DOI: 10.34185/1991-7848.itmm.2024.01.029

ПОРІВНЯННЯ МОЖЛИВОСТЕЙ СЕРВІСНО-ОРІЄНТОВНОЇ АРХІТЕКТУРИ ТА МІКРОСЕРВІСНОЮ АРХІТЕКТУРОЮ У СТВОРЕННІ ІНФОРМАЦІЙНИХ СИСТЕМ

Молодець Б.В., Булана Т.М.

Анотація: *Робота присвячена аналізу можливостей архітектур таких як: сервісно-орієнтованої архітектури та мікросервісна архітектура. В рамках роботи описані переваги та недоліки цих архітектур, наведені можливі схеми реалізації та відмінност,. В якості провайдера хмарних послуг було обрано Amazon Web Service. У результаті аналізу було отримані рекомендації коли доречно використовувати той чи інший підхід. Мікросервіси дозволяють гнучко масштабувати систему, додаючи або видаляючи окремі сервіси в залежності від потреб. Сервісно-орієнтована архітектура сприяє високій модульності та зручному взаємодії між компонентами системи. Враховуючи швидко зростаючі технології та вимоги до інформаційних систем, використання мікросервісної та сервісно-орієнтованої архітектур дозволить створити динамічну та ефективну систему, здатну швидко адаптуватися до змінних умов та вимог користувачів.*

Ключові слова: *інформаційна система, моніторинг якості повітря, архітектура програмного забезпечення, SOA, MSA*

Вибір архітектури програмного забезпечення є ключовим етапом у процесі розробки будь-якого проекту. Коректна обрана архітектура визначає ефективність, гнучкість та масштабованість системи.

Мікросервісна архітектура - це підхід до розробки програмного забезпечення, де програмний додаток розбивається на невеликі, самостійні модулі, відомі як мікросервіси. Кожен мікросервіс виконує конкретну функцію та має власний набір інтерфейсів API для взаємодії з іншими мікросервісами. Мікросервіси взаємодіють між собою через API, що дозволяє їм співпрацювати та обмінюватися даними [1].

Для розгортання мікросервісної архітектури на платформі Amazon Web Services можна скористатися наступним: сервісами контеризації Amazon Elastic Compute Cloud (EC2) або Amazon Elastic Kubernetes Service (EKS) , які дозволяють запускати та керувати контейнерами для ваших мікросервісів, Amazon Simple Storage Service (S3) в якості сховища для зберігання об’єктів , такі як файли, зображення та інше, Amazon CloudFront в якості швидкому та

ефективному доставці вмісту веб-застосунку, Amazon ALB для балансування навантаження, Amazon DynamoDB та Amazon Aurora як нереляційну та реляційну бази даних відповідно та Amazon ElastiCache для підвищеня продуктивності та масштабованість застосунку шляхом кешування даних в оперативній пам'яті та забезпечуючи швидкий доступ[2].

До переваг мікросервісної архітектури можна віднести наступне[3]:

- Гнучкість: Мікросервісна архітектура дозволяє розділити додаток на невеликі, незалежні сервіси, що надає більшу гнучкість у розробці, тестуванні та впровадженні нових функцій;
- Швидкість розгортання: Можливість незалежного розгортання кожного сервісу дозволяє прискорити процес впровадження змін та оновлень.
- Масштабованість: Кожен сервіс може бути масштабований незалежно, що дозволяє оптимізувати ресурси та витрати.

До недоліків мікросервісної архітектури можна віднести наступні пункти:

- З більшою кількістю сервісів у системі стає складніше керувати конфігураціями та налаштуваннями.
- З більшою кількістю сервісів також ускладнюється моніторинг їх роботи та відлагодження випадків неполадок.
- Більше сервісів означає більше залежностей між ними.
- Інтеграція між сервісами може вимагати значних зусиль у вирішенні питань сумісності, забезпеченні безпеки та забезпеченні спільного управління.
- Зменшується швидкодія системи через накладні розходи на мережеві виклики між сервісами (в тому числі при обміні даними).

Альтернативним рішенням, яке було обрано для розробки інформаційної системи, стала сервісно-орієнтовна архітектура (SOA).

Основна ідея SOA полягає в тому, щоб розділити функціональність додатків на окремі сервіси, які можуть бути реалізовані та підтримувані незалежно один від одного [4].

