# Transforming Monolithic Systems to a Microservices Architecture

Muhammad Hamza
Software Engineering,
LUT University,
Finland
+358504766307
muhammad.hamza@lut.fi

DOI: 10.1145/3573074.3573091
http://doi.org/10.1145/3573074.3573091

## ABSTRACT

**Context:** microservices architecture enables organizations to develop an application as a suite of loosely coupled small and independent services that can be developed, tested, and deployed independently. Various organizations are re-architecting their existing monolithic systems with microservices architecture. However, re-architecting the entire system can bring some indefinite challenges. **Objective:** the goal of this research project is to investigate (i) the need for migration from monolithic to microservices architecture, (ii) architectural description for adopting microservices, (iii) refactoring tools and methods (iv) potential challenges while transforming to microservices, and (v) effective patterns and strategies to adopt microservice successfully. **Methodology:** the industrial empirical (interviews, case study, and questionnaire survey) approach will be used to meet the research objective. **Possible outcomes:** the expected outcomes would be (i) an evidence-based decision-making framework for transforming monolithic architecture (ii) strategies to refactor the monolithic architecture (iii) prioritization-based taxonomy of challenges while transforming to microservices. Finally, (iv) the development of a decision model for selecting patterns and strategies for successful implementation of the microservices system.

## Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques – *Software organizations and properties*

## General Terms

Human Factors, Performance, Design, Verification

## Keywords

Monolithic architecture, Microservice architecture, Methods, Challenges, Patterns, Strategies

## 1. INTRODUCTION

Applications in traditional monolithic architecture are developed as a single unit wherein all the functions are tightly coupled. However, several issues such as slow delivery, high code coupling, scaling, and deployment arise when the code base of monolithic architecture grows exponentially. Therefore, microservices architecture is gaining significant momentum in IT enterprises day by day [1]. Companies like Guardian, Amazon, Netflix, Uber, Etsy, and many others migrated their products from a monolithic architecture to microservices systems [2].

Microservices architecture consists of a collection of small autonomous services that can be developed, tested, and deployed independently. The application in a microservices architecture is developed as a suite of loosely coupled services that represent a single business capability within a bounded context [3]. Different teams can develop each service and use different technology stacks. Moreover, these services are easier to maintain and more fault-tolerant, i.e., failure of one service could not halt the entire system. Microservices' architectural style promotes isolation, autonomy, share-nothing philosophy, and service choreography. The architecture is supposed to increase agility, developer productivity, application scalability, reliability, and maintainability. It is also a suitable choice for iterative development processes like agile and DevOps [4]. Organizations that deliver their software through the microservice paradigm can improve productivity, resilience, scalability, and business functionality.

However, for all the benefits microservice-based architecture offers, the journey for an organization to migrate to a microservices architectural style can be challenging and perilous. In fact, legacy software engineering processes and lack of knowledge are the main hurdles that companies face in adopting microservices [1][4]. Similarly, some technical challenges include migrating from legacy monolithic systems to scalable microservices architecture, defining and implementing software delivery pipelines, and change management that is affected due to this migration. Moreover, organizations typically have to redefine their software development method, up-skill human resources, and need to redefine business scope and values.

So far, various research studies have been published on microservice architecture with different contexts and development processes. For example, Soldani et al. [5] highlighted that the challenges of microservices-based applications are due to their intrinsic complexity. The challenges are associated with the design, development, and operations of independent and asynchronously evolving microservices. They also provided a taxonomic classification comparing the existing gray literature on the pains and gains of Microservices, from design to development. Pahl et al. [6] conducted a systematic mapping study that highlights most of the research conducted on technological review, test environments, and use case architectures. Moreover, they found no study covering the empirical evaluation of microservices in different size organizations.

However, the current literature does not examine empirically the process of monolithic to microservices transformation in different organizations, and the challenges faced during migration and while executing microservices architecture [6]. In addition, there is a lack of strategies and guidelines to evaluate the capability of an organization concerning to migration process. Furthermore, there is a lake of a decision model for selecting the patterns and strategies for the successful execution of the microservice system.

Seeking the importance of microservices architecture, this research project is proposed to understand information/metrices that are required for an organization before migration to microservices; strategies to refactor the existing monolithic architecture to microservices. Moreover, this research work will understand the challenging factors faced by organizations during the migration process and in the microservices system. Ultimately, the research work will provide a practically robust decision model with the aim to recommend the pattern and strategies for the successful implementation of microservices architecture.

# 2. RESEARCH OBJECTIVE AND QUESTIONS

## 2.1 Research objective and questions

This research project will be accomplished by addressing the three main and their subsequent research questions. The project is divided into three phases for explicit understanding.

### *2.1.1 Phase-1:*

The ultimate objective of phase 1 is to empirically investigate, why organizations migrate from a monolithic architecture to microservices. What were the challenges in their existing monolithic architecture that the organizations believed to solve by adopting the microservices architecture? Furthermore, what information/metrics do organizations collect before taking the decision to migrate to microservices? Do organizations perform a migration by themselves or by hiring an outsourced consultancy? In addition, we will investigate how and what strategies organizations employ for refactoring the monolithic architecture. This phase will further investigate whether an organization achieved a higher degree of technological diversity as expected from microservice architecture.

