DOI: 10.58168/CISMP2024_501-506
УДК 004.87

МИКРОСЕРВИСЫ: ПОДХОД К ПОСТРОЕНИЮ МАСШТАБИРУЕМЫХ СИСТЕМ

Д.В. Арапов, В.Р. Алексеев
vlad_alekseev696@mail.ru

ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

Аннотация. В данной работе рассматривается микросервисная архитектура как современный подход к разработке приложений. Описываются ключевые принципы – это параллельная разработка разработчиков, использование Docker и Kubernetes и независимость микросервисов. Выделяет преимущества: гибкость и устойчивость к сбоям. Сравнение двух архитектур, когда и зачем переходить на микросервисы.

Ключевые слова: микросервис, архитектура, монолит, Docker, Kubernates, приложение.

MICROSERVICES: AN APPROACH TO BUILDING SCALABLE SYSTEMS

D.V. Arapov, V.R. Alekseev
vlad_alekseev696@mail.ru

Voronezh State University of Forestry and Technologies named after G.F. Morozov

Abstract. This paper examines microservice architecture as a modern approach to application development. The key principles described are parallel development by developers, the use of Docker and Kubernetes, and the independence of microservices. Highlights advantages: flexibility and resilience to failures. Comparison of two architectures, when and why to switch to microservices.

Keywords: microservice, architecture, monolith, Docker, Kubernates, application.

Введение

Все чаще для разработки приложения выбирают микросервисную архитектуру. Компании хотят более гибкие и масштабируемы приложения, что микросервисы могут как раз предоставить. Из-за таких преимуществ как раз и выбирают микросервисы. Если компании важно быстро реагировать на изменения и внедрять новые функции, поэтому микросервисы открывают множество возможностей для создания.

Микросервисная архитектура

Микросервисная архитектура или микросервисы – это архитектурный подход к разработке, где приложение разделяется на небольшие сервисы, которые отвечают только за свой блок и может разрабатываться независимо от

других сервисов. Такая архитектура позволяет брать большое монолитное приложение и разделить его на маленькие управляемые сервисы.

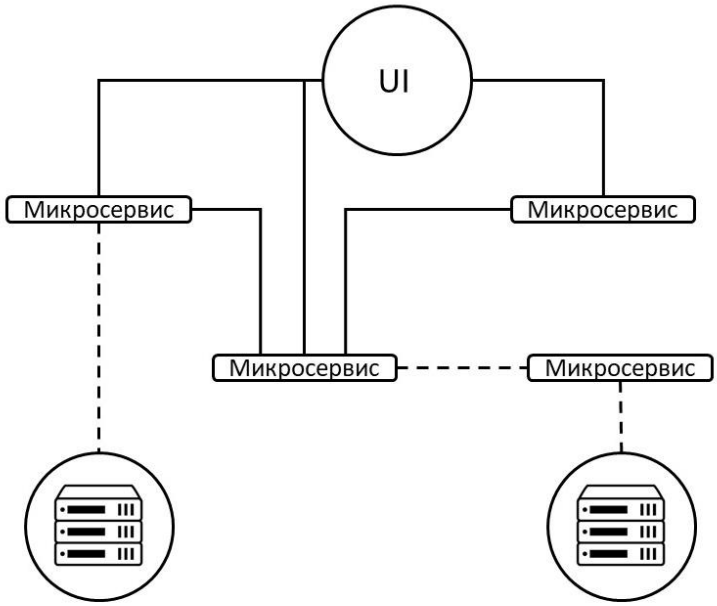

Рисунок 1 – Микросервисная архитектура

Преимущество микросервисов

Микросервисы имеют множество преимуществ, которые упрощают разработку. К этим преимуществам относятся:

1. Масштабируемость – микросервисы можно масштабировать независимо от других, что позволяет распределить ресурсы туда, где больше в них нуждаются;
2. Гибкость – каждый микросервис может быть разработан на разных языках программирования, что позволяет выбирать оптимальные технологии для каждой задачи. Например, для высокопроизводительных частей можно использовать C++, а для остальных — Java;
3. Устойчивость к сбоям – так как сервисы практически не связаны между собой, если даже один из сервисов выйдет из строя – это не повлияет на всю систему. Такой сбой можно исправить, не затрагивая остальные компоненты;
4. Распределенная разработка – благодаря микросервисам, разработка продукта может быть распределена между несколькими командами, каждая из которых будет работать над отдельным сервисом.

