Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 225 (2023) 3649–3659

Procedia Computer Science

www.elsevier.com/locate/procedia

27th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2023)

Challenges In Optimizing Migration Costs From On-Premises To Microsoft Azure

Florin Olariu[a,*], Lenuța Alboaie[a]

[a] Faculty of Computer Science, Alexandru Ioan Cuza University, Strada General Henri Mathias Berthelot Nr. 16, Iași 700259, România

Abstract

The current paper analyzes the feasibility of a modular web application's migration procedure from on-premises to the cloud. Our focus is on identifying cost savings and options for hosting the application in the cloud. The research specifically examines the impact of architectural decisions records (ADR) on a modular monolith use case, utilizing .NET Core for the backend and Angular for the front end, with Clean Architecture as the design pattern.

We investigate different cloud models (IaaS, PaaS, Serverless) considering project management's triple constraints (time, cost, performance). The study demonstrates that a modular monolith can be migrated with varying effort and costs depending on the chosen cloud model and technology stack. We explore optimizations such as resource utilization, licensing fees, and cost reduction through infrastructure reservations. Our findings show that the migration cost can range from a 20% increase with IaaS to approximately 70% cost reduction with the Serverless strategy compared to the on-premises environment, using equivalent resources.

We also explore methods to lower expenses for each model, including resource modifications, Linux operating systems, and longer resource reservations. Considering limitations, we propose a two-stage migration strategy: initially lifting and shifting the application to IaaS with cost optimization, and subsequently migrating to PaaS for scalability and simplified resource management in the long term.

Peer-review under responsibility of the scientific committee of the 27th International Conference on Knowledge Based and Intelligent Information and Engineering Systems

Keywords: Cloud Migration; Modular monolith; Lift and Shift; Clean Architecture; Optimizing costs; Optimizing resources; Azure Pricing.

* Corresponding author.
E-mail address: olariu@gmail.com

Peer-review under responsibility of the scientific committee of the 27th International Conference on Knowledge Based and Intelligent Information and Engineering Systems
10.1016/j.procs.2023.10.360

1. Introduction

In today's rapidly evolving technological landscape of Cloud/Edge Computing, businesses face intense pressure to stay competitive by leveraging the latest tools and technologies. One of the most significant challenges faced by many organizations is managing the costs of maintaining on-premises infrastructure. The capital expenditure [1] involved in building and maintaining data centers can be prohibitively expensive, particularly for smaller businesses. This is where a cloud computing platform, such as Microsoft Azure, can help. By migrating to Cloud, businesses can significantly reduce their IT costs while using the latest cloud-based technologies. Businesses can lower their spending on hardware, power, cooling, and other infrastructure-related costs by finding a balance between on-premises infrastructure and a cloud-based one. With Azure, organizations may switch from a CapEx model to an operating expenditure (OpEx) model, where they only pay for the storage, computation, and bandwidth they use.

According to Michael S. Dobson [2] and reiterated multiple times within books like “The fast forward MBA in Project Management” [3] or “Project Management: A Systems Approach to Planning, Scheduling, and Controlling” [4], three main constraints, costs, time, and performance is a triangle of limitations that should be carefully analyzed for the migration process. Understanding the triple constraint is essential to ensuring project success, as it helps project managers make informed decisions about how to allocate resources and manage expectations. Based on these considerations, in section 2 we examine a few of the most popular architectural styles in commercial (business-oriented) applications to identify common behaviors and patterns for migrations.

In section 3 we analyze how they might be transferred from an on-premises environment to Azure while keeping an eye on factors like prices, transition times, and technical difficulties. For our use case, we propose a modular monolith [5], called Hermit Portal that uses .NET Core for the backend and Angular for the front end. The backend is using Clean Architecture [6] as an architectural design pattern.

Flexera [7] a pioneer in cloud management and optimization, published in their annual report [8] that across all enterprises, the optimization of current cloud use has climbed to 59% in 2022. Also, in [9] we meet the Microsoft vision that projects a decrease in our costs from on-premises by simply moving our application to the cloud. In our proposal, in terms of costs, we have explored variations of each cloud model, trying to optimize among other resource usages, licensing fees, and ways of reducing costs by reserving infrastructure upfront for a specific period. For cost estimates, we have made use of the Azure TCO calculator [10], Azure Pricing Calculator [11], and Azure Pricing pages. There are many companies, that have adopted cloud-based architectures at various levels [12] and in this article, we analyze and demonstrate that in this process are many technical challenges and constraints that can be reflected in the cost level.

