: (097) 750-23-39, https://orcid.org/0000-0001-7983-3598

DEVOPS INTEGRATION EXPERIENCE: CASES OF LARGE IT PROJECTS

Abstract. The article is devoted to the study of the main aspects of integration for the DevOps concept, which is widely used by leading IT companies that develop and provide software release services. The article describes the main components of DevOps integration, culture, automation, measurement and exchange, which combines a set of continuous processes - continuous integration, continuous deployment and continuous delivery.

The purpose of the article is to study the main components that can be used in the DevOps concept for large IT projects, as well as to conduct a study of work processes such as continuous delivery, continuous integration and continuous software deployment. The main aspects of continuous integration, consisting of automated build, unit testing, quality assurance testing, artefact generation, feedback loop, and configuration management knowledge, are considered and analysed. Based on these aspects, the article describes the characteristics of the main processes that continuous integration contains, which consists of building source code, self-testing, preparing an artefact, quality assurance, security testing, and publishing an artefact.

The article analyses the use of the DevOps concept by large IT companies such as Netflix and Microsoft. Having analysed both large IT projects, it can be concluded that the DevOps concept can be used for both streaming services (Netflix)

and cloud platforms (Microsoft Azure), where the same parameters were analysed: architecture, integration, infrastructure, automation and orchestration, monitoring and logging, security. After that, we analysed potential challenges and problems, where automation and continuous processes using microservice architecture can serve as an effective solution, which will reduce deployment time from months to minutes, as well as increase the reliability and speed of releases.

Keywords: integration, DevOps, continuous integration, continuous delivery, continuous deployment, IT projects.

Постановка проблеми. Для сучасного IT-середовища великі проєкти стикаються з численними викликами, пов'язаними з ефективністю розробки, надійністю, масштабованістю та швидкістю випуску продуктів. Інтеграція методології DevOps, яка спрямована на автоматизацію та поліпшення взаємодії між командами розробки (Dev) та операцій (Ops) є ключовим підходом для вирішення цих викликів [1]. Інтеграція DevOps у великих IT-проєктах стає все більш важливою для забезпечення ефективності, гнучкості та розгортання програмного забезпечення [2]. Проте, впровадження DevOps у великих IT-проєктах може супроводжуватися низкою проблем, які можуть впливати на успішність та ефективність цього процесу. До таких проблем відноситься наступне.

1. Організаційні обмеження (бар'єри), які містять відсутність взаєморозуміння між командами (традиційно команди розробки та операцій працюють окремо, що створює бар'єри для ефективної комунікації) та опір змінами (де у великих організаціях часто спостерігається опір новим підходам та методології, шо ускладнює впровадження DevOps);
2. Складність інтеграції інструментів та процесів, що обумовлюється різноманіттям інструментів (вибір та інтеграція різних DevOps інструментів CI/CD, моніторинг, управління конфігураціями, що може бути складним завданням для великих IT-проєктів з різноманітними вимогами), а також сумісністю систем (велика кількість застарілого обладнання, програмного забезпечення та систем, які потребують інтеграції з сучасними DevOps інструментами);
3. Масштабованість та продуктивність, де забезпечення ефективного масштабування процесів DevOps для підтримки великих проєктів є складним завданням, що потребує ретельного планування та впровадження для підтримання високої продуктивності систем у масштабованих середовищах з великими обсягами даними;
4. Безпека, де інтеграція DevOps повинна враховувати питання безпеки на всіх етапах життєвого циклу розробки, що може вимагати додаткових ресурсів та змін у процесах;
5. Навчання та підготовка персоналу, де основними проблемами є дефіцит кваліфікованих кадрів (Впровадження DevOps вимагає наявності

фахівців з відповідними навичками, що може бути проблемою у великих проєктах через дефіцит таких фахівців) та навчання персоналу (де необхідно проводити постійне навчання та підвищення кваліфікації персоналу для ефективного використання DevOps інструментів та методологій).

Інтеграція DevOps у великі IT проєкти є складним, але необхідним процесом для досягнення підвищеної ефективності, швидкості розробки та надійності продуктів. Успішне подолання перерахованих вище проблем вимагає комплексного підходу, включаючи зміну організаційної культури, вибір та інтеграцію відповідних інструментів, забезпечення масштабованості та безпеки процесів, а також навчання та підготовку персоналу. Вирішення цих проблем дозволить організаціям повною мірою скористатися перевагами методології DevOps.

Аналіз останніх досліджень та публікацій. Література з останніх досягнень про інформаційні системи в переважній більшості присвячується вивченню управління проєктами з акцентом на проєкти з розробкою програмного забезпечення. Однак завдяки цифровій трансформації все більш організацій та компаній широко впроваджують міжфункціональні команди, що поєднує в собі розробку програмного забезпечення з виконанням завдань програмного забезпечення. Водночас з цим концепція DevOps спрямована на ефективне управління розробкою та операційною діяльністю, а також на плавне управління напружених відносин всередині команд, що є результатом неоднорідного складу навичок, обов'язків та стилів роботи [3].

