УДК 004.7 **doi: 10.26906/SUNZ.2024.1.058**

В. Д. Дюльгер, А. Р. Сорокін

Харківський національний університет радіоелектроніки, Харків, Україна

АНАЛІЗ МЕТОДІВ ІНТЕГРАЦІЇ ТА УЗГОДЖЕННЯ МІКРОСЕРВІСІВ В ХМАРНІЙ АРХІТЕКТУРІ

Анотація. Метою даної роботи є проведення аналізу сучасних методів інтеграції мікросервісів в хмарній архітектурі. Розділи роботи включають вивчення сучасного ландшафту хмарних архітектур та мікросервісів, їхню сутність та переваги в контексті хмарних середовищ. Детальний аналіз викликів та особливостей інтеграції мікросервісів включає розділи про контейнеризацію та оркестрацію, API-взаємодію та гібридні рішення, а також мікросервісну шину. Робота визначає роль та функціонал кожного методу, розкриває їхні переваги та виклики, що допомагає розуміти оптимальні стратегії інтеграції для розробників та організацій у сучасному інформаційному просторі.

Ключові слова: хмарні технології, мікросервісна архітектура, контейнеризація, serverless.

Вступ

В сучасному інформаційному суспільстві, характеризованому стрімким розвитком технологій, хмарні архітектури та мікросервіси визначають новий стандарт в галузі розробки програмного забезпечення. Розповсюдження хмарних рішень та використання мікросервісної архітектури відкривають безліч можливостей для оптимізації та розширення функціональності програмних продуктів. Зокрема, важливим аспектом стає аналіз методів інтеграції та узгодження мікросервісів в хмарній архітектурі. Спрямовані на взаємодію компоненти системи вимагають особливої уваги у зв'язку з різноманітністю сервісів, їхньою динамічністю та необхідністю забезпечення надійності та ефективності функціонування. Дослідження вказаної теми відкриває можливість вдосконалення стратегій розгортання мікросервісів у хмарних середовищах та сприяє оптимізації їхньої взаємодії для досягнення максимальної ефективності в різноманітних інформаційних проектах.

Мета статті – проведення аналізу методів інтеграції та узгодження мікросервісів в хмарній архітектурі та огляд перспектив. Дослідження спрямоване на виявлення оптимальних стратегій розгортання мікросервісів, їхньої ефективної взаємодії та забезпечення надійності в хмарних середовищах. Результати дослідження можуть внести вагому вклад у покращення розробки та управління програмним забезпеченням в контексті сучасних технологічних тенденцій.

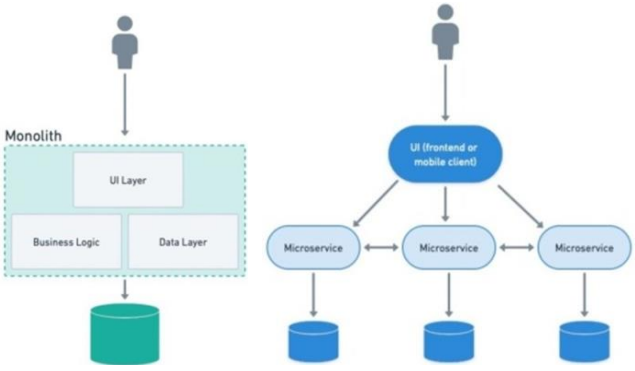
Результати досліджень

1. Сучасний ландшафт хмарних архітектур та мікросервісів. Хмарні архітектури базуються на принципах, таких як віртуалізація ресурсів, самообслуговування, широкий доступ та масштабованість. Вони надають користувачам можливість використовувати ресурси за вимогою, оптимізуючи використання обчислювальних потужностей та забезпечуючи гнучкість [2].

Публічні хмарні платформи, такі як AWS, Azure та Google Cloud, пропонують широкий функціонал для розгортання та управління ресурсами. Приватні хмари, в свою чергу, надають більший контроль та безпеку. Аналіз переваг кожного типу платформи допомагає підібрати оптимальний підхід для конкретного проекту.

