Aalto-yliopisto
Aalto-universitetet
Aalto University

Master's Programme in Computer, Communication and Information Sciences

Framework for Seamless Cloud-to-Cloud Service Migration

Roope Paajanen

Master's thesis
2024

Aalto-yliopisto
Aalto-universitetet
Aalto University

Author Roope Paajanen

Title of thesis Framework for Seamless Cloud-to-Cloud Service Migration

Programme Master's Programme in Computer, Communication and Information Sciences

Major Communications Engineering

Thesis supervisor Prof. Manner Jukka

Thesis advisors M.Sc. Töllinen Torsti, M.Sc. Heikkinen Anssi

Collaborative partner A Finnish financial company

Date 31.7.2024 **Number of pages** 84+2 **Language** English

Abstract

The agility, scalability, and innovative services cloud providers offer have created an inviting market for companies still managing an on-premises infrastructure. Many companies in various industries have started to utilise cloud computing's capabilities and started moving their IT infrastructure to the cloud. This thesis is written in collaboration with a Finnish financial company, that is executing a cloud migration, transitioning from a hybrid environment utilising on-premises infrastructure and AWS to fully a Microsoft Azure environment.

Despite the growing popularity of cloud computing, existing academic literature and documentation provided by cloud providers tend to focus on migrating from on-premises environment to the cloud. This creates a knowledge gap for companies already utilising cloud services but seeking to transition to a different cloud platform.

This thesis studies cloud-to-cloud service migration from the perspective of a cloud developer working in the financial company. The research conducted examines various cloud migration methodologies, tools, and case studies from the industry. Additionally, the research gathers company specific insight on cloud-to-cloud migration by discussing the company's cloud migration strategy and by conducting interviews with the company's cloud experts. The findings culminate into a four-step cloud-to-cloud migration framework designed to guide organisation's developers through the migration lifecycle, ensuring a structured and efficient service transition from AWS to Microsoft Azure.

Keywords Cloud Migration, Cloud-to-Cloud Migration, Migration Strategies, Migration Framework, Cloud Computing

Aalto-yliopisto
Aalto-universitetet
Aalto University

Tekijä Roope Paajanen

Työn nimi Pilvisiirtymäkehys pilvipalveluiden väliselle siirtymälle

Koulutusohjelma Tieto-, tietoliikenne- ja informaatiotekniikan maisteriohjelma

Pääaine Tietoliikennetekniikka

Työn valvoja Prof. Manner Jukka

Työn ohjaajat M.Sc. Töllinen Torsti, M.Sc. Heikkinen Anssi

Yhteistyötaho Suomalainen finanssialan yritys

Päivämäärä 31.7.2024 **Sivumäärä** 84+2 **Kieli** Englanti

Tiivistelmä

Pilvipalveluiden tarjoama ketteryys, skaalautuvuus ja innovatiiviset palvelut ovat luoneet houkuttelevan markkinan yrityksille, jotka edelleen hallinnoivat paikallista infrastruktuuria. Monet yritykset eri toimialoilla ovatkin alkaneet hyödyntämään pilvipalveluiden mahdollisuuksia ja siirtäneet IT infrastruktuurinsa pilveen. Tämä opinnäytetyö on tehty yhteistyössä suomalaisen finanssialan yrityksen kanssa, joka toteuttaa pilvisiirtymää hybridiympäristössä, jossa hyödynnetään paikallista infrastruktuuria ja Amazon Web Servicen pilvialustaa, siirtyen kokonaan Microsoft Azure pilvialustalle.

Huolimatta pilvipalveluiden kasvavasta suosiosta, olemassa oleva akateeminen kirjallisuus ja pilvipalveluntarjoajien dokumentaatio keskittyvät pääasiassa paikallisista ympäristöistä pilvialustalle siirtymiseen. Tämä luo ongelmatilanteita yrityksille, jotka jo käyttävät pilvipalveluita, mutta aikovat siirtyä eri pilvialustalle.

Tässä opinnäytetyössä tutkitaan pilvialustojen välistä siirtymää finanssialan työskentelevän pilvikehittäjän näkökulmasta. Tutkimuksessa tarkastellaan erilaisia pilvimigraatiomenetelmiä, työkaluja ja alan tapaustutkimuksia. Lisäksi tutkimuksessa kerätään yrityskohtaisia näkemyksiä pilvisiirtymästä hyödyntämällä yrityksen pilvisiirtymästrategia raportteja ja haastattelemalla yrityksen pilvikehittäjiä. Tuloksien pohjalta opinnäytetyö esittelee nelivaiheisen pilvisiirtymäkehyksen, joka on suunniteltu ohjaamaan organisaation kehittäjiä pilvisiirtymäprosessin läpi, varmistaen jäsennellyn ja tehokkaan pilvipalvelusiirron Amazon Web Services alustalta Microsoft Azure alustalle.

Avainsanat Pilvisiirtymä, Pilvestä pilveen siirtymä, Pilvisiirtymästrategia, Pilvisiirtymäkehys, Pilvipalvelut.

Table of contents

Preface and acknowledgements

Finalising and submitting my master's thesis marks the end for my five-year long journey through the Finnish academic world. This time period has been filled with growth, challenges, and countless learning experiences that have shaped my academic and personal development.

I want to thank my family, friends, and partner for their unwavering support throughout my studies, especially during these uncertain times. Additionally, I want to thank my peers from both Åbo Akademi University in Turku and Aalto University in Espoo for making the journey in the academic circles unforgettable.

Lastly, I want to thank my supervisor Jukka Manner and advisors Torsti Töllinen, Anssi Heikkinen, and Tommi Pystynen. Your guidance, feedback, and persistent support have been instrumental in the completion of this thesis.

Helsinki, 31 July 2024
Roope Paajanen

Symbols and abbreviations

Abbreviations

AWS	Amazon Web Services
SaaS	Software as a Service
PaaS	Platform as a Service
IaaS	Infrastructure as a Service
IOT	Index-Organised Table
DMS	Database Migration Service
CDC	Changed Data Capture
CI/CD	Continuous Integration and Continuous Delivery
DR	Disaster Recovery
RTO	Recovery Time Objectives
RPO	Recovery Point Objectives
ESG	Environmental, Social and Governance
POV	Point-Of-View
CAPEX	Capital Expenditures
OPEX	Operational Expenditures
VPN	Virtual Private Network
KDS	Kinesis Data Streams
AML	Amazon Machine Learning
Azure AD	Azure Active Directory
AWS SCT	AWS Schema Conversion Tool
T-SQL	Transact-SQL
SCMS	Secure Cloud Migration Strategy
CPU	Central Processing Unit
API	Application Programming Interface
SAST	Static Application Security Testing
IaC	Infrastructure as a Code

Tables

Images

1. Introduction

In the dynamic spectrum of digital transformation, cloud computing has surpassed beyond being a mere technological advantage to becoming a cornerstone of corporate strategy. The agility, scalability, and countless innovative services offered by cloud computing platforms are essential for businesses seeking to maintain a competitive edge in today's fast-paced market. These platforms empower developers with a diverse set of tools, enabling seamless management and deployment of applications, while also accelerating the evolution of more efficient, robust, and versatile solutions. The competition among leading cloud service providers has fostered an environment of continuous improvement, urging them to innovate and deliver superior functionalities. Consequently, developers find themselves in a pivotal position, carefully evaluating these technologies to tailor the best possible environment for their unique application needs.

In the broader spectrum, while individual developers explore the nuances of tools and services, large-scale enterprises embark on a more complex journey. They engage in a comprehensive evaluation of cloud platforms, ensuring alignment with their overarching organisational goals, infrastructural prerequisites, regulatory compliance, data security, and operational continuity.

The growing popularity of cloud technologies have resulted in a situation where many companies are shifting their on-premises (locally hosted) infrastructure to the cloud, hosted by a cloud provider. The process of migrating infrastructure, applications, and services to the public cloud (cloud-migration) can become a complex project depending on the size and readiness of the local environment. However, companies and cloud providers are aware of this, and invest resources to make the process as efficient as possible. Cloud platforms offer courses and tools to ensure that locally managed applications and services can be efficiently moved to their particular cloud platform.

However, there is limited investment or research focused on moving applications, services, and infrastructure between cloud platforms. It is commonly believed that once an application is in the cloud, it is easy to move, but that is rarely the case. Similar to local on-premises environments, cloud infrastructure can quickly become complex with various dependencies.

1.1 Problem Statement

This study takes a closer look at a Finnish financial institution's cloud migration strategy. In the fall of 2023, the company unveiled a forward-thinking cloud strategy. The strategy defines a clear objective to transition from their existing hybrid network– a blend of public cloud services hosted in Amazon Web Services and local deployments in their proprietary data centres – to a cloud ecosystem hosted on Microsoft Azure by 2030.

The implementation of new applications during the past decades has piled up to a situation where 15% of the company's applications are either partially or completely hosted in public cloud [10]. As of February in 2023, the company's main cloud computing partner has been Amazon Web Services for almost a decade, with several hundred active accounts as well as a large spectrum of on-premises installations. Today the company is facing their next issue, how to create a smooth transition to the new cloud platform by 2030. Not only is the company challenged with technical challenges resulting from the transition to a new platform provider, but it must also factor in concerns regarding data security, regulatory compliance, the continuity of services, and the training of its workforce. The size of this transition is magnified by company's dominant position in the Finnish financial sector, introducing unique market-specific factors that must be considered.

In the broader context, the company is responsible for managing and operating all applications on the target cloud platform. However, regarding the migration of applications to a platform managed by a new cloud provider, the migration responsibility will fall to either an individual developer or a small team, who may not possess all the necessary skills to execute a successful transfer of the application to the new platform. These circumstances might create awkward and fragile situations where decisions of the correct tools and ways of working might be unclear for the one executing the migration.

The process of migrating from on-premises infrastructure to the public cloud is both well-researched and supported, with a solid foundation in academic literature and a clear place in the company's cloud strategy. However, there is relatively little research conducted or tools provided to assist companies and developers in implementing effective cloud-to-cloud migration, possibly due to its specificity or the perception that organisations would not frequently switch cloud providers once they have migrated to the cloud. Due to this, the cloud-to-cloud migration process is often left for the individual developer, team, or company to solve.

1.2 Objectives of the Study

The aim of this research is to gain a comprehensive understanding of cloud-to-cloud migration process and create a customised cloud migration framework tailored to the company's context. The framework aims to reduce the costs and complexities associated with cloud-to-cloud migration for the company's development teams. The study intends to examine various aspects of this transition, including overarching organisational strategies as well as specific obstacles encountered by developers.

The research questions are:

- *How can a cloud-to-cloud migration be executed efficiently and successfully in the company's environment?*

- *What does a development team need to take into account when executing a cloud-to-cloud migration?*

- *What are the critical processes, challenges, and best practices involved in executing a cloud-to-cloud migration, specifically from AWS to Azure, within a large financial institution?*

The primary goal of this thesis is to present a comprehensive cloud migration framework that will function as a guide and tool for developers or small teams within the company. This framework is specifically designed to streamline the cloud-to-cloud migration process to Azure Cloud and provide clear and actionable steps for each phase of the migration journey. The framework is intended to assist teams in navigating the complexities of migration, ensuring that they remain aligned with best practices and meet organisational goals.

1.3 Research Methodology

This study adopts a comprehensive approach to analyse the company's cloud migration strategy with a particular focus on facilitating a smooth transition to Azure cloud by 2030. The methodology is designed to uncover the multi-dimensional challenges associated with such a significant shift, studying technical obstacles, data security, regulatory compliance, service continuity, and workforce training needs. The data collected from the analysis will be utilised to create a robust and tailored cloud migration framework, specifically customised for the company's developers. This thesis or framework will not focus on regulations or constraints on cloud or software development in

the Finnish financial market. Instead, the thesis will concentrate on the technical aspect of migrating software between different cloud providers.

Qualitative Analysis

At the core of the thesis' methodology is an in-depth qualitative assessment of the company's current cloud setup and future migration initiatives. This includes examining internal documents, strategic plans, and project reports to measure the present situation, future aspirations, and the steps planned to connect these two phases. This examination is crucial for pinpointing migration motivations, anticipated obstacles, and devised mitigation strategies.

Stakeholder Interviews

To gain insights from those directly involved in the migration process, semi-constructed interviews will be conducted with a range of stakeholders within the company. This includes software developers and tech leads who have been participating in the architecture, design, development, or management of an application that is completely or partially running on Amazon Web Services (AWS). The objective is to gather first-hand accounts of experiences, expectations, and concerns related to the cloud migration effort. These interviews should provide valuable perspectives and insights on the technical and organisational challenges faced during the transition to Azure.

Comparative Analysis

The study will also involve a comparative analysis with industry best practices for cloud-to-cloud migration. By analysing how similar transitions can been managed, the study seeks to gather practical insights that are used to create a cloud migration framework for the financial company's developers.

Development of a Cloud Migration Framework

Based on the findings from the qualitative analysis, stakeholder interviews, and comparative review, a cloud migration framework specific to the company's needs will be developed. This framework will provide a structured approach for developers to navigate the complexities of migrating applications to Azure, addressing key areas such as technical architecture, data migration, security protocols, compliance commitment, and employee training. The framework will be designed to be adaptable, allowing for customisation based on the specific requirements of different applications and teams within the company.

1.4 Structure of the Thesis

Firstly, this thesis starts by covering general information about cloud services and cloud providers by discussing the general benefits cloud solutions offer. Secondly, the thesis will compare various cloud migration methods, practises, and tools to gain a comprehensive understanding of best practises in the industry. Thirdly, this thesis will analyse the financial institution's objective in migrating to Azure Cloud, including the reasoning behind this decision and an analysis of the migration strategy. Fourthly, the thesis will discuss insights from stakeholder interviews covering different practical scenarios an individual developer might face when migrating software between different cloud providers.

Finally, the thesis will provide a comprehensive framework for an individual developer or a team in the financial company, to validate their cloud migration process and help their journey moving applications between the previously used cloud platform AWS to the target cloud platform Microsoft Azure.

1.5 Results

The proposed cloud migration framework consists of four phases. The first phase is the Preparation and Concept phase, which provides the necessary information to establish a solid understanding of the starting point and prepare for the migration. This phase assists the team in detecting areas of improvement and avoiding potential issues during later phases.

The second phase is the Blueprint Design phase. In this phase, the thesis provides a comprehensive migration roadmap for the development team, helping them select the appropriate cloud service counterparts on the new cloud platform.

The third phase of the framework is Execution and Transition. In this phase, the framework guides and assists the developer in successfully and efficiently executing the migration process itself by choosing the correct migration strategy for their cloud service.

The fourth and final phase of the framework is Post-Migration and Optimisation. This phase focuses on the application lifecycle after the initial migration process is executed. It aims to ensure that the migrated infrastructure remains optimised and aligned with business needs, and that the development team has processes in place to identify and address future migration needs.

2. Cloud Computing and Cloud Migration

This section of the thesis covers various aspects of cloud computing and migration, providing a comprehensive foundation necessary for understanding the final solution presented in the thesis. Firstly, the section begins with an exploration of cloud computing fundamentals and the different cloud service models. This foundational knowledge is crucial for understanding the details of cloud migration strategies and methodologies covered later in the thesis.

Secondly, the section covers detailed perspectives on various migration strategies and methodologies, showcasing industry case studies that illustrate practical applications and outcomes. By covering these critical aspects of cloud technologies and cloud migration, this thesis aims to provide a broad understanding of the complexities and best practices involved in cloud-to-cloud migrations, particularly from AWS to Azure.

2.1 Cloud Computing

This chapter explores the impact of cloud technologies on the IT industry, providing an essential foundation for understanding cloud migration. It begins with an overview of major cloud service providers, highlighting their competitive landscape and market dynamics. The discussion extends to the three primary service models offered by cloud providers: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), detailing their functionalities, benefits, and use cases. Innovation and emerging technologies within the cloud ecosystem, such as AI, machine learning, and IoT, are also examined to illustrate their transformative potential across various sectors. Finally, the geographical reach and strategic placement of data centres by cloud providers are analysed, emphasising their role in ensuring service availability, reducing latency, and complying with regional regulations. This comprehensive overview sets the stage for a deeper dive into the strategies, methodologies, and case studies of cloud migration, providing a robust framework for understanding the complexities and opportunities of migrating from AWS to Azure.

2.1.1 Overview of Cloud Service Providers

Cloud service providers universally deliver essential functionalities like computing power, data storage, and internet-based applications. These services empower corporations and developers to rapidly enhance software and applications, driving efficiency and innovation without initial capital expenditure. Nevertheless, distinct attributes between providers guide the selection process for partnering corporations.

In 2023, Synergy Research Group reported that enterprise investment in IaaS, PaaS, and hosted private cloud services globally surpassed $227 billion [2]. Echoing this growth, Richard Johnson from Yahoo Finance forecasts the cloud computing market's value to soar from $495.3 billion in 2022 to approximately $2.495 trillion by 2032 [3]. Despite this rapid market expansion, a handful of key players predominantly dominate the sector. Amazon Web Services leads, consistently holding 32-34% market share over the past five years. In contrast, Microsoft Azure's market share has surged to 23%, up from 13% since Q4 2017. Google Cloud also maintains a significant presence, securing an 11% market share in 2022.

This section sets the stage by introducing the competitive landscape of cloud service providers, highlighting the market's growth potential and the pivotal role of major players, thereby laying the groundwork for a detailed exploration of their unique offerings and strategic positions in the subsequent sections.

2.1.2 Service Models and Offerings

On an architectural level, Cloud computing service offers three different types of services for users: Infrastructure as a service (IaaS), Platform as a service (PaaS) and Software as a Service (SaaS). M. Ikram and F. Hussain states their view of cloud service model hierarchy in their article “Software as a Service (SaaS) Service Selection Based on Measuring the Shortest Distance to the Consumer's Preferences”: “Software as a Service (SaaS) is a type of cloud service that runs and operates over the Platform as a Service (PaaS), which in turn works on the Infrastructure as a Service (IaaS)”[4]. The cloud services are in fact layered on top of each other, as shown in Figure 2. These different platforms offer users a different level of customisation and ownership status depending on different needs, from raw computing power to complete software solutions. See Cloud service model customisability and responsibility separation differences in Figure 1.

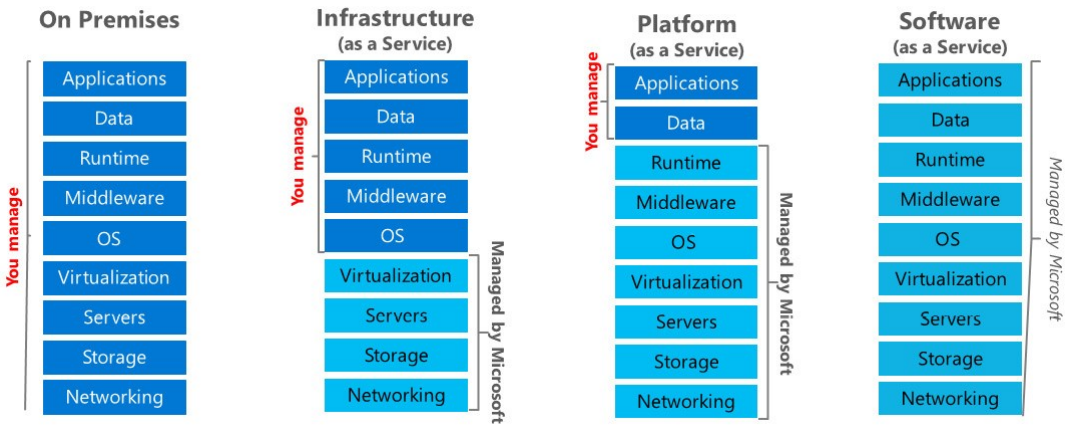

Figure 1, Separation of Responsibilities in the Azure Model [37]

Infrastructure as a Service

Infrastructure as a Service, or IaaS, provides the low-level service in cloud model hierarchy and provides the most fundamental resources such as virtual machines, storage, and networks. IaaS is the least supported cloud computing model providing full customisability for end user. The IaaS service model allows users to rent IT infrastructure on a pay-as-you-go basis, offering flexibility and scalability without the need to invest in physical hardware. Examples of IaaS providers include AWS EC2, Microsoft Azure Virtual Machines, and Google Compute Engine.

Platform as a Service

Platform as a Service, or PaaS, builds on IaaS by providing a platform that includes operating systems, databases, and tools for application development, deployment, and management. Unlike SaaS, which delivers ready-to-use software, PaaS offers an environment where developers can build, deploy, and manage their own applications without worrying about the underlying infrastructure. This environment includes operating systems, development tools, and database management systems. PaaS allows developers to focus on the creative side of app development, offering the flexibility to innovate and customise applications while the service provider manages the maintenance of the underlying resources. This model accelerates development, encourages innovation, and simplifies operations by abstracting and automating the management of hardware and system software. Popular PaaS offerings include Microsoft Azure App Services, Google App Engine and AWS Fargate.

Software as a Service

Software as a Service or SaaS is the top layer service in cloud model hierarchy and simultaneously the most common of these cloud models for an end user. SaaS is a cloud computing model where applications are hosted and managed by the service provider and made available for the user through internet rather than being installed on their local machine. This often makes a SaaS solution the least customisable, as it is primarily designed for end-user with a pre-optimised set of features and limited configuration options. Some well-known SaaS solutions widely used are for example different email inboxes, communication channels such as Teams and Slack.

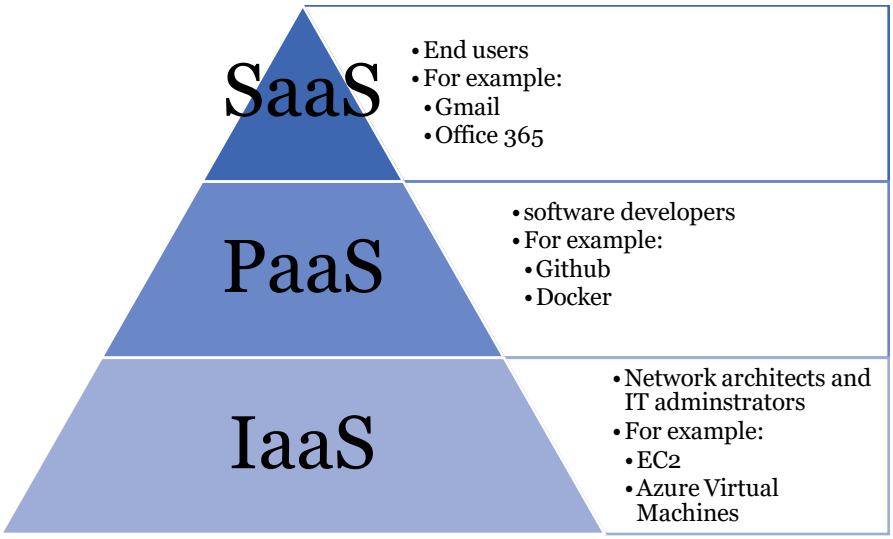

Figure 2, Representing a Pyramid of the Three Different Cloud Service Models.