**RQ1. Why and how do organizations migrate from monolithic to microservice?**

**RQ1.1. Why do organizations migrate from monolithic to microservices?**

This RQ will be used to empirically investigate the need for organizations to migrate from monolithic to microservices architecture. How do organizations assess their existing monolithic architecture is no longer sustained?

**RQ1.2. What information/metrics were collected before migration to microservices?**

The aim of this research question is to investigate how organizations planned the migration process and what information/metrics were collected before migration to microservices. Do the organizations plan the migration themselves or hire an outsourced consultancy?

**RQ1.3. How do organizations refactor monolithic architecture to microservice?**

Under this research question, we will study how organization refactors their existing monolithic architecture to microservices. What tools and strategies are employed when refactoring the monolithic architecture?

**Phase-1: Research methodology**

To address RQ1, we will use a systematic mapping study (SMS) to explore the migration process from the state-of-the-art literature. SMS is considered to be one of the most comprehensive research methods to understand the state-of-the-art literature concerning a particular topic [7].

In addition to this, we will conduct an empirical qualitative (interview) [8] approach to address RQ1. We will conduct structured and semi-structured interviews with the industrial practitioners to gain a deeper insight into how they planned and migrated their monolithic system to microservices architecture.

**Possible outcomes:**

The possible outcomes of phase 1 could be a decision-making framework that will help the organizations to evaluate their capability of migration from monolithic to microservices architecture. This phase will serve as an initial feed for phase-2 and 3.

### *2.1.2 Phase-2:*

In phase 2 of the research project, (i) we will identify the key challenges that organizations face while transforming their monolithic system to microservices architecture. (ii) We will further analyze microservices systems of the organizations and open source microservices projects to identify the key issues and their causes in each microservices design area.

**RQ2. What are the challenges faced by practitioners while migrating to microservices?**

This research question aims to identify the key challenges faced by practitioners while transforming the monolithic system into a microservices architecture. Furthermore, we will identify the challenges and their causes in open-source microservice systems. The issues and their causes will be identified in each design area of the microservices system. The identified issues and causes will be validated with industrial practitioners with respect to their microservices systems.

**Phase 2: Research methodology**

We will conduct structured and semi-structured interviews with practitioners to identify the challenges faced while transforming to microservices architecture. In addition to this, we will use mining software repositories (MSR) [9] to identify the key issues and causes in each design area of the microservices system. MSR is used to analyze the rich data available in software repositories and Q&A. The identified issues and their causes will be validated by microservices system experts by conducting the questionnaire survey study.

**Possible outcomes:**

This phase will provide a set of critical challenges, faced by the organizations while transforming the monolithic system into a microservices architecture. In addition to this, phase 2 will identify the key challenges and their causes in the open-source microservices system and be validated with organizational microservices systems. Based on the identified challenges, we will develop a decision-making framework with the aim to assist the experts to consider the most significant challenges on a priority basis.

### *2.1.3 Phase-3:*

In phase 3 of the research project, we will identify the patterns and strategies from the state-of-the-art practice. Patterns are the reusable solution to commonly occurring problems or challenges. To address the migration process and microservices architecture issue, a limited number of patterns and strategies were proposed by the research and practice community. For example, 'decompose an application by subdomain', 'business capability, or 'services per team' are some patterns for decomposing the monolithic application. Similarly, there are set of patterns for other design areas of microservices architecture. Each pattern has its own advantages and disadvantages in the microservices system. For an instance, if the project is large and requires more people, then, the 'services per team' pattern could negatively impact it. Therefore, selecting the right patterns and strategies for each design area of a microservices architecture is perilous for practitioners.

To this end, we will develop a decision model that will assist practitioners to select the best suitable set of patterns and strategies for the microservices system life cycle. The model will provide a holistic view of patterns and strategies for their system need. The developed model will be evaluated by the practitioners by conducting case studies and be improved based on the suggestions. RQ3. How to develop a decision model for selecting the patterns and strategies?

**RQ3.1. what are the patterns, strategies, and anti-patterns for microservices architecture?**

The aim of this research question is to investigate the patterns, strategies, and anti-patterns from state-of-the-art industry practices.

**RQ3.2. What would be the practically robust decision model for selecting patterns and strategies?**

This research question will be used to develop a decision model for selecting the patterns and strategies to achieve the set of quality attributes. The developed model will be evaluated in the industry to check its effectiveness.

