February 07, 2024

Research Proposal

Master's Thesis in Technology (DTEK1002)

Florian Dejonckheere

fwdejo@utu.fi

Abstract

Working title: *A Framework for Writing Distributed Applications using the Modular Monolith Software Architecture*

The advent of cloud computing has fundamentally changed the way software is deployed and managed. Nowadays, developers define the software architecture in function of the underlying infrastructure, and often decompose the application into a set of smaller applications based on logical boundaries in the domain logic, which are deployed independently of each other.

This separation of concerns brings along a number of advantages such as scalability, fault isolation and independent deployment, over more traditional software development practices such as monolithic codebases. The adoption of a microservice-based architecture also introduces a number of challenges for the developers. Developers need to be aware of the distributed nature of the application, and need to take into account the additional complexity that comes with it. This affects in turn the development velocity and efficiency. The resulting software also increases in complexity, complicating maintenance and debugging.

In this thesis, we investigate the modular monolith software architecture, a hybrid model between monolithic and microservice-based architectures that aims to combine the best of both worlds. We analyze the pattern and its advantages and disadvantages, and compare it to its architectural predecessors. We identify the challenges and opportunities that arise when adopting the modular monolith architecture for dynamic languages compared to statically-typed languages. Finally, the architecture is implemented as a proof-of-concept, and evaluated in the context of a real-world use case. We discuss the results and draw conclusions about the effectiveness of the architecture.

Aims and goals

The thesis can be divided into two parts: a literary part and a more practical part. For the literary part, I first analyze the origins and characteristics of the modular monolith software architecture, and compare it to similar software architectures. Then, I investigate how the identification of module boundaries could be improved through using automated technologies. Finally, I design a solution in the form of a tool for developers that automatically identifies optimal module boundaries in a monolithic codebase.

For the practical part, I apply the proposed solution to a real-world use case, and evaluate how accurate and efficient it is, and how it impacts the development process. The use case I intend to use for the proof-of-concept is called NephroFlow Link, part of the NephroFlow Product Suite.

NephroFlow Link is a middleware application written in Ruby that acts as a data broker between connected dialysis machines and the NephroFlow cloud platform. It is currently designed as a monolithic application, and is deployed as a single process on a single server. However, the application is becoming increasingly complex which slows down the development velocity and increases the maintenance burden on the developers. Furthermore, the requirements in regard to scalability and extensibility are increasing rapidly, which leads me to believe the application would benefit from a modular monolith architecture. I intend to develop the proposed solution using NephroFlow Link as a case study.

Research questions

Research Question 1: What is the modular monolith architecture, and what sets it apart from monolithic and microservices architectures?

Motivation: Explore why the modular monolith architecture is becoming increasingly popular, and why it is considered a viable alternative to monolithic and microservices architectures.

Research Question 2: Which challenges and opportunities arise when considering adoption of the modular monolith architecture for an existing codebase?

Motivation: Investigate how restructuring an existing codebase as a modular monolith would impact the development process and resulting product.

Research Question 3: How can automated techniques effectively identify optimal module boundaries in a modular monolith architecture?

Motivation: Investigate technologies based on dependency analysis and semantic clustering to automatically determine optimal module boundaries in a modular monolith architecture. Compare to manual modularization efforts in terms of accuracy, efficiency, development velocity.

Methodology

Based on the principles of the modular monolith architecture, module boundaries are manually identified within the application. This is done by experienced developers that have an intimate knowledge of both the codebase and the application domain. Next, using the proposed solution, module boundaries are automatically identified within the application.

To evaluate the effectiveness of the proposed solution, key criteria are identified:

1. **Accuracy**: how close do the module boundaries of the automated techniques align with the manually identified ones? The degree of alignment is measured using metrics such as Jaccard similarity.
2. **Modularity**: how modular are the boundaries identified by the automated techniques? The degree of modularity is measured using metrics such as the modularity Q, the clustering coefficient, and the Fenton and Melton metric.
3. **Efficiency**: how much effort is required to implement the automatically identified module boundaries compared to manual modularization? Feedback from experienced developers is gathered using surveys and interviews.
4. **Adaptibility**: how well do the modules react to changes in the codebase and evolving requirements? The impact is measured empirically by introducing changes in the environment or requirements, and observing how well the boundaries adapt to the change using metrics such as code churn and defect density.

By measuring these key criteria quantitatively and qualitatively for the module boundaries of the automated and the manual modularization, I can assess the effectiveness of the automated techniques in a thorough and objective manner.

Contents

References

Abgaz, Y., Mccarren, A., Elger, P., Solan, D., Lapuz, N., Bivol, M., Jackson, G., Yilmaz, M., Buckley Jim, & Clarke, P. *Decomposition of Monolith Applications Into Microservices Architectures: A Systematic Review: Vol. PP.*

Almeida, J., & Silva, A. R. *Monolith Migration Complexity Tuning Through the Application of Microservices Patterns.*

Alshuqayran, N., Ali, N., & Evans, R. *A Systematic Mapping Study in Microservice Architecture.*

Anand, V., Garg, D., Kaufmann, A., & Mace, J. *Blueprint: A Toolchain for Highly-Reconfigurable Microservice Applications.* Association for Computing Machinery.

Andrade, B., Santos, S., & Silva, A. R. *From Monolith to Microservices: Static and Dynamic Analysis Comparison.*

Bacchiani, L., Bravetti, M., Giallorenzo, S., Mauro Jacopo, Talevi, I., & Zavattaro, G. *Microservice Dynamic Architecture-Level Deployment Orchestration* (Vol. LNCS-12717). Springer International Publishing.

Barde, K. *Modular Monoliths: Revolutionizing Software Architecture for Efficient Payment Systems in Fintech* (Vol. 71).

Ghemawat, S., Grandl, R., Petrovic, S., Whittaker, M., Patel, P., Posva, I., & Vahdat, A. *Towards Modern Development of Cloud Applications.* Association for Computing Machinery.

Gonçalves, N., Faustino, D., Silva, A. R., & Portela Manuel. *Monolith Modularization Towards Microservices: Refactoring and Performance Trade-offs.*

Jin, W., Liu, T., Cai, Y., Kazman, R., Mo, R., & Zheng, Q. *Service Candidate Identification from Monolithic Systems Based on Execution Traces* (Vol. 47).

Kendall, S. C., Waldo, J., Wollrath, A., & Wyant, G. *A Note on Distributed Computing*. Sun Microsystems, Inc.

Ruoyu Su, & Xiaozhou Li. *Modular Monolith: Is This the Trend in Software Architecture?.*

Villamizar, M., Garcés, O., Castro, H., Verano, M., Salamanca, L., Casallas, R., & Gil, S. *Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud.*

Wolfart, D., Assunção, W. K. G., Silva, I. F. da, Domingos, D. C. P., Schmeing, E., Villaca, G. L. D., & Paza, D. d. N. *Modernizing Legacy Systems with Microservices: A Roadmap.* Association for Computing Machinery.