2.1.3 Innovation and Emerging Technologies

As of early 2024 several different technologies are facing a revolution which creates new possibilities for different business sectors to come up with new fascinating implementations. These advancements are not merely iterative; they represent foundational shifts in how industries operate, innovate, and deliver value to its customers. Among these transformative technologies, artificial intelligence (AI), machine learning (ML), Quantum Computing, Internet of Things (IoT), 5G Networking, Augmenter and Virtual Reality (AR/VR), Distributed Ledger Technology (DLT) and cloud computing stand out as key drivers of change, particularly in the financial sector. The report by Propsons and Galaskis at the World Economic Forum in 2020 underscores this point, highlighting that investments in AI and cloud computing are crucial for

financial organisations aiming to address and solve critical business challenges effectively [8].

However, the adoption of new technologies in large organisations is accompanied by significant costs. According to key insights from the Propsons and Galaskis report, the widespread accessibility of emerging technologies is set to intensify competition among financial institutions. Their success depends on their agility in integrating new technologies into their operational processes. The report further suggests that maximizing the benefits of these technological advances will require breaking down traditional industry boundaries, thereby increasing opportunities for innovation. Moreover, a notable trend is the growing reliance on third-party providers to operate and manage these new technologies, introducing a level of dependency that organisations must carefully consider when incorporating these solutions into their infrastructures.

2.1.4 Geographical Reach and Data Centres

The geographical reach and strategic placements of data centres and regions are critical for cloud computing, development, and consumer experience. Cloud providers like AWS and Microsoft Azure strategically position their data centres worldwide to ensure high availability, low latency, and compliance with regional regulations.

Firstly, one of the cloud computing's primary advantages against traditional on-premises data centres is its global accessibility, allowing users to access resources from anywhere with an internet connection. This is facilitated by an extensive network of data centres in multiple regions and availability zones across different countries and continents. The wide distribution ensures service redundancy and operational continuity even during outages.

Secondly, customer and data centre proximity have a crucial impact on application performance. A well-placed data centre can reduce latency and enhance the user experience, especially with real-time application like video conferencing and online gaming. Deploying applications to data centres geographically close to consumers minimises delays, ensuring a faster response time.

Thirdly, compliance with regional data privacy laws is another key aspect. Different regions have specific regulations, such as the EU's GDPR, which requires data to be stored and processed within the EU. Cloud providers maintain data centres in various countries to meet these regulations, ensuring data sovereignty and compliance.

In conclusion, strategically placed data centres enhance cloud service effectiveness by ensuring global accessibility, minimising latency, and meeting regulatory compliance. This strategy improves user experience and ensures the resilience and reliability of cloud services, making them essential for modern businesses.

2.1.5 Conclusion

In Summary, this chapter has covered fundamental aspects of cloud computing in the IT industry, with a focus on understanding the landscape of cloud service providers, their offering models, and emerging technologies. The discussion covered the major offering models and their levels of customisation and control: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Additionally, the chapter emphasises the significance of emerging technologies in cloud computing and the crucial role of agile development methods in effectively utilising these advancements, especially in the financial sector. This foundational understanding of cloud computing sets the stage for exploring specific cloud migration strategies, particularly from AWS to Azure, in the following sections.

2.2 Cloud Migration - Comparative Analysis

In this part the thesis will cover and analyse different aspects of cloud migration. Firstly, the chapter will cover different fundamentals of cloud migration and briefly cover the most common cloud migration methods. Secondly, this part of the thesis will analyse the different migration strategies found in academic and industrial examples. Thirdly, the paper will discuss the different techniques and tools provided by the cloud providers. Fourthly, the paper will examine the synthesis of best practices for cloud migration derived from academic literature, industry guidelines, and case studies of AWS to Azure migrations. Lastly, this chapter will cover common challenges faced during a cloud migration.

2.2.1 Introduction

The role of this part in the thesis is to investigate one of the tree important topics in the literature review. In this chapter the thesis will dive into different aspects of cloud migration strategies, differences between cloud providers so this information can be later used when implementing the cloud-to-cloud framework itself. This part is very crucial for the final outcome of this thesis, since it is very central to understand the different cloud technologies and how they differ between the different providers. By investigating different strategies, we can analyse their possible affect in the context of the company the framework is developed. By creating a better understanding of the

recommended ways of working and which tools to use, this thesis is able to provide the best possible solution in the mentioned context.

2.2.2 Fundamentals of Cloud Migration

Cloud-to-cloud migration refers to the process of transferring data, and workloads from one cloud provider to another. To migrate the application there are several migration strategies for different purposes and scenarios. The selection of migration strategy is determined through an analysis of the application's characteristics, complexity of the migration process, and the objective of the migration. The process consists of steps where the application needs, technical and operational challenges, outcomes, cost efficiency, performance and scalability are assessed. The goal is to identify a strategy that ensures a seamless migration, but also optimises the application functionality. Although the migration methods have the same goal – to move software from cloud provider to another, they have fundamental differences in what steps are taken, and how much effort is put into them. In this section the thesis covers different common cloud migration strategies and discusses the variation they provided for different application complexities.

2.2.3 Migration Strategies

The idea of an efficient migration strategy is that it helps the developer or development team to execute the application migration as time and resource effectively as possible, while simultaneously following the limitations created by the business solution. The cloud migration strategies consist of various technical and business variables that affects the decision on which cloud migration protocol to follow. Every method is tailored for their own specific purpose and contain their own pros and cons. In this chapter the thesis will cover some of the most known and used migration strategies.

AWS 6r's

Stephan Orban from Amazon Web Services recommends six different migrating method for moving applications to Cloud [27]. These methods are specially focused on moving applications from on-premises to the cloud, but the same principals can be used to move applications between cloud platforms.

The first strategy is "Retire", that happens as well be the least demanding method of the six different strategies. As the name states the application on AWS does not need to be migrated at all, since it can be shut down instead. Despite the absence of data from the company regarding obsolete AWS services that are no longer in active use, it is reasonable to assume that an

organisation of this magnitude will possess legacy applications that have been partially overlooked and are now facing retirement as the migration to Azure proceeds. Moreover, harmonising the portfolio of managed applications is in a central part of the company's strategic objectives, thus leading to the deprecation of unused applications.

The second strategy is "Retain". As the development team plan their migration to Azure, they may encounter dependencies on AWS that cannot yet be transitioned to Azure. For instance, they might be utilising a service that has not been migrated to Azure, thus complicating the migration process unnecessarily. In such circumstances, the team could initiate an preliminary migration plan. This would involve mapping existing dependencies, gradually reducing them, and preparing for tests in a hybrid environment to ensure that the system maintains full functionality throughout the transition.

The third migration method is "Rehost", which is commonly referred to as lift and shift is one of the most common migration strategies. As the name states, the methods typically consist of processes where an application is lift from its previous environment and moved to another platform. Rehost is one of the most simple and quickest way to migrate an application since it does not require any major code or architecture changes. This leads to a fairly easy compliance and security management, since these properties have to be only configured to the new resource. However, depending on the complexity on the previous platform, rehosting an application might not take full advantage of cloud native applications features such as scalability. In addition, rehosting an application avoids the management of technical debt, that may increase dramatically when problems are migrated. Azure provides various different tools to execute a rehost migration process. For instance, Azure Migrate offers various of different tools that can be utilised, such as Migration and Modernisation for migrating servers [38]. Additionally, Azure Files can be utilised to facilitate a lift-and-shift process. Rehosting an application could be done by using Azure Files. Azure Files provides a mechanism where cloud-based virtual machines can access file shares that mirror those found in an external environment [73]. Rehost in the context of the framework might be a great solution for container-based applications and virtual machines.

The fourth common migration method is "Repurchase". This approach involves replacing an internally managed service with third-party software purchased from a vendor. During the company's migration to Azure, this could apply to proprietary systems such as Customer Relationship Management (CRM) or Enterprise Resource Planning (ERP) systems that are currently hosted on AWS. Additionally, the organisation uses AWS-specific

services like Amazon Redshift for data warehousing, Amazon WorkMail and Amazon Chime for email and communications, and AWS Shield and AWS WAF for security and compliance. Each of these tools has counterparts on Azure, which the team should consider as potential replacements when migrating their infrastructure. Choosing Azure's offerings such as Azure Synapse Analytics as an alternative to Redshift and Azure Sentinel for advanced security management could lead to improved integration, enhanced functionalities, and potentially reduced operational costs.

The fifth migration method covered in this chapter is “Replatform”. Replatforming does not differ much from rehosting. The core fundamental is very similar, the goal is to migrate the service without completely rewriting the application. However, when replatforming a service the goal is to take advantage of the capabilities provided by the new cloud platform provider. This includes small changes depending on the service, without completely overhauling its core architecture. Typical services that could face replatforming could be self-managed Oracle or SQL database on AWS to Azure SQL Database for PostgreSQL. Replatforming is often a recommended migration method due to the benefits with small investments. With replatforming the service might see several technical improvements such as performance, scalability, and reliability. Additionally, replatforming can lead to better disaster recovery capabilities, which are critical in the company’s context.

The sixth and last migration method is “Refactor”. Refactor is the most time-consuming migration method and is usually executed with a clear intent to improve the service as cloud migration is a good opportunity to harmonise applications and services. Refactoring an application gives the possibility to add features cloud native application tend to have, such as scalability, performance and good cost management, that can be hard to achieve with other migration methods. Refactor method could be the right strategy if the application is facing issues as bottlenecks, technical debt or is in need of new cloud-native features. In a scenario like this refactoring the application while migrating it to Azure could help the team solve multiple issues simultaneously.

Four Methods for Migrating Legacy Applications

Muhammad Ehsan Rana and Wan Nurhayati Wan Ab Rahman describes four different cloud migration methods in their paper “A Review of Cloud Migration Techniques and Models for Legacy Applications: Key Considerations and Potential Concerns” in 2018 [72]. Even the methods are focusing on migrating legacy applications from on-premises to cloud, can the same fundamental ideology be used in cloud-to-cloud migration.

In the article the authors discuss four different migration methods for legacy applications. The first method is Forklift Migration, which is similar to Rehost method recommended by AWS. In Forklift Method the whole application is moved as it is and instead of moving it part by part. This method clearly has its limitations, and the authors recommend having a clearly planned rollback method if the migration is not proceeding as planned.

The second method is called Hybrid Migration, which focuses on moving and verifying application part by part. With this method possibility for unexpected behaviour can be minimised. However, the second method is more time consuming compared to the first one, since every component must be individually wrapped temporary and tested in the new environment.

The third method discussed by the authors is Phased Migration, where the application is migrated to the new platform in smaller phases. There are different ways to dictate the details and length of each phase, but the fundamental idea is that the application is divided into smaller parts, that are managed and tested individually. The authors don't recommend this method to applications that cannot afford to have long downtime.

The fourth and last migration method is Big Bang Method, which resembles refactor method which the thesis covered earlier. The fundamental idea of this method is to reimplement the application completely from beginning and replace old legacy dependencies with new tools.

2.2.4 Phasing Cloud Migration

Phasing cloud migration is a crucial part of a migration strategy. Cloud migration is separated into different sections to ensure a more manageable and structured process, allowing for thorough evaluation and optimisation of specific requirements at each stage, mitigating risks, and minimising disruption to ongoing operations.

In this chapter the thesis will cover seven different migration phases Tom Laszewski and Prakash Nauduri discuss in their book "Migrating to the Cloud" [29]. The book works as a reference guide for cloud migration for development teams moving to Oracle cloud, covering a seven-phased cloud migration that can be compared to the well-known waterfall methodology, where first phase "Requirements Gathering" is replaced with "Assessment". In addition to Assessment, the migration methodology consists of six other phases which are Analysis and Design, Migration, Testing, Optimisation, Deployment and Post-Production Support. See Figure 3 for a visualised representation of the migration process and phases. Although the book focuses on

migrating client and server applications to the Oracle cloud, the fundamentals of the cloud migration phasing can be utilised in AWS to Azure migration.

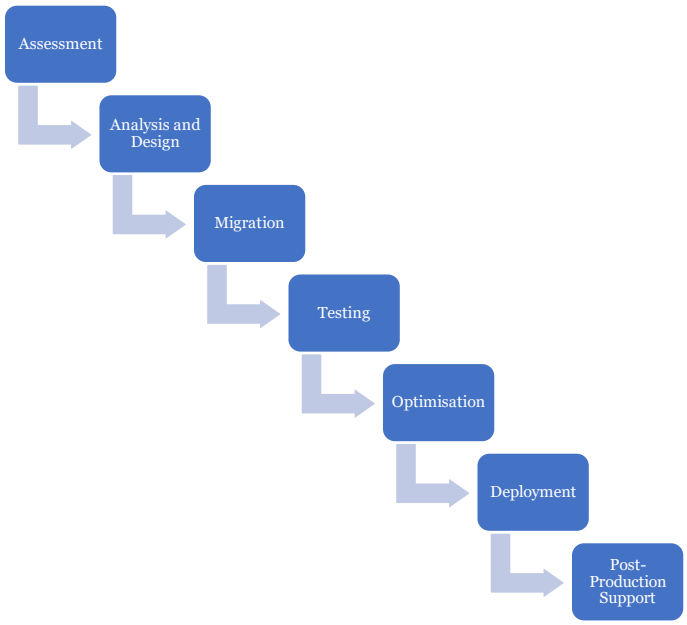

Figure 3, Migration Phases [29]

Assessment

The first phase of the cloud migration process is the Assessment phase, where various aspects of the current environment are thoroughly evaluated. During this phase, key elements such as project management information, potential costs, migration approaches, and tools are evaluated. The primary objective of the Assessment phase is to understand the key drivers behind the migration and create an inventory and clear understanding of the current environment and migration readiness. In addition, Assessment phase is used to identify challenges and requirements that might influence the migration, including expiring licenses, urgency, and necessary application changes.

Additionally, Assessment phase involves making critical decisions regarding migration tools and assessing the need for training requirements for the IT staff. It also includes evaluating the resources required for the target database and other associated resources. Furthermore, an estimate of the migration effort is often developed during this phase, providing a clearer understanding of the scope and complexity of the migration project. The Assessment phase is estimated to span from days to weeks depending heavily on the application size and complexity and migration readiness.

Analysis and Design

The second phase of cloud migration is Analysis and Design. The purpose of the phase is to determine the implementation details on the target platform. During the second phase a plan for the migration is created that utilises the features and functionalities provided on the new cloud platform, while simultaneously addressing differences between the source and target platforms. The objective is to utilise the features Azure provides as efficiently as possible, while ensuring minimal disruption and optimal performance for migrated applications.

One of key components in "Analysis and Design" phase is service mapping, where services used on earlier cloud provider is identified and mapped to their counterparts on the new platform. In the context of AWS and Azure this would mean for instance understanding how AWS EC2 map to Azure Virtual Machines, AWS S3 to Azure Blob Storage and AWS RDS to Azure SQL.

Additionally, during the second phase data migration strategy is planned with focus on data integrity, security, and minimal downtime. This consists of choosing appropriate data transfer methods, handling data format differences and ensuring that data is consistent and secure during the transfer process.

Another critical part of Analysis and Design phase is considering application compatibility between the two platforms. This involves assessing application dependencies, middleware and frameworks to ensure that the application functions correctly on the new platform. Adjustments might be required in configuration files, scripts and code to fulfil platform-specific requirements.

In addition, security and compliance requirements should be planned in Analysis and Design phase. Both AWS and Azure offer security features, but their configuration differ. Identifying and planning encryption, password management and network security are vital for setting up Azure environment to match or exceed the current security level in AWS.

Another part that should be planned during Analysis and Design phase is performance optimisation. The migration of AWS native functionalities, identified earlier in assessment phase, must be planned to fully leverage Azure cloud's capabilities. This involves selecting appropriate instance types, storage options, and networking configurations to ensure that the application will run efficiently and cost-effectively in Azure.

Migration

After the assessment of the current portfolio and planning the migration strategy, the next step is to execute the migration where the actual transfer of objects and data occurs. In the book [29] Migration phase is separated into three different categories which are Database Schema Migration, Data Migration, and Application Migration. The processes and steps around migration itself are designed to enhance the migration process and ensure the decisions and tools are correct. The information and knowledge about application complexity and migration readiness should be used to define the proper methods and timeline how the application should be migrated. Which parts of the application need to be completely rewritten, and which parts can be moved with little to no modifications.

Testing

After all the necessary components are migrated to the new host platform the next logical step would be to verify the migration successfulness. The authors state in their book [29] that application and database testing after migration tends to be require most effort.

Although many testing tasks can be executed with cloud platforms offering automated testing tools or with relatively simple scripting, some parts include abstract aspects like the business logic behind an application, which requires a more nuanced approach to validate accurately and ensure the integrity of the system.

However, as most of the testing involves tasks like testing of functions, triggers, application interaction with cloud native tools, database maintenance scripts and data verification, most of the testing and verification can be executed with automated tools. Automated testing for instance in CI/CD pipelines creates a dedicated phase where the application or new feature is validated. Not only creates this a standardised level of quality, but as well reassures the application's functionality for the developer executing the migration.

Optimisation

Migrating application, databases and other components across cloud platforms might affect the application's performance. According to the authors [29] one of the reasons might be the high optimisation done in source cloud platform that possibly cannot be migrated directly as it is. Some of the optimisation possibilities can be detected already during the migration phase itself, but the authors [29] state that having a two-to-three-month

optimisation period after the initial migration is very common. However, most bottlenecks must be resolved quickly during or soon after the migration to avoid limitation in performance and bad customer experience.

According to the authors [29] there are a number of steps one can follow to cover the most important aspects of optimisation in cloud environment. Firstly, many solutions on source cloud platform consist of various different cloud native solutions which helps the application to utilise the features cloud platforms offer. These native solutions tend to be complicated to migrate as they are, and often require a refactoring process to utilise the cloud native solutions provided by the target cloud platform. This should be planned already during the assessment phase, since modifications after the migration might cause issues with dependencies and will require more time and effort.

Secondly, the cloud platforms handle workloads differently. According to the authors [29] many teams and developers might underestimate the required workload or has a misunderstanding how it is handled in the target system. Microsoft Azure offers a number of tools to manage workloads such as Azure Virtual Machine Scale Sets and Azure Kubernetes Service.

Deployment

Deployment phase is the second last stage of the migration process. During the deployment phase the application should be migrated to the target cloud platform and tested thoroughly to validate that the software is running as intended. The authors [29] discuss the importance of hardware and software licensing, purchase, installation and configuration, since these are central topic in an on-premises to cloud migration. However, in the context of this thesis and the financial corporation, most of these issues are solved on a higher organisational level than where the developer working with the cloud migration is focusing on. This being said, in the context of this thesis these issues are irrelevant and most of the automated deployment pipelines will be created and supported by other internal teams within the company.

However, deployment phase still is an important stage of the migration process. Other central aspects covered by the authors [29] can be relevant although there are fundamental differences between the environments the authors base their conclusions on and the environment the financial company and this thesis is focusing on. Aspects covered by the authors such as rollback compatibility and backup and recovery are important aspects when deploying into production in the new environment for the first time. Tools and services such as Azure Backup, Azure Site Recovery (ASR), version control tools such as GitHub and Azure DevOps are just an example of the vast catalogue offered by Azure to tackle backup and rollback related issues.

Post-Production Support

Last stage in the cloud migration phases provided be authors [29] is Post-Production Support. Although all the migration phases would we be executed carefully, mistakes in new environment can and likely will arise. It is crucial to have an idea of the tools and methods that can be used to detect and solve issues in the new deployment. Additionally, some core features might have been tailored for the new cloud environment, that may require clarification for consumers. According to the authors the most common issues may arise due to the unfamiliarity with the new environment, applications not performing as expected, and data representation such as formatting or time and time required to administrate or manage the new environment. Tools like Azure Monitor, Azure Application Insights and Azure Service Health can be used to detect and raise possible issues with the installation in Microsoft Azure Cloud. Additionally, tools like Azure Portal can be utilised to find documentation or instructions on how specific tools communicate or explain error codes.

2.2.5 Migrating Databases

Database schema Migration is the process of transferring tables, indexes and views to a new cloud platform. In the book [29] the authors state that migration tools today are well equipped to execute database schema migrations quickly and accurately. The authors have listed the most important details to consider during a database schema migration. Firstly, ensuring schema completeness before migration is crucial to avoid errors during schema creation on the new platform. A single application can use several different schemas and databases, which can create a circular dependency between these services. These dependencies may not be migrated simultaneously, that can lead to confusing errors when the new schema or database is deployed on the new platform. Recompiling and verifying invalid objects are crucial to ensure a successful migration.

Secondly, identifying and possibly clearing the database from automatically filled default values is an important part of planning a cloud migration. The system function used for example add the current time to a column could for example in AWS be `CURRENT_TIMESTAMP` which might not exist or has a different syntax on Azure. This may lead to a table creation failure or an invalid depended object error.

Thirdly, the authors mention the usage of clustered indexes to enhance cloud migration. Utilising a physically sorted data storage matched with the logical order (indexes) the database can reduce time to return sorted data and to

access data by co-locating the index and the actual object. When migrating an Index-Organised Table (IOT) from AWS to Azure it is crucial to understand the equivalent features and their differences. In this context, the source database could be for example Amazon RDS that supports IOTs, or Amazon Aurora that does not directly support IOTs but uses clustered indexes to achieve similar performance benefit. In contrast, in Azure the alternative services would be Azure AQL Database, Azure Database for PostgreSQL, or Azure Database for MySQL that supports clustered indexes to receive similar functionality to IOTs.

Fourthly, a common way to enhance database performance is to partition database tables into smaller segments. This makes the database management easier as well as reduces the in the overall amount of data scanned. However, the authors state that migration tools does not generally migrate partition-related configurations when the database itself is migrated. Therefore, it is crucial to take possible partition configurations into account after migrating the database and before the data migration.

Lastly, the authors recommend focusing on user and role management and object names when working on a database migration with a team. Users having access to right resources and having proper naming guidelines may seem obvious, but avoiding simple and unnecessary mistakes can enhance the efficiency and overall experience of the migration process.

After database data is migrated successfully, next step recommended by the authors would be to migrate database stored program. Migrating database stored program includes transferring stored procedures, triggers and tables representing the query result. These functionalities are often used to implement the business logic into the relational database. In Microsoft SQL Server and Sybase, procedures and triggers are a core services to manage CRUD (Create, Read, Update, Delete) operations. However, programs stored in databases might create inflexibility when it comes to migrating the service to another platform. To enhance the migration process, the authors have recommended a few methods.

Firstly, in the book the authors recommend to clean and optimise the code. This can be executed by conducting code reviews where procedures and functions can be verified and tested for efficiency. Through code review even old and unused features can be identified and removed before the migration process. In the context of AWS to Azure migration some of the cloud native tools can be utilised to support the migration process. Although the converted stored procedures and triggers must be tested thoroughly to validate successfulness and efficiency.

Secondly, according to the authors databases that are heavily using procedures and triggers it is common to see utilising nested stored procedure calls. This creates dependencies between different databases and procedures. Automated migration tools are rarely equipped to handle and detect dependencies created by nested stored procedures correctly. This can lead to confusing error messages, since the source procedure initiating the nested call will not likely now how to handle the error from target procedure. To avoid this, the authors recommend paying close attention, detect and carefully test nested procedure calls before and after the migration process.