Хмарні сервіси стають важливою складовою при створенні та розгортанні програмного забезпечення. Вони дозволяють зменшити час розробки, спростити процеси тестування та забезпечити високу доступність додатків. В свою чергу, мікросервісна архітектура розглядає додаток як сукупність невеликих, незалежних сервісів. Її переваги включають легку масштабованість, незалежність компонентів та покращену ефективність розробки.

Порівняння мікросервісів з монолітними рішеннями (рис. 1) вказує на переваги, такі як легка масштабованість окремих компонентів та спрощення розвитку, що робить мікросервіси більш гнучкими та підходящими для сучасних вимог ринку.



а б

Рис. 1. Монолітна (а) та мікросервісна (б) архітектура розгортання програмного забезпечення

Мікросервіси дозволяють декомпозицію системи на невеликі фрагменти, що полегшує розвиток та управління окремими компонентами. Це призводить до збільшення гнучкості та масштабованості всієї системи [4].

2. Сутність мікросервісної архітектури полягає в її гнучкості – здатності змінювати та адаптуватися до нових умов та вимог. Кожен мікросервіс може бути розгляданий як окрема одиниця, що може бути легко масштабована горизонтально для забезпечення оптимальної продуктивності. Мікросервісна архітектура надає високий рівень незалежності від конкретних хмарних інфраструктур. Це дозволяє

 © Дюльгер В. Д., Сорокін А. Р., 2024

ефективно використовувати можливості різних хмарних сервісів без значних модифікацій коду мікросервісів [5]. В хмарних середовищах мікросервіси можуть використовувати еластичність для автоматичного масштабування ресурсів в залежності від завдань та навантаження. Це сприяє оптимізації використання ресурсів та підвищує ефективність систем. Хмарні платформи надають інструменти для керування та моніторингу мікросервісами, що спрощує розгортання, керування життєвим циклом та підтримку цієї архітектури. Це робить хмарні середовища ідеальними для впровадження мікросервісної архітектури.

3. Виклики та особливості інтеграції мікросервісів в хмарній архітектурі В умовах хмарної архітектури важливо вирішити завдання забезпечення ефективної та безперебійної комунікації між розподіленими мікросервісами. Перехід до хмари може викликати проблеми зі швидкістю та надійністю мережевого обміну.

Однією з ключових особливостей є впровадження ефективних засобів безпеки для захисту мікросервісів в хмарних середовищах. Це включає управління доступом, шифрування та моніторинг безпеки [1]. Необхідно узгоджувати інтерфейси та протоколи між мікросервісами для ефективної взаємодії в хмарних умовах, де різноманітні технології можуть викликати проблеми сумісності.

Забезпечення ефективного моніторингу та управління ресурсами мікросервісів в хмарі важливо для забезпечення стабільності та високої доступності системи [1]. Інтеграція мікросервісів в хмарній архітектурі вимагає розробки гнучких та відмовостійких стратегій, щоб ефективно реагувати на зміни в середовищі та вирішувати можливі проблеми.

4. Аналіз сучасних методів інтеграції мікросервісів в хмарній архітектурі Контейнеризація та оркестрація стали ключовими компонентами для інтеграції мікросервісів у хмарних середовищах [3].

Використання контейнерів, таких як Docker, дозволяє ізолювати мікросервіси та їх залежності, забезпечуючи консистентність середовищ тестування та розгортання. Інструменти оркестрації, такі як Kubernetes, надають можливість автоматизованого керування та масштабування контейнеризованих мікросервісів. Це полегшує управління та підтримку великої кількості сервісів у хмарних обчислювальних середовищах.

Переваги:

- Ізоляція та портативність — контейнеризація дозволяє ізолювати мікросервіси, забезпечуючи консистентність середовища в різних стадіях розробки;

- Спрощення розгортання та масштабування — ркестраційні інструменти, такі як Kubernetes, автоматизують процеси розгортання та масштабування, що покращує ефективність управління.

