Microservices: An Architectural Approach to Software Development

CS4991 Capstone Report, 2022
Wen Ip
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
wenip3@gmail.com

Abstract
The recent implementation of a microservices architectural style for a financial advisor application at a large banking company had its advantages, as well as drawbacks. The implementation resulted in the potential for easy scalability and product development work that coincides more closely with business needs. However, implementing microservices also increased management complexity and posed a challenge for the standardization of microservices across teams. My team and I explored a more efficient strategy for managing and standardizing microservices to maximize the potential of microservices in applications. The drawbacks and advantages of microservices will be used to recommend potential implementation methods for microservices that will improve software development process while using this architecture and enhance the functionality and performance of microservice applications. Microservices demonstrated that they can be easily integrated with other applications without having to create redundant code; but also showed the need for standards across software development and microservice implementations.

1. Introduction

Microservices is a new approach to the way an application is structured and built. Software development techniques and processes have evolved to deliver high quality software solutions. These solutions have become more complex with larger architectural designs to expand functionality and scalability. The architecture style of building a software application determines its scalability, resilience, and business functionality. It is important to determine the architectural design of the software product to fit its needs and purpose.

2. Literature Review

According to O'Connor, et. al., (2017), microservice architecture is an emerging approach that revolves around the idea that self-contained components of functionality make up a larger system [1]. It allows for an application to be built upon a collection of individual services that can be arranged, organized, and maintained into multiple applications.

Unlu, et. al. (2022) [2]. posit that the potential of implementing this architecture transforms the way companies utilize cloud technology, as well. Microservices have been paired with the emerging popularity of cloud technology. With the use of automation on cloud platforms, microservices can be deployed and scaled quickly. Each microservice serves a business need or functionality, meaning an application can easily be altered by choosing to include a microservice or not.

Since microservices is an emerging approach, many companies have adopted this method, but there is not much research

about its common practices. Researching the advantages and disadvantages of microservices can help companies decide whether they want to implement microservices or further develop their approach to implementing microservices.

3. Process Design

I was assigned to create an application from scratch for my summer internship. As a software engineering intern, I learned software development techniques adopted by the company, developed the proposed application with a team, and presented a demo of the application to stakeholders.

3.1 Review of Project

Over the course of ten weeks, I worked on an application for a large banking company. The five-person development team included software engineering interns with a scrum master and product owner. The team worked in 2-week sprints using scrum methodologies.

The purpose of the application we developed is to enable financial advisors to give feedback on how single-page applications are meeting their business needs. The application uses microservice architecture and can be imported into different financial advisors' platforms as a button on the taskbar of the page.

3.2 Technical Aspects

The microservice used Spring Boot framework and layers. The database was built on MongoDB. The frontend side of the project used Angular, Typescript, HTML and CSS. The backend side of the project used Java. Other IDEs' used to create the project were Visual Studio Code, Postman, Gradle, and IntelliJ.

3.3 Challenges

The team was new to the framework and technologies used for this application, which created a learning curve for using the technologies and connecting the layers of the application.

Another challenge was that the integration can be different for various applications because of the way these applications are already set up. Grabbing URL links or advisor IDs may vary greatly between applications, which can cause confusion between engineering teams when projects are switched over to new teams.

3.4 Advantages

The implementation of this widget allows users to submit feedback on the application so engineers can better understand their users' needs. Using a microservice architecture for this widget allows engineers to customize the integration of the widget to their application. Microservice architecture also makes it easy to integrate the widget with different applications. Different applications have different ways of collecting information related to the feedback, such as page URLs and advisor IDs. Engineers can pass parameters to the microservice when it is imported into their application.

3.5 Disadvantages

Since microservices can be implemented into multiple applications, a concern would be how to develop, integrate and maintain these microservices. It can be difficult to integrate microservices into different applications if these existing applications are developed differently and teams must spend more time creating unique solutions for integration and maintenance of each of these applications. If a microservice architecture is used for an application, it would be important to create models and standards for how these microservices are implemented so that is easier to integrate and change the architecture to better mold to business needs.

4. Results

By the end of the ten weeks, the application had connected layers from frontend to the database and successfully implemented a microservice architecture. The team was also able to import the microservice into other applications locally, but not on the cloud. The team attempted to import the microservice through JFrog Artifactory, but due to the time constraints, was unable to get approvals and complete this task in time. Theoretically, the application should be able to function on its own and be integrated into other existing applications.

5. Conclusion

This project allows us to analyze the implementations of microservices for an application. One of the most important outcomes is that the microservice can be customizable to other applications if imported. This shows the potential of microservices to reduce redundancy and increase efficiency in creating applications built on microservices.

While microservices can be adapted to many different applications, complications in the maintenance of these applications could arise if certain standards are not set for microservice integration. When microservice architecture is implemented for application, it is important to focus on and research software development techniques and standards to minimize integration issues and maintenance time.

6. Future Work

If the application we designed is chosen for further work by another team, they should be able to continue their development work with the documentation notes from JIRA and Confluence created by my team. Future work for the application includes implementing and testing integration of the microservice using JFrog Artifactory. With complete integration through Artifactory, an analysis can be done on calls made by the microservice to see how it functions and to review the performance of the microservice component after integrating it with other applications.

Since the use of microservices are relatively new technology, it would be helpful to continue to research and discuss implementation of microservices to aid decisions to use this type of architecture.

7. UVA Evaluation

The CS program prepared me to use skills that allowed me to excel at my internships. The program provided a variety of software development skills, including programming different types of applications, understanding data structures, and software development techniques that are relevant in the current industry.

8. Acknowledgments

I would like to thank my professors, teaching assistants, and classmates for creating a safe learning environment where I could learn and collaborate. I also want to thank the team members and mentors from my internship for working with me over the summer and encouraging growth in my career.

References

[1] O'Connor, R., Elger, P. and Clarke, P., 2017. Continuous software engineering-A microservices architecture perspective. *Journal of Software: Evolution and Process*, 29(11), p.e1866.

[2] Unlu, H., Bilgin, B. and Demirors, O., 2022. A survey on organizational choices for microservice-based software architectures. *Turkish Journal of Electrical Engineering and Computer Sciences*, 30(4), pp.1187-1203.