Thirdly, extensive use of temporary tables is a problem that the authors recommend solving by refactoring the database during the migration process. There are a few reasons behind the usage of temporary tables. The authors state in their book that older databases can have restrictions on the number of tables that could be joined in a query efficiently. In AWS a developer might have stored different data types in different tables which are used side by side in the same application. According to the authors, migration tools are usually effective in transferring temporary tables and their links as they are configured in the source database. However, the migration may affect the communication performance in the target environment. Analysing and redesigning workflows could reduce dependencies between temporary tables and make the database more efficient by utilising cloud native solutions the target host provides.

2.2.6 Migrating Data

After database schema migration is done following stage recommended by the authors [29] would be to migrate the data itself. In addition to migrating the data to the correct new platform, data migration is also a way to validate the database schema migration successfulness. The authors recommend keeping the migration process as automated as possible by using different migration tools and scripts. However, database size should affect the migration process planning and chosen tools. When migrating several terabytes of data, database features such as partitioning and compression are possible ways to enhance the migration efficiency.

Database sizes vary, which should affect the decision making on the chosen migration strategy and tools. The authors recommend using automated tools to migrate data. However, databases over the size of two terabytes usually require careful planning on the right strategy, tools, and use of database features such as partitioning and compression. A method recommended by the authors is to manually migrate smaller portions of data to validate and debug the migration process and database configurations.

Common challenges when migrating data according to the authors are narrow migration window and limited system resources. However, there is few methods one can follow to make migration more efficient. The methods the authors recommend are parallel data extraction from source database and loading into the target database, multithreaded process for data loading, avoiding index maintenance during data loading, and reduction of I/O operations using staging areas via named pipelines for data transfer between source and target database.

Generally, data migration is separated into offline and online data migration. The distinct difference between these methods is how the source and target database communicate during the process. In an online data migration, the data is migrated in real-time, maintaining a continuous connection between the source and target database. Online migration can easily create additional load on the source system and impact the application performance. Therefore, it is recommended by the authors to use online migration for smaller databases. In contrast, in offline migration the database is extracted to flat files and then uploaded into the target database. Since during an offline migration the extraction and loading process are disconnected, the migration process can be separated into smaller time windows, where the migration can be executed outside peak hours. Cloud providers provide various tools for both methods.

The book discussed in this chapter covers additionally recommendation and tools to use during a data migration. The authors recommend a Changed data capture method (CDC). Configuring a CDC during the data migration allows the changes made in the source database to be replicated automatically to target database. Utilising a CDC can make a migration process of larger databases more efficient. The authors recommend two different ways to set up a CDC for data migration. First method recommended by the authors is log mining. According to the book log mining is the most used method technique for implementing CDC. Log mining consists of reading the transaction logs from the source database and extract as they are executed. After the changes in source database have been captured, the changes are either immediately transferred and executed at the target database or stored and transmitted later at a chosen time. Second method recommended by the authors is using triggers. Triggers can be implemented into the source database to capture changes and write them to a staging table, afterwards the changes are replicated into the target database. According to the authors, setting up and configuring triggers in the source database can be a challenging task depending on the database complexity. Additionally, triggers can significantly impact the source database performance, which is why triggers are not the most popular technique to set up CDC.

2.2.7 Migrating Applications

Migrating applications to the new cloud platform can be described as the core task of the whole migration process. The authors Laszewski and Nauduri separates migrated application in their book [29] into two categories that are customised applications and generic applications. According to the book customised application tend to be in-house developed application tailored to support specific business applications. Additionally, customised applications tend utilise different platform and environment specific native solutions to leverage best possible performance and integration. In addition, customised solution might reflect the developers' skill set and preferences, that varies among software professionals.

Customised applications lead to customised solutions, which can result in complicated and highly interdependent systems that can become increasingly complex over time. With this in mind, customised application will likely require native solution and refactoring when migrating to a new platform where badly managed version control, documentation and knowledge transfer will make the process less effective.

In contrast, generic applications, also stated in the book [29] as packaged applications, typically is less complex and dependent on other solutions. As the authors state, generic applications migration should not be a complex task, even though the application must be tested thoroughly on target platform. In the context of cloud-to-cloud migration, migrating a generic application can reflect the process of migrating a container-based application between cloud platforms.

2.2.8 Verifying Cloud Migration Success

Verifying the success of migrating applications, databases, or other components between cloud platforms, such as AWS to Azure, is a fundamental aspect of a migration strategy. This verification is essential to ensure that the system operates correctly and efficiently in the new environment. This part of the process involves testing and validation to verify the application's performance, functionality, and data integrity. In this section the thesis will cover four different methodologies and practises that can be used to verify migration success.

Functional Validation

Functional validation focuses on ensuring that all the application features and functionalities work correctly in the target environment. It involves assessment of application behaviour, usability, and compliance requirements.

On of the main components of Functional Validation is functional testing. The goal with functional testing is to assure that all features and functionalities of the application work in the target cloud environment as they worked in the source environment. In the paper "Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments." [30] the authors emphasise the importance of a well-structured framework to manage and pace complex cloud migrations, especially when working with legacy software. Testing in continuous integration and continuous delivery (CI/CD) pipelines is an example of automating the testing. Additionally, the developer could use as well other testing tools such as Postman for API testing or Selenium for user interface testing, ensuring that the core application functionality maintains its intended behaviour.

Another common method to validate application functionality is User Acceptance Testing (UAT). UAT is a method used to validate that the application meets all the set requirements, expectation, and complies with the relevant industry standards. UAT validates functionality from an end-user perspective, identifying issues not caught in earlier testing phases. UAT can be executed with demonstration sessions with stakeholders or in other feedback collection methods. However, automated and parallel UAT have been demonstrated enhancing efficiency in detecting functional inconsistencies, synchronisation failures, and non-deterministic behaviours in multi-tenant SaaS systems in the paper "Evaluating the User Acceptance Testing for Multi-tenant Cloud Applications" [33]. Tools like Selenium or Junit can be automated to test various UAT scenarios. The tools can be utilised to simulate user interaction as well as automate backend and API tests to gain a comprehensive validation of all application functionalities and application logic.

Performance and Compatibility Validation

Performance and compatibility validation ensures that the application functions efficiently under various conditions and remains compatible across different environments, such as devices, browsers, operating systems, and network conditions.

A central method verifying application performance and migration success is performance testing. Performance testing can be used to ensure that the application functions as it should in different conditions and that the cloud native features are utilised effectively. Performance testing has different methods to test application adaptability in different scenarios, such as load testing and stress testing. The idea is to increase the load on the application suddenly to simulate peak traffic and detect possible bottlenecks. Additionally, load

testing can be used to detect application's costs during periods of less usage and generally see how well the application adapts to sudden changes.

Since cloud platforms are scalable and flexible, allowing for dynamic resource allocation based on workload demands, one could think that performance testing is not required with cloud native application. However, this is not accurate. As Mukherjee, Wang, and Krishnamurthy demonstrate in their study "Performance Testing Web Applications on the Cloud" [32], systematic experimentation is essential to eliminate bottlenecks unrelated to the server being tested. This ensures that the true performance impact of the cloud platform can be accurately determined. Cloud solutions consist not only of cloud-native components and pre-configured servers but also of customised solutions that can bottleneck under high loads. Performance testing is crucial for detecting these issues.

Compatibility testing is a method for verifying that applications, services, and other components function correctly across different environments, such as various devices, browsers, operating systems, and network conditions, both during and after migration. In the context of cloud-to-cloud migration compatibility testing plays a vital role in ensuring that the applications migrated across cloud providers continue to operate in the new environment. With compatibility testing the developer responsible for the migration can ensure that the application maintains a consistent user experience.

Azure Cloud offers tools such as Azure DevTest Labs and can additionally utilise tools such as BrowserStack and Sauce Labs to setup various test environments to ensure application compatibility across different web browsers and devices. In addition, Azure Cloud offers tools for network testing. However, these tests are irrelevant for the purpose of this thesis, since the company's network configurations will likely be built and supported by external parties.

Data and Recovery Validation

Data and recovery validation focuses on ensuring the integrity and recoverability of data post-migration, ensuring that applications can recover swiftly from failures and maintain business continuity.

One of the methods for verifying data accuracy during and after a cloud-to-cloud migration is Data Integrity testing. The article "A secure and efficient data migration over cloud computing "[31] discusses the use of hash-based techniques for validating data integrity during cloud database migration. Comparing data between the source and target databases can help identify

discrepancies. Techniques such as record count verification and hash-based data validation are essential to detect any data corruption or loss.

Disaster Recovery (DR) Testing is a method for verifying application recovery processes. It ensures that the migrated application and its data can be restored or rolled back to a previous working state, minimising the time the application remains unusable. DR Testing ensures that the mechanisms and procedures set up for an application recovery works as indented and that the application can meet Recovery Time Objectives (RTO) and Recovery Point Objectives (RPO). According to Andrade and Nogueira's study on cloud-based disaster recovery solutions [35], performance evaluation is central for validating the effectiveness of DR strategies. This includes metrics such as availability, response time, and throughput ensuring that the DR solution not only recovers data but also maintains an optimal post-recovery performance level.

In Azure, tools like Azure Site Recovery and Azure Backup are core services when implementing and testing DR strategies. Azure Site Recovery helps orchestrate the replication, failover, and recovery of applications, ensuring that critical business functions continue to operate during disruptions. Azure Backup provides secure and efficient backup solutions to safeguard data, enabling quick restoration in the event of data loss or corruption. Combining these tools in the development teams DR testing toolkit, these tools can facilitate extensive DR testing, ensuring that applications can swiftly recover from failures and maintain business continuity.

Ongoing Monitoring and Validation

Ongoing monitoring and validation are critical components of verifying the success of cloud-to-cloud migration. This process ensures that the applications perform optimally in their new environment and that any potential issues are quickly detected and resolved. By providing real-time insights into system performance and user activity, monitoring and logging help maintain application stability and reliability post-migration.

Regression Testing is a testing methodology used in version control tools to ensure that changes do not affect the previous functionalities. Regression Testing can include for example re-running previous tests that existing functionalities remains unaffected. In cloud-to-cloud migration Regression Testing plays a vital role. During the migration a lot of configurations and small adjustments are being made, depending on the complexity and application migration readiness that can affect the application functionalities on target cloud platform.

The study by Godefroid, Lehmann, and Polishchuk [34] emphasises the importance of differential regression testing for REST APIs in cloud environments. Their approach compares different versions of APIs to detect regressions effectively, which is essential for maintaining functionality and performance across cloud migrations. Automated regression testing tools can streamline this process, ensuring that applications continue to operate as expected in post-migration. During a cloud-to-cloud migration automated Regression tests could be utilised to validate API consistency and detect possible configuration issues.

Monitoring and logging are critical components on the process of verifying the success of cloud-to-cloud migration. Cloud platforms offer different services that can be used to ensure that the applications can perform optimally in their new environment and that any possible issues can be quickly detected and resolved. By providing real-time insights into system performance and user activity, monitoring and logging help maintain application stability and reliability post-migration.

Continuous monitoring involves tracking the health of the applications and infrastructure in real-time by measuring the application's performance and availability. Monitoring helps development teams detect any anomalies, performance bottlenecks, and potential failures early. Tools like Azure Monitor and AWS CloudWatch are examples of such tools, and they can easily be integrated with other cloud services, offering metrics, alerts, triggers and dashboards to visualise the application's health and performance data.

Log Management involves collecting, storing, and analysing log data from different parts of the application and infrastructure. The log data provides detailed records of system events, errors, and user activities. Azure Log Analytics and AWS CloudTrail are tools that can be used to gather and analyse logs. The logs can be utilised troubleshooting issues, perform root cause analysis, and ensure compliance with regulatory requirements.

Conclusion

Verifying the success of migrating applications, databases, or other components between cloud platforms is crucial to ensure seamless operation in the new environment. This involves a comprehensive approach that includes functional validation, performance and compatibility validation, data and recovery validation, and ongoing monitoring. Cloud providers offer tools for building tests as well as managing and creating different scenarios to ensure the complete success of the cloud-to-cloud migration process.

By employing the covered methodologies and leveraging appropriate tools, development teams can ensure a successful cloud-to-cloud migration, maintaining application stability, performance, and reliability in the new cloud environment.

2.2.9 Migration Example Studies

In this chapter the thesis will cover some example case studies of cloud migrations between on-premises, cloud, and multi-cloud. Companies rarely execute their migration on one go, but instead migrates to a new platform piece by piece. In this chapter the thesis will cover different migration cases with different environments, to acquire a broader understanding how companies in different sizes execute cloud migration and what tools they utilise.

Netflix to AWS – 2008-2016

The video streaming platform Netflix introduced video-on-demand online service in 2007 in addition to their DVD rental service. During this time Netflix built two data centres to support the streaming needs. However, Netflix faced issues constantly with their relational databases, and once even a data centre failure shut down Netflix operations for three days. Netflix was growing in a rapid speed, which lead Netflix migrating to AWS in 2008. The software issues Netflix faced with their on-premises datacentres were fixed by rebuilding the whole infrastructure with AWS native solutions and services [51]. Architecturally, Netflix migrated from a monolithic app to hundreds of micro-services, and denormalized and their data model, using NoSQL databases [52].

Today, Netflix is well known partner with AWS and is utilising several different cloud native tools to optimise their services. Netflix utilises Amazon Kinesis Data Streams (Amazon KDS) to gather and analyse large volumes of data in real-time, while storing this data in Amazon S3 buckets for further analysis. AWS Lambda streamlines processes, replacing outdated procedures with automated data manipulation, backup decisions, and security checks. Amazon Route 53 provides a solid DNS infrastructure, ensuring seamless streaming by efficiently routing traffic during server failures or region overloads. Amazon Redshift enables Netflix to analyse vast amounts of data quickly, refining content recommendations and optimising streaming through its columnar storage for complex queries. Amazon Machine Learning (AML) personalises viewer experiences by analysing user behaviour and preferences, offering tailored content recommendations. Amazon Kinesis powers Netflix's real-time analytics, processing diverse streaming data to improve content recommendations and streaming quality. Finally, Amazon Elastic Compute Cloud (EC2) ensures scalability, powering the

computational engines behind content delivery, transcoding, and other compute-intensive tasks, providing a smooth streaming experience for Netflix's global audience [51].

After seven years in 2016 Netflix had finally migrated all their services to AWS. One of the primary reasons for the extended timeline was the sheer scale of Netflix's operations, which included millions of users and vast amounts of data. The migration required re-architecting Netflix's entire infrastructure to leverage the cloud effectively. Each component of the service had to be evaluated, tested, and transitioned without disrupting the user experience. Another significant factor was the need for Netflix to develop new tools and processes to handle the migration. This included creating automated testing environments, deployment strategies, and monitoring systems to ensure smooth operation during and after the transition. The process also involved extensive training for the engineering teams to adapt to the new cloud environment and utilise AWS services efficiently. Additionally, the complexity of maintaining high availability and performance standards during the migration impacted the timeline. Netflix had to ensure that their services remained operational 24/7, even as they moved critical workloads to the cloud. This required a phased approach, migrating services incrementally and continuously testing for stability and performance [51][52][53].

Capital One to AWS – 2012-2020

Capital One partnered up with AWS and migrated to cloud in 2020, which made the company the first financial company in the United States to move completely to cloud. This shift was driven by the investments into software and cloud development to gain greater agility, improved customer experiences, and enhanced data security [55].

Capital One's vice president of software engineering states that the key solutions for their technological advancements during the past years are in the methodologies and tools the company utilises. The vice president (VP) emphasises the importance of agile and efficient CI/CD pipelines to enable products to scale when needed. Additionally, the mentality of "Do the hard part first" has been crucial in the company's cloud migration journey. This has enabled the company to focus on solving the complicated issues migrations might face in the financial industry and helped them leverage the new and modern architecture and tools cloud platforms provide. [55]

The VP comments that the cloud migration to AWS has not been easy, but rather a multi-year project that has required investments and resources from every part of the business. The VP states that best software practices, such as managing technical debt, continuously verifying goals, and integrating

defined operations and built-in controls into their Minimum Viable Product (MVP) and Minimum Desirable Product (MDP) processes, played a crucial role in addressing the complexities effectively. [55]

To become a cloud-first organisation, Capital One had to rethink not only technology development and operations but also talent management. As a result, they pushed education efforts far and wide by introducing the AWS Navigator initiative, putting you in touch with an expert solutions architect for workshops. The Monthly "Invest in Yourself" days and being involved with AWS Certification programs have proven to be central in their workforce training. Capital One increased its technology team to 11,000 employees and switched to agile development and DevOps practices; they also implemented Operational Readiness Reviews in accordance with the requirements of AWS. Powered by this transformation, the company could not only innovate quickly but also operate its technology at scale effectively. [54]

Waze to multi-cloud – 2017

Waze is a company that provides a platform for drivers to communicate with each other and even with law enforcement of possible situations on the roads that might affect traffic [50]. Waze was launched in 2009 and has utilised AWS cloud solutions from early on to support its application's flexibility, facilitating its rapid user growth. Later in 2013 Waze was bought by Google [49]. In 2017, Waze initiated a strategic migration from AWS to a multi-cloud environment involving both AWS and Google Cloud Platform (GCP). The migration was managed only by a small team of two people and was executed quickly in two months impacting 80 million users. The objective was to enhance system resilience and reduce the risk of downtime, a lesson learned from Amazon outages in 2015 [46] and 2017 [47].

The primary goal during the migration was to ensure that the migration process would not affect the developers' daily tasks at Waze. The migration was planned to execute in the background, so that the two cloud platforms can start to communicate through a VPN (virtual private network) as it was one single cloud provider. The communication network between cloud providers was first executed with AWS EC2, AWS Direct Connect, VPN appliances as a middleman between the cloud providers, GCP Cloud VPN, and GCP Compute Engine. This allowed the virtual machines on both cloud platforms to communicate with each other and work as one cloud platform. For the source VM the target VM on the other cloud provider is just another internal IP address. However, the solution had some flaws, having a VPN middleman led to less performance specially during high traffic. To tackle this issue Google has introduced a service called Google Cloud Interconnect, that enables a company as Waze to physically connect their AWS Region with GCP. With GCP

Interconnect, Waze does not need to rely on VPN connections, which improves performance and reliability. This direct connection allows for seamless communication between AWS and GCP, ensuring efficient data transfer and robust application performance even during peak usage times. [48]

After a successful migration the developer responsible for the application could choose either cloud platforms' services or tools to continue working with. To solve deployment issues in multi-cloud environment Waze introduced Spinnaker to manage their multi-cloud setup to their developers, an open-software cloud deployment tool developed by Netflix and managed by several other companies such as Microsoft and Google. With Spinnaker Waze does not need to manage both GCP and AWS individually, but instead Spinnaker can deploy and manage an application on several cloud providers simultaneously. With Spinnaker the developers don't need to deploy the application to several platforms and manage them with their individual tools, instead everything can be done from Spinnaker directly.

With these tools Waze was able to execute a very effective cloud migration. The Waze speaker emphasises that a strong rollback strategy and efficient tools have enabled quick deployments and fast migrations without significantly impacting developers' workflow [48].

2.2.10 Migration Tools

As well as a normal cloud migration, a cloud-to-cloud migration can be a complicated and demanding process depending on the application and infrastructure complexity and migration readiness. To support organisations in cloud migration the cloud providers like Microsoft Azure and AWS offer tools to simplify the migration process. Additionally, there are some third-party vendors that offer cloud migration tools to different cloud platforms. The article "Cloud Migration Tools: Overview and Comparison" compares various cloud migration tools utilised in the industry, which will be covered in this chapter [36]. See Table 1 below for an overview of the different cloud migration tools.

Table 1, Summarized Different Cloud Migration Tools [36]

Tool/service name	Vendor name	Nature of tool/service	Cloud type support (public, private, hybrid)	Cloud platform support	Support migration of		
					Data	Application	Process
Azure Migrate	Microsoft	Dynamic	All	Azure	Y	Y	Y
AWS Cloud Data Migration	Amazon	Dynamic	All	AWS	Y	Y	Y
Live Migration	Cloud Endure	Dynamic	All	Azure, AWS, GCP ...	Y	Y	Y
* DynaCenter	Racemi	Dynamic	All	Azure, AWS	Y	Y	Y
* One Hybrid Cloud	CloudVelox	Dynmaic	All	AWS	Y	Y	Y
Cloud Migration and Operations	Dynatrace	Dynamic	All	Azure, AWS, OpenStack ...	Y	Y	Y
MigrationWiz	BitTitan	Dynamic	All	All	-	Y	Y
Science Logic	ScienceLogic	Dynamic	All	Azure, AWS, IBM...	Y	Y	-
App Dynamic	Cisco	Dynamic	All	Azure, AWS	Y	Y	-

* Provided by the source, was not able to verify functionality in Europe.

Azure Migrate – Microsoft

As expected, Azure Migrate tool is one of the alternatives for cloud migration between AWS and Azure. Azure Migrated is provided for free and is also well equipped with various integration in Azure cloud such as “Discovery and assessment”, “Migration and modernisation”, “Data Migration Assistant”, and many more [38]. See Table 1 for more detailed description of Azure Migrate’s service catalogue. Azure Migrate offers a one-shop portal for every tool and process you might need in a cloud migration process, such as assessing, starting, running and tracking your migration process and migrated services. Microsoft Azure supports a variety of different services, everything from ordinary servers, SQL databases, web apps, virtual desktops and data.

Additionally, Azure Migrate includes assessment features that help measuring an application's readiness for migration, estimate the application size, estimate potential costs, and generate a cross-server dependency analysis and recommend optimisation strategies. These features can be used even if the application is currently hosted on a different cloud platform, such as AWS [39].

Table 2, Microsoft Azure Migrate Integrated Tools [38]

Tool	Assess and migrate	Details
Azure Migrate: Discovery and assessment	Discover and assess servers including SQL and web apps	Discover and assess on-premises servers running on VMware, Hyper-V, and physical servers in preparation for migration to Azure.
Migration and modernization	Migrate servers	Migrate VMware VMs, Hyper-V VMs, physical servers, other virtualized servers, and public cloud VMs to Azure.
Data Migration Assistant	Assess SQL Server databases for migration to Azure SQL Database, Azure SQL Managed Instance, or Azure VMs running SQL Server.	Data Migration Assistant is a standalone tool to assess SQL Servers. It helps pinpoint potential problems blocking migration. It identifies unsupported features, new features that can benefit you after migration, and the right path for database migration.
Azure Database Migration Service	Migrate on-premises databases to Azure VMs running SQL Server, Azure SQL Database, or SQL Managed Instances	Azure Database Migration Service offers a fully managed solution that facilitates smooth database transitions from various sources to Azure's data platforms, minimising downtime (online migrations). [71]
Web app migration assistant Assess on-premises web apps and migrate them to Azure.	Azure App Service Migration Assistant is a standalone tool to assess on-premises websites for migration to Azure App Service.	Use Migration Assistant to migrate .NET and PHP web apps to Azure
Azure Data Box	Migrate offline data	Azure Data Box products can be utilised to move large amounts of offline data to Azure.