Незважаючи на те, що безліч організацій застосовують практику DevOps, завдяки суттєвим перевагам, які позитивно впливають на швидкий вихід (розгортання) та надійність, існують також потенційні виклики та проблеми, які безпосередньо впливають на реалізацію концепції DevOps. В роботі [4] авторами проводилося дослідження, де обговорювалися проблеми, пов'язані з культурою DevOps та її практикою. Автори зробили акцент на тому, як DevOps працює в організації, надає детальне визначення концепції та визначає культурні проблеми, з якими можуть стикатися організації ще на етапі впровадження концепції DevOps. З цією метою авторами була розроблена модель DevOps Culture Challenges Model (DC2M) з метою покращення співпраці, розуміння та зменшення бар'єрів між командою розробників та операційною групою.

Інтеграція DevOps передбачає собою розробку платформи, спрямованої на підвищення ефективності управління програмними проєктами, де платформа може містити в собі стратегію, яка використовує безперервну інтеграцію (БІ) та безперервну доставку (БД). В дослідженні [5] авторами надана ефективна платформа на основі безперервної інтеграції та конвеєра безперервної доставки для компіляції вихідного коду, аналізу коду, виконання коду аж до його розгортання на основі повністю автоматизованого способу.

З розвитком інформаційних технологій великого значення набувають хмарні обчислення та технології, де концепція DevOps також відіграє важливе значення. Оскільки організації та компанії дедалі частіше переміщують свою інфраструктуру та додатки в хмару, традиційний ізольований підхід до розробки та експлуатації програмного забезпечення вже не відповідає вимогам гнучких та масштабованих систем. Деякі дослідження спрямовані на вивчення основних принципів та переваг концепції DevOps, наприклад, в роботі [6] автором розглядається, як концепція DevOps адаптувалася та трансформувалася у відповідь на унікальні виклики і можливості, що надаються хмарним середовищем. На основі аналізу галузевих тенденцій, конкретних прикладів та досліджень автором описано ключові аспекти, практики та майбутні перспективи DevOps в епоху хмарних технологій.

Останні досягнення в галузі машинного навчання відкрили нові можливості, які дозволяють комбінувати операції, процеси та розробки в поєднанні зі штучним інтелектом (ШІ), що дозволить автоматизувати деякі робочі функції. В роботі [7] авторами проводилося дослідження з оптимізації DevOps за допомогою ШІ для спрощення хмарної безперервної інтеграції та безперервного розгортання. Автором висувається оптимізована ШІ модель DevOps в якості запропонованої моделі, яка є ефективною в процесі розробки та дистрибуції, де в загальному підсумку підсумовуються результати додатків в поєднанні з робочим процесом DevOps для усунення розриву між розробкою моделі машинного навчання і оперативним розгортанням, що дозволяє використовувати підхід ШІ DevOps для сучасних розробок програмного забезпечення з використанням хмарних платформ БІ та БД.

Метою статті є вивчення основних складових компонентів, які можуть використовуватися в концепції DevOps для великих IT-проєктів, а також проведення дослідження робочих процесів, таких як безперервна доставка, безперервна інтеграція та безперервне розгортання програмного забезпечення. Для досягнення поставленої мети необхідно вирішити наступні задачі: на основі літературного огляду визначити основні компоненти інтеграції DevOps та охарактеризувати принципи, які можуть використовуватися при розробці, тестуванні та розгортанні програмного забезпечення; охарактеризувати основні складові концепції DevOps та основні робочі процеси; проаналізувати комбінацію безперервної інтеграції та безперервного розгортання програмного забезпечення в процесі розробки, яку використовують сучасні IT-компанії; з'ясувати, які переваги та недоліки може містити в собі концепція DevOps для великих сучасних IT-проєктів.

Виклад основного матеріалу. Концепція DevOps виникла для того, щоб подолати розрив між розробкою програмного забезпечення та розгортанням цього програмного забезпечення у виробництво для великих IT-компаній, що займаються розробкою та наданням продуктів програмного забезпечення [8].

Тому методологія DevOps передбачає собою співпрацю між групою розробників та спеціалістів з експлуатації для створення програмного забезпечення з метою швидкого розгортання IT-рішень на ринку.

Основною метою DevOps є використання безперервних процесів розробки програмного забезпечення, наприклад, безперервна доставка, безперервне розгортання та мікросервіси для підтримки гнучкого життєвого циклу розробки програмного забезпечення. Згідно з метою забезпечується сприяння співпраці між розробкою та операціями, частково шляхом автоматизації завдань зі створення та розгортання коду та тестування, щоб скоротити час розробки програмного забезпечення, забезпечити безперервну доставку програмного забезпечення та підвищити стабільність програмного забезпечення з метою задоволення потреб клієнтів [8, 9].

DevOps складається з таких компонентів, як культура, автоматизація, вимірювання та обмін, що набуває популярності завдяки своєму постійному підходу – безперервній інтеграції (БІ), безперервному розгортанню (БР) та доставці (БД) програмного забезпечення. БІ вимагає від розробника фіксувати код декілька разів на день з наступною автоматичною збіркою, тестуванням та негайним зворотним зв'язком з виробником щоразу, коли виявляється будь-яка помилка [11].

В контексті DevOps важливою є можливість надійного оновлення системи в робочому стані, де DevOps передбачає собою крос-функціональну співпрацю та автоматизацію між розробкою та експлуатацією програмного забезпечення. Прийняття та впровадження DevOps в компаніях є нетривіальним через необхідні зміни в технічних, організаційних та культурних аспектах. [12].