To determine if a specific cloud model is a good fit for the application and if we can bring overall costs down, we will concentrate on the following service models: Serverless (Azure functions or FaaS), PaaS, and IaaS. We explored variations of each cloud model, trying to optimize among other resource usages, licensing fees, and ways of reducing costs by reserving infrastructure upfront for a specific period. Therefore, we have determined that moving to the cloud can wind up costing anywhere between 20% more with IaaS to approximately 70% less with the Serverless strategy when employing equal resources to the on-premises environment. Between these two extremes, we have attempted to employ various methods to lower the expenses for each model, such as resource modification, the use of Linux operating systems rather than Windows, or resource reservations for a longer duration. Due to other limitations, we've determined that the migration strategy, for types of systems similar to the one we used, should be carried out in two stages, first lifting, and shifting the application to IaaS (applying costs optimization to licensing and reservation of resources) in the short term and, on the other hand, migrating to PaaS in the long term to benefit from the scalability and simplicity of resource management.

2. Architectural Patterns Towards Clean Architecture Principles

At this moment, there are numerous architectural patterns that can be used in different contexts. To replicate a database, for instance, or to connect several peripheral devices to a bus (master-slave drives), we must use the master-slaves pattern Similarly, if we need to do Android development or manage notification services, we must use the event-bus pattern [13].

On the other hand, when we need to structure business applications that need to be decomposed into a group of subtasks, each subtask being able to a become an abstraction for a specific level, we need to make use of the **layered pattern**.

Back in 1967, Melvin Conway said that: “the organizations that design systems are compelled to create designs that are exact replicas of these organizations' communication arrangements”. This statement, quoted in multiple publications (e.g. [14], or [15]) gained prevalence and nowadays is also known as “Conway’s Law”.

Since, many companies are developing applications, by simply separating their developers by layers (presentation layer – frontend developers, application layer and business layer – backend developers, data layer – database administrators, also known as DBAs), this architectural pattern becomes a common option for almost all the business applications. Many architectural patterns, developed over the past 20 years, include more and more features, such as reduced coupling from both a coding and an infrastructure standpoint. Here are a few of them that we can emphasize: Onion Architecture [16], Hexagonal Architecture [17], Command and Query Responsibility Segregation [18], and Clean Architecture [6]. Robert C. Martin noticed that all the architectures we’ve discussed above can be added under a big umbrella of Clean Architecture and noticed that although the specifics of each of these architectures differ slightly, they are all extremely similar. The separation of concerns is the shared goal of all of them. Each of them separates the software into layers to do this. Each contains at least two layers, one for user and system interfaces and the other for business rules. The author also noticed that the above architectures produce a system with the following benefits: Independent of the framework, Testable, Independent of UI, and Independent of databases. Based on these observations in [6] is proposed the architectural diagram for all the situations:

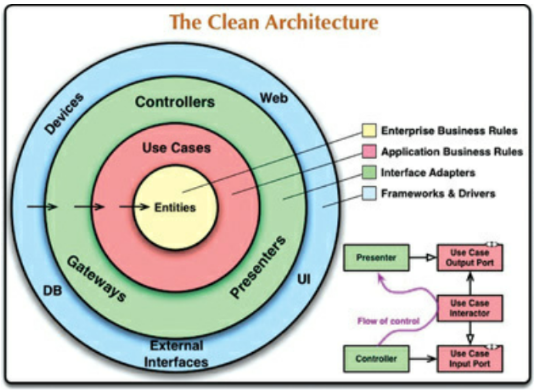

Figure 1: The Clean Architecture pattern proposed by Robert C. Martin [6]