AWS Cloud Data Migration - Amazon

One of biggest competitors in cloud computing field AWS offers their own cloud migration tools and services as well. Similar to Azure Migrate, AWS mainly supports incoming migrations, which limits its applicability within the context of this thesis focused on cloud-to-cloud migration between different platforms. However, AWS offer tools to assess dependencies and cloud infrastructure in the current AWS environment.

For migration AWS offers tools for a three-step migration process: Assess, Mobilize, and Migrate & Modernize [40]. AWS offer a versatile toolset for the various phases, such as tools for optimisation and licensing and a migration evaluator for the assessment phase. Mobilize and Migrate & Modernize phases offer a vast catalogue of different tools as well for different application types and complexities.

Additionally, other AWS tools can be used to map out AWS infrastructure and dependencies. The article "Visualize your AWS Infrastructure with Amazon Neptune and AWS Config" [41] outlines a method for visualizing AWS application infrastructure using AWS Config and Amazon Neptune. Creating a visual representation of all AWS services utilised by an application can enhance the developer's understanding, which is crucial for those responsible for migrating the application to a new cloud platform. A clear visualization of the native cloud services employed on the previous platform can significantly assist in the assessment and planning stages of the cloud-to-cloud migration.

Live Migration – Cloud Endure

Cloud Endure, now part of AWS, offers robust live migration capabilities through continuous block-level replication, ensuring minimal RPO for all applications and databases. Its automated machine conversion process allows Windows and Linux machines to natively boot and run in AWS, handling all necessary configuration and activation changes. Unlike traditional disaster recovery solutions, Cloud Endure minimises the required resources in the staging area, leveraging AWS's lightweight compute and storage to keep costs low until fully provisioned workloads are needed. The orchestration engine automates the launch of operational workloads in the target AWS region, achieving minimal RTO by cloning disks and provisioning necessary resources. [42]

Cloud Migration and Operations – Dynatrace

The Dynatrace cloud migration tool assists organisations in planning, accelerating, and optimising their cloud migration process. It provides insights

into existing infrastructure, maps dependencies, and helps architect microservices for an efficient migration strategy. During the migration, Dynatrace minimises performance degradation, quickly identifies root causes of issues, and ensures optimal performance. Post-migration, it enhances application performance with causal AI, automation, and deep DevOps integration, offering comprehensive observability and security. [43]

MigrationWiz – BitTitan

MigrationWiz by BigTitan offers automated migration services for emails, archives, and documents. It supports a wide range of source and destination platforms, making it popular among small to medium-sized businesses for its ease of use and comprehensive migration support. [44] MigrationWiz is not targeted for bigger enterprises with complex cloud infrastructure wanting to leverage the cloud native functionalities on PaaS or IaaS platforms.

Science Logic – ScienceLogic

According to the article [36] Science Logic would be one of the key tools when planning and executing a cloud migration. However, according to ScienceLogic's own website [45] their focus has shifted more towards cloud monitoring, dependency mapping, performance optimisation and automation. These are all core processes in cloud migration but cannot independently assist a company or development team in their migration process. By providing monitoring, dependency mapping, and performance optimisation, ScienceLogic supports the broader goals of cloud transformation and operational excellence.

2.2.11 Common Challenges and Mitigation Strategies

Cloud migration can be a very complex and difficult task to execute, and as one can assume there are different challenges depending on the application complexity, migration resources, and application migration readiness. In the following chapter the thesis will cover some of the most common challenges and mitigation strategies against them.

Cloud Migration Security

Firstly, one of the most common concerns when migrating to the cloud and between vendors is security. How should data and applications be migrated, stored and managed as securely as possible. Dalal N. Alharthi developed a three-step Secure Cloud Migration Strategy (SCMS) in his article [56] that covers several different aspects of software testing to ensure migration successfulness. see Figure 4 for visual representation of SCMS. Additionally,

Alharthi emphasises the importance of cloud migration preparation and migration readiness and adaption, which are all topics this thesis has also focused heavily on.

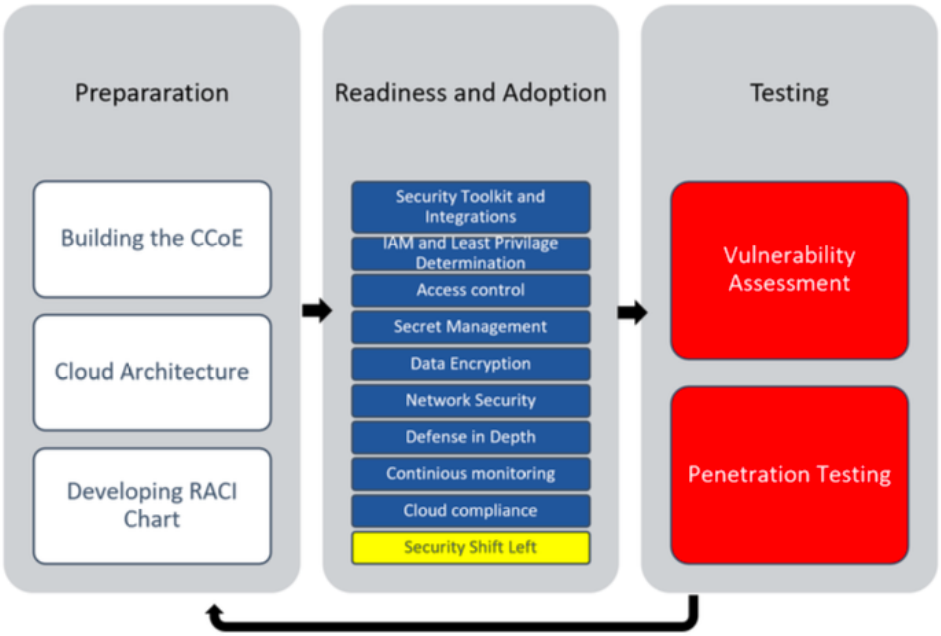

Figure 4, Secure Cloud Migration Strategy (SCMS) [56]

Risk Management and Sustainable Cloud Migration

Secondly, another common concern among companies and developers executing a cloud migration is risk management. Managing risks during cloud migration involves evaluation of cost, environmental impact, data quality, and service continuity. A systematic risk management strategy can identify and analyse risks in the different phases of migration. The authors in the article [57] have divided a sustainable cloud migration into four different dimensions: Economic, Environmental, Social, and Technology. See Figure 5 for visual representation of the four sustainable migration dimensions. Economically sustainable cloud migration evaluates if the migration is economically worth for the organisation. Understanding the costs of outsourcing IT infrastructure for a company will require cost analysis, that this section of sustainable cloud migration covers. Additionally, continuous monitoring after the initial migration is executed will help the development team to decrease their cloud costs [58].

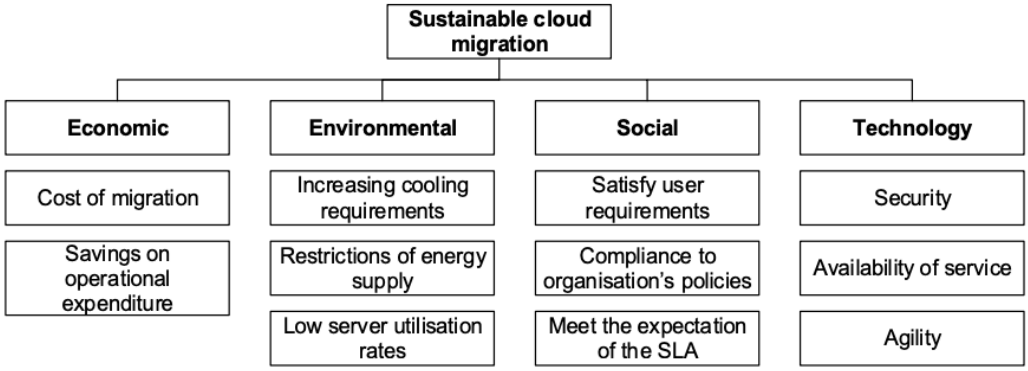

Figure 5, Sustainable Migration Dimensions [57]

An aspect that does not always get as much attention but is as important is Environmental Dimension. Data centres and cloud applications don't use any less energy than an on-premises installation, and when outsourcing infrastructure management, the customer does not have the same ability to make customised solutions to the data centres to manage electricity source, usage or utilisation of the heat produced by data centres. A common issue is that an application is using more resources than it would need, which leads to unnecessary energy usage. The authors recommend focusing on the environmental questions when choosing the cloud provider, such as how the heat generated by data centres are going to be utilised and is the data centre located near customers so no unnecessarily long communication lines needs to be used. Additionally, the authors emphasise the importance of comparing different solution and hardware, if possible, when choosing the cloud components and services for the application. Additionally, scaling down services when not utilised will both save energy and the environment, as well as lower the cost.

Third dimension covered in the article is social dimension. When migrating software between platforms often some parts of the application must be modified or changed. A common problem many developers face is that these modifications do not meet the requirements from users, organisation, or other relevant stakeholders. The author states, that having a clear goal, and requirements set before initiating the migration process is crucial for the final product. Additionally, when utilising a cloud platform, the application and its performance is dependent on a third party. An outage or technical failure at the cloud provider data centre cloud make the hosted application unavailable for its customer. Since the past decade AWS have had outages a few times that have impacted several million people for several days in the United States [46][47]. A customer utilising a cloud platform has no control over potential outages at the cloud provider's data centres; consequently, they will inevitably face the repercussions of application unavailability on their own customers. The authors do not cover the possibility of having a hybrid cloud

solution on several cloud platforms to minimise the possible risk of outage, but examples of these solutions are covered in the thesis in chapter 2.2.9 Migration Example Studies.

The fourth and last dimension is the technology dimension that aims to ensure that the application is scalable, maintainable and secure at the target cloud platform. The cloud computing's main advantage is its ability to share resources and infrastructure to multiple clients. However, using shared space on a third-party platform involves certain risks. These risks include falsification of messages, where someone could tamper with data being sent over the cloud, leading to false information dissemination. There is also the risk of hardware interception, where unauthorized access to the cloud's physical components could result in data breaches. Additionally, information leaks could occur due to weak security measures, and traffic redirection, where network traffic is rerouted without authorization, potentially allowing malicious actors to intercept or manipulate data. These risks highlight the need for robust security measures in cloud migration strategies.

Technical and Operational Challenges

During the process of moving applications across platforms several technical and operational challenges can arise. Every platform has its own infrastructure that may differ in context to others. Two critical areas where these types of challenges might arise are application compatibility and performance optimisation.

Compatibility issues may occur when designing an architecture blueprint of the application infrastructure on the target platform. The tools provided by the source platform might differ or might not exist at all on the target platform. Especially older legacy application might have custom made solutions that have not been updated or managed can affect the application's migration readiness. The authors [58] recommend conducting a comprehensive application assessment to identify compatibility issues before starting on the migration itself. Having a clear understanding of the application's migration readiness on the source platform can make the migration more efficient and additionally may have a positive impact on the application performance on the target platform.

A common technical and operational challenge during a software migration is performance optimisation. The application might utilise source platform native services that cannot be automatically migrated to the target platform. The application is likely utilising tools for auto-scaling, load balancing and performance monitoring that needs to be implemented on the target platform as well. Not utilising similar tools on the target platform will likely affect

the application's performance and user experience. To avoid an issue with performance optimisation during and after cloud migration the authors [58] recommends conducting a comprehensive analysis of the tools used to manage application performance and additionally conduct performance testing to identify bottlenecks on the target platform.

Conclusion

This chapter covered three common problems that tend to arise during a cloud migration, as well as some mitigation strategies how to solve them. As in many other parts in this thesis, the sources discussed mainly on-premises to cloud migration, where old legacy infrastructure might cause some problems that cloud platforms don't necessarily tend to contain. However, the problems covered in this chapter: Security, Risk Management and Sustainable Cloud Migration, and Technical and Operational problems are common issues also during cloud-to-cloud migration, since they don't require an old complex on-premises infrastructure to arise. The three common cloud migration challenges and mitigation methods can be found from the Table 3.

Table 3, Three Common Cloud Migration Challenges and Mitigation Methods

Migration aspect	Challenge	Mitigation
Security	Ensuring data and applications are migrated, stored, and managed securely.	Implement a comprehensive Secure Cloud Migration Strategy (SCMS) and prepare thoroughly for migration readiness and adaptation.
Risk Management and Sustainable Cloud Migration	Managing cost, environmental impact, data quality, and service continuity.	Employ a systematic risk management strategy that evaluates economic, environmental, social, and technological sustainability.
Technical and Operational	Addressing application compatibility and performance optimisation issues.	Conduct comprehensive application assessments and performance testing to identify and mitigate compatibility issues and performance bottlenecks.

2.2.12 Conclusion

In conclusion, this section of the thesis has covered different aspect of cloud-to-cloud migration focusing on the transition between AWS and Azure. The analysis has covered fundamental cloud migration strategies, techniques and

tools essential for a successful cloud migration. With a detailed inspection of various migration strategies such as AWS's 6Rs, and the cloud migration phasing approach discussed by Laszewski and Nauduri, the importance of a well-structured and planned migration strategy has been emphasised. Each strategy, method and phase addresses operational challenges, ensuring that the migration process is executed both efficiently and effectively.

The case studies covered of companies like, video streaming platform Netflix, finance institution Capital One, and driver communication platform Waze illustrated real-world applications of these strategies, highlighting the problems the companies faced, and the tools and methods used to implement solutions. These case studies highlight the importance of careful planning, vision, ensured rollback methods, and the utilisation of appropriate tools and technologies to manage the complexities of cloud migration.

Security, risk management and sustainability, and technical and operational challenges have been identified as three common and critical problems arising during cloud migration process. Firstly, the Secure Cloud Migration Strategy (SCMS) proposed by Dalal N. Alharthi provides a structured approach to address these concerns, ensuring that data and applications are migrated securely and efficiently. Secondly, the study of economic, environmental, social, and technological dimension of sustainable cloud migration offers a holistic perspective of its impact on cloud migration. Thirdly, the thesis covered the importance of assessing technical and operational challenges, such as application compatibility and performance optimisation to plan, execute, and manage a cloud migration. Application assessments and performance testing are essential to identify and mitigate potential issues, ensuring seamless functionality on the new platform.

In summary, this section of the thesis provides a thorough study of the best practises with real-life examples for cloud-to-cloud migration, emphasising the importance of strategic planning, risk management, and continuous optimisation. The insights and methodologies presented aims to provide an essential background and set of tools for the final framework to facilitate an efficient and successful cloud-to-cloud migration in the financial company.

3. Case Study

3.1 Qualitative Analysis

As stated before, the financial company this thesis is written with has always been in the forefront of digital transformation. Until this point the infrastructure is separated into a hybrid cloud setup where the system is partially running on AWS and partially on-premises. The company's main target as today is move everything to cloud. In other words, harmonising the on-premises infrastructure is first priority in the company's strategy. The target is to have everything in cloud before 2030 and eventually on Azure cloud.

3.1.1 Reasoning Behind the Cloud Strategy

Deploying new software to on-premises will always become legacy sooner or later, and the financial company in this case is fully aware of it. In a big corporation people's role and responsibilities change, and it can be complicated to pass on the needed information to keep the development and management of on-premises software efficient. Instead of managing a whole software installation in company's datacentre, a developer could utilise cloud-based IaaS and PaaS solutions, so time could be used more efficiently to the development of the product itself, instead of managing it.

Cloud Migration Reduces the Required Manpower

According to recent studies, the migration from on-premises installations to cloud-based platforms, such as Google Cloud, can lead to significant resource allocation differences. For instance, Kolanuvada Varma's and Giruba Beulah's research from 2022 [12] demonstrates that managing an on-premises installations demands a substantial amount of manpower in contrast to cloud-based application. In the context of Varma's and Beulah's study they calculated that the on-premises installation required four times as much manpower to be managed when compared to the same application on Google-Cloud. [12] This difference in manpower demand can be further illustrated by the data presented in the Figure 6 below. The pie chart shows the distribution of manpower required for on-premises versus cloud-based installations.

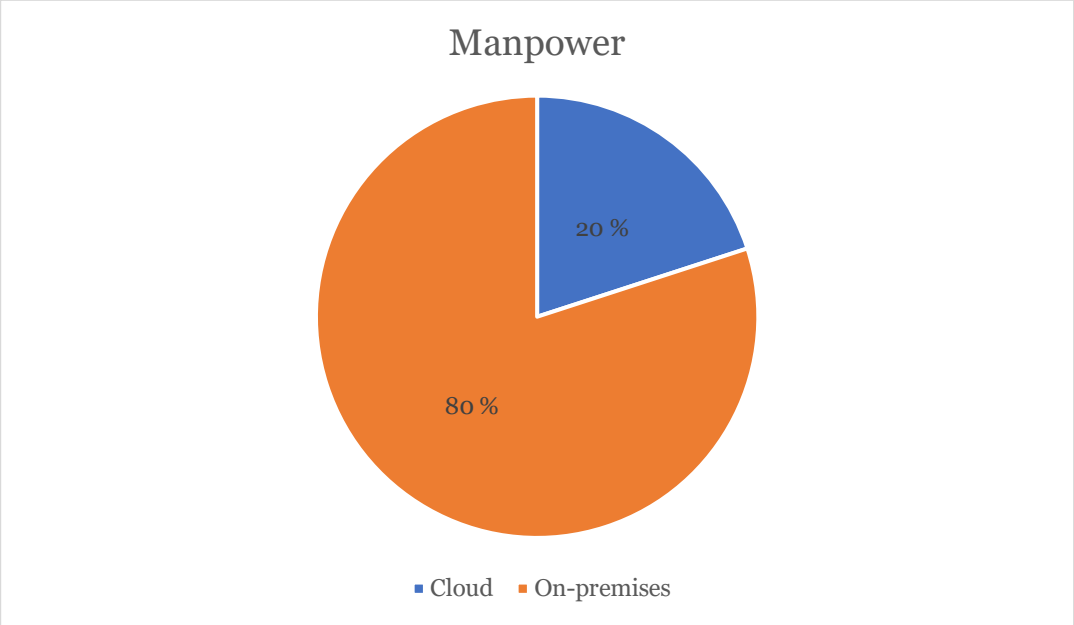

Figure 6, Manpower in On-Premises vs Cloud [12]

Furthermore, the same study observes a decrease in reported issues following migration to the cloud. The study compared the same application running on-premises to its deployment on Google Cloud. As shown in the figure 7 the on-premises version was responsible for 83% of reported issues, while the cloud deployment generated only 17% of recorded issues [12].

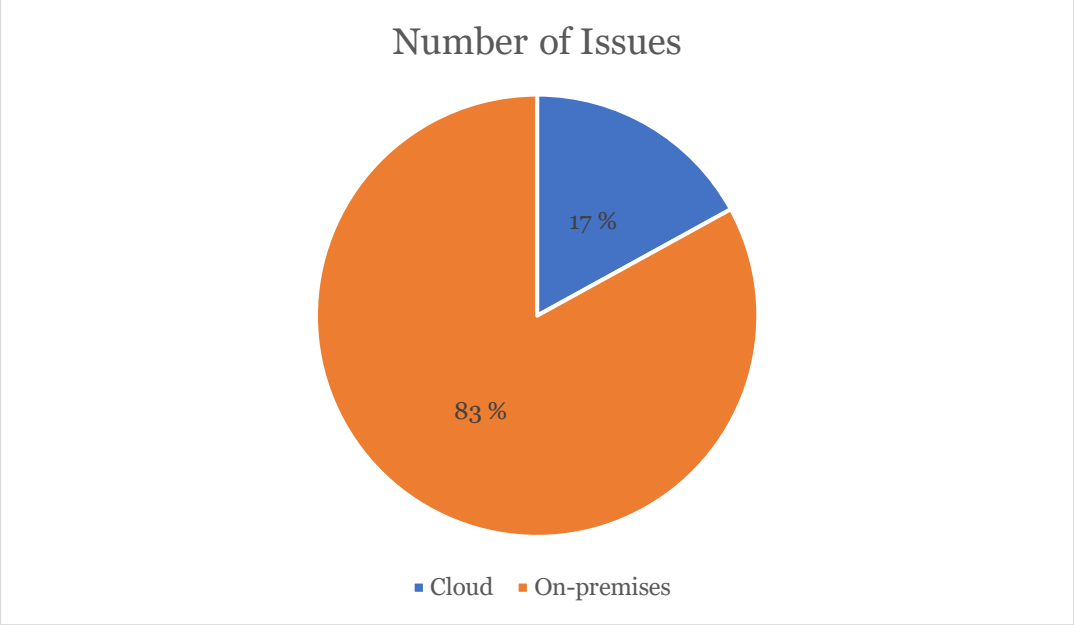

Figure 7, Number of issues in On-Premises and Cloud [12]

The impact of migration on processing time is also noted in the study. Prior to migration, the software relied on a Relational Database Management

System (RDBMS) for data storage and retrieval. However, the RDBMS had limitations, including a restriction on opening multiple connections with the source database for bootstrapping historical data. This limitation resulted in overhead on the source database, leading to increased overall data extraction job time.

Upon migration to Google BigQuery, this issue was effectively mitigated. Google BigQuery provides batch and stream APIs, enabling rapid extraction of data without the constraints encountered with the previous RDBMS. As a result, the processing time for the software improved significantly.

The research indicates that processing time for the software decreased to a one-third from on-premises installation when migrated to the mentioned Google BigQuery. [12]

Lastly, with all these positive impacts the cloud migration had on the software, the same research indicates that the new version deployed on Google-Cloud doubled the number of consumers the software had.

3.1.2 Cloud Positioning

The company has decided already to invest in Microsoft Azure as their leading public cloud provider, but in this chapter the thesis will investigate the different reasons that have led to this decision. The company has evaluated different public cloud vendors with five different Point-Of-Views (POVs) to help with the decision which to invest in. The five different POVs are ESG (Environmental, Social and Governance), Cost, Regulatory, Technology and Ecosystem. However, there are some decided limitations and goals what the company's cloud strategy should include. Firstly, Google Cloud Platform will not be included in the comparison since the lack of adoption within the company at this point. Secondly, the development of earlier multi-cloud approach will not be continued, and a new sole public cloud provider will be adopted. The multi-cloud will still be present until full transition to one provider is executed. Having specified these conditions, the company opted to primarily focus on comparing Amazon Web Services (AWS) and Microsoft Azure as the leading candidates for their future cloud provider.

ESG - Azure's edge over AWS with significant Finnish investments

ESG or Environmental, Social and Governance compares the providers on their environmental impact, how to decision would affect investments in Finland and which solution would be most in line with the company's values. AWS has established a region in Stockholm, Sweden, eliminating the immediate need to expand their European footprint by establishing another region

in neighbouring Finland. The Stockholm region offers a sufficiently low latency of 12ms to Finland [14], which might seem adequate for most applications. However, this setup restricts certain applications that, due to financial regulations, must be hosted within Finland's borders.

Contrastingly, Microsoft is taking significant strides by constructing a full region in Finland with datacentres in Espoo, Vihti and Kirkkonummi. The scheduled launch for the Finland region is in 2027. This development promises to address regulatory challenges and indicates a commitment to sustainability. The new Finnish region is projected to be entirely emission-free and designed to recycle heat energy back into the local community. Moreover, the establishment of a dedicated cloud region within Finland is anticipated to unlock new investment opportunities, marking a significant milestone in cloud computing infrastructure in the region.