Окрім того, DevOps визначається як культурний та технологічний підхід до інтеграції завдань, знань та навичок, пов'язаних із плануванням, створенням та виконанням дій в рамках однієї міжфункціональної команди, відповідальної за один або декілька цифрових продуктів. Тому концепція DevOps, як правило, складається з трьох основних принципів:

1) принцип потоку, який відповідає за тісну співпрацю між розробкою та операціями для досягнення високої якості, а також для наскрізної відповідальності за програмні продукти;

2) принцип зворотного зв'язку, шо стосується використання гнучких значень та автоматизації для досягнення безперервної доставки програмного забезпечення;

3) принцип безперервного навчання та середовища довіри підтримує культуру організаційного навчання на успіхах та невдачах.

У традиційній розробці програмного забезпечення існує чітке розмежування між такими видами діяльності, як планування, аналіз, проектування та кодування, які є необхідними для розгортання виробництва. Після завершення роботи з розробки операційна група відповідає за керування підтримкою та

обслуговуванням, коли виникає необхідність внесення певних змін в запущене програмне забезпечення.

Однак, в середовищах, що швидко змінюються, IT-службам доводиться координувати та узгоджувати зміни у своїх відділах, оскільки розробники працюють зі складними та динамічними середовищами, де від них очікується постійне впровадження нових рішень та інновацій, щоб створювати складні критично важливі для бізнесу та безпеки програмне забезпечення. Тісна співпраця між компонентами розвитку та операційними функціями IT необхідні для покращення доставки, якості, надійності та стійкості програмного забезпечення та забезпечення швидкого усунення помилок. Багатофункціональні команди DevOps потребують співпраці між різними експертами для досягнення цих цілей. Тому для того, щоб максимізувати ці переваги більшість сучасних компаній впроваджують спеціалізовані групи DevOps, щоб поступово відмовитися від обслуговування та підтримки ізольованих IT-відділів.

Для того, щоб краще зрозуміти сутність концепції DevOps розглянемо автоматизацію процесів, що забезпечує якість коду та зменшує кількість дефектів. На рис. 1 показано схему, згідно якої описуються всі компоненти інтеграції DevOps, що є доволі трудомістким процесом. Технічні навички в поєднанні з гнучкими навичками здатні доповнювати ключові компоненти, які відносяться до успішної інтеграції концепції DevOps, як показано на рис. 1.

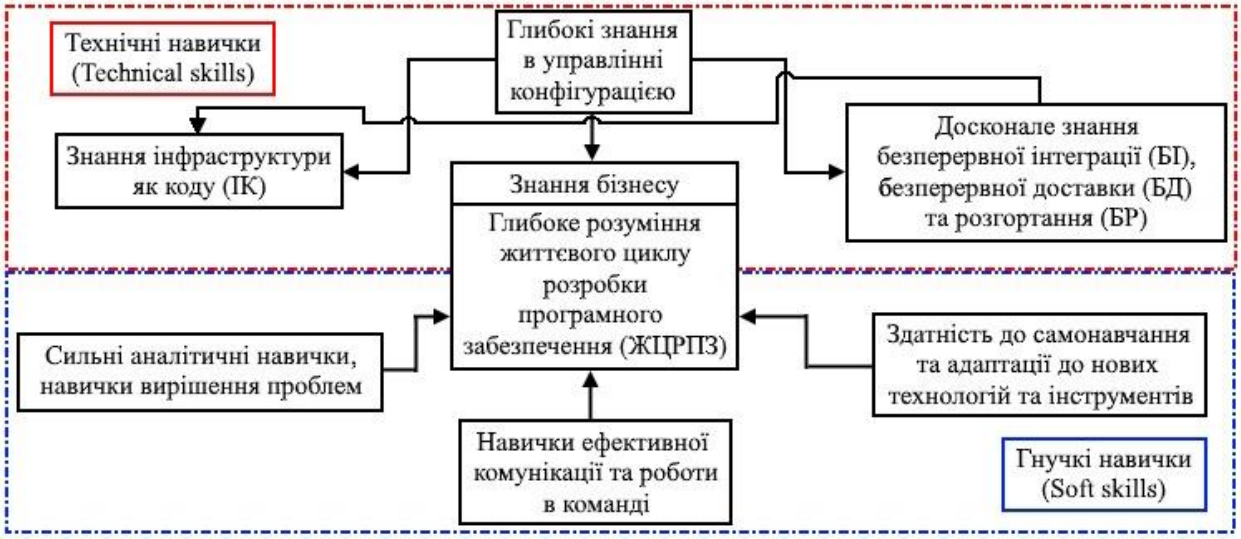

Рис. 1 Складові компоненти інтеграції концепції DevOps

Джерело: сформовано автором на основі [13]

Як видно з рис. 1 схема складається з окремих компонентів, які доповнюють гнучкі та технічні навички в поєднанні з глибоким розумінням життєвого циклу програмного забезпечення (ЖЦРПЗ), куди також входять глибокі знання в управлінні конфігурацією. Глибоке розуміння ЖЦРПЗ передбачає собою розуміння та знання всіх етапів, від планування та проєктування до

тестування та розгортання. Таке розуміння дає змогу ефективно співпрацювати з командами розробки, контролю якості та операцій з метою забезпечення плавної, ефективної розробки та випуск програмного забезпечення.

Володіння принципами інфраструктури як коду (ІК) в концепції DevOps передбачає собою опис та керування інфраструктурою за допомогою коду на основі інструментів, наприклад, Chef, Terraform, Puppet тощо. ІК дозволяє автоматично створювати та конфігурувати інфраструктуру, що робить її більш ефективною, надійною та масштабованою.

