FAKULTÄT FÜR INFORMATIK

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Refactoring an Orchestration Microservice for Simplicity of Control Flow Definition and Customizability.

Rajendra Kharbuja

FAKULTÄT FÜR INFORMATIK

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Refactoring an Orchestration Microservice for Simplicity of Control Flow Definition and Customizability.

Refactoring einer Orchestrierungsmicroservice zur Vereinfachung der Kontrollfluss Definition und Anpassbarkeit.

Author:	Rajendra Kharbuja
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	Dr. Holger Wittges
Submission Date:	

I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, Rajendra Kharbuja

Acknowledgments

Abstract

Contents

1 Introduction

1.1 Motivation

Software Architecture definition is an inevitable and crucial step for software development process. By defining an architecture, we define elements, their behavior, their structural composition to construct a system as a whole and various interfaces to characterize communication required between the elements. [Mic] The objective of building an architecture is not only to provide the desired functionalities but also to calibrate the quality of the software across various non-functional attributes such as availability, usability, performance etc. Modifiability is one among the various other quality attributes controlled by an architecture. [Len03]

Software development process as well as development pattern play an equal role to define quality attributes. The quality attributes can be broadly classified as a)Structural and b)Process quality attributes. The structural quality attributes such as testability, modifiability, security etc are affected by the development pattern followed by developers. Adoption of a specific design pattern to solve a problem may be one example of a significant development approach. Similarly, the quality attributes are affected differently, based on the development methodology used. For example, an agile methodology will help to improve maintainability due to its approach of getting fast customer response.[Chanp]

Software change is unavoidable for many reasons. The changes are driven by difference between the software's ability and expectations along time. Additionally, the focus on growing the software require it to undergo changes. Another reason can be to nullify the technical debt occurred along the development stages due to various compromising decisions made.[Byr09] A research carried out by Agile Manifesto, whose result is shown by figure1.1 suggests that a surprisingly higher percentage of projects actually fail to meet the requirements of the user.

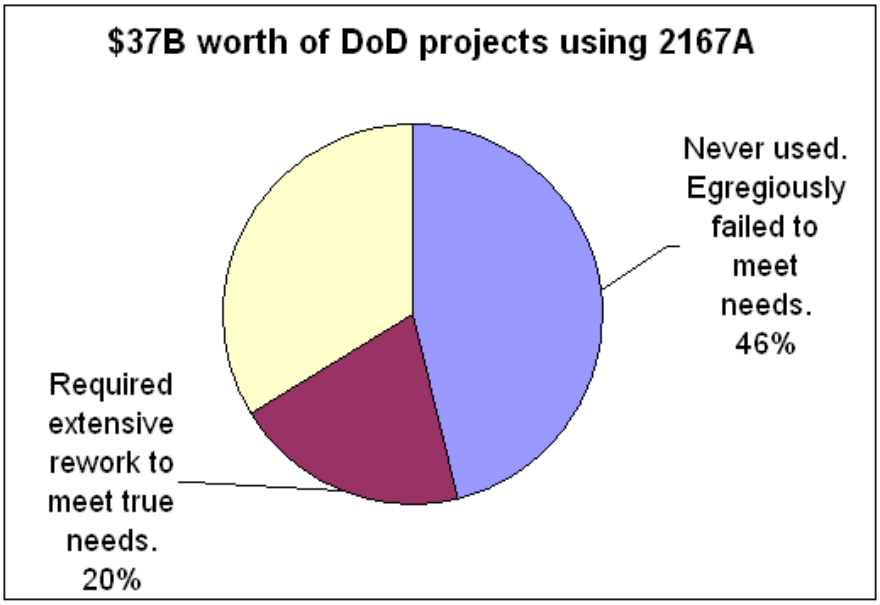

Figure 1.1: Percentage of Projects with Requirement mismatch [Miy]

However, responding to change is not free and impose significant impact on software architecture and development methodology. The purpose of architecture and development methodology should be to incorporate those changes and lower the technical debt as swiftly as possible.[Nor15] Any variation to software architecture affects coupling and cohesion and thus impacting understandability as well as complexity of the software.[Byr09]

The software development methodologies have come a long way undergoing a lot of modifications. Few of them to list would be waterfall model, spiral model, model and iterative model.[np08] Nevertheless, we have agile modeling, which works around some basic principles. Accepting change requirement at any stage of development and releasing software regularly are among the principles which supports to reduce the the cost of change.[And14]

Accordingly, software architecture has changed in number of ways and in number of direction throughout the history of evolution. It has passed through monolithic, layered, distributed object, domain driven development, hexagonal, service oriented architecture etc. [Som04] Nonetheless, we have microservices architecture with one of the major driving concepts being to improve cohesion and decrease coupling among components. This eventually, apart from accomplishing various other major goals, also upgrades the rate and efficiency of customizability.[New15]

Microservices are achieved by breaking down a system along business domains following Single Responsibility Principle. As the individual microservices achieved in such a way become the autonomous components, there is a need for a separate kind of component micro service in order to fulfill high-level business goals, which is Orchestration microservice. [New15] Regardless of following agile development methodology and microservice architecture, orchestration has been a painful and cumbersome area, considering the complexities being suffered when one has to make changes at any given point. There are a lot of technological and conceptual procedures around orchestration to make it less painful but at the same time there is no single agreement upon which one to follow.[Flo08]

List of Figures

Bibliography

[And14] J. G. Andrew Stellman. *Learning Agile*. O'Reilly Media, Inc., 2014.

[Byr09] J. C. C. Byron J. Williams. *Characterizing Software Architecture Changes: A Systematic Review*. July 2009. URL: http://www.ic.unicamp.br/~wainer/outros/systrev/10.pdf.

[Chanp] D. Chappell. *THE THREE ASPECTS OF SOFTWARE QUALITY: FUNCTIONAL, STRUCTURAL, AND PROCESS*. n.p.

[Flo08] B. P. Florian Daniel. *Insights into Web Service Orchestration and Choreography*. Tech. rep. International Journal of E-Business Research, 2008.

[Len03] R. K. Len Bass Paul Clements. *Software Architecture in Practice*. 2nd. Vol. 4. Addison Wesley, 2003.

[Mic] Microsoft. *Software Architecture and Design*. URL: https://msdn.microsoft.com/en-us/library/ee658098.aspx.

[Miy] C. Miyachi. *How Software Architecture Learns*. URL: http://www.methodsandtools.com/archive/softwarearchitecturelearning.php.

[New15] S. Newman. *Building Microservices Designing Fine-Grained Systems*. O'Reilly Media, Inc., 2015.

[Nor15] L. Northrop. *Trends and New Directions in Software Architecture*. 2015. URL: http://resources.sei.cmu.edu/asset_files/Webinar/2015_018_100_438676.pdf.

[np08] n.p. *SELECTING A DEVELOPMENT APPROACH*. Mar. 2008. URL: http://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf.

[Som04] I. Sommerville. *Software Engineering: Seventh Edition*. Pearson Education, 2004.