Cost - Azure excels in contractual affordability, AWS in service fees

Another POV the company used to evaluate the candidates was cost. This is rather important, since cloud migration costs are underestimated in the financial sector due to the management of applications dependencies, legislation, support from non-IT departments, re-architecting applications, and hiring external contractors [15].

The financial company in the context of this thesis has already nine years of history with AWS which has over time led to vast investments. As it is today, the company has made an enterprise agreement to bring the cost lower with various discounts compared with the normal service fees. However, no deeper partnership program has been discussed nor does it seem to be likely to happen. Despite that, AWS offers generally lower service specific costs compared with their competitors. In addition, AWS is seen as a forerunner in cloud usage and cost optimisation tooling [10].

In comparison, Microsoft Azure service fees are generally higher than AWS. Gartner Magic Quadrant CIPS report from 2022 even states that Azure customers have complained of cost increase over time without clear reasoning [16]. Even though Azures cost management capabilities generally are behind cost management tools AWS offers there are some possibilities for discounted prices due to enterprise-wide agreement the company investigated in this thesis has made with Microsoft.

Ecosystem - Azure's inherent Microsoft integration versus AWS's open interoperability

Both Azure and AWS are competitive in their own environments and to find the best compatibility one needs to evaluate the company's current tools and software. AWS is generally considered as number one when it comes to independent software vendors (ISV) and SaaS providers. In addition, AWS uses and support various different open-source products. This has led to the implementation of AWS related products to different systems that are already in use in the company. To migrate these products to Azure will require investments to solve these dependencies.

In the context of open-source ecosystems, Microsoft Azure is often not the primary choice for organisations. Nevertheless, for environments heavily reliant on Microsoft products, Azure emerges as a formidable option. Microsoft SaaS products are dominant in the financial company's infrastructure where they serve as a daily operational backbone for all employees. The migration to Azure could set the stage for an integrated environment, significantly boosting the efficiency of inter-application communication within the company's infrastructure.

Technology - AWS leads based on technological advancements

AWS has generally wider and the most comprehensive IaaS and PaaS offering in cloud space [16]. AWS is a clear leader in modern technologies and services like serverless, NoSQL, data mining and analytics. In addition, the wide AWS service catalogue offers traditional technologies like relational databases, virtual machines etc.

The rather long history between the financial company and AWS has culminated in the deployment of numerous applications and long-term investments. As of early 2024, the company manages a diverse portfolio of productised platforms on AWS. The availability of resources and knowledge on AWS development is broad, ensuring that employees have ready access to the tools and information necessary for their work. Moreover, AWS certifications and various training programs are not only widely accessible but also actively utilised, enhancing the technical proficiency and readiness of the company's personnel.

In comparison the company's productization on Azure is still at its early stages as of early 2024. The current development stack predominantly consists of technologies that do not align with Microsoft's ecosystem. While Azure offers support for these technologies, they are not seamlessly integrated as core elements of the Azure platform. Furthermore, Azure presents a more

limited selection of services for the open-source technologies that the company frequently employs. Presently, there exists a noticeable gap in Azure development expertise within the company. Nonetheless, there has been a significant surge in Azure certification among employees, with a remarkable increase of over 430% in Microsoft Azure certifications from September 2023 to March 2024 [17].

Compliance - Azure capable of meeting existing regulatory standards

AWS does not provide a region that would adhere the current regulatory and contractual obligations that some of the company's application require. This would limit the company's vision of full cloud migration. In contrast, Microsoft Azure is set to offer a local region in Finland, fully compliant with all necessary regulatory and contractual conditions, including geographical requirements and the provision of Finnish operations personnel.

In conclusion, while Amazon Web Services demonstrates superior capabilities in development and cost management, the strategic benefits and alignment with the company's values and vision offered by Microsoft Azure's Finnish cloud region carry more weight in the decision-making process. Nevertheless, the transition to Azure will necessitate maintaining a multi-cloud approach for several years, given the significant effort and resources required to migrate from AWS.

3.1.3 Key Driver and Goals for Cloud

Developing an on-premises application has its own limitations. As businesses strive for greater agility in their operations and development processes, the constraints imposed by on-premises solutions become increasingly evident. On-premises environments, by their very nature, are less flexible and scalable compared to cloud environments. This lack of flexibility can significantly impact a business's ability to respond to market changes rapidly and to innovate at a pace that meets customer demands. Additionally, the capital expenditure required for infrastructure maintenance and upgrades in on-premises settings often limits resources that could otherwise be allocated to development and innovation efforts [18].

The company has set four different main drivers and goals for cloud [10]. In this chapter the thesis will covers these key areas in detail, providing a comprehensive understanding of the company's strategic direction in leveraging cloud technology. In addition, this chapter will compare the goals to different reports and academic articles to evaluate the effectiveness and alignment of the company's strategy with current industry trends and best practices. The

four key areas are agility in cloud, developer productivity, time-to-market and cost-to-serve and quality.

Agility in Cloud

Cloud environments are inherently more agile due to their scalable, flexible, and dynamic nature, enabling rapid deployment and adaptation to change requirements. This agility is a critical asset in today's fast-paced business environment, where the ability to respond swiftly to market changes can provide a significant competitive advantage.

The company's key areas in agility in cloud is autonomy within developers and applications during software deployment to the cloud, aiming to harness DevOps practices and automation to achieve these objectives efficiently [10].

The paper "Delivering software with agility and quality in a cloud environment" published by IBM claims that the integration of DevOps principles within cloud environments significantly enhances both the agility and quality of software delivery [19]. According to the article companies that are "born in the cloud" such as Netflix, have faced quick expansion in their business operations and consistent advancements in the quality of the services, reportedly through the strategic application of DevOps methodologies.

Increase Developer Productivity

One of the company's four goals with cloud migration is to increase an individual developer's productivity. As we stated before managing an on-premises installation can be time consuming due to their less flexible and scalable structure when compared to cloud solutions. As we mentioned how Varma's and Beulah's article states that managing a on-premises installation may take up to four times more manpower when compared to its cloud-based counterpart [12]. By decreasing the number of working hours needed to manage the current hybrid-environment, the company is trying to find new cloud-based capabilities and services to boost the developer's productivity. To put into simpler terms, the goal is to let the developers focus on the application development rather than wasting resources on understanding and managing an older on-premises application.

Time-to-Market.

The third goal the company has set for their cloud migration journey is time-to-market, in other words provide tools for developers and teams to publish new applications and features as fast as possible. The company want their IT department to react faster to different business opportunities and adapt

efficiently to new applications and solutions. More precisely this can be achieved by using cloud-based SaaS and PaaS solutions. SeetharamaTantry's article "Impact Analysis of Legacy System Migration to the Cloud Environment: A Focused Study" reports that migrating legacy software can have a significant impact on the application's deployment process [21]. The study argues that migrating legacy systems to the cloud enables organisations to adjust to evolving business requirements and control costs, while maintaining interoperability and facilitating integration across diverse programming languages.

On the contrary, an individual developer and the whole developer community has its own preferences and new tools are launched on the market constantly. By concentrating all applications under one cloud provider to make the deployment process more effective the variety of development tools are decreased to the catalogue offered by the cloud provider. Oliver Gass's, Hendrik Meth's, and Alexander Maedche's article from 2014 "PaaS Characteristics for Productive Software Development: An Evaluation Framework" discusses the developer experience on different PaaS platforms [22]. The study indicates that present cloud-based platforms vary significantly in the quantity and quality of their technical and non-technical features, offering a wide range of diversity. One of the three key findings made in the paper was how cloud-based platforms facilitate rapid application development through the use of reusable templates, components, and other building blocks. This enhances the developer's time efficiency when deploying minor features and adjustments. However, when implementing major changes the time efficiency cloud-based platform can offer starts to align with the time efficiency on a local development environment. The report states that when developing major changes using cloud-based IDE's the development can turn into painful, slow, and frustrating activity. In addition, the paper reports developer's mentioning in interviews that cloud providers do not offer tools that would be more flexible than custom code that still is necessary in most use cases.

Cost-To-Serve and Quality

The fourth goal the company has set focuses on optimising service delivery costs while maintaining or improving quality. It's anticipated that transitioning to the cloud will generate substantial costs, primarily because of the need to manage operations across various platforms concurrently throughout the migration period, as well as the investment required to develop the skills and capabilities necessary for deploying applications on a new platform. Nonetheless, it is projected that these costs will eventually offset the initial outlays incurred during the migration phase, once the new cloud infrastructure is fully operational.

Cloud platforms provide the customers the ability to shift from capital expenditures (CAPEX) such as hardware to run their application to operational expenditures (OPEX). This has as well been in the company's interest. To goal is to create more on-demand capacity more available and create a more transparent cost structure.

Furthermore, the company has planned to prioritise test automation and automated security validation between different environments in Azure. Automated testing and validation are crucial components maintaining high-quality standards helping the company ensure the application robustness and mitigate risks on the target platform.

Moving to the new cloud platform will create knowledge gaps within the company's IT department, since the new platform will require new skills. To solve this issue, the company is investing in building cloud expertise by offering Azure certification courses and certification tests for its employees. As noted by the company's Head of Development and Technology, "company has significantly increased its cloud expertise compared to the starting point: more than 800 employees have embarked on the Microsoft Azure learning path, and the number of certified Azure experts is growing day by day. In the coming years, company will become the largest cloud trainer in Northern Europe." [70] This initiative not only enhances the technical capabilities within the company but also positions it as a leader in cloud education and innovation in the region.

3.1.4 Cloud Migration Strategy Timeline

The complete cloud migration is calculated to take in total seven years to execute. This time window is estimated to consist of everything from the point of publicly announcing the migration to the end goal, which is to have every application running on cloud.

Until the beginning of 2024 cloud development has not been fully utilised. The reasoning can be the Finnish and European finance regulations that limits the possible full advantage what geographically spread cloud computing could offers. A lightweight landing zone to Amazon Web Services was created only nine years ago in 2015. During this time has Amazon Web Services mainly been used only by external employees and company's testing and experimentation team. After this several individual employees and teams have been started slowly deploying parts of their applications to AWS with the goal of minimising the amount of manual maintenance an on-premises installation can demand [10].

In 2018 the finance corporation deployed a new version of AWS landing zone, to keep up with the raising popularity of cloud development. At this point the corporation had published a new shared cloud development strategy: “Public Cloud primary platform for everything”. Firstly, this resulted in an expanded adoption of AWS within various projects and organisations. Secondly, the strategy precipitated a rise of cloud models like SaaS within various business domains. Thirdly, this new strategy initiated the beginning of various cloud trainings around the organisation [10].

The rise of cloud computing within the company was evident. During the following years 2020 and 2021 a new ICT-strategy was published: “Cloud as the primary platform”. The new strategy raised hybrid-IT solutions leveraging SaaS and custom cloud in public cloud’s popularity within the corporation. However, the number of migrated applications has created some inconsistence in cloud transition processes. The cloud migration is mainly based on refactoring instead of lift and shift or other known migration strategies or other conventional migration approaches [10].

Diverse backgrounds among developers have led to varied approaches to the refactoring process. Consequently, this diversity often yields differing outcomes concerning the application's testability, reusability, completeness, understandability, adaptability, fault proneness, complexity, maintainability, and stability. This difference is due to the fact that different refactoring methods have been shown to improve certain quality characteristics of software [11].

No change in the migration strategy has been done during the years 2022 and 2023. Most of the cloud transitions are still based on re-factoring, with some minor re-platforming efforts being done. However, it is safe to say that no common migrating strategy from on-premises to AWS was planned nor executed in the corporation to make the transition as smooth as possible. However, during the years 2022 and 2023 the company made a significant effort on offering AWS courses and certificates for employees to establish a wider competence in cloud computing and AWS. The work done during the years 2015 and 2023 had a very positive impact on cloud computing leading to a hybrid environment with multi-cloud in the company [10].

The company’s eight-year journey in cloud development and transition has established a solid foundation for the forthcoming Azure cloud migration. Clear goals have been outlined to solidify the expected outcomes in the years ahead. During the years 2023-2024 approximately 15% of all the company’s applications are running in public cloud. [10] This same time window is used to productize and create the basic capabilities to Azure’s Sweden region and make the platform ready for incoming products. The company’s next goal is

to have everything running in public cloud in 2030, and eventually in Microsoft Azure.

3.1.5 Conclusion

In conclusion, this qualitive analysis highlights the history and future plans of the company's cloud strategy. The case study covers topics such as what has been the reason in the first place to move to cloud, in what was succeeded and what could be done better when moving to Azure. The company's new cloud strategy emphasises the importance of a systematic and phased approach, that will move the company's whole hybrid infrastructure to a fully cloud-based by 2030. The key drivers in the company's new cloud strategy are increased agility, enhanced developer productivity, faster time-to-market, and optimised cost-to-serve without compromising quality.

The company's decision to invest in Microsoft Azure as the primary cloud provider was driven by several factors, such as Microsoft's investments in establishing a local region to southern Finland being the most central. Having a local datacentre aligns with the company's ESG and financial industry's regulatory requirements. The decision also reflects a strategic move to leverage Azure's integration capabilities, especially within an ecosystem heavily reliant on Microsoft products. Additionally, the qualitive analysis reveals a strong correlation between cloud migration and reduced operational costs, increased efficiency, and enhanced service delivery based on research done in the industry. The analysis underscores the critical role cloud platforms have in enabling businesses to scale operations, optimise resources, and innovate rapidly in response to market demands.

Overall, the qualitative analysis provides insight into the complexities involved in a large-scale migration project. It serves as a practical example for other organisations planning a similar transition, emphasising the importance of strategic planning and continuously adapting to evolving technologies and requirements.

3.2 Stakeholder Interviews

In this chapter the thesis will present findings made from the stakeholder interviews. The interviews were 30 minutes long semi-structured interviews with employees with a software development background in the financial company. The target interviewees are persons who have developed a product or service that is currently running completely or partly on AWS. The goal of these interviews is to gather an understanding of the technical readiness to conduct a migration to Azure Cloud, and what issues they have taken into consideration.

The interviews were held with four different stakeholders. The interviewees were chosen based on their technical background, history with first steps when implementing new features to AWS in 2017 and by their knowledge and background with the company's context. In addition, as the company employs several thousand within the IT department, the interviewees was handpicked from different areas to gather inputs within different contexts. The first participant interviewed holds the position of lead developer on the company's API Management team and brings twenty years of experience in software solutions within the financial services sector. This extensive background provides a robust understanding of API infrastructure and its development on the utilised cloud platforms. The second participant in the interviews is the lead developer responsible for the company's customer self-service channels. The team is managing four AWS application and seven more to be deployed to production. The third participant is a tech lead, from the same department as the previous interviewee. The third participant have over 14 years of experience in the company, and a higher perspective on the organisations cloud infrastructure. The fourth participant is a tech lead in the "everyday economy" department responsible for various everyday banking components. The organisation in known for being a forerunner in cloud within the corporation.

The interview questions were categorised to four different parts. The first section was expectations and preparations, to gather an understanding of the starting point where the team or teams within an organisation. The second part of the interviews focused on technical challenges and solutions, to gather insight of the assumed problems the team might be facing when migrating to another cloud platform and how the team have plan to solve these issues. The third part of the interview covered support and resources; with this chapter the interview try gather understanding on the developers needs from this framework. What kind of problems does the developer or team this framework solves for them. What kind of solution would bring most value in the context of their migration process. Lastly, the fourth part of the interview was post-migration. The last part of the interview focuses on how the team or developer had planned to measure their migration success, how have they planned to compare the new solution against the older one in AWS.

The interview questions can be found from the appendices section.

3.2.1 General Findings from the Interviews

A clear insight from the interviews was that teams had not put any resources to plan their cloud-to-cloud migration. Most of the interviewees reported that

they had set goals to get a fundamental understanding of Azure cloud tools, but no direct plan on by who, when and how the migration would be executed. The lack of initiative and decisions at this point is understandable, since the focus is now mainly on moving the older legacy software from on-premises to public cloud. However, the cloud-to-cloud migration will be executed at some point in the future, even it might not be relevant in the near future.

Additionally, a general finding from the interviews was that AWS usage significantly varies across different parts of the company. Factors such as the company's size, various business areas, and the autonomy of individual teams have led to a diverse array of working methods and recommended tools. This diversity is evident in the interviews when examining the general cloud competence and the specific AWS technologies employed by these teams. The extent of AWS applications ranges from only a few per organisation to entire organisational infrastructures reliant on the current cloud provider.

3.2.2 Expectations and Preparations

As slightly covered in the previous chapter, the teams and organisations across the company are focusing now on creating a fundamental understanding how Azure should be operated. This is done through various Microsoft Azure pilots, courses, and certifications, such as Azure Fundamentals, Azure Developer and Azure Solution Architect. Although the general knowledge and skill within cloud computing is related between different platforms, and previous experience on other platform most likely can help one to get used to with a new cloud provider, there is a clear difference between AWS and Azure from a developer's perspective.

From the first interview with the API organisations lead developer we found that from developers' perspective at this point is crucial to learn the vocabulary used on the new platform. This is a heavily linked with the communication used in the context of Azure platform. The knowledge sharing moments such as discussions with colleagues or external experts demands a common language in the format of understanding the vocabulary and tools in the context of Azure.

Insights from the third interview with the organisation tech lead highlighted that Azure pilot sessions revealed gaps in Azure knowledge even among competent AWS developers. The interviewee noted that these nuanced differences between the platforms, although subtle, were unexpected and led to confusion, impacting effectiveness in the new environment.

In addition, a study "An Investigation of Microsoft Azure and Amazon Web Services from Users' Perspectives" by several authors states that even though AWS and Azure share a lot of similarities, there are certain features and services that differ clearly. There are clear differences in pricing, availability and specific cloud configurations that might affect the adaptation process on the new cloud platform [25]. These differences might require additional learning or adjustment.

3.2.3 Technical Challenges and Solutions

In the second chapter of the interviews, the goal was to gather an understanding of the technical challenges teams are anticipating with the upcoming cloud-to-cloud migration, and the possible solutions they have investigated. The questions were set up as direct as possible to gather exact answers how the team has planned to solve specific issues.

Most of the interviewees reported that they had questions and concerns how Azure accounts and networking was going to be set up. The concerns were related to upcoming changes with the used infrastructure, and how a transition between different regions were planned to execute. Even though the task of this thesis is not to solve these problems technically, the unawareness how the new platform bureaucracy and region configurations is something the final framework produced in this thesis can possibly clarify the development teams.

Even though the overall philosophy of cloud computing is generally more or less the same among both providers, some of the interviewees reported about confusion of finding direct equivalents for certain AWS services on the new cloud platform. The third interviewee stated that few of their current tools on AWS does not have a direct corresponding service on Azure. Developers in the interviewee's organisation are currently using services like AWS Fargate and AWS Step Functions that Azure does not support. On Azure a most similar service for containerization could be Azure Kubernetes Service (AKS) that is generally more complex service with Kubernetes concepts like pods, policies, and deployments. In the context of Function as a Service (FaaS) and serverless computing, the service on Azure that most closely compares to AWS Step Functions is Azure Functions. AWS step functions are a user-friendly way to set up microservice processes without needing a proper background in programming or software development. Azure functions does not support as an easy set up process and demands a more manual deployment to achieve the same results. However, a manually implemented solution is more manageable and customisable than for example AWS stepfunctions, but a change like this in the developer's toolbox can take a moment to get used to. However, the second interviewee anticipates that even though the

change in used platform might cause demand some delays, the migration of container-based application itself should not be that complicated process.

Additionally, downtime in databases and applications during the migration process was topic raised by several participants. A common concern was how the migration process should be executed with minimal downtime for end-users, minimal disruption to ongoing operations and efficient deprecation on the previous platform. Many interview participants anticipates that the migration process is be quick and mostly automated for specific components, such as databases with the help of for example Azure Migrate. However, the same participants report concerns with migrating AWS native applications, where the amount rebuilding required for the migration can be unknown in the beginning of the migration process. More specifically, the interviewees are not concerned about the complexity of technical implementation, but rather about the number of applications that will require partial or complete reconstruction during the migration process.

3.2.4 Support and Resources

The third chapter of the interviews focused on support and resources development teams would see beneficial during the different phases when migration applications to Azure. The questions were built specifically to address the questions the framework developed in this thesis should answer; by filling which gaps, could the framework enhance the migration process and development experience for a development team.

Many participants were still in the early stages of their journey on Azure, which led to unawareness of the future obstacles they might face. However, some hypotheses of the possible workload and problems could be made. Participant reported that during the migration process their development teams would gain most value with clear guidelines on deployment, monitoring, and best practises to navigate efficiently through the migration process.

The participants expressed a desire for the framework to provide guidelines tailored to the specific configurations and unique needs of their application, which would effectively guide developers through various migration phases. Additionally, the participants emphasised the importance of these guidelines aligning with the broader context of the company. The participants also noted that providing concrete examples of the appropriate tools to use at each phase would address most potential issues. These examples should clarify how and when to implement these tools effectively, ensuring alignment with the specific needs of their application and the overall objectives of the company. Additionally, many wished for professional support during the migration from someone who has already experience with migrating their

application. These same participants suggested that a platform for sharing knowledge of successful and unsuccessful migration attempts would bring a lot of value for teams who are planning their application migration.

3.2.5 Post Migration

The last chapter of the interviews focused on last phase of the migration. In this part of the interview, the questions were designed to gather insight of Azure adaptation, measurement of migration successfulness, and support needed after the application is deployed on Azure Cloud.

The majority of the participants reported that the biggest concern during post migration is cost management. The participants specified, that the main concern is not related to the application's cost on Azure, but rather the cost generated by running the application on several platforms during the deprecation phase on the previous platform. By enhancing the deprecation process on AWS, the teams can mitigate concerns regarding the concurrent accumulation of costs across both platforms, thereby allowing for a more efficient allocation of resources towards the new platform.

Additionally, several participants raised automation as a core topic on Azure. For instance, the first participant reported that the amount of automation has significantly increased within the company since AWS had been taken into use in 2017. Now when the company's automation and deployment culture and competence has evolved during the past eight years the company is investing in new deployment pipelines in Azure based on GitHub version control and deployment triggers [26]. The change in deployment and automation tools raises some concerns among the interview participants. The main concerns are how one should operate in the new production environment with new deployment pipelines. The fourth participants report that most of the concerns could be alleviated with clear instructions on what the developer should take into consideration when deploying an application or new features to production. The operating model in production must be made clear to raise the general competence level, and therefore make production deployments safe and robust.

According to few of the participants measuring the migration successfulness is no easy task. However, the third and second participant states in the interview that migration efficiency measurement is important to align with the company's cloud strategy, in this case time-to-market effectiveness. The second participant even reports that the migration should not take more than a few days per component.

Generally, the participants want clear guidelines on deployment, monitoring, and best practices to navigate the technical landscapes of Azure. In addition to documentation and instructions, the participants believe that platforms for sharing knowledge among peers could bring a lot of value for several development teams migration processes.

3.2.6 Challenges and Proposed Solutions

In this chapter the thesis will discuss some of the issues identified during the interviews, and propose possible solutions used in the framework. It is important to note that at this stage, the analysis will focus solely on the solutions themselves, without comparing them against the broader company context or considering varying migration strategies. This approach allows for a focused examination of each issue and its respective resolution strategy, ensuring clarity and depth in addressing the core challenges.