**Phase 3: Research Methodology**

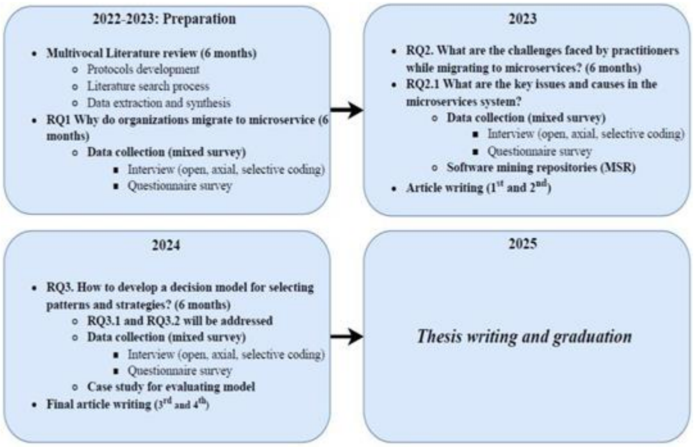
To address the RQ3, we will identify patterns and strategies from the state of the practice by conducting interviews with microservices system practitioners. In addition, we will find the patterns and strategies proposed by experts and available in the form of a report and online Q&A repositories.

In the next step, the proposed decision model will be developed for selecting patterns and strategies for microservices architecture design areas. Decision models in software architecture are used to map the component of problem space to the component of solution space [11]. X. Xu et al [11] proposed a decision model that assists developers and architects in selecting appropriate patterns for blockchain-based applications. Similarly, the study presents decision models for cyber-foraging systems that map functional and non-functional requirements to architectural tactics for cyber-foraging. The proposed decision model will be developed by following the guidelines of developed decision models in other domains of software engineering.

Finally, a case study approach will be used to evaluate the effectiveness of the proposed model with microservices practitioners. The model will be modified based on the suggestion of practitioners. The final version of the model will be available for practitioners' practices.

## 3. SCHEDULE

The overall schedule of this research study is presented in Figure 1. In the beginning, I will conduct a systematic mapping study to understand the state-of-the-art literature published over the year on our research questions. The industrial data collection process will begin at the end of November 2022. Mapping the RQs on the timeline is preliminary, the approach and research questions may change as my research progress.



**2022-2023: Preparation**

- **Multivocal Literature review (6 months)**
  - Protocols development
  - Literature search process
  - Data extraction and synthesis
- **RQ1 Why do organizations migrate to microservice (6 months)**
  - **Data collection (mixed survey)**
    - Interview (open, axial, selective coding)
    - Questionnaire survey

**2023**

- **RQ2. What are the challenges faced by practitioners while migrating to microservices? (6 months)**
- **RQ2.1 What are the key issues and causes in the microservices system?**
  - **Data collection (mixed survey)**
    - Interview (open, axial, selective coding)
    - Questionnaire survey
  - **Software mining repositories (MSR)**
- **Article writing (1st and 2nd)**

**2024**

- **RQ3. How to develop a decision model for selecting patterns and strategies? (6 months)**
  - **RQ3.1 and RQ3.2 will be addressed**
  - **Data collection (mixed survey)**
    - Interview (open, axial, selective coding)
    - Questionnaire survey
  - **Case study for evaluating model**
- **Final article writing (3rd and 4th)**

**2025**

*Thesis writing and graduation*

**Figure 1. Preliminary timeline of the research**.

## 4. REFERENCES

[1] D. Taibi, V. Lenarduzzi, and C. Pahl, "Processes, Motivations, and Issues for Migrating to Microservices Architectures: An Empirical Investigation," IEEE Cloud Comput., vol. 4, no. 5, pp. 22–32, Sep. 2017, doi: 10.1109/MCC.2017.

[2] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables devops: Migration to a cloud-native architecture," Ieee Software, vol. 33, no. 3, pp. 42–52, 2016.

[3] J. L. Martin Fowler, "Microservices a definition of this new architectural term." [Online]. Available: https://martinfowler.com/articles/microservices.html

[4] D. Trihinas, A. Tryfonos, M. D. Dikaiakos, and G. Pallis, "Devops as a service: Pushing the boundaries of microservice adoption," IEEE Internet Computing, vol. 22, no. 3, pp. 65–71, 2018.

[5] J. Soldani, D. A. Tamburri, and W.-J. V. D. Heuvel, "The pains and gains of microservices: A Systematic grey literature review," Journal of Systems and Software, vol. 146, pp. 215–232, 2018, doi: https://doi.org/10.1016/j.jss.2018.09.082.

[6] C. Pahl and P. Jamshidi, "Microservices: a systematic mapping study.," CLOSER (1), pp. 137–146, 2016.

[7] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in 12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12, 2008, pp. 1–10.

[8] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, "Selecting empirical methods for software engineering research," in Guide to advanced empirical software engineering, Springer, 2008, pp. 285–311.

[9] A. E. Hassan, "The road ahead for mining software repositories," in 2008 Frontiers of Software Maintenance, 2008, pp. 48–57.

[10] G. A. Lewis, P. Lago, and P. Avgeriou, "A decision model for cyber-foraging systems," in 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), 2016, pp. 51–60.

[11] X. Xu, H. D. Bandara, Q. Lu, I. Weber, L. Bass, and L. Zhu, "A decision model for choosing patterns in blockchain-based applications," in 2021 IEEE 18th International Conference on Software Architecture (ICSA), 2021, pp. 47–57.