Знання та навички володінням БІ, БД та БР вимагають чіткого налаштування цих процесів, що впливає на автоматизацію створення, тестування та випуск програмного забезпечення. Завдяки цим процесам можливо скоротити час випуску програмного забезпечення та покращення його якості. БІ, БД та БР являє собою практику автоматичної доставки та розгортання додатків у виробниче середовище після успішного проходження всіх етапів БІ, що показано на рис. 2.

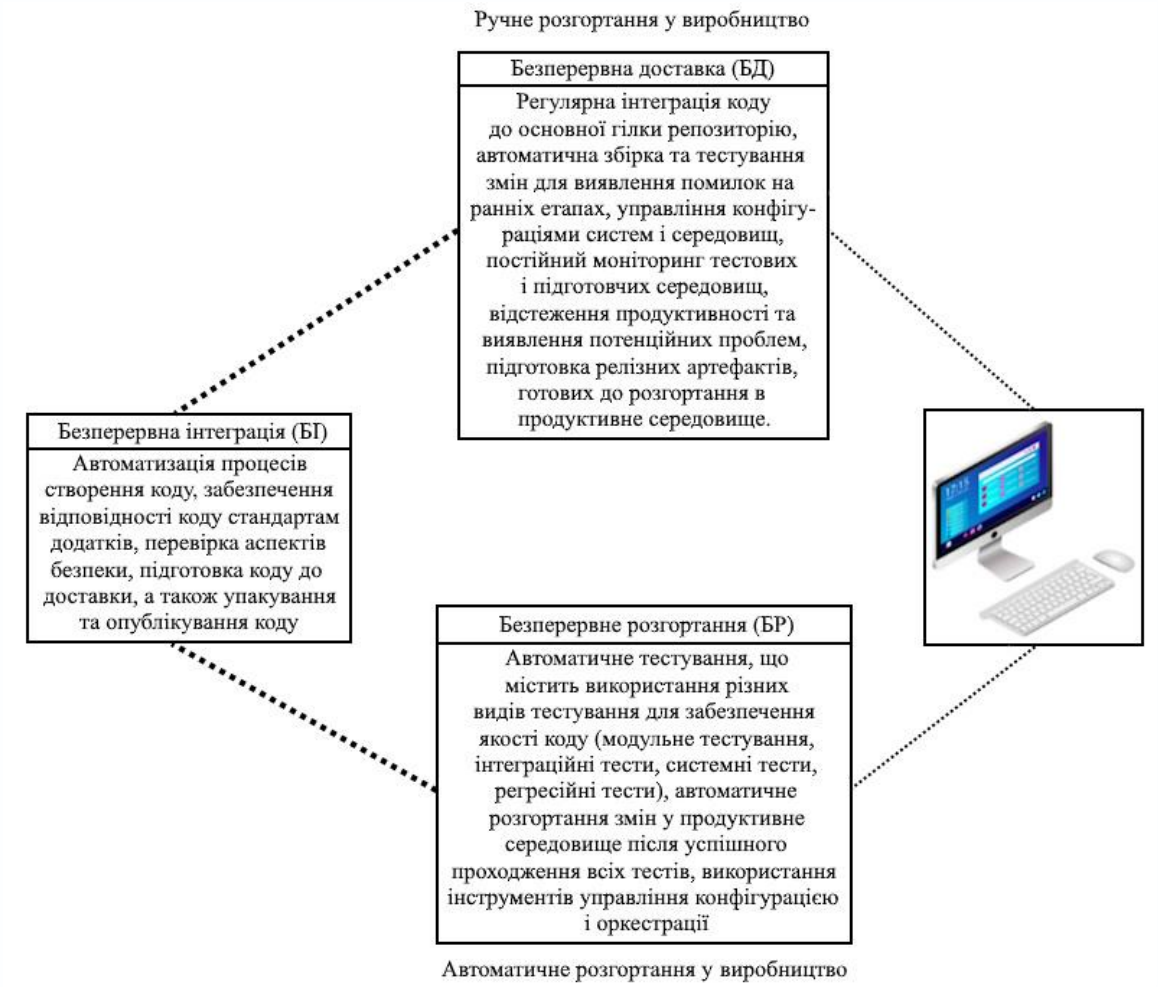

Рис. 2 Основні робочі процеси безперервної доставки, безперервної інтеграції та безперервного розгортання

Джерело: сформовано автором на основі [13]

Однією з відмінних рис є те, що безперервна доставка розгортає код безперервно за допомогою ручних тригерів, в той час як безперервне розгортання автоматизує розгортання без втручання людини. Якщо процеси безперервної доставки та розгортання не потребують особливих підходів, то безперервна інтеграція передбачає розробку більш детальної стратегії.

Основна ідея CI полягає в автоматизації процесу створення коду, перевірці відповідності коду стандартам додатків, оцінці факторів безпеки, підготовці коду до доставки, його пакування та публікації. Тому БІ висувається в ролі конвеєра, який здатний об'єднувати в собі всі компоненти в одне ціле, що дозволяє переводити компоненти з одного етапу на інший у випадку, коли вони будуть готові. Це дозволяє виявляти та виправляти помилки на ранніх стадіях розробки, що в свою чергу забезпечує швидку та надійну доставку нових функцій та можливостей. Завдяки скороченню часу між розробкою нового функціоналу та передачею його користувачам, БІ також впливає на мінімізацію ризиків та проблем, які можуть бути пов'язані з розгортанням.