In Figure 1 we can identify some concentric circles representing different areas in the software. The fact that the outer circles are mechanisms, and the inner circle is composed of policies is what we need to emphasize in this situation. The rule that makes this system work is what Robert C. Martin called: the ***“Dependency rule: Source code dependencies must point only inward, toward higher-level policies”*** [6]**.** The Dependency Rule, coined by Robert C. Martin, has been highlighted by Jeffrey Palermo in Onion Architecture and Alistair Cockburn in Hexagonal Architecture as well. Complying with simple rules may seem straightforward, but it plays a crucial role in positively impacting critical aspects such as Application testability, Application maintainability, and Application flexibility (changing external databases, and UI implementations because some of them can become obsolete). In Figure 2 we depicted a high-level overview of the Hermit Portal by using a C4 Model [19]:

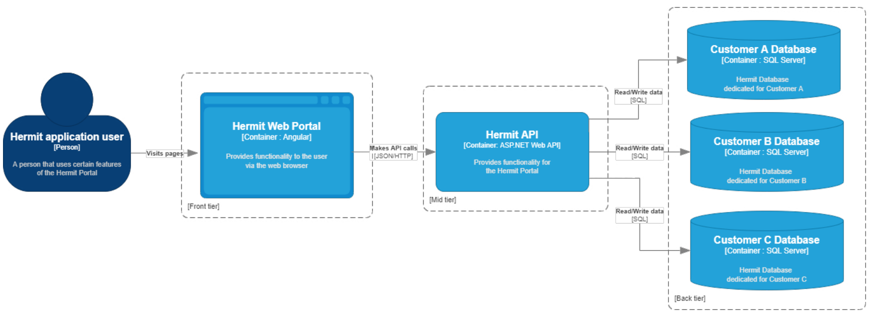

Figure 2: High-level overview of the Hermit Portal architecture using the C4 Model

In conclusion, based on Robert C. Martin's proposed pattern, Hermit API serves as the core module's representation, and the Hermit Web Portal and customer databases serve as the exterior (UI and databases) representations.

3. Case study – migrating a modular monolith based on a Clean Architecture pattern from on-premise to Microsoft Azure

According to Business Formation Statistics [20], since the beginning of the previous decade, there has been a continuous increase in the number of business applications. In 2010 2.50 million new business applications were submitted. Ten years later, in 2020, this number increased by 75% to 4.38 million. Additionally, it was a 24.7 percent gain from 2019 and by far the largest growth in the previous ten years. This results in making use of capital expenditure which means a high upfront cost for the hardware and dedicated infrastructure personnel being able to manage the infrastructure. In [9] the Microsoft perspective was that we will be able to determine why migration is most prevalent and that's because cost savings are the primary driver of cloud migration, even at the basic IaaS level. Any business can spend less on infrastructure management, server or VM provisioning, and hardware maintenance by utilizing more managed infrastructure services.

On the other hand, after agreeing to move their programs to the cloud, any organization will choose IaaS over more advanced options like PaaS primarily because it will be more appropriate for them.

The quickest route to the cloud is to transition to a setting that is comparable to your current on-premises setting because it has a lower learning curve. The need to migrate to the cloud can arise from various reasons. Some of the most known reasons are scalability needs, cost reductions, backup and recovery across regions, minimal IT infrastructure overhead, and easy monitoring [21].

Migrating to the cloud should be a decision that takes into consideration a lot of factors, as the overall process is far from risk-free. Depending on the cloud computing model level we might have added development costs that need to be done for the migration to be feasible.

Additionally, we must consider the expense of migrating the infrastructure and its dependencies, the training of the staff, any potential downtime, and the danger of data loss if things are not well planned.

In this article, we will concentrate on the cost savings that the Azure Cloud provider could offer us for a specific use case application.

3.1. Use case of Hermit Portal application.

To evaluate various cloud solutions and their potential cost reduction benefits, we will make use of a system that was developed into an IT company as a portal for HR department management. In this paper will reference this system as Hermit Portal, given the fact that we are using actual statistics of the production environment, we have changed the name of the application to comply with the NDA and this explains why Figure 2 shows a mixture of C4 Level 1 and C4 Level 2 [19]. The Hermit Portal is a SaaS web application, built with multi-tenancy approach. From the infrastructure perspective, it is built using the 3-tier architecture, and everything is structured under its own domain, as a modular monolith [5]. Multi-tenancy is achieved using the one database per tenant model, which at the time of writing represents more than 300 databases in production. As a technology stack, we are using .NET Core for the backend, Angular for the frontend, and SQL Server for storage.