Недоліки:

- Складність конфігурації — налаштування контейнерів та їхніх параметрів може виявитися складним завданням, особливо для великих систем;

- Велика кількість компонентів — використання оркестраторів може призвести до значного збільшення кількості компонентів у системі, що може збільшити складність управління.

API-взаємодія є важливим аспектом для успішної інтеграції мікросервісів в хмарній архітектурі.

Стандартизовані та добре задокументовані API дозволяють ефективно обмінюватися даними та взаємодіяти між мікросервісами. Використання REST або GraphQL дозволяє підтримувати гнучкі та ефективні точки доступу [3]. Інтеграція гібридних рішень дозволяє комбінувати використання власних та зовнішніх мікросервісів. Це дає можливість підтримувати специфічні функції власного бізнесу та одночасно використовувати зовнішні сервіси для покращення функціональності.

Переваги:

- Спрощена взаємодія — Використання API дозволяє просто та ефективно забезпечити взаємодію між мікросервісами, знижуючи залежності;

- Гнучкість — Гібридні рішення дозволяють комбінувати внутрішні та хмарні ресурси, забезпечуючи гнучкість і резервування.

Недоліки:

- Проблеми безпеки — забезпечення безпеки при обміні даними через API може бути складною задачею, особливо при великій кількості зовнішніх взаємодій;

- Можливість виникнення залежностей — гібридні рішення можуть призводити до виникнення складних залежностей між внутрішніми та зовнішніми компонентами.

5. Мікросервісна шина (Microservices Bus). Мікросервісна шина є іншим ефективним методом інтеграції мікросервісів у хмарних архітектурах. Шина дозволяє забезпечити взаємодію між мікросервісами, служить посередником для обміну повідомленнями та забезпечує асинхронну комунікацію між компонентами системи [4]. Застосування шини полегшує розширення функціональності та додає гнучкості у систему, а також спрощує впровадження змін у окремих мікросервісах. Аналіз та вибір оптимального методу інтеграції залежить від конкретних потреб проекту, розміру системи та вимог до її ефективності та масштабованості.

Переваги:

- Централізоване керування — мікросервісна шина дозволяє централізовано керувати взаємодією мікросервісів та зменшити зв'язок між ними;

- Спрощення розширення — додавання нових сервісів може бути спрощеним завданням, оскільки вони можуть взаємодіяти через шину.

Недоліки:

- Однопоточність — використання централізованої шини може призвести до однопоточності, де збільшення навантаження може стати обмежуючим фактором;

- Потенційний одиничний пункт відмови — централізована шина може стати одиничним пунктом відмови, якщо не враховані відповідні механізми для врегулювання цього ризику. Цей аналіз допомагає визначити оптимальний метод інтеграції мікросервісів, враховуючи конкретні потреби та обмеження конкретного проекту чи організації.

6. Узгодження мікросервісів у хмарному середовищі. У хмарній архітектурі, де мікросервіси можуть бути розташовані в різних місцях, забезпечення синхронізації даних стає критичним завданням. Розгляд методів для обміну та оновлення даних між мікросервісами, таких як подійно-орієнтована архітектура чи використання асинхронних повідомлень, є важливим для забезпечення консистентності [6]. Аналіз концепції CAP (Consistency, Availability, Partition tolerance, рис. 2) допомагає визначити, як досягти балансу між доступністю, консистентністю та стійкістю при роботі з розподіленими даними. Використання методів реплікації та версіонування даних може покращити консистентність в мікросервісній архітектурі [6].