Основные компоненты микросервисов

1. Контейнеризация – используют Docker, он автоматизирует развертывание приложений в контейнерах, полезен тем, что в него можно упаковать микросервисы и развернуть их в независимых контейнерах;

2. Оркестрация контейнеров проходит с помощью системы Kubernetes, она предназначена для автоматизации развёртывания и управления контейнерными приложениями;

3. API-шлюз действует как центральная точка входа для клиентов, которая управляет запросами, аутентификацией и направляет запросы в соответствующий микросервис;

4. Кэширование, часто кэш хранит используемые данные рядом с микросервисом, что повышает производительность за счет сокращения повторяющихся запросов. Например, в приложении можно кэшировать результаты поиска товаров, чтобы, когда были повторные запросы к этому же товару, информация возвращалась быстрее, из-за того, что пропускаем обращение к базе данных;

5. Реестр и обнаружение сервисов, отслеживает местоположение и автоматически находит доступные микросервисы, что позволяет динамически управлять масштабированием системы.

Отличие монолитной от микросервисной архитектуры

Эти две архитектуры представляют собой разные подходы к разработке программного обеспечения. Конечно, микросервисная архитектура считается более современным подходом к созданию архитектуры, но и монолитная архитектура всё ещё остается актуальной. Например, если имеется небольшой проект, где ограниченный функционал и сложность управления низкая, это может подойди для стартапов, так как с помощью монолитной архитектуры можно намного быстрее выпустить продукт на рынок. Но если будет происходить рост и усложнение проекта, тогда уже можно планировать переход на микросервисную архитектуру для более сложного функционала.

Значимое отличие от микросервисной архитектуры – это то, что все компоненты объединены в единое целое, то есть интерфейс, доступ к базе данных и другие элементы, хранятся в одном сервисе.

Рисунок 2 – Монолитная архитектура

Когда стоит переходить на микросервисы

1. Рост команды разработчиков - когда число людей на проекте растёт, появляется потребность в параллельной разработке. В монолитной архитектуре, если разработчик захочет в одном из сервисов сделать изменение из-за такого придется останавливать всё приложение, что будет очень замедлять и затруднять разработку. Так как микросервисы расположены независимо друг от друга, это позволяет вносить изменения на определённые сервисы, что не влияет на работу всего приложения;
2. Проблемы с масштабированием – у каждого сервера приложения разные требование к масштабированию, то есть микросервисы могут расширять только сервисы, которые нуждаются в этом. Например, если в приложении логика работы с сервисом А нагружена намного больше, то можно масштабировать только микросервис, который отвечает за него;
3. Использования разных технологий – также может быть такая проблема, что одного языка программирования уже не хватит из-за каких-либо функций, но микросервисы позволяют использовать это;
4. Высокая сложность кода – со временем код становится слишком сложным и в монолитной архитектуре он может стать трудно поддерживаемым из-за этого осуществляется переход на микросервисы.

Заключение

Микросервисная архитектура – это мощный инструмент для создания современных приложений, который даёт разработчикам справляться с проблемами, которые есть в монолитной архитектуре. Однако переход на микросервисы, требуют больших навыков, но такая архитектура открывает новые возможности для разработки.

Список литературы

1. Habr. Микросервисы. – URL: https://habr.com/ru/articles/249183/ (дата обращения: 01.10.2024).

2. Полуэктов А.В., Макаренко Ф.В., Ягодкин А.С. Использование сторонних библиотек при написании программ для обработки статистических данных // Моделирование систем и процессов. – 2022. – Т. 15, № 2. – С. 33-41.