In the table below, we have the average usage of the application, per month.

Table 1. Usage of Hermit portal and total bandwidth

UA - Number of unique users per month (avg)	50.000
SA - Number of sessions per user/month (avg)	5
BA - Bandwidth usage per session in MB (avg)	6
BT - Total bandwidth used/month (UA*SA*BA)	1.5TB

The on-premises infrastructure for the self-service portal is composed of two web servers, two application servers, and two database servers, configured as an always-on cluster group, for high availability. The actual configuration of the on-premises VMs can be found in Table 2.

Table 2. Server configuration for on-premises Hermit Portal

Server	OS	CPU	RAM	Storage
WebServer 1 (WS1)	Windows	4 cores (Intel Xeon E2683 v3) 2.0 GHz	8 GB	100GB
WebServer 2 (WS2)	Windows	4 cores (Intel Xeon E2683 v3) 2.0 GHz	8 GB	100GB
AppServer 1 (AS1)	Windows	4 cores (Intel Xeon E2683 v3) 2.0 GHz	8 GB	100GB
AppServer 2 (AS2)	Windows	4 cores (Intel Xeon E2683 v3) 2.0 GHz	8 GB	100GB
DbServer 1 (DS1)	Windows	4 cores (Intel Xeon E2683 v3) 2.0 GHz	32 GB	1.6TB
DbServer 2 (DS2)	Windows	4 cores (Intel Xeon E2683 v3) 2.0 GHz	32 GB	1.6TB

3.2. Migrating to IaaS

IaaS is a pay-as-you-go type of cloud service that offers computing, storage, and network resources. Some examples of services are virtual machines and storage [21]. In IaaS, the cloud provider is responsible for network, storage, servers, and virtualization. The rest of the concerns, from the operating system to the application and data are the responsibility of the customer. Using IaaS does not exclude the need for infrastructure specialists in the organization, but it facilitates a leaner migration to cloud solutions using methods like lift and shift.

Lifting and shifting to the cloud is one of the easiest and most accessible ways to migrate software systems to the cloud. It is also called rehosting and it basically implies copying an application from an on-premises environment to the cloud. Since the application may be deployed unchanged to a new environment with the same specifications (if all dependencies have already been resolved and are reachable from within that cloud provider), the migration expenses are quite minimal [22].

Using the lift and shift strategy, we can utilize the Azure TCO calculator to assess the possible cost savings over the course of five years.

Since our use case is based on the Windows operating system for the VMs, we need to mention the fact that one of the main costs will be represented by the software licenses for Windows or SQL Server. Azure offers the possibility to reuse the on-premises licenses for Windows OS and SQL Server via Azure Hybrid. This will result in bringing costs down in our Azure infrastructure. One point of attention is that Azure offers discounts if the VMs are reserved upfront for a specific timeframe (1 year / 3 years). The discounts applied are 41% for the 1-year reservation and 64% for the 3-year reservation. Using the Azure TCO calculator, we can see that for the 5-year timeframe it automatically makes use of the 3-year and 1-year discounts. As we will see in the following sections, not all Azure services have the option of reserving resources. Below we can see a monthly cost calculation for one of the F4s v2 VMs (4 vCPU, 8 GB RAM) that we will use for our migrated infrastructure.

Table 3. Cost comparison between pay-as-you-go and reserved resources.

Machine type	Cost per month including Azure Hybrid Benefit		
	Pay-as-you-go	1 year reserved	3 years reserved
F4s v2 VMs	124,77 EUR	73,93 EUR	45,03 EUR

Table 4. Cost comparison over 5 years, for similar infrastructure on-premises and IaaS cloud.