Ключові аспекти БІ складаються з наступного:

- автоматизовані збірки, де розробниками можуть доволі часто фіксуватися зміни коду, запускаючи автоматизовані збірки, які компілюють код, запускають тести якості коду та генерують артефакти, які можна розгорнути;
- модульне тестування (юніт-тестування), де системи БІ виконують модульні тести для перевірки окремих компонентів кодової бази, а завчасне виявлення дефектів допомагає підтримувати якість коду;
- тестування забезпечення якості, де БІ гарантує, що зміни в коді можуть легко інтегруватися з існуючим кодом;
- генерація артефактів, де процес БІ може створювати готові до розгортання артефакти (контейнери, блоки, двійкові файли тощо);
- цикл зворотного зв'язку, де розробники безперервно можуть отримувати інформацію про зміни в коди, що потребує оперативного вирішення конфліктних ситуацій та проблем;
- знання управління конфігурацією, де інженерам необхідно знати принципи керування конфігурацією для централізованого керування конфігурацією інфраструктури та програмного забезпечення, що забезпечує узгоджену конфігурацію в усіх системах, знижуючи при цьому ризик помилок та підвищуючи надійність.

Далі розглянемо загальну структуру робочих процесів, які містить безперервна інтеграція, як показано на рис. 3. Безперервна інтеграція складається з шести основних процесів, які взаємопов'язані між собою й доповнюють один одного. В табл. 1 наведена характеристика цих процесів.

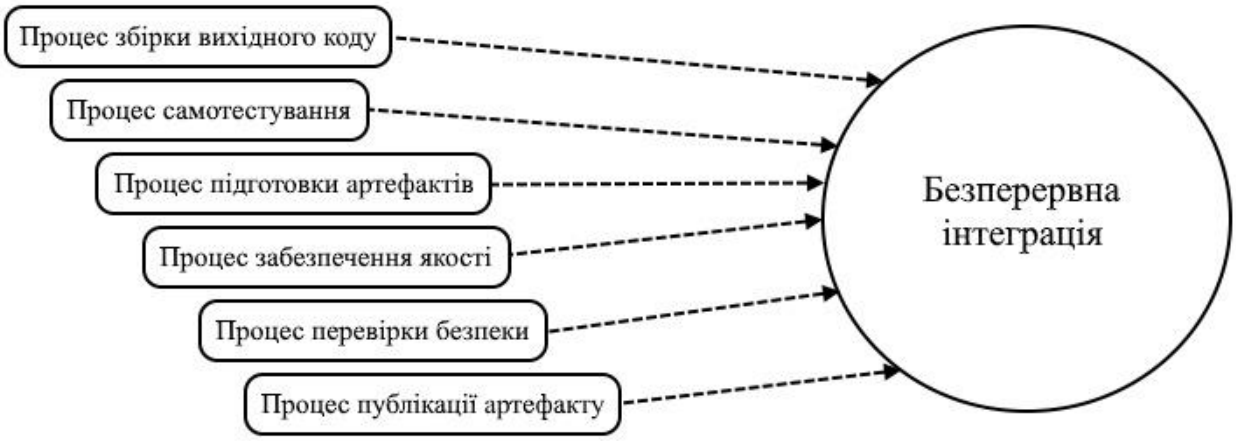

Рис. 3 Загальна структура робочих процесів безперервної інтеграції
Джерело: сформовано автором на основі [13]

Таблиця 1.

Характеристика основних процесів безперервної інтеграції

Процеси	Характеристики
Побудова вихідного коду	Цей початковий крок передбачає компіляцію та створення вихідного коду.
Самоперевірка	Для перевірки функціональності коду виконуються автоматичні тести. Ці тести охоплюють модульне тестування, тестування якості коду та інші перевірки якості. Це гарантує відповідність коду стандартам і умовам кодування.
Підготовка артефакту	Тут код упаковано в артефакти, які можна розгортати. Ці артефакти можуть бути двійковими файлами, контейнерами або іншими формами, придатними для розгортання.
Забезпечення якості	Процеси контролю якості, включаючи ручне тестування та прийняття користувачами тестування, переконайтеся, що артефакти відповідають бажаним стандартам якості.
Перевірка безпеки	Сканування безпеки та оцінка вразливості виконуються для виявлення та усунення будь-яких ризиків безпеці.
Публікація артефакту	Нарешті перевірені та захищені артефакти публікуються у відповідному середовищі.

Джерело: сформовано автором

В межах великих IT-проектів DevOps використовується для автоматизації процесів розгортання, моніторингу, тестування та управління інфраструктурою. Розглянемо деякі IT-проєкти, які на сьогодні широко використовують інтеграцію DevOps.

1. Інтеграція DevOps для Netflix

Архітектура. Американська компанія Neflix надає стрімінговий сервіс фільмів та серіалів широко почала впроваджувати хмарні практики DevOps, в

якій використовується архітектура мікросервісів, де кожен компонент додатку працює незалежно. Це дозволяє швидко впроваджувати інновації та надавати нові функції для своєї користувацької бази (завдяки інструментам з відкритим вихідним кодом), а розробникам це дозволяє працювати над окремими частинами системи без ризику впливу на інші системи [6].

