Final Year Project

Performance Analysis of Design Patterns in Microservice Architecture

Rajit Banerjee

Student ID: 18202817

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science with Data Science

Supervisor: Professor John Murphy

UCD School of Computer Science

University College Dublin

13th November 2021

Chapter 1: Project Specification

1.1 Problem Statement

Microservice architecture is a style of designing software systems to be highly maintainable, scalable, loosely-coupled and independently deployable. Moreover, each service is built to be self-contained and implement a single business capability. Design patterns in software engineering refer to any general, repeatable or reusable solution [1] to recurring problems faced during the software design process. The aim of this project is to analyse the performance of a number of microservice design patterns (based on metrics such as query response time, CPU/RAM usage, cost of hosting and packet loss rate), and evaluate their benefits and shortcomings depending on the business requirement and use case. A non-exhaustive list of design patterns that could be explored is as follows:

- API Gateway
- Chain of Responsibility
- Asynchronous Messaging
- Database or Shared Data
- Event Sourcing
- Command Query Responsibility Segregation (CQRS)
- Saga
- Circuit Breaker
- Strangler (Decomposition)
- Consumer-Driver Contract Test
- Externalise Configuration
- Aggregator
- Branch

For the aforementioned design patterns, sufficiently complex simulations will be designed for the performance engineering experiments. The project will also look at some common issues in microservices, and how they compare with traditional monolithic architectures.

1.2 Background

Microservices have gained traction in recent years with the rise of Agile software development and a DevOps [2] approach. As software engineers migrate from monoliths to microservices, it is important to make appropriate choices for system design and avoid "anti-patterns". Although no one design pattern can be called the "best", the performance of systems can be optimised by following design patterns suited to the use case, with the right configuration of hardware resources.

1.3 Related Work

Due to their popularity, microservices have been written about extensively in books like [3], [4], [5]. Articles such as [6], [7], [8], [9], [10] discuss the intricacies of microservice architecture as well as the trade-offs between various common design patterns. In [11], the performance problems inherent to microservices are explored, with evaluations performed using a custom-built prototyping suite. Akbulut and Perros [12] dive into the performance analysis aspect of microservices that is being proposed in this project, where they consider 3 different design patterns.

1.4 Datasets

Any data that is to be used or analysed in this project will be generated during the course of experiments. There are no dependencies on additional datasets.

1.5 Resources Required

A non-exhaustive list of resources is specified below, following preliminary needs assessment.

- Languages/Frameworks: Node.js + Express.js, React.js
- Tools: Git, Docker, Apache JMeter
- Database: MongoDB
- Compute: Linux server (maintained by the UCD School of Computer Science), possibly a High-Performance Computing Cluster

Chapter 2: **Introduction**

Introduce your vision of the project here. Describe the domain of the project, and the intended application. A well-written report will answer three key questions: What am I doing in this project? Why is it worth doing? How do I plan to go about it? In this introductory section, offer a concise answer to the What, and follow-up with a compelling account of the Why. Leave the How to a subsequent section. Do not try to do too much in any single section of the report. By providing details in a logical order, you will show that you have a plan for the report and the project.

Chapter 3: Related Work and Ideas

A key task of this first report is to establish a baseline against which your later work will be judged. Your FYP project does not exist in a vacuum, and its central problem, or a variant thereof, will have been tackled by others before you. In this section, you should describe how previous approaches have tackled the problem, and clearly articulate the state of the art (or SOA) for your project.

For research-oriented projects, this task will be time-consuming but relatively straightforward. You should read past works on the subject, summarise the main points, pros and cons, and root out the previous works that they cite in turn. You may use Wikipedia as a secondary source only, which is to say that it can be a useful first port of call on many topics but not a source that should be liberally cited. Rather, use Wikipedia as a hub for gathering references to primary work in the field (original papers and reports), then read and summarise those. Do not quote a work that you have not read, unless you are quoting someone else's view of that work. Never use another writer's words as your own. Place any extracts from another's work in double quotes, and attribute the quotation to its author with a citation. It is a very low act to plagiarise another's work and take credit for their words, so tread carefully. Even unintentional plagiarism is still plagiarism.

For more application-oriented projects, you are still expected to survey other solutions, either for the given problem or for similar problems, and also consider applications that share functionality or design principles with your own. In short, this section is the core of your report regardless of what kind of project you do.

Chapter 4: Project Work Plan

In this section you will present a work plan for the remainder of your project. Show that you have considered the issues carefully, and that you can be trusted to lead a research or development effort. Be as specific as you can about the time you expect to allocate to each work component, and the dependencies they have to each other. A Gantt chart is helpful in this respect, but do show some sense in how you present your plan. A naïve understanding makes for a simplistic plan.

A key part of a successful project is evaluation. It is not enough to just state that your project is a success, or that your friends seem to like it. You must have a plan for evaluating the end result. How you evaluate will depend on the nature of your project, and you should have a serious conversation with your supervisor about evaluation before you get to this stage. Will your work yield quantitative results that can be compared to past work or to established benchmarks? Does your work consider different configurations of a system or a solution that you can compare to each other, allowing you to empirically find the best one? Do you have a sample user pool for your planned application, and are they willing to give you structured qualitative and quantitative feedback (e.g. via a questionnaire)? However you plan to evaluate your project, please sketch your intentions here.

Bibliography

[1] Wikipedia. “Software design pattern,” [Online]. Available: https://en.wikipedia.org/wiki/Software_design_pattern (visited on 26th Oct. 2021).

[2] Amazon Web Services. “What is DevOps?” [Online]. Available: https://aws.amazon.com/devops/what-is-devops/ (visited on 26th Oct. 2021).

[3] C. Richardson, *Microservices Patterns: With Examples in Java*. Manning Publications, Oct. 2018, Book.

[4] M. Kleppmann, *Designing Data-Intensive Applications*. O’Reilly Media, Mar. 2017, Book.

[5] S. Newman, *Building Microservices: Designing Fine-Grained Systems*. O’Reilly Media, Dec. 2014, Book.

[6] M. Kamaruzzaman. “Effective microservices: 10 best practices,” [Online]. Available: https://t.co/ZM78yg19OR?amp=1 (visited on 26th Oct. 2021).

[7] ——, “Microservice architecture and its 10 most important design patterns,” [Online]. Available: https://towardsdatascience.com/microservice-architecture-and-its-10-most-important-design-patterns-824952d7fa41 (visited on 26th Oct. 2021).

[8] S. Kappagantula. “Everything you need to know about microservices design patterns,” [Online]. Available: https://www.edureka.co/blog/microservices-design-patterns (visited on 26th Oct. 2021).

[9] M. Udantha. “Design patterns for microservices,” [Online]. Available: https://dzone.com/articles/design-patterns-for-microservices-1 (visited on 26th Oct. 2021).

[10] J. Lewis and M. Fowler. “Microservices.” (25th Mar. 2014), [Online]. Available: https://martinfowler.com/articles/microservices.html (visited on 26th Oct. 2021).

[11] K. Cully, “Performance problems inherent to microservices with independent communication and resiliency configuration,” M.S. thesis, University College Dublin, Ireland, Mar. 2020.

[12] A. Akbulut and H. G. Perros, “Performance Analysis of Microservice Design Patterns,” *IEEE Internet Computing*, vol. 23, no. 6, pp. 19–27, 2019. DOI: 10.1109/MIC.2019.2951094.