У сервісно-орієнтовній архітектурі функціональність додатків розбивається на окремі сервіси, які можуть бути незалежними один від одного. Завдяки розділенню функціональності на окремі сервіси, сервісно-орієнтовна архітектура дозволяє краще масштабувати систему, оскільки можна масштабувати тільки ті сервіси, які потребують додаткових ресурсів [5].

Основними відмінностями сервісно-орієнтовна архітектура від мікросервісної архітектури є наступні пункти:

- Сервісно-орієнтовна архітектура зазвичай створює більш великі та важкі сервіси, які можуть містити різноманітну функціональність; Мікросервісна архітектура,

навпаки, розглядає функціональність як набір дрібних, відокремлених сервісів, які керуються принципом "один сервіс - одна функція".

- У сервісно-орієнтовній архітектурі, великі сервіси можуть мати складні залежності один від одного, що може робити їх важко масштабувати та підтримувати. У мікросервісній архітектурі, кожен сервіс незалежний і може бути масштабованим та підтримуваним окремо.
- У сервісно-орієнтовній архітектурі сервіси можуть використовувати різні протоколи та стандарти для взаємодії між собою. У мікросервісній архітектурі, зазвичай використовуються легкі та стандартні протоколи, такі як HTTP або REST, для взаємодії між сервісами.
- У сервісно-орієнтовній архітектурі керування багатьма великими сервісами може бути складним завданням. У мікросервісній архітектурі, хоча кількість сервісів може бути більшою, кожен сервіс менший і простіший у розгортанні та управлінні.

У контексті розробки системи моніторингу якості повітря, мікросервісна та сервісно-орієнтована архітектура відіграють значущу роль. Мікросервіси дозволяють гнучко масштабувати систему, додаючи або видаляючи окремі сервіси в залежності від потреб. Сервісно-орієнтована архітектура сприяє високій модульності та зручному взаємодії між компонентами системи. Враховуючи швидко зростаючі технології та вимоги до систем моніторингу повітря, використання мікросервісної та сервісно-орієнтованої архітектур дозволить створити динамічну та ефективну систему, здатну швидко адаптуватися до змінних умов та вимог користувачів.

ЛІТЕРАТУРА / REFERENCES

1. Dragoni, N. et al. (2017). Microservices: Yesterday, Today, and Tomorrow. In: Mazzara, M., Meyer, B. (eds) Present and Ulterior Software Engineering. Springer, Cham. DOI: 10.1007/978-3-319-67425-4_12

2. Villamizar M., Garcés O., Ochoa L., Castro H., Salamanca L., Verano M. M., Casallas R., Gil S., Valencia C., Zambrano A., Lang M. (2016). Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures. DOI: 10.1109/CCGrid.2016.37.

3. Simple microservices architecture on AWS - Implementing Microservices on AWS (amazon.com) [Електронний ресурс] – Режим доступу до ресурсу: https://docs.aws.amazon.com/whitepapers/latest/microservices-on-aws/simple-microservices-architecture-on-aws.html

4. What is SOA (Service-Oriented Architecture)? [Електронний ресурс] – Режим доступу до ресурсу: https://aws.amazon.com/what-is/service-oriented-architecture/
5. Lewis G., Smith D., Chapin N., Kontogiannis K. (2010). Proceedings of the Fourth International Workshop on a Research Agenda for Maintenance and Evolution of Service-Oriented Systems (MESOA 2010). 107.

Comparison of the capabilities of service-oriented architecture and microservice architecture in the creation of information systems

Molodets Bohdan, Bulana Tetiana

Abstract: *The paper analyzes the capabilities of such architectures as service-oriented architecture and microservice architecture. The paper describes the advantages and disadvantages of these architectures, provides possible implementation schemes and differences. Amazon Web Service was chosen as a cloud service provider. As a result of the analysis, recommendations have been made on when it is appropriate to use one or the other approach. Microservices allow for flexible scaling of the system by adding or removing individual services depending on the needs. Service-oriented architecture promotes high modularity and convenient interaction between system components. In view of the rapidly growing technologies and requirements for information systems, the usage of microservice and service-oriented architectures will allow to create a dynamic and efficient system that can quickly adapt to changing conditions and user requirements.*

Keywords: *information system, air qualitymonitoring , software architecture, SOA, MSA*