Інтеграція. Незважаючи на те, що інтеграція DevOps для Netflix є складним процесом, який включає в себе впровадження найкращих практик та інструментів для автоматизації, моніторингу та покращення всього життєвого циклу розробки програмного забезпечення, стрімінговий сервіс є відомим прикладом компанії, яка успішно впровадила DevOps підхід, що дозволяє їм швидко та безпечно випускати нові функції та оновлення. Тому Netflix використовує інструменти, такі як Jenkins, Spinnaker, GitHub для БІ та БР, що дозволяє автоматизувати процеси збору, тестування та впровадження коду.

Інфраструктура. Netflix використовує підхід інфраструктура як код (ІК), де інфраструктура керується та налаштовується через код. Інструменти, такі як Terraform та AWS CloudFormation допомагають автоматизувати процеси розгортання інфраструктури. Успіх DevOps у Netflix можна пояснити їхнім сильним акцентом на автоматизації, відмовостійкості, та децентралізації. Використовуючи хмарні платформи, такі як AWS, вони досягли масштабованої інфраструктури, яка обробляє мільярди запитів щодня.

Автоматизація та оркестрація. Автоматизація є важливим аспектом DevOps Netflix, де використовуються інструменти для автоматизації завдань розгортання, масштабування та управління інфраструктурою (наприклад, Spinnaker, що використовується для автоматизації процесів розгортання додатків).

Моніторинг та логування. З метою забезпечення надійної роботи системи Netflix використовує розширені рішення для моніторингу та логування. Деякі інструменти, наприклад, Prometheus, Grafana та власний набір інструментів з найменуванням Atlas використовуються для моніторингу продуктивності та відстеження проблем.

Безпека. Важливим аспектом інтеграції DevOps є безпека, де Netflix впроваджує безпекові практики на всіх етапах життєвого циклу розробки, що включає в себе використання таких інструментів, як OWASP ZAP для тестування безпеки додатків.

Потенційні проблеми. Проблема висувається в необхідності швидко доставляти новий контент та оновлення для мільйонів користувачів по всьому світу. Рішенням цієї проблеми може слугувати автоматизація, БІ, БД та БР з використанням мікросервісної архітектури, що дозволить скоротити час розгортання з місяців до хвилин, а також сприятиме підвищенню надійності та швидкості релізів.

2. Інтеграція DevOps для Microsoft Azure

Інтеграція DevOps в Microsoft Azure передбачає впровадження найкращих практик та використання інструментів, що надаються Azure, для автоматизації, моніторингу та покращення всього життєвого циклу розробки програмного забезпечення. Azure пропонує широкий спектр сервісів, які підтримують DevOps підхід, що дозволяє організаціям ефективно управляти своїми додатками та інфраструктурою [6].

Архітектура. Подолання розриву між розробкою та операційною діяльністю Microsoft Azure охопила DevOps, щоб забезпечити комплексну платформу для хмарних обчислень. Azure DevOps пропонує набір інструментів для підтримки всього життєвого циклу DevOps, включаючи контроль вихідного коду, збірку, розгортання та управління випусками. Хмарна платформа компанії Microsoft підтримує мікросервісу архітектуру через такі сервіси: Azure Kubernetes Service (AKS) в якості сервісу для розгортання, управління та масштабування додатків, Azure Service Fabric в якості платформи для створення та управління масштабованими та надійними мікросервісами, Azure Functions Overview в якості сервісу з обчислювальними ресурсами в хмарному середовищі Microsoft Azure, що дозволяє розробникам писати та розгортати додатки.

Інтеграція. Інтеграція DevOps в Microsoft Azure забезпечує організаціям високу ступінь автоматизації, надійності та швидкості для випуску нових функцій, що є важливим для успіху в сучасному конкурентному середовищі. Azure надає всі необхідні інструменти та сервіси для ефективного впровадження DevOps практик. Завдяки тісній інтеграції з популярними інструментами розробки, такими як Visual Studio, Azure DevOps забезпечує безперешкодну співпрацю між командами розробників та операційними командами. Azure DevOps пропонує повний набір інструментів для БІ та БР: Azure Pipelines в якості сервісу для автоматизації процесів збірки, тестування та розгортання коду; Azure Repos в якості хмарного сервісу для управління репозиторіями; Git. Azure Artifacts, в якості менеджера артефактів для управління залежностями та артефактами.

Інфраструктура. Azure підтримує підхід ІК за допомогою наступних інструментів: Azure Resource Manager (ARM) Templates в якості інструменту для управління ресурсами за допомогою шаблонів; Azure Bicep, в якості мови для написання інфраструктури у вигляді коду, яка є спрощеною версією ARM шаблонів; Terraform, в якості підтримки інфраструктури як коду для управління ресурсами.

Автоматизація та оркестрація. Інструменти для автоматизації та оркестрації: Azure Automation, в якості сервісу для автоматизації процесів управління та конфігурації інфраструктури; Azure Logic Apps в якості платформи для автоматизації робочих процесів.

Моніторинг та логування. Microsoft Azure надає розширені рішення для моніторингу та логування, що забезпечується завдяки використанню інструментів та сервісів: Azure Monitor, що використовується в якості комплексних рішень для моніторингу додатків та інфраструктури; Azure Log Analytics в якості сервісу для збору та аналізу логів; Azure Application Insights в якості інструменту для моніторингу продуктивності додатків.