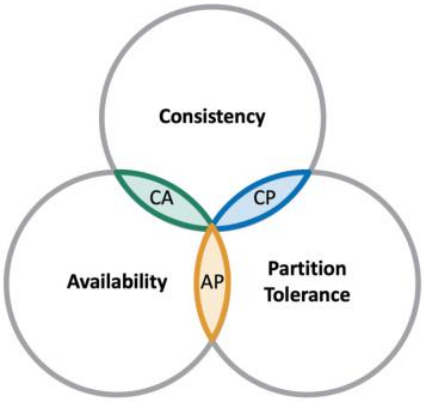


Рис. 2. Концепція Consistency, Availability, Partition tolerance

7. Перспективи розвитку та майбутні тренди. Хмарна архітектура та мікросервіси є динамічним полем розвитку, і майбутні тренди обіцяють захоплюючі перспективи для подальшого вдосконалення інтеграції мікросервісів у хмарі. Деякі ключові аспекти, які можуть визначати майбутній шлях розвитку, включають:

- розширення обчислювальних можливостей на "краї" мережі (Edge) та в туманних обчисленнях (Fog) стає важливим трендом. Це дозволяє обробку даних ближче до їх джерела, покращуючи продуктивність та знижуючи затримки [7];

- розвиток технологій контейнеризації, таких як Kubernetes, та Serverless архітектур дозволяє ефективно використовувати ресурси, спрощує розгортання та забезпечує гнучкість роботи мікросервісів;

- тренд до більшої декомпозиції на рівні функціональності, коли мікросервіси стають ще меншими та більш спеціалізованими, допомагає забезпечити гнучкість, але вимагає удосконаленої системи управління та моніторингу;

- інтеграція штучного інтелекту та машинного навчання в мікросервісні архітектури дозволяє створювати інтелектуальні та автономні системи, які можуть адаптуватися до змін у реальному часі [5];

- підвищення безпеки включає в себе вдосконалення методів автентифікації, контролю доступу та захисту даних у розподілених середовищах, щоб запобігти загрозам кібербезпеки.

Майбутнє інтеграції мікросервісів у хмарній архітектурі обіцяє новаторські рішення та розвиток технологій для підтримки високоефективних та гнучких систем. Вивчення цих тенденцій може допомогти готувати архітекторів та розробників до майбутніх викликів і можливостей у цьому еволюційному області.

Висновки

В статті розглянуто використання мікросервісної архітектури в поєднанні з хмарними технологіями, що відкриває нові горизонти для розвитку програмного забезпечення та обслуговування сучасних бізнес-процесів. Ці підходи дозволяють підприємствам бути гнучкими, швидкореагуючими та конкурентоспроможними в умовах постійних змін технологічного середовища та ринкових вимог.

СПИСОК ЛІТЕРАТУРИ

1. Newman, S. (2015). "Building Microservices: Designing Fine-Grained Systems." O'Reilly Media.
2. Lewis, J., & Fowler, M. (2014). "Microservices: a definition of this new architectural term." https://martinfowler.com/articles/microservices.html
3. Richardson, C. (2018). "Microservices Patterns: With Examples in Java." Manning Publications.
4. Wiggins, A. (2016). "Microservices in Action." Manning Publications.
5. Dragoni, N. et al. (2017). "Microservices: yesterday, today, and tomorrow." Communications of the ACM, 60(6), 85-93.
6. Armbrust, M., Stoica, I., Zaharia, M., Fox, A., Griffith, R., Joseph, A. D., ... & Zaharia, M. (2010). "A view of cloud computing." Communications of the ACM, 53(4), 50-58.
7. Dikaiakos, M. D., Katsaros, D., Mehra, P., Pallis, G., & Vakali, A. (2009). "Cloud computing: Distributed internet computing for IT and scientific research." Journal of Internet Services and Applications, 1(1), 7-18.

Received (Надійшла) 22.11.2023
Accepted for publication (Прийнята до друку) 24.01.2024

Analysis of methods for integrating and coordinating microservices in cloud architecture

V. Diulher, A. Sorokin

Abstract. The purpose of this work is to analyze modern methods of integrating microservices into cloud architectures. Sections of the paper include the study of the modern landscape of cloud architectures and microservices, their essence and advantages in the context of cloud environments. A detailed analysis of the challenges and features of microservices integration includes sections on containerization and orchestration, API interaction and hybrid solutions, and the microservice bus. The work defines the role and functionality of each method, reveals their advantages and challenges, which helps to understand the optimal integration strategies for developers and organizations in the modern information space.

Keywords: cloud technologies, microservice architecture, containerization, serverless.