The first problem this chapter will cover is service alignment and correspondence between AWS and Azure platforms. Finding corresponding service on the cloud provider is the most fundamental issue the framework should solve. The framework should include a detailed service mapping guide that identifies AWS services and their Azure counterparts, including recommended substitutions or adaptations where direct equivalents do not exist.

The second issue raised in the interviews was specifically the migration of container-based applications. As the interview participant stated, there are some concerns about migration container-based applications from AWS to Azure. Firstly, migrating container-based application from AWS Fargate to Azure Kubernetes Service (AKS) is understandably a concern among developers, since the two applicationsdiffer fundamentally. AWS Fargate is a serverless compute engine developers can use to run container-based application, without needing to manage servers or clusters. AWS Fargate is most ideal for users who want to focus on building and scaling applications without the overhead of managing the infrastructure. It simplifies the deployment process by handling the provisioning and scaling of the underlying infrastructure. In comparison, AKS is a container orchestration services, which developers can use to automate deployments, scale applications depending on the workload, and manage a container-based application. AKS is generally thought as a heavier tool, since it can be used to manage the nodes (Virtual machines) that form the Kubernetes cluster to optimise the cluster configurations leading to more balanced performance, cost, and complexity. AKS is suitable for teams that want more control over their container orchestration and are looking for the flexibility to scale and manage a wide range of applications.

To solve this issue, the framework must obtain the same point of view as a developer worked previously on AWS Fargate would have. The framework should provide best practises when it comes to migrating a container-based application between platforms and with instructions how one should navigate in a new environment with a more complex tool.

The third issue raised by interview participants was the migration of specific serverless based services. The issue is particularly related to the migration of AWS Step Functions, which provides a different solution to orchestrate the serverless function with other AWS applications. The framework should provide a solution that helps user to map the equivalent AWS components in Azure such as AWS API Gateway to Azure API Management and the different databases. Additionally, the framework can utilise Azure Logic Apps Designer, to assist developers to recreate their workflow.

The fourth problem raised in the interviews is application dependency management. In a company at this size and number of applications it is managing on AWS, application dependencies are a common issue among developers. The dependencies must be mapped during the first phases of cloud migration either by utilising application's documentation or by utilising cloud native tools such as AWS Application Discovery Service to identify and map the starting point. Additionally, Microsoft Azure provides tools like Azure Migrate which can be used to generate dependency analysis that could help the developer to compare application dependencies between the source and target cloud platforms. Lastly, the framework provided in this thesis should include instructions on AWS and Azure service mapping that can be utilised to compare alternative services and tools across the cloud platforms.

The fifth problem which the interviewees raised was the database migration with minimal downtime. This affects more or less all applications that will be migrated. When moving the hosted service to another platform, a delay is inevitable, but the problem is how a development team can minimise the downtime during the migration process. The framework should include best practises how the development team can prepare a database for migration, and what the execution steps for a successful cloud-to-cloud service migration are.

The sixth problem detected in the interviews was related to post-migration phase, as how should the development team set up and operate Microsoft Azure. Many participants were concern about rising costs when migrating to Azure and are in need of tools and resources how they should manage resources and operate monitoring tools to ensure operational efficiency with reasonable costs. The framework should provide information of the

equivalent AWS tools and resources on Azure, to help the development team to understand fundamental differences between these platforms.

The seventh and last issue raised by the interview participants was the change in deployment pipelines. The deployment pipelines are going to be built by a different team, dedicated to build, and monitor production pipelines. Therefore, the framework does not need to create or investigate any deployment pipeline used in Azure, but instead guide the development teams to right resources.

3.2.7 Conclusion

The stakeholder interviews conducted in this thesis provided significant insights into the readiness and challenges associated with the planned cloud-to-cloud migration from AWS to Azure within the financial company. The semi-structured interviews involved four various key technical leaders, developers and architects across different departments, highlighting several critical areas for consideration, ranging from preparation and expectation to technical challenges, support needs, and post-migration concerns.

Firstly, the interviews showed that there is a significant deficiency in cloud-to-cloud migration planning among the teams. However, at this stage the cloud-to-cloud migration is not a priority. Instead, the teams are concentrating on acquiring fundamental knowledge of Azure through certifications and pilot programs. The lack of a clear migration strategy indicates a need for structured planning and a detailed roadmap to ensure smooth transition to the new cloud platform.

Secondly, the interviews revealed a substantial variability in AWS usage across the different departments, reflecting the diverse nature of the different business areas, autonomy of individual teams, and ways of working. The diversity has resulted in different levels of cloud competence and varying use of AWS technologies, highlighting a need for tailored migration strategies that address specific requirements and challenges.

Thirdly, technical challenges emerged as a noticeable theme in the interviews, especially the differences between AWS and Azure services. Frequently mentioned concerns about finding direct equivalents for AWS services, managing container-based applications, and minimising downtime during migration highlights the necessity for detailed service mapping guides, comprehensive application assessments, and best practises for performance optimisation.

Fourthly, support and resources were identified from the interviews to be crucial elements for a successful migration. Interview participants emphasised the importance of clear guidelines on deployment, monitoring, and best practises, as well as professional support from experienced peers. The need for a knowledge-sharing platform or session emerged, indicating a collaborative approach to overcome migration hurdles.

Lastly, post-migration concerns centred around cost management and the operation of new deployment pipelines. The need for clear instructions on managing resources, setting up monitoring tools, and operating in the new production environment was evident. Participants also stressed the importance of measuring migration success and ensuring ongoing support to maintain operational efficiency on Azure.

In Conclusion, the stakeholder interviews highlighted key gaps in planning and migration readiness for the financial company's cloud migration from AWS to Azure. To gain an overview of the interview key findings and proposed solution, please refer to Table 4.

Table 4, Interview Key Findings and Solutions

Aspect	**Key Findings**	**Solutions**
Planning	Lack of comprehensive migration strategy.	Develop a detailed migration roadmap.
AWS Usage Variability	Diverse AWS usage across departments.	Migration strategies to specific team needs
Technical Challenges	Issues with service equivalency, container management, and downtime.	Provide service mapping guides, conduct application assessments, and establish optimisation best practises.
Support and Resources	Need for guidelines, peer support, and knowledge-sharing platforms.	Clear deployment guidelines, a knowledge-sharing platform, professional support.
Post-Migration	Challenges with cost management and new deployment pipeline operations.	Offer detailed instructions on resource management, monitoring setup, and performance evaluation.

4. The Cloud-to-Cloud Migration Framework

The following chapter in the thesis will cover the cloud-to-cloud migration framework built based on the findings discussed earlier in the literature review. The framework is built upon the findings from cloud migration tools, methodologies and case studies from the industry as well as on findings from the interviews with financial company's cloud developers and the company's cloud strategy. The framework aims to provide a structured and systematic approach to ensure a seamless transition from AWS to Azure, addressing the unique challenges and requirements identified in previous chapters. By utilising the best practices and methodologies explored, the framework will offer a practical guide to achieve a successful and efficient cloud migration, ensuring business continuity, data integrity, and optimal performance in the new cloud environment. The cloud-to-cloud migration framework is designed to help developers in the financial company to migrate applications and services from AWS to Azure effectively, efficiently, and securely.

4.1 Migration Phases

The framework is structured into four distinct phases: Preparation and Concept, Blueprint Design, Execution and Transition, and Post-Migration and Optimisation. The four-step process is based on the original phases outlined by Laszewski and Nauduri [29] [Figure 6]. By narrowing these phases into broader categories, the framework adopts a more generalised and versatile approach. This customisation allows for application across a wider range of cloud services and scenarios to be migrated with the framework. The four-phase structure is designed to be flexible and adaptable, ensuring that it can accommodate various migration contexts and challenges while maintaining the core principles of a successful cloud migration. Each phase consists of specific steps and best practices to ensure a seamless migration. To gain an overview of the migration phases and the key tasks, please refer to Figure 8.

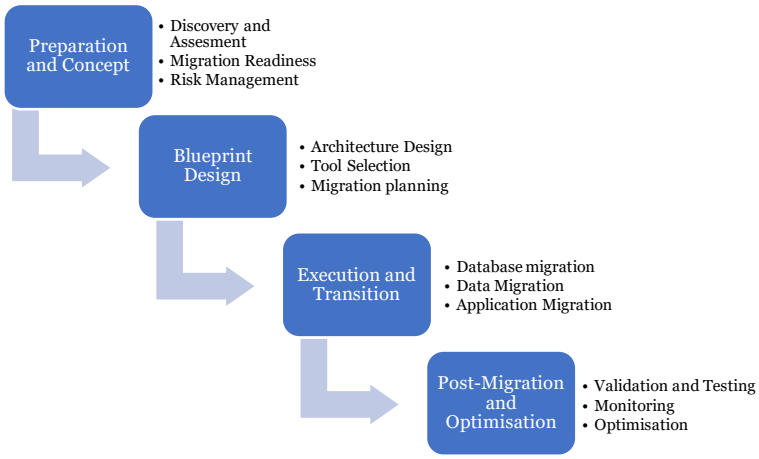

Figure 8, Cloud-to-Cloud Migration Phases

4.1.1 Preparation and Concept Phase

The framework's four phases begin with Preparation and Concept phase. The primary objective of this phase is to lay a solid foundation for the migration process by thoroughly understanding the current environment and infrastructure, setting clear goals, and preparing the necessary resources. This phase involves a comprehensive assessment of the current environment, an analysis of the application's migration readiness, and strategic planning for risk management and stakeholder engagement. Given its critical role, the Preparation and Concept phase requires substantial resources and time to ensure a deep understanding of the project's architecture and readiness for migration.

Discovery and Assessment of Current Environment

AWS offers hundreds of different tools of some can be utilised when assessing the current cloud environment with their own purpose and advantage. Since it can be difficult to choose the right tools to assess the current infrastructure, the thesis will discuss some of the recommended and popular tools AWS provides that can be utilised to understand the current application environment and dependencies. The thesis divides the tools into three categories: Dependency Mapping, Security and Compliance, and Documentation and Review.

Application hosted by the financial company's development teams tend to consist of several microservices. A clear image of the dependencies the application has with other microservices is crucial when assessing the infrastructure on the source platform. Simply by managing and developing the application from the start will provide an overall good understanding of the application infrastructure. Unfortunately, that is not always the case with

developers responsible for the migration. Additionally, AWS does not offer any tools that would automatically generate a bird's-eye view of the application infrastructure. However, other tools can be used to gather information of the application reaction points and what services it triggers, which can be used to draw an architecture image of the services, if one does not already exist.

AWS offers a debugging tool called AWS X-Ray which can be utilised to understand and map AWS application dependencies. AWS X-Ray is a tool that can be used to trace requests through the application and different AWS services it triggers. The service main purpose is to trace, debug and detect anomalies. Fortunately, the service also offers a service map and trace map functionalities, where the chosen AWS account services and triggers that have been used during a chosen timeframe will be shown. See Figure 9 below for an example of service mapping with AWS X-Ray. Additionally, the tool can be used to see how to application reacts to different scenarios and detect possibly performance issues and bottlenecks. X-Ray can be hard to use to draw the complete architecture of an application, in case no previous information or knowledge is available. However, it can help to understanding small details of the architecture by tracing calls into services that are not used often or otherwise unknown by the developer.

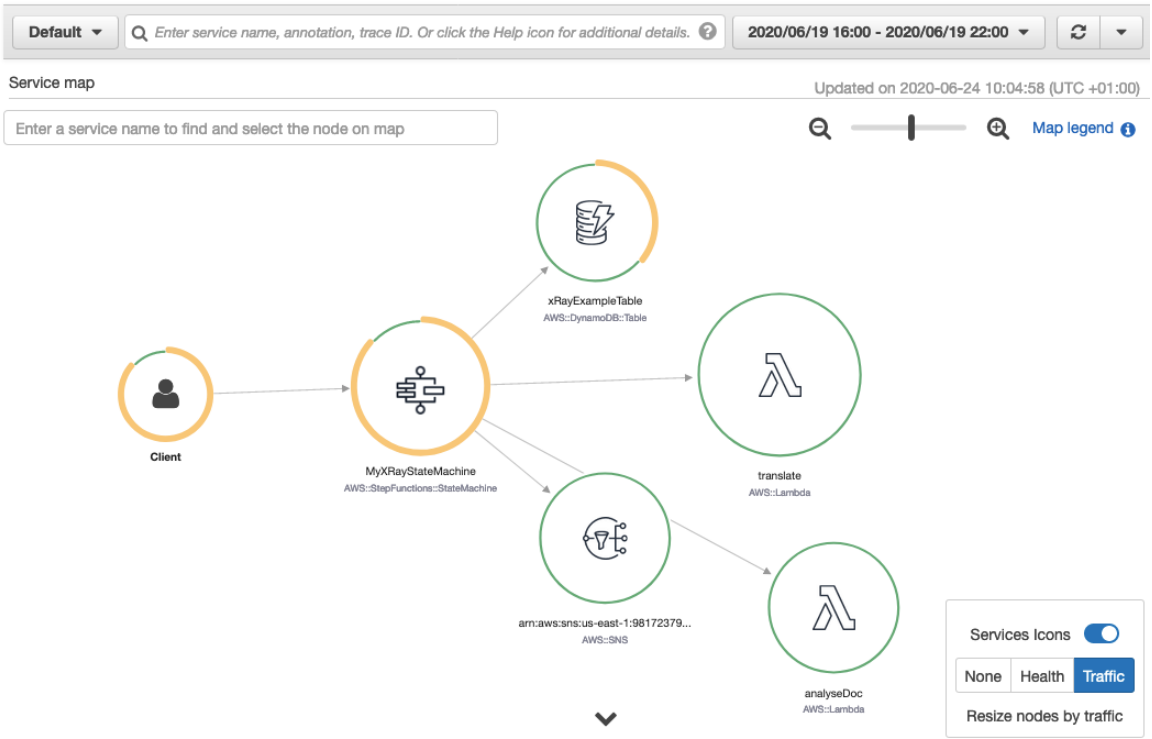

Figure 9, AWS X-Ray Service Map [59]

In comparison, Microsoft Azure offers a tool called Azure Migrate that is designed for migrating on-premises applications to Azure. However, due to the tight competition between cloud providers, Microsoft have added features as well for discovering, assessing, and migrating AWS and GCP applications to Azure. Azure Migrate can be used to discover various AWS instances under one AWS account, such as software inventory, web apps, SQL Server instances, and databases. This automated discovery tool helps migration team to build a comprehensive understanding of the existing infrastructure and aids in planning a seamless migration [60].

A part of assessing the source platform is security and compliance review. Analysing application compliance not only provides a broader understanding of the application but also give a better understanding of the application's migration readiness. AWS IAM (Identity and Access Management) can be utilised to gain a comprehensive understanding of the permissions and roles associated with each service, ensuring that all security and access controls are thoroughly evaluated.

Although AWS provides some tools that can be utilised to gather information how the services work and communicate with each other, the most important source of information is people who have worked with the application, or documentation where the architecture is explained. By organising meetings or demonstration sessions where the application's inventory and architecture on both source and target platform can be thoroughly discussed.

In conclusion, the discovery and assessment phase are crucial for understanding the current cloud environment and planning a successful migration. The tools and services discussed, such as AWS X-Ray and Azure Migrate, provide essential insights into application dependencies, security, and infrastructure. For a detailed description of these tools, please refer to Table 5.

Table 5, Discovery and Assessment Tools and Services

Step	**Tools and Services**	**Description**
Dependency Mapping	AWS X-Ray	Traces requests through the application to identify performance bottlenecks and service dependencies.
	AWS Migrate	Discover, assess and migrate applications from AWS to Azure.
Security and Compliance	AWS IAM	Reviews IAM policies and roles to understand the current security posture.

Documentation and Review	Confluence	Read and write detailed documentation of the current environment, including resource configurations and performance metrics.
	Review with Stakeholders - Demonstration session	Conducts review and demonstration sessions with stakeholders to validate findings and ensure alignment with business objectives.

Migration Readiness

A migration readiness assessment is a process of gaining insights of the application's strengths and weaknesses on the current platform and developing a plan how to enhance the identified problems on the target platform. During a migration readiness review, performance and cost-related metrics are of utmost importance. By thoroughly analysing the application's health data on AWS, the development team can gain valuable insights into potential areas for optimisation, ensuring a smooth and efficient migration process.

Performance metrics is a central detail when assessing an application's migration readiness. Amazon CloudWatch is a popular tool for monitoring and gathering performance metrics such as CPU and memory usage. Gathering application performance metrics can give an understanding of the application's migration readiness.

Additionally, cost analysis is a central method for assessing application's migration readiness on the source platform. This involves evaluating the costs associated with each service on the source platform. Understanding these costs can help the development team identify potential areas for cost savings on the target platform. While high costs on the source platform may not necessitate immediate modifications, they offer insights into possible adjustments needed on the target platform to optimise resource utilisation and reduce expenses.

In Addition to AWS Services, Azure Migrate offers features for planning cost efficiency for the migration. Azure Migrate offers a similar automated discovery tool for cost analysis as it does for infrastructure assessment. It provides tools for estimating the cost of running workloads in Azure, helping organisations to budget effectively and make informed decisions about their migration strategy. Additionally, Azure Migrate can generate detailed migration

plans that outline the steps required to move each service to Azure, including any necessary modifications or optimisations.

In conclusion, a thorough migration readiness assessment is crucial for identifying the strengths and weaknesses of applications on the current platform and developing an effective plan for migration. For more information on the tools and services used in migration readiness, please see Table 6 below.

Table 6, Migration Readiness Tools and Services

Step	**Tools and Services**	**Description**
Performance Metrics	Amazon CloudWatch	Monitors performance metrics like CPU usage, memory usage.
Cost Analysis	AWS Cost Explorer	Analyses the current cost structure and identifies opportunities for cost savings.
	AWS Budgets	Monitors and manages costs to understand the financial impact of the current environment.

Risk Management

Risk Management is a central topic in every migration phase. However, by conducting actions to detect risk in earlier phases the probability of sudden failures can be minimised. As covered earlier in the thesis the article [57] have divided a sustainable cloud migration into four different dimensions: Economic, Environmental, Social, and Technology. In the context of the financial company, developer responsible for the migration, and the final framework, only a few of these topics are central. Decisions on the cloud provider have already been done by others in the company, additionally the whole migration process will likely be costly. However, these areas can be affected during the migration by conducting proper performance analysis on the application and detect possible issues and bottlenecks that can be solved on the target platform. By enhancing the application's performance both cost and climate impact can be enhanced.

The focus when assessing risk management during the initial phase on social and technological dimensions. The risks included in these dimensions arise with poorly written software. A poorly scalable, maintainable with lacks in security will phase technical issues that leads to poor performance and consumer experience.

Conclusion

In conclusion, the idea of the initial phase of the migration process is to gain an understanding on which AWS services the application infrastructure is built on, how well it is performing, and identifying possible risks the application would have during a migration

A catalogue of the AWS services used by the application is a recommended starting point, since the performance analysis and discussions will concentrate on the services the application utilises. A catalogue of the AWS infrastructure should contain everything the application utilises like for examples the virtual machines running in for example AWS EC2, storage and database services like DynamoDB, S3 Buckets, Glacier Archives, networking components such as VPCs, gateways, subnets, and VPN connection and lastly functions, API Gateways and load balancers. Unfortunately, AWS does not provide any tools that can be utilised to generate a bird's-eye view of tee architecture automatically. However, some other tools such as AWS X-Ray and Azure Migrate can be utilised to help with the process. Although, the best source for creating a clear picture of the infrastructure is an expert with experience with the cloud service to be migrated.

4.1.2 Blueprint Design Phase

The second phase of the migration process is the Blueprint Design Phase where the main goal is to establish a detailed and actionable migration plan. This involves creating a comprehensive, step-by-step plan for migrating the application from AWS to Azure by leveraging the service maps, identified risks, and assessed migration readiness from the Preparation and Concept phase.

This plan should include a detailed mapping of existing AWS services to their Azure counterparts, selected appropriate migration strategy, and a plan for security, compliance, and rollback measures. To build a plan for the migration, the thesis divides the Blueprint Design Phase into three dimensions: Service Mapping, Migration Strategy Selection, and Security, Compliance, and Rollback Planning.

Service Mapping

The first dimension in Blueprint Design phase is service mapping the AWS services with their counterparts on Azure. To execute a successful service mapping between AWS and Azure a catalogue of utilised AWS services done in the previous chapter is required. For instance, EC2 instances in AWS can be mapped to Azure Virtual Machines, while AWS RDS databases might

correspond to Azure SQL Database. Due to the limitations of the thesis, the framework won't be able to cover all the services AWS and Azure offers.

Insight from the interviews dictates that some of the popular services at the financial company do not have an identical counterpart on Azure, such as AWS Fargate in comparison to AKS, AWS Stepfunctions in comparison to Azure Functions. The decision of the chosen service on the target platform should be based on specific needs and requirements the solution should solve, and not by an assumption that a similar service on the target platform will solve the same issues. For instance, AWS Fargate is a different service than AKS, and cannot be compared since they offer different solutions for different purposes. Instead, a more similar service on Azure could be Azure Container Apps, that is a lighter version of AKS, where the service lets the user to deploy and scale applications without requiring access to cluster and node controls. In the case of AWS StepFunctions, Azure functions is not the equivalent service, and they should not be used to solve the same issues. Instead, Azure provides a service called Logic Apps that imitates the same functionalities both in backend process logic as well as in developing workflows.

Since the thesis cannot cover every application and their counterpart, a recommended method is to study the different services and compare them to the requirements set for the solution.

Microsoft tools and documentation is a recommended source to compare AWS services to Azure alternatives. Comparing the AWS Service catalogue generated with Azure Migrate and AWS X-Ray discovery and assessment tools to AWS to Azure services comparison [61] will help connecting source platform services to their counterparts on target platform. The article [61] compares services that are roughly comparable. Every AWS service or Azure will not likely be listed, nor does every tool have a feature-to-feature similar service on the alternative platform.

Migration Strategy

Decision on Cloud-to-cloud migration strategy should be based on the results from application assessment, migration readiness, and complexity. The desired migration strategy should be one of the six strategies recommended by Stephen Orban from AWS [27]. The spectrum of the six migration strategies covers all the level of complexities and scenarios an ordinary application managed by the financial company's development team would need. For a visual representation of the cloud migration strategy spectrum, please see Figure 10 below.