Безпека. Azure забезпечує широкий набір інструментів для безпеки завдяки таким інструментам та сервісам: Azure Security Center, шо використовується в якості централізованої панелі для управління безпекою та виявлення загроз; Azure Active Directory (AD) в якості сервісу для управління ідентифікацією та доступом; Azure Key Vault, який є інструментом для управління секретами та ключами шифрування.

Потенційні проблеми. До потенційних проблем відносяться надійність та швидкість розгортання, а також масштабування та підтримка високого рівня надійності. Для вирішення цих потенційних проблем впроваджуються практики DevOps, а саме автоматизація БІ БР та моніторинг, а також використання ІК, автоматизації тестування, розгортання, що впливає на зменшення кількості інцидентів, підвищує швидкість розгортання, поліпшує загальний стан стабільності платформи, а також забезпечує надійність сервісів з можливістю швидкого масштабування.

Організації, які використовують Azure DevOps мають можливість підвищення ефективності шляхом скорочення термінів виходу на ринок й посиленням співпраці в рамках своїх проєктів з розробки програмного забезпечення. Тому успішне впровадження практик DevOps для хмарного середовище, що демонструє те, як організації та компанії можуть використовувати хмарні платформи для прискорення інновацій, покращення масштабованості та підвищення якості обслуговування клієнтів.

Висновки. Під час проведеного дослідження охарактеризовано основні компоненти інтеграції DevOps, культура, автоматизація, вимірювання та обмін, що поєднує в собі комплекс безперервних процесів – безперервна інтеграція, розгортання та доставка. Основні принципи, які містяться в концепції DevOps охоплює в собі принципи потоку, зворотного зв’язку та безперервного навчання.

Охарактеризовано основні складові концепції DevOps, які складаються з технічних (знання інфраструктури як коду, знання безперервної інтеграції, безперервної доставки та розгортання, а також знання в управлінні конфігурацією) та гнучких (аналітичні навички, навички ефективної комунікації та роботи в команді, а також здатність до самонавчання та адаптації до нових технологій) навичок, які поєднуються з глибоким розумінням життєвого циклу розробки програмного забезпечення. Знання та навички володінням БІ, БД та БР вимагають чіткого налаштування цих процесів, що впливає на

автоматизацію створення, тестування та випуск програмного забезпечення. Розглянуто основні робочі процеси безперервної доставки, безперервної інтеграції та безперервного розгортання, де було з'ясовано, що безперервна доставка розгортає код безперервно за допомогою ручних тригерів, в той час як безперервне розгортання автоматизує розгортання без втручання людини.

Розглянуто та проаналізовано основні аспекти безперервної інтеграції, що складається з автоматизованої збірки, модульного тестування, тестування забезпечення якості, генерації артефактів, циклу зворотного зв'язку, знань управління конфігурацією. На основі цих аспектів розглянуто характеристику основних процесів, які містить безперервна інтеграція, яка складається з побудови вихідного коду, самоперевірки, підготовки артефакту, забезпечення якості, перевірки безпеки та публікації артефакту.

Проаналізовано використання концепції DevOps великими IT-компаніями, такими як Netflix та Microsoft. Проаналізувавши обидва великі IT-проєкти можна зробити висновок, що концепція DevOps може використовуватися як для стрімінгових сервісів (Netflix), так і для хмарних платформ (Microsoft Azure), де було проаналізовано ідентичні параметри: архітектура, інтеграція, інфраструктура, автоматизація та оркестрація, моніторинг та логування, безпека. Після чого було проаналізовано потенційні виклики та проблеми, де ефективним рішенням може слугувати автоматизація та безперервні процеси з використанням мікросервісної архітектури, що дозволить скоротити час розгортання з місяців до хвилин, а також сприятиме підвищенню надійності та швидкості релізів.

Література:

1. Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., ... & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*, 217-230. Retrieved from https://doi.org/10.1016/j.infsof.2019.06.010 [in English],

2. Faustino, J., Adriano, D., Amaro, R., Pereira, R., & da Silva, M. M. (2022). DevOps benefits: A systematic literature review. *Software: Practice and Experience*, *52*(9), 1905-1926. https://doi.org/10.1002/spe.3096

3. Wiedemann, A., Wiesche, M., Gewald, H., & Krcmar, H. (2023). Integrating development and operations teams: A control approach for DevOps. *Information and Organization*, *33*(3), 1-17. https://doi.org/10.1016/j.infoandorg.2023.100474

4. Khan, M. S., Khan, A. W., & Khan, J. (2022). DevOps' Culture Challenges Model (DC2M): A Systematic Literature Review Protocol. *Evolving Software Processes: Trends and Future Directions*, 201-217. https://doi.org/10.1002/9781119821779.ch10

5. Ferdian, S., Kandaga, T., Widjaja, A., Toba, H., Joshua, R., & Narabel, J. (2021) Continuous Integration and Continuous Delivery Platform Development of Software Engineering and Software Project Management in Higher Education. *Jurnal Teknik Informatika dan Sistem Informasi p-ISSN*, *2443*, 2210. http://dx.doi.org/10.28932/jutisi.v7i1.3254

6. Datla, V. (2023). The Evolution of DevOps in the Cloud Era. *Journal of Computer Engineering and Technology (JCET)*, *6*(1), 7-12. https://iaeme.com/Home/issue/JCET?Volume=6&Issue=1