3. Atlassian.com. Микросервисы [понятие и основные преимущества]. – URL: https://www.atlassian.com/ru/microservices (дата обращения: 01.10.2024).

4. Кадровое агентство IT and Digital. Что такое микросервисы и как они работают? – URL: https://itanddigital.ru/microservice (дата обращения: 01.10.2024).

5. Skillbox.ru. О микросервисной архитектуре простыми словами. – URL: https://skillbox.ru/media/code/o-mikroservisnoy-arkhitekture-prostymi-slovami/ (дата обращения 02.10.2024).

6. Бакаев Д.Н., Стукало О.Г., Денисенко В.В., Скрыпников А.В., Савченко И.И., Зиновьева В.В. Информационный инструментарий проектного управления развитием промышленных предприятий // Моделирование систем и процессов. – 2022. – Т. 15, № 2. – С. 14-24.

7. Microservices.io. What are microservices? – URL: https://microservices.io/ (дата обращения 08.10.2024).

8. Wikipedia. Microservices. – URL: https://fr.wikipedia.org/wiki/Microservices (дата обращения 08.10.2024).

9. Rad Hat. Comprende ce que sont les microservices. – URL: https://www.redhat.com/fr/topics/microservices (дата обращения 10.10.2024).

10. Сазонова С.А., Епифанов Е.Н., Асминин В.Ф., Мозговой Н.В., Осипов А.А., Дружинина Е.В., Кораблин С.Н. Моделирование возможной обстановки при пожаре на объектах с массовым пребыванием людей // Моделирование систем и процессов. – 2022. – Т. 15, № 1. – С. 85-96.

11. Amazon Web Services. Que sont les microservices? | AWS – URL: https://aws.amazon.com/fr/microservices/ (дата обращения 11.10.2024).

References

1. Habr. Microservices. – URL: https://habr.com/ru/articles/249183/ (date of access: 01.10.2024).

2. Poluektov A.V., Makarenko F.V., Yagodkin A.S. Using third-party libraries when writing programs for processing statistical data // Modeling of systems and processes. – 2022. – T. 15, No. 2. – P. 33-41.

3. Atlassian.com. Microservices [concept and main advantages]. – URL: https://www.atlassian.com/ru/microservices (date of access: 01.10.2024).

4. IT and Digital recruitment agency. What are microservices and how do they work? – URL: https://itanddigital.ru/microservice (date of access: 01.10.2024).

5. Skillbox.ru. About microservice architecture in simple words. – URL: https://skillbox.ru/media/code/o-mikroservisnoy-arkhitekture-prostymi-slovami/ (date of access: 02.10.2024).

6. Bakaev D.N., Stukalo O.G., Denisenko V.V., Skrypnikov A.V., Savchenko I.I., Zinovyeva V.V. Information tools for project management of the development of industrial enterprises // Modeling of systems and processes. – 2022. – T. 15, No. 2. – P. 14-24.

7. Microservices.io. What are microservices? – URL: https://microservices.io/ (date of access: 08.10.2024).

8. Wikipedia. Microservices. – URL: https://fr.wikipedia.org/wiki/ Microservices (date of access: 08.10.2024).

9. Rad Hat. Comprende ce que sont les microservices. – URL: https://www.redhat.com/fr/topics/microservices (date of access: 10.10.2024).

10. Sazonova S.A., Epifanov E.N., Asminin V.F., Mozgovoy N.V., Osipov A.A., Druzhinina E.V., Korablin S.N. Modeling of a possible fire situation at facilities with large numbers of people // Modeling of systems and processes. – 2022. – T. 15, No. 1. – P. 85-96.

11. Amazon Web Services. Que sont les microservices? | AWS. – URL: https://aws.amazon.com/fr/microservices/ (date of access: 11.10.2024).