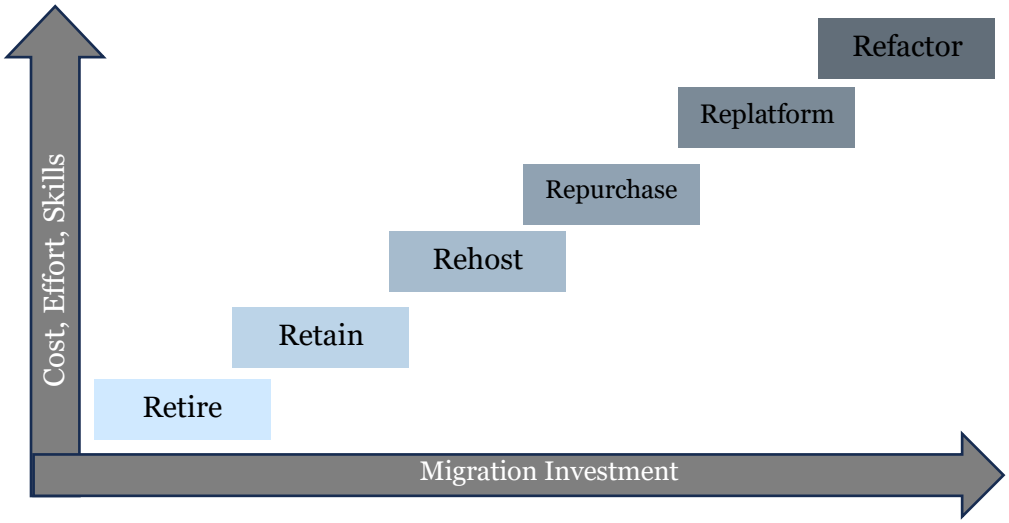

Figure 10, AWS 6R Migration Strategies

Based on the application assessment results, the migration team can identify the number of changes required for the migration to Azure. In some cases, the application might be obsolete, and can be deleted while rest of the infrastructure is migrated. Or alternatively, the application might be ready for migration, but an external factor blocks the execution of cloud migration, and the application must be retained on the source platform until further notice. The last four methods are actual migration methods, where the application will be moved to the target platform. The last four methods vary from an ordinary "lift-and-shift" where for instance a container-based application can be migrated as it is. Consequently, a far more complex application utilising AWS cloud native feature might require complete replatforming or refactoring, where the cloud native services must be rebuilt on the target platform.

The migration will likely cause downtime for the application. Understanding the requirements for acceptable downtime is crucial for the migration team, and these should be clearly defined during the application assessment phase. The decision on the appropriate migration strategy should be guided by disaster recovery (DR) plans, recovery time objectives (RTO), and recovery point objectives (RPO).

To manage and minimise the downtime effectively, the chosen migration strategy can be tailored to specific needs. An application can be divided into migration phases, where moving small parts of the application sequentially will reduce the impact the migration has on its customers. Alternatively, an application can also be set up to run in a hybrid cloud until the migration is completed. These setups allow for a gradual transition and testing in the new environment while maintaining operational continuity. However, these methods are very costly and time consuming, since it requires to be managed

on two separate cloud platforms simultaneously. Instead, a parallel run would allow the migration team to duplicate the instance of the application on the target platform (Azure) while the original remains operational on the source platform (AWS). Once the application is migrated, tested and validated on the target environment, the switch can be made with minimal downtime. Parallel run approach will potentially be more resource-intensive and costly, but it ensures the highest level of availability and business continuity.

Security, Compliance, and Rollback Planning

Due to the strict financial regulations, having a robust security, compliance and rollback plan for the migration is a central topic. Not only does it ensure that the solutions on the target platform meets requirements set by regulations, but additionally guides the developer and validates the correctness of the final version on the target platform. Additionally, having clear rules, standards, and ways of working will enable the developers to work fast, and confirm that correct features are developed with appropriate tools.

Microsoft Azure provides various tools to enhance security that can be utilised to implement a comprehensive identity and access management and data protection strategies. Tools like Azure Security Center can be utilised to assess the state of security, get recommendations, and protect against external threats. Other tools such as Azure Active Directory (Microsoft Entra ID) can be used to restrict access for only authorized users to specific resources, and Azure Key Vault to store and protect sensitive data such as keys, secrets, and certificates.

Ensuring compliance with industry standards and regulations is critical for financial organisations. Azure provides tools such as Azure Compliance Manager and Azure policy to manage and enforce compliance. Compliance requirements can be set up in Azure with tools like Azure Compliance Manager, that provides solutions to assess and monitor compliances with regulatory requirements. Azure Compliance Manager offers also built-in assessments for various regulations such as GDPR (General Data Protection Regulation), HIPAA (Health Insurance Portability and Accountability Act) and ISO standards (International Organisation for Standardization standards). In Addition, Azure Policy can be utilised to create, assign, and manage policies to enforce customised organisational standards and ensure that resources are compliant with the requirements.

Depending on the application complexity, cloud migration can be very time consuming. What we learned from Waze cloud migration case study was that a robust rollback strategy will help the developers to migrate and add features

to target platform at a fast pace without worrying about breaking the application. This let the migration team at Waze to migrate their whole infrastructure with 80 million consumers from AWS to hybrid cloud environment utilising both AWS and GCP in only two months. The financial group will not be able to migrate all the AWS services to Azure in only two months, regardless how much resources would be put into it due to European and Finnish finance regulations. However, having a robust rollback strategy including backups, pipeline testing, testing rollback procedures, version control and establishing a timeline for a rollback execution will make the migration process both more secure as well more comfortable for the developers. In the end this will make the migration process faster and save both time and expenses.

In conclusion, the main objective in the second phase of the migration process, the Blueprint Design Phase is to ensure and create a structured and executable plan for migrating application from AWS to Azure. The phase involves mapping AWS services with their Azure counterparts, selecting appropriate migration strategy based on the application complexity, downtime requirements, and migration resources, and creating a plan for security, compliance, and rollback. By utilising Azure tools this chapter covers, the migration team can ensure that the application utilises the right Azure services, meets the set requirements, and is migrated efficiently and securely with minimal downtime. For a summarising table of the tools and services covered in the Blueprint Design Phase, please refer to Table 7 below.

Table 7, Blueprint Design Phase Tools and Services

Step	Tools And Services	Description
Security Measures	Azure Security Center, Azure AD, Azure Key Vault	Implements identity and access management, network security, and data protection.
Compliance Requirements	Azure Compliance Manager, Azure Policy.	Ensures adherence to industry standards and regulations.
Rollback Planning	Backups, Pipeline Testing, Testing Rollback Procedures, Version Control, Timeline for Rollback Execution.	Develops a robust strategy to quickly recover from migration issues.

4.1.3 Execution and Transition Phase

The Execution and Transition Phase is the third phase in the migration process. The primary objective is to execute the migration plan effectively,

ensuring data integrity and minimal disruption. This phase involves setting up the target environment, and executing data, database and application migration. Additionally, this chapter covers the right methods and tools to conduct thorough testing and validation to ensure that the application is performing on a required level on the target platform. The phase is divided into four dimensions: Infrastructure Setup, Data Migration, Database Migration, and Application Migration.

Infrastructure setup

Setting up the Azure infrastructure should be the first step in the process. This involves configuring virtual networks, storage accounts, and necessary compute resources. Properly set up infrastructure ensures that the target environment is prepared to host the migrated application and data. The target platform should reflect the design and planning done in earlier phases, ensuring all configurations meet the requirements identified during the assessment phase. By utilising results from assessment tools covered earlier, the target platform's infrastructure can be configured to reflect similar structure as on the source platform. Central activities to setup the infrastructure include creating Azure Virtual Networks, setting up Azure Storage accounts, and configuring Azure Virtual Machines to match the configurations on the source platform.

Database Migration

Firstly, the data and database migration is generally divided into online and offline migrations. Online migration maintains a continuous connection between source and the target database allowing a real-time transfer. Online migration minimises the downtime and ensures data consistency but will have an impact on the database performance. This approach is suitable for smaller databases that can withstand higher traffic and require minimal downtime. Offline migration involves extraction of the data files from source database to flat files and uploading them to target cloud platform. This approach is ideal for larger dataset since the migration process separates the extraction and loading processes from each other and allows for flexible scheduling for example during off-peak hours. Offline data migration tends to generally take longer time and more careful data planning to ensure consistency. The migration team in the financial company must evaluate their database migration requirements, such as uptime, performance importance, time, and resources to make the decision between offline and online migration.

Secondly, database migration consists of database schema migration, which involves transferring database tables, indexes, and view to target cloud

platform. To avoid error during the schema migration the most essential part is to ensure schema completeness on the source platform. More complex applications tend to often utilise multiple schemas and databases, which may have their interdependencies. These dependencies must be carefully managed and identified before executing the migration process to prevent sudden errors during the migration. AWS provides tools like the AWS Schema Conversion Tool (AWS SCT) that can be utilised to ensure schema completeness and identify any issues on the source platform (AWS) that need to be addressed before migration.

Thirdly, the database on the source platform might utilise default values and system functions. A function like `CURRENT_TIMESTAMP` in AWS might not have an exact counterpart on the target platform Azure. The article [29] from 2011 states that automated migration tools are not able to solve default values, although migration tools have likely been improved during the past thirteen years, Azure migration tools don't mention any details about this. By identifying and adjusting the values and functions on the target platform the migration team can ensure a smooth and efficient database migration.

Fourthly, databases like Amazon Aurora and Amazon RDS on the source platform might utilise clustered indexes to optimise database performance. Clustered indexes align the data storage with logical order, improving data retrieval performance. In order to migrate clustered indexes to Azure, a detailed report of clustered indexes on the AWS account can be generated with AWS Schema Conversion Tool (AWS SCT), which can be recreated in Azure with Azure SQL Server Management Studio or Transact-SQL (T—SQL) scripts. [61][62]

Lastly, bigger databases (>2TB) are often partitioned into smaller segments, which improves the database management and performance. However, the partition configurations are often not migrated automatically with migration tools. For the migration team, it is crucial to review and configure the source platform specific configurations on the target platform once the migration is executed.

Data Migration

Data migration involves the process of transferring the actual application data to the target platform. After the initial database migration is done, data migration can be utilised to validate the database schema migration. By utilising automated tools like Azure Data Factory to migrate AWS S3 bucket to Azure storage [63] or Azure Data Box to migrate offline data [64].

To avoid the common data migration challenges such as narrow migration window and limited system resources, the migration team can utilise the following methodologies [29]. Methods like parallel data extraction from source database and parallel uploading to target database, multithreaded processes for data loading, and avoiding index maintenance during data loading can be utilised to make the data migration process more efficient.

Such as databases, data can be migrated both through offline and online migration. The migration team must evaluate which method suits their requirements. Generally, online migration is recommended, since the process is mostly automated and quicker. Offline migrations can be optimal for larger data sets, since the migration can be scheduled during off-peak hours to minimise the impact on both source and target system.

Application Migration

Migrating the application itself from AWS to Azures might be the most complex phase in the cloud migration process. Application complexity and migration readiness varies vastly depending on the application infrastructure on the source cloud platform. The assessment phase should have provided enough information for the migration team how they should proceed with the migration, and which strategy they should follow.

The migration strategies are divided into six different processes, depending on the application requirements [figure 8]. The two least migration effort requiring strategies Retire and Retain won't be covered, since that either means that the application is going to be deprecated, or the migration is decided to be executed later. However, the four other migration strategies, Repurchase, Rehost, Refactor, and Rebuild are central migration strategies in the framework.

Repurchase involves replacing the current application on source cloud platform with a commercially available software product. This strategy is suitable for applications where instead of migrating and maintaining the application an external solution would be more cost-effective and time efficient. Example of an application could be a custom-made email service, which could be replaced with Azure Email Communication Service. Or alternatively switching a custom-made CRM solution to Microsoft Dynamics 365.

Rehost or "lift-and-shift" involves moving the application as it is. Rehost strategy is most suitable for container-based applications that can be migrated as they are without needing to refactor any cloud native services. However, the migration process will require investments in configuring the target environment to match the source environment as much as possible. Insight

from the interview shows that there is uncertainty in which container manager service should be utilised. A popular tool in the financial company's AWS infrastructure has been AWS Fargate, and the best match to it would be Azure Container Apps. Azure Container Apps provides a low-maintenance and scalable environment to deploy and run container-based applications. In case more control of the containers is required, AKS provides a more complex container orchestration tool to manage and scale individual nodes in environment.

Replatform involves minimal changes to the application to fit to the Azure environment. This strategy suits best for application that are utilising some cloud native services that cannot be migrated automatically, but instead will require some manual implementation on the target platform. Based on findings in the discovery and assessment phase, alternative application on the target platform can be mapped with the used ones on source platform.

Refactor being the most intensive and migration effort-requiring strategy, involves a complete redevelopment of the application. This strategy is suitable when the application cannot be efficiently migrated using the other methods or when significant improvements and optimisation requirements were detected in the Discovery and Assessment phase. Rebuilding the application will allow to take full advantage of Azure's cloud native capabilities, ensuring that the application is utilising best tools and services for performance, scalability, and security from ground up.

4.1.4 Post-Migration and Optimisation Phase

The Post-Migration and Optimisation phase's main goal is to ensure that the migrated application operates smoothly and efficiently in the new environment. This phase involves validating the migration's success, monitoring performance, and continuously optimising the environment. The post migration optimisation effort will not end after a specific task, but instead continue as long as the application is being developed and has consumers. In this chapter the thesis will cover methodologies and tools a migration team should follow to meet organisational and stakeholder requirements set for the application.

Testing and Validation

Testing and validation are integral to confirming that the migrated applications meet all requirements and standards. This involves conducting comprehensive tests such as functionality testing to verify that all the application features and functionalities work correctly in the target environment, performance testing to confirm that the application scales according to application

traffic, and security testing to validate that the migrated environment meets all the security requirements and identifies any vulnerabilities.

There are different alternatives how to execute validation testing in Azure. The framework recommends implementing automated functionality testing in CI/CD pipelines for example in the format of unit testing. By integrating functional tests into CI/CD pipelines, each feature can be automatically tested as code is moved through different stages of development, ensuring early detection of issues and maintaining consistent functionality. As in the article [30] the authors emphasise the importance structured test framework to manage complex cloud migration. The deployment process becomes more secure and efficient by having a clear step for verifying that all the application functionalities are working as they should. Additionally, tools like Postman and Selenium can be utilised for API and user interface testing. With Postman the development team will be able to create and execute API tests to verify that the application services communicate successfully with each other, while Selenium can simulate user interactions to ensure the front-end functionality is preserved.

One of the cloud computing's advantages is the application's ability to scale horizontally, dynamically adjusting when the application is facing high traffic. Conversely, the application should scale down during off-peak hours, for instance during nights when the application does not have that many active users. This creates a situation where the application can maintain high performance a good user experience, even when the application would be facing more traffic, as well as scale down to save energy and costs when needed. Contrary to the belief that performance testing is unnecessary for cloud-native applications due to their inherent scalability, systematic performance testing remains essential. The study [32] emphasises the importance of thorough experimentation to eliminate bottlenecks unrelated to the server being tested, thereby accurately determining the cloud platform's true performance impact. Cloud solutions are often composed of both cloud-native components and customised elements, which may bottleneck under high loads. Performance testing validates the application's ability to perform in various conditions.

Azure offers various tools for performance testing such as Azure Load Testing. Firstly, with Azure Load Testing the migration team can simulate a high number of users accessing the application to test its performance under peak load conditions. Secondly, Azure Load testing offers tools for stress testing, which can be utilised to push the application beyond its normal operational capacity to see how it handles extreme conditions and identify possible breaking points. Thirdly, Azure Load Testing support endurance testing, which can be used to run test over an extended period to ensure the

application can handle sustained usage without performance issues. Fourthly, Azure Load Testing can be utilised to run spike tests that simulates sudden increases in load to observe how the application handles abrupt traffic changes. [65]

Security testing is a very central field of software testing, especially in the financial industry. Article [57] divided sustainable cloud migration into 4 dimensions, where technology dimension contains security aspect. However, as the author states as well [57] security flaws will have not only an impact on the technological performance of the application, but as well an impact on the social dimension. Data leaks and application disturbance creates a bad reputation for the application and the company managing it, which can be costly. One of the fundamental features of cloud platforms is their inherent security and the fact that consumers do not need to manage the underlying infrastructure. However, entrusting a cloud provider with full access to your applications means that the development team must rely on the platform's security measures.

The article [56] emphasises the importance of vulnerability assessment and security Shift Left approach when migrating to a new cloud platform. Azure provides various tools and services to conduct thorough security testing. For instance, vulnerability assessment can be conducted with a tool like Azure Security Center, which identifies and addresses security weaknesses the application has in the new environment as well as provides recommendations how mitigate the security issues. Additionally, adopting the security Shift Left approach to development cycle, for instance by integrating Static Application Security Testing (SAST) with tool like SonarQube into CI/CD pipelines, the tool will analyse the code for vulnerabilities during the build process. By implementing automated testing into the deployment pipeline, developers can be ensured that their application can move forward to production.

Performance Monitoring

After the initial migration, validation, and testing are completed, continuous performance monitoring is necessary to maintain optimal performance. User preferences or user amount might change, or possibly flaws in dependencies might create performance issues that needs actions taken. Performance monitoring's main task is to follow the application's health after the application is deployed to detect anomalies in for example response times, error rates, and resource utilisation.

Azure Monitor provides a variety of tools, services and a dashboard to track, manage, and detect performance issues and manage resources. With comprehensive Azure Monitor utilisation, the migration team will be able to

identify and address issues before they impact end-user. Additionally, Application Insights, which is a part of Azure Monitor, provides deep insight into application performance and user activity, that helps assessing application health, but as well identify possible user patterns and possible scenarios where user exit the application. In addition to performance and user tracking, Azure Monitor can track resource utilisation metrics such as CPU (central processing unit), memory, and disk usage. By monitoring these metrics, the migration team can adjust their resource allocation dynamically to meet the demand, thereby optimising costs and performance. In addition to Azure Monitor, Azure Cost Management and Azure Advisor can be used to gather recommendation on cost optimisation and resource management.

The reason why performance and resource utilisation statistic are gathered is to enable the migration team to make proactive issue resolution. For instance, a critical response time can have a predefined threshold, which wen exceeded triggers a notification. This can be used to either alert developers about possible bottlenecks or either trigger the application to scale up resources or restart service. The main goal being that the system notices a performance issue in the application, triggers a notification, and automatically tries to mitigate the issue immediately.

To implement effective performance monitoring, it is essential to set up and configure monitoring tools according to key performance indicators (KPIs), setting required threshold alert, and establish automated actions for common issues. Azure Monitor configurations should be regularly reviewed and adjusted to match the required configurations as the application and its usage evolves.

Optimisation

The cloud application should be continually optimised and improved based on the findings from Performance Monitoring. Optimising the application will ensure that the application performance will continue to improve while possibly reducing costs. Key optimisation strategies include auto-scaling, cost management, and performance tuning.

Auto-scaling is a fundamental methodology in cloud application optimisation. With properly configured auto-scaling the application will be able to adjust resources based on real-time demand. Azure provides various auto-scale capabilities through services like Azure Virtual Machine Scale Sets [66] and Azure App services [67]. These tools can automatically increase or decrease the number of running instances in response to the application's load and ensure that resources are used efficiently.

One of the common problems when migration to a new cloud platform is managing costs [57]. Regularly reviewing and optimising resource usage is essential for maintaining low cost on a cloud platform. Azure Cost Management and Billing tools provide comprehensive insights into spending patterns and potential savings opportunities. Azure Cost Management and Billing can also provide saving plans which can recommend new payment methods for Azure services that are consistently utilised to save costs. Additionally, Azure can be set to trigger budget alerts if predefined threshold has been exceeded to help forecast costs and ensure that spending is kept within the required limits. All these tools combined will help the migration team to optimise their application on the new cloud platform, although there are several other cost management features in Azure this thesis does not cover.

Lastly, Performance tuning involves fine-tuning application configurations to enhance overall performance. Based on results from performance monitoring migration team can identify possible areas of improvement to enhance application performance and cost efficiency. For instance, using Azure AQL Database's built-in performance tuning features, such as automatic indexing and query performance insights, the migration team can significantly enhance the database's performance. Additionally, utilising Azure Cache for Redis can be utilised to reduce the load on databases and improve application response time by caching frequently accessed data.

In conclusion, having a structured approach how to verify migration successfulness, monitor performance, and optimise cloud applications is a central part of cloud migration process. By implementing the covered optimisation strategies, such as verifying and test application functionalities in automated CI/CD pipelines, setting up monitoring and alert systems, regularly reviewing performance data and configurations, and making adjustment based on the reviewed insights will improve both the application's performance and reduce costs. For a summary of the tools and services covered in the Post-Migration and Optimisation Phase, please refer to Table 8.

Table 8, Post-Migration and Optimisation Phase Tools and Services

Step	**Tools and Services**	**Description**
Testing and Validation	Automated Functional Testing, Performance Testing, Security Testing	Ensuring that the migrated applications meet all requirements and standards.
Performance Monitoring	Azure Monitor, Application Insight	Continuously tracking performance metric to identify and address issues.

Optimisation	Auto-Scaling, Azure Cost Management, Performance Tuning	Improving efficiency and reducing costs by leveraging Azure's capabilities

4.2 Evaluation of the Framework

The four-step cloud-to-cloud migration framework covers the cloud migration lifecycle spectrum from the beginning to end in a well-structured way by using a cloud migration structure and recommended tools in the industry. A developer at the financial company should be able to efficiently, securely and effectively be able to migrate the AWS application to Azure by following this framework regardless the application's complexity.

The findings from the literature review are used in a broad setting to cover all the different phases of cloud migration and common problems in the industry. The thesis focuses specially on the issues raised in the interviews which can be seen in Table 4. Firstly, by providing a structure step-by-step guide for cloud-to-cloud migration, the framework offers a solution to the anticipated problems of missing a clear migration strategy. Secondly, by disproving the issue of not finding AWS service counterparts on Azure solves a central issue the cloud experts in the company had. Most if not all AWS services have a counterpart on Azure. Reason why this was raised in the interview was likely due to the limited research done by the interviewees on cloud-to-cloud migration. Thirdly, technical challenges such as container management and downtime management during migration were solved by providing the right Azure tools and methodologies how these issues should be mitigated. Fourthly, establishing a knowledge sharing platform is not directly a problem that is solved in the framework, although possible solutions are discussed in chapter 5.1. Future Research. Lastly, the framework discusses the challenges in managing costs and new deployments during post-migration. The framework includes guidance on post-migration activities, such as setting up performance monitoring, managing resources, and evaluating performance. Additionally, it provides key metric and methodologies for assessing migration successfulness and continuously improve efficiency in the new cloud platform.

Since the thesis studies a broad subject, some gaps in study are inevitable. The framework provides a well-structured starting point for the financial company, but requires future enhancements, as well as an experienced cloud developer to execute the migration. Due to the material restrictions of the financial company's resources, the framework is solely basing the final product on insights from the industry case studies, migration instructions, cloud provider's documentation, and stakeholder interviews. This leaves gaps in how the framework should be fused into the company's cloud migration

strategy, and with what company specific tools the cloud-to-cloud migration should be implemented with.

Future work at the company should include careful planning on how the framework can be included into the company's cloud migration strategy, decide what role it has in it, and how the learnings from migration should be spread across the company. Generally, in the field of cloud computing, the next steps for improving the framework should include more specific details on how Azure environment should be set up, how IaC (Infrastructure as a Code) tools such as Azure ARM templates and CI/CD pipelines should be implemented and managed, and examples how cloud application should be implemented in Azure to create a more complete instruction how cloud migration should be managed. However, many of these technological gaps in the framework will most likely be solved by other internal teams within the company. As for instance a centralised CI/CD deployment pipeline is likely being developed by dedicated team, so the teams executing the migration can focus on developing new features on Microsoft Azure.