7. Vemuri, N., Thaneeru, N., & Tatikonda, V. M. (2024). AI-Optimized DevOps for Streamlined Cloud CI/CD. *International Journal of Innovative Science and Research Technology*, *9*(7), 504-510. http://dx.doi.org/10.5281/zenodo.10673085

8. Senapathi, M., Buchan, J., & Osman, H. (2018). DevOps capabilities, practices, and challenges: Insights from a case study. In *roceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018*, 57-67. https://doi.org/10.1145/3210459.3210465

9. Luz, W. P., Pinto, G., & Bonifácio, R. (2019). Adopting DevOps in the real world: A theory, a model, and a case study. *Journal of Systems and Software*, *157*, 1-16. https://doi.org/10.1016/j.jss.2019.07.083

10. Shahin, M., & Babar, M. A. (2020, June). On the role of software architecture in DevOps transformation: An industrial case study. In *Proceedings of the International Conference on Software and System Processes*, 175-184. Retrieved from https://doi.org/10.1145/3379177.3388891 [in English].

11. Agarwal, A., Gupta, S., & Choudhury, T. (2018). Continuous and integrated software development using DevOps. In *2018 International conference on advances in computing and communication engineering (ICACCE)*, 290-293. https://doi.org/10.1109/ICACCE.2018.8458052

12. Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., ... & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*, 217-230. https://doi.org/10.1016/j.infsof.2019.06.010

13. Bodnar, L., Bodnar, M., Shulakova, K., Vasylenko, O., Tsarov, R., & Siemens, E. (2024). Practical Experience in DevOps Implementation. In *Proceedings of the 12th International Conference on Applied Innovations in IT (ICAIIT)*. 12(1), 33-39. https://doi.org/10.25673/115639

References:

1. Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., ... & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*, 217-230. Retrieved from https://doi.org/10.1016/j.infsof.2019.06.010 [in English]

2. Faustino, J., Adriano, D., Amaro, R., Pereira, R., & da Silva, M. M. (2022). DevOps benefits: A systematic literature review. *Software: Practice and Experience*, *52*(9), 1905-1926. Retrieved from https://doi.org/10.1002/spe.3096 [in English]

3. Wiedemann, A., Wiesche, M., Gewald, H., & Krcmar, H. (2023). Integrating development and operations teams: A control approach for DevOps. *Information and Organization*, *33*(3), 1-17. Retrieved from https://doi.org/10.1016/j.infoandorg.2023.100474 [in English]

4. Khan, M. S., Khan, A. W., & Khan, J. (2022). DevOps' Culture Challenges Model (DC2M): A Systematic Literature Review Protocol. *Evolving Software Processes: Trends and Future Directions*, 201-217. Retrieved from https://doi.org/10.1002/9781119821779.ch10 [in English]

5. Ferdian, S., Kandaga, T., Widjaja, A., Toba, H., Joshua, R., & Narabel, J. (2021) Continuous Integration and Continuous Delivery Platform Development of Software Engineering and Software Project Management in Higher Education. *Jurnal Teknik Informatika dan Sistem Informasi p-ISSN*, *2443*, 2210. Retrieved from http://dx.doi.org/10.28932/jutisi.v7i1.3254 [in English]

6. Datla, V. (2023). The Evolution of DevOps in the Cloud Era. *Journal of Computer Engineering and Technology (JCET)*, *6*(1), 7-12. Retrieved from https://iaeme.com/Home/issue/JCET?Volume=6&Issue=1 [in English]

7. Vemuri, N., Thaneeru, N., & Tatikonda, V. M. (2024). AI-Optimized DevOps for Streamlined Cloud CI/CD. *International Journal of Innovative Science and Research Technology*, *9*(7), 504-510. http://dx.doi.org/10.5281/zenodo.10673085 [in English]

8. Senapathi, M., Buchan, J., & Osman, H. (2018). DevOps capabilities, practices, and challenges: Insights from a case study. In *roceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018*, 57-67. Retrieved from https://doi.org/10.1145/3210459.3210465 [in English]

9. Luz, W. P., Pinto, G., & Bonifácio, R. (2019). Adopting DevOps in the real world: A theory, a model, and a case study. *Journal of Systems and Software*, *157*, 1-16. Retrieved from https://doi.org/10.1016/j.jss.2019.07.083 [in English]

10. Shahin, M., & Babar, M. A. (2020, June). On the role of software architecture in DevOps transformation: An industrial case study. In *Proceedings of the International Conference on Software and System Processes*, 175-184. Retrieved from https://doi.org/10.1145/3379177.3388891 [in English]

11. Agarwal, A., Gupta, S., & Choudhury, T. (2018). Continuous and integrated software development using DevOps. In *2018 International conference on advances in computing and communication engineering (ICACCE)*, 290-293. Retrieved from https://doi.org/10.1109/ICACCE.2018.8458052 [in English]

12. Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., ... & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*, 217-230. Retrieved from https://doi.org/10.1016/j.infsof.2019.06.010 [in English]

13. Bodnar, L., Bodnar, M., Shulakova, K., Vasylenko, O., Tsarov, R., & Siemens, E. (2024). Practical Experience in DevOps Implementation. In *Proceedings of the 12th International Conference on Applied Innovations in IT (ICAIIT)*. 12(1), 33-39. Retrieved from https://doi.org/10.25673/115639 [in English].