4.3 Conclusion

The cloud-to-cloud migration framework developed in this thesis provides a complete guide for migrating application and services from AWS to Azure based on the company insight and findings from the industry. The framework addresses the company specific challenges raised in the interviews, while keeping the solutions in line with organisational goals and requirements. The final product provides a cloud-to-cloud migration guide for financial company's cloud developers ensuring a structured and efficient transition while maintaining business continuity, data integrity, and optimal performance.

The framework is divided into four distinct phases – Preparation and Concept, Blueprint Design, Execution and Transition, and Post-Migration and Optimisation. Each phase covers detailed steps and best practises for each its specific needs, such as dependency mapping, migration strategy selection, and testing methodologies. Additionally, the framework showcases cloud native tools such as AWS X-Ray, Azure Migrate, and Azure Monitor that can be utilised to ensure a robust and secure migration process.

The framework emphasises the importance of thorough planning, continuous monitoring, and optimisation, which are crucial for successful cloud migration. Additionally, the framework highlights the need for ongoing adaptation and fine-tuning to meet changing business needs and technological advancements. This approach covers the whole cloud-to-cloud migration lifecycle spectrum from source platform discovery methods to post-migration

optimisation, which can serve as a valuable reference for other organisations facing similar cloud-to-cloud migration challenges.

5. Summary

The thesis presents a framework for cloud-to-cloud migrations, especially focusing on transitioning cloud applications and services from AWS to Microsoft Azure within a Finnish financial company. The study covers gaps that the academic literature and documentation provided by the cloud vendors has, which often tend to focus on migrations from on-premises infrastructure to cloud platforms. The solution this thesis provides covers various cloud migration strategies, tools, and methodologies to solve common issues often faced during cloud migrations, such as application compatibility, data integrity, compliance, and security standards.

The developed framework is divided into four phase which are:

1. **Preparation and Concept**, during which groundwork for the upcoming cloud migration is done. This involves processes such as assessing current environment, risk management, and setting clear goals for the migration.

2. **Blueprint Design**, which involves creating a migration plan, mapping used AWS services to their Azure counterparts, and addressing security and compliance measures.

3. **Execution and Transition**, during which the migration itself is executed, ensuring that databases, data, and application are transferred efficiently and securely.

4. **Post-Migration and Optimisation**, which focuses on ongoing monitoring and optimisation after the initial migration is executed. This phase provides tools and methodologies for maintaining application performance and adapt to changing requirements.

5.1 Future Research

The framework focuses on structured planning, continuous monitoring, and adaption, which supports the migration team during the complex and dynamic cloud migration. The framework functions as a practical guide for developers in the financial company, and possibly even external developers and teams facing similar issues, ensuring a smooth and secure transition while maintaining business continuity. The study covers the importance of tailored migration strategies, since cloud solutions tend to vary in complexity and cloud nativity level due to the different organisational requirements and developer preferences.

Due to the broad scope cloud migration contains and the limitation a master's thesis has, there are inevitable gaps and improvements in the framework. The solution provided in the thesis offers a general cloud-to-cloud migration process but is missing concrete examples for instance how Microsoft Azure infrastructure should be set up. In a larger company e.g. the financial company this thesis studies, network, deployment pipelines, and infrastructure related challenges are solved and implemented by independent teams. Due to this, a cloud developer's responsibility is limited to only manage and migrate the application to target platform according to the company's technical documentation. The thesis is concentrating on the singular developer's or team's challenges they'll likely face, and not solve altogether all cloud-to-cloud related issues. Future research should focus on integrating more specific tools, methodologies, and example studies, particularly for the setup and optimisation of Azure environments. Additionally, developing a more detailed approach for CI/CD pipeline implementation and performance optimisation in Azure could further enhance the framework. As cloud technology evolves, continuous updates and refinements to the framework will be necessary to maintain its relevance and applicability.

In the context of the financial company, the next step would be to discuss how and when the cloud-to-cloud migration framework would be implemented into the company's cloud strategy. The framework should be enhanced with demo sessions and possible pilot tests, to achieve a greater understanding of the framework's strengths and weaknesses. Additionally, a central topic raised in the interviews was the need for knowledge sharing platform to discuss experiences and recommendations for cloud-to-cloud migration. The framework does not provide any direct solution how a knowledge-sharing platform should be created. However, the company has had similar "guilds" for other projects requiring a feedback platform among colleagues. Establishing a cloud-migration guild could be an effective way to facilitate discussions, share insights, and disseminate experiences with cloud migration and cloud-to-cloud migration processes and tools.

Overall, this thesis and framework offers valuable insights and practical tools for organisations undertaking cloud-to-cloud migration. While focusing on the challenges and requirements in financial company, the same methodologies and tools can widely be used across different companies and industries facing a similar task.

References

[1] Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing — The business perspective. Decision Support Systems, 51(1), 176-189. University of Florida, United States. Persistent Systems, Inc., CA, United States. https://doi.org/10.1016/j.dss.2010.12.006

[2] Synergy Research Group. (2023). Cloud spending growth rate slows but Q4 still up by $10 billion from 2021; Microsoft gains market share. Retrieved 25.07.2024, from https://www.srgresearch.com/articles/cloud-spending-growth-rate-slows-but-q4-still-up-by-10-billion-from-2021-microsoft-gains-market-share

[3] Johnson, R. (2023, July 18). Cloud computing market to be worth USD 2,495.2 billion by 2032, growing at 17.8% CAGR. Acumen Research and Consulting. Yahoo Finance. Retrieved July 25, 2024, from https://finance.yahoo.com/news/cloud-computing-market-worth-usd-130600292.html

[4] Ikram, M., & Hussain, F. (2018). Software as a Service (SaaS) Service Selection Based on Measuring the Shortest Distance to the Consumer's Preferences. 403-415. https://doi.org/10.1007/978-3-319-75928-9_36.

[5] Matsepe, N., & Lingen, E. (2022). Determinants of emerging technologies adoption in the South African financial sector. South African Journal of Business Management. https://doi.org/10.4102/sajbm.v53i1.2493.

[6] Hashizume, K., Rosado, D., Fernández-Medina, E., & Fernández, E. (2013). An analysis of security issues for cloud computing. Journal of Internet Services and Applications, 4, 1-13. https://doi.org/10.1186/1869-0238-4-5.

[7] Financial company, Annual Finance Report 2023

[8] Propson, D., & Galaski, R. (2020). Forging new pathways: The next evolution of innovation in financial services [online]. World Economic Forum. Retrieved 25.07.2024 from https://www.weforum.org/reports/forging-new-pathways-the-next-evolution-of-innovation-in-financial-services

[9] Varia, J. (2010, October). Migrating your existing applications to the AWS cloud: A phase-driven approach to cloud migration. Amazon Web Services. Retrieved 25.7.2024 from https://aws.amazon.com/blogs/aws/new-whitepaper-migrating-your-existing-applications-to-the-aws-cloud/

[10] Internal source – Cloud strategy 2023

[11] Kumari, N., & Saha, A. (2014). EFFECT OF REFACTORING ON SOFTWARE QUALITY. USICT, Dwarka, Delhi, India. 37-46. https://doi.org/10.5121/CSIT.2014.4505.

[12] Varma, K., & Se, G. (2022). Efficient Scalable Migrations in the Cloud. 2022 IEEE/ACIS 7th International Conference on Big Data, Cloud Computing, and Data Science (BCD), 3-6. https://doi.org/10.1109/bcd54882.2022.9900725.

[13] Internal source – Cloud Positioning 2023

[14] Jeff Bar (2018, December) "Now Open – AWS Europe (Stockholm) Region" AWS News Blog. Retrieved 25.7.2024 from https://aws.amazon.com/blogs/aws/now-open-aws-europe-stockholm-region/

[15] F. Kaya, M. v. d. Berg, R. Wieringa and M. Makkes. The Banking Industry Underestimates Costs of Cloud Migrations. 2020 IEEE 22nd Conference on Business Informatics (CBI), Antwerp, Belgium, 2020, pp. 300-309, doi: 10.1109/CBI49978.2020.00039.

[16] Raj Bala, Dennis Smith, Kevin Ji, David Wright, Miguel Angel Borrega. 2022 Gartner Magic Quadrant for Cloud Infrastructure and Platform Services 2022 report - Gartner Research

[17] Internal source – Microsoft Enterprise Skills Initiative for memberships

[18] Kompella, L. (2021). Transitions in Information Systems Development: SME's Issues and Challenges. In Proceedings of the 23rd International Conference on Enterprise Information Systems SciTePress, pages 413-420. DOI: 10.5220/0010516904130420

[19] Oliveira, F., Eilam, T., Nagpurkar, P., Isci, C., Kalantar, M., Segmuller, W., & Snible, E. (2016). Delivering software with agility and quality in a cloud environment. IBM IBM Journal of Research and Development. https://doi.org/10.1147/JRD.2016.2517498.

[20] S. Kolb, J. Lenhard and G. Wirtz, "Application Migration Effort in the Cloud - The Case of Cloud Platforms," 2015 IEEE 8th International Conference on Cloud Computing, New York, NY, USA, 2015, pp. 41-48, doi: 10.1109/CLOUD.2015.16.

[21] H. SeetharamaTantry, Murulidhar N.N, K. Chandrasekaran (2020). Impact Analysis of Legacy System Migration to the Cloud Environment: A Focused Study. International Journal of Advanced Trends in Computer Science and Engineering. https://doi.org/10.30534/ijatcse/2020/21912020.

[22] Gass, O., Meth, H., & Maedche, A. (2014). PaaS Characteristics for Productive Software Development: An Evaluation Framework. IEEE Internet Computing, 18, 56-64. https://doi.org/10.1109/MIC.2014.12.

[23] Mahdi Fahmideh Gholami, Farhad Daneshgar, Graham Low, Ghassan Beydoun (2016). Cloud migration process—A survey, evaluation framework, and open challenges. Journal of Systems and Software. Pages 31-69, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2016.06.068.

[24] Amazon Web Services. (n.d.). Prioritization and migration strategy. Retrieved 25.7.2024, from https://docs.aws.amazon.com/prescriptive-guidance/latest/application-portfolio-assessment-guide/prioritization-and-migration-strategy.html#migration-r-type

[25] Al-Sayyed, R., Wadi' A. Hijawi, Bashiti, A. M., AlJarah, I., Obeid, N., & Al-Adwan, O. (2019). An investigation of microsoft azure and amazon web services from users' perspectives. International Journal of Emerging Technologies in Learning (Online), 14(10), 217-241. doi: https://doi.org/10.3991/ijet.v14i10.9902

[26] Internal source – Azure & GitHub development feature for Software Project creation process (Confluence)

[27] Stephen Orban (2016).6 Strategies for Migrating Applications to the Cloud. AWS Cloud Enterprise Strategy Blog. Retrieved 25.7.2024 from https://aws.amazon.com/blogs/enterprise-strategy/6-strategies-for-migrating-applications-to-the-cloud/

[28] Strauch, S., Andrikopoulos, V., Karastoyanova, D., & Vukojevic, K. (2015). Migrating e-science applications to the cloud: Methodology

and evaluation. In O. Terzo & L. Mossucca (Eds.), Cloud computing with e-science applications (Chapter 5, pp. 89-114). CRC Press/Taylor & Francis.

[29] Laszewski, T., & Nauduri, P. (2011). Migrating to the cloud: Oracle client/server modernization (1st ed.). Syngress.

[30] Rodriguez, Juan & Crasso, Marco & Mateos, Cristian & Zunino, Alejandro & Campo, Marcelo & Salvatierra, Gonzalo. (2013). Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments. Publisher: IGI Global.

[31] Rao, G. M., Srinivas, K., Sayyad, S., Venkatesh, K., & Dadheech, P. (2021). A secure and efficient data migration over cloud computing. IOP Conference Series: Materials Science and Engineering, 1099(1), 012082. https://doi.org/10.1088/1757-899X/1099/1/012082

[32] J. Mukherjee, M. Wang and D. Krishnamurthy, "Performance Testing Web Applications on the Cloud," 2014 IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops, Cleveland, OH, USA, 2014, pp. 363-369, doi: 10.1109/ICSTW.2014.57.

[33] Hugo Santiago C. Pinto, V.; R. Oliveira, R.; F. Vilela, R. and R. S. Souza, S. (2018). Evaluating the User Acceptance Testing for Multitenant Cloud Applications. In Proceedings of the 8th International Conference on Cloud Computing and Services Science - CLOSER; ISBN 978-989-758-295-0; ISSN 2184-5042, SciTePress. DOI: 10.5220/0006664000470056

[34] Godefroid, P., Lehmann, D., & Polishchuk, M. (2018). Differential Regression Testing for REST APIs. Proceedings of the ACM Symposium on Cloud Computing, 234-245. Publisher: New York, NY, USA: ACM. DOI: 10.1145/3395363.3397374.

[35] Andrade, E., & Nogueira, B. (2019). Performability evaluation of a cloud-based disaster recovery solution for IT environments. Journal of Grid Computing, 17(3), 603-621. https://doi.org/10.1007/s10723-018-9446-2

[36] Balobaid, A., Debnath, D. (2018). Cloud migration tools: Overview and comparison. In Services - SERVICES 2018 (Vol. 10975). Springer International Publishing AG. https://doi.org/10.1007/978-3-319-94472-2

[37] Trabelsi, H. R., & Laniel, M.-A. (2023). A developer's guide to cloud apps using Microsoft Azure: Migrate and modernize your cloud-native applications with containers on Azure using real-world case studies (1st ed.). Packt Publishing, Limited. EISBN: 9781804613801.

[38] Microsoft. (n.d.). Azure Migrate: Cloud migration and modernization. Retrieved 2.7.2024 from https://learn.microsoft.com/en-us/azure/migrate/migrate-services-overview

[39] Microsoft. (n.d.). Tutorial: Assess AWS VM migration to Azure. Retrieved 3.7.2024 from https://learn.microsoft.com/en-us/azure/migrate/tutorial-assess-aws

[40] Amazon Web Services. (n.d.). How to migrate your applications to the AWS Cloud. Retrieved 03.07.2024, from https://aws.amazon.com/cloud-migration/how-to-migrate

[41] Raizada, R., & Dhavle, A. (2021). Visualize your AWS Infrastructure with Amazon Neptune and AWS Config. AWS Database Blog. Retrieved 03.07.2024, from https://aws.amazon.com/blogs/database/visualize-your-aws-infrastructure-with-amazon-neptune-and-aws-config/

[42] CloudEndure. (n.d.). CloudEndure: Disaster recovery and cloud migration. Retrieved 03.07.2024, from https://www.cloudendure.com/

[43] Dynatrace. (n.d.). Cloud migration solutions. Retrieved 03.07.2024, from https://www.dynatrace.com/solutions/cloud-migration/

[44] BitTitan. (n.d.). Cloud migrations made easy. Retrieved 03.07.2024, from https://www.bittitan.com/

[45] ScienceLogic. (2023). Cloud Transformation Strategy & Solutions. Retrieved 03.07.2024 from https://sciencelogic.com/blog/cloud-transformation-101-everything-you-need-to-know

[46] Max Smolaks (2015). AWS suffers a five-hour outage in the US. DatacenterDynamics. Retrieved 08.07.2024, from https://www.datacenterdynamics.com/en/news/aws-suffers-a-five-hour-outage-in-the-us/

[47] Sebastian Moss (2017). Amazon Web Services US-East-1 goes down. DatacenterDynamics. Retrieved 08.07.2024, from

https://www.datacenterdynamics.com/en/news/amazon-web-services-us-east-1-goes-down/

[48] Tarcic, N. (2017, March 9). Waze: Migrating a multi million user app to Google Cloud Platform (Google Cloud Next '17) [Video]. YouTube. Google Cloud Tech. https://www.youtube.com/watch?v=woazNHiP-r0&ab_channel=GoogleCloudTech

[49] Rushe, D. (2013, June 11). Google buys Waze map app in bid to outsmart Apple. The Guardian. Retrieved 08.07.2024, from https://www.theguardian.com/technology/2013/jun/11/google-buys-waze-maps-billion

[50] Waze. (n.d.). About Waze. Retrieved 08.07.2024, from https://www.waze.com/about

[51] Mahendra, K. (2024, March 27). Netflix AWS migration: From data centers to the cloud. Bacancy Technology. Retrieved July 8, 2024, from https://www.bacancytechnology.com/blog/netflix-aws-migration

[52] Saraf, N. (2022, October 3). Netflix's story of migration to AWS. LinkedIn. Retrieved 08.07.2024, from https://www.linkedin.com/pulse/netflixs-story-migration-aws-nilay-saraf/

[53] Izrailevsky, Y., Vlaovic, S., & Meshenberg, R. (2016, February 12). Completing the Netflix cloud migration. Netflix. Retrieved July 8, 2024, from https://about.netflix.com/en/news/completing-the-netflix-cloud-migration

[54] Amazon Web Services. (n.d.). Capital One: Going all in on AWS. Retrieved July 8, 2024, from https://aws.amazon.com/solutions/case-studies/capital-one-all-in-on-aws/

[55] Perkel, A. (2024, May 23). Our cloud migration journey. Capital One. Retrieved July 8, 2024, from https://www.capitalone.com/software/blog/cloud-migration-journey/

[56] Alharthi, D. N. (2023). Secure Cloud Migration Strategy (SCMS): A Safe Journey to the Cloud. University of Arizona, Tucson, AZ, United States. DOI: https://doi.org/10.34190/iccws.18.1.1038

[57] Rahman, A. A. L. A., Islam, S., Kalloniatis, C., & Gritzalis, S. (2017). A risk management approach for a sustainable cloud migration. Journal of Risk and Financial Management, 10(4), 1-19. https://doi.org/10.3390/jrfm10040020

[58] Staevsky, N., & Gaftandzhieva, S. (2023). Cloud migration: Identifying the sources of potential technical challenges and issues. International Journal of Advanced Computer Science & Applications, 14(12). https://doi.org/10.14569/IJACSA.2023.0141204

[59] Smith, B. (2020, September 14). Introducing AWS X-Ray new integration with AWS Step Functions. AWS Compute Blog. Retrieved 18.07.2024 from https://aws.amazon.com/blogs/compute/introducing-aws-x-ray-new-integration-with-aws-step-functions/

[60] Microsoft. (2024, May 4). Tutorial: Discover AWS instances with Azure Migrate: Discovery and assessment. Microsoft Learn. Retrieved July 24, 2024, from https://learn.microsoft.com/en-us/azure/migrate/tutorial-discover-aws

[61] Amazon Web Services. (n.d.). AWS Schema Conversion Tool. Amazon Web Services. Retrieved 21.7.2024 from https://aws.amazon.com/dms/schema-conversion-tool/

[62] Microsoft. (2024, January 25). CREATE INDEX (Transact-SQL). Microsoft Learn. Retrieved July 22, 2024, from https://learn.microsoft.com/en-us/sql/relational-databases/indexes/create-clustered-indexes?view=sql-server-ver16

[63] Microsoft. (2024, May 15). Data migration guidance for Amazon S3 to Azure Storage. Microsoft Learn. Retrieved July 22, 2024, from https://learn.microsoft.com/en-us/azure/data-factory/data-migration-guidance-s3-azure-storage

[64] Microsoft. (n.d.). Azure Data Box overview. Microsoft Learn. Retrieved 22.07.2024, from https://learn.microsoft.com/en-us/azure/databox/

[65] Microsoft Azure. (n.d.). Azure Load Testing. Retrieved 22.07.2024, from https://azure.microsoft.com/en-us/products/load-testing

[66] Microsoft. (2024, June 14). Overview of virtual machine scale sets. Microsoft Azure. Retrieved 23.07.2024, from https://learn.microsoft.com/en-us/azure/virtual-machine-scale-sets/overview

[67] Microsoft. (2024, June 26). Automatically scale an app in Azure App Service. Microsoft Azure. Retrieved 23.07.2024, from https://learn.microsoft.com/en-us/azure/app-service/manage-automatic-scaling?tabs=azure-portal

[68] Microsoft. (2024, March 21). Monitor and tune performance. Microsoft Learn. Retrieved 23.07.2024, from https://learn.microsoft.com/en-us/azure/azure-sql/database/monitor-tune-overview?view=azuresql

[69] Microsoft. (n.d.). Azure Cache for Redis. Microsoft Azure. Retrieved 23.07.2024, from https://azure.microsoft.com/en-us/products/cache

[70] Internal Source - Company Intra Page “Miten pilvisiirtymä etenee ja mitä pilvipalveluista on hyvä tietää?” (29.1.2024). Retrieved 25.07.2024

[71] Microsoft. (2023, August 29). What is Azure Database Migration Service? Microsoft Learn. Retrieved 26.07.2024, from https://learn.microsoft.com/en-us/azure/dms/dms-overview

[72] Rana, M. E., & Wanabrahman, W. N. (2018). A review of cloud migration techniques and models for legacy applications: Key considerations and potential concerns. Journal of Computational and Theoretical Nanoscience, 15(2), 111-124. https://doi.org/10.1166/asl.2018.11142

[73] Microsoft. (2024, May 10). What is Azure Files? [Article]. Microsoft Learn. Retrieved 30.7. from https://learn.microsoft.com/en-us/azure/storage/files/storage-files-introduction

Appendices

Interview Questions

1. Expectations and Preparations

- Can you outline any preparations or adjustments your team has started making or plans to make in anticipation of the migration? (Azure certs?)
- What resources or knowledge do you believe are essential for a smooth transition?

2. Technical Challenges and Solutions

- What technical challenges do you anticipate in migrating applications from AWS to Azure?
- Have you identified any specific AWS services or features that might not have direct equivalents in Azure? How do you plan to address these discrepancies?
- What tools, if any, have you considered or used to facilitate cloud service migrations?

3. Support and Resources

- What kind of support and resources would you find most valuable during a platform migration?
- Are there any gaps in knowledge or resources that, if filled, could make the migration easier or more efficient?
- In considering the migration process for your application from AWS to Azure, what specific challenges or needs do you hope this framework will address for you?

4. Post-Migration

- Once the migration is complete, what are your biggest concerns regarding the adaptation to Azure?
- How do you plan to measure the success of the migration for your applications?
- What kind of ongoing support or resources would be helpful to ensure smooth operations and development on Azure?

Interview Participant Invitation Message

Seeking Volunteers for Cloud Migration Interview

Hello *Developers*,

I'm conducting interviews for my master's thesis "Framework for Seamless Cloud Service Migration: A Developer and Software Perspective," aiming to gather insights that will help me craft a comprehensive migration framework.

I'm looking for five to ten developers and tech leads who are (or have been) directly involved in developing, implementing, or maintaining applications on AWS that are facing migration to Azure over the next six years. Your insights will be valuable in identifying expectations, challenges, solutions, and the kind of support needed for this transition.

If you're willing to share your experiences and contribute to this initiative, please respond to this message, or reach out directly. The interview will take approximately 30 minutes, and your participation will be central in shaping our cloud-to-cloud migration strategy.

The thesis is a collaboration between Aalto University and *confident*.

Thank you for considering this opportunity.

Best,
Roope Paajanen

* Company name hidden