Category	**On-premises (EUR)**	**Azure IaaS (EUR)**	**Azure IaaS Reserved (EUR)**	**Azure IaaS Hybrid (EUR)**	**Azure IaaS Reserved Hybrid (EUR)**
Compute	90.051,79	137.840,59	121.487,39	43.579,58	27.600,91
Hardware	*39.036,14*	N/A	N/A	N/A	N/A
Software	*6.894,70*	N/A	N/A	N/A	N/A
Electricity	*6.200,15*	N/A	N/A	N/A	N/A
Virtualization	*9.031,80*	N/A	N/A	N/A	N/A
Database	*28.889,10*	N/A	N/A	N/A	N/A
Data Center	8.577,60	0	0	0	0
Networking	19.980,40	803,82	803,82	803,82	803,82
Storage	1.190,95	7.028,60	7.028,60	7.028,60	7.028,60
IT Labor	10.878,40	10.878,40	10.878,40	10.878,40	10.878,40
Total	**130.679,23**	**156.551,41**	**140.198,28**	**76.299,77**	**46.311,79**

According to Table 4, lifting and shifting to equivalent cloud VMs with Azure Hybrid enabled would result in cost savings of €84.367,44, over the course of 5 years. This translates into approximately 65% cost reduction on infrastructure, compared to on-premises. An important note is that the Azure infrastructure, for our use case, could end up costing almost €27.000 more than the on-premises setup, over the course of 5 years, if we would not make use of Azure Hybrid and the reservation of resources. As we can notice, the biggest improvement that we can make, in terms of costs, is to make use of the Azure Hybrid benefit, which basically wipes the software licensing costs (assuming we already have those licenses in on-premise). Reserving the hardware for 1-3 years is another option to reduce costs. This however needs to thoroughly be evaluated, in terms of the future lifecycle of the application and the load that we expect, since we will have those devices “locked” and we will have to pay for them no matter what, for the entire period of the reservation. This basically results in less flexibility from the infrastructure perspective.

3.3. Migrating to PaaS

PaaS is a cloud solution that offers customers the infrastructure necessary to deploy their web applications. PaaS offers the same capabilities as IaaS plus a few additional ones. For example, the cloud provider also takes care of the operating system, the middleware, and the runtime, which reduces the overall time needed to be invested by the team, to handle infrastructure concerns and increases the speed of the delivery process [23].

The biggest advantage of using PaaS is that the development team needs to focus only on the development part and the deployment of the application. There is no need to configure web servers, like IIS, install certain runtimes, or take care of the underlying updates and patches needed for these infrastructure components.

The same applies to the part of the database, where, for this cost estimation we will use Azure’s managed SQL solution, two instances of General-Purpose Single Gen5 4 vCores.

Table 5. Cost comparison over 5 years, for similar infrastructure on-premises and PaaS cloud.

Category	Cost on-premises (EUR)	Cost Azure PaaS (EUR)	Cost Azure PaaS Hybrid (EUR)
Compute	90.051,79	96.604,50	65,319.70
Hardware	*39.036,14*	N/A	N/A
Software	*6.894,70*	N/A	N/A
Electricity	*6.200,15*	N/A	N/A
Virtualization	*9.031,80*	N/A	N/A
Database	*28.889,10*	N/A	N/A
Data Center	8.577,60	0	0
Networking	19.980,40	803,82	803,82
Storage	1.190,95	6.416,60	6.416,60
IT Labor	10.878,40	0	0
Total	**130.679,23**	**103.825,30**	**72.540,53**

We used two B3 Basic App Services with 2 instances each to match our on-premise configuration (4 core, 7 GB RAM). There are no resource reservations available for Basic plans, so comparing IaaS with PaaS is difficult. Azure currently only offers reservations for Premium V3 plans, which have more resources than we need (e.g., 4 cores, 16 GB RAM) and can increase costs. Based on monitoring findings (by using Application Insights) and the new configuration, we can assess cost savings for both on-premises and cloud. We will focus on evaluating the Azure Hybrid benefit for the cloud solution, as it's the most suitable when migrating from existing infrastructure with acquired licenses.

Table 6. Cost comparison over 5 years, for adjusted infrastructure on-premises, PaaS, and IaaS cloud.

Category	On-premises (EUR)	On-premises adjusted (EUR)	Azure IaaS Reserved Hybrid adjusted (EUR)	Azure PaaS Hybrid adjusted (EUR)
Compute	90.051,79	94.523,00	22.444,01	55.931,00
Hardware	*39.036,14*	*43.540,59*	N/A	N/A
Software	*6.894,70*	*6.894,70*	N/A	N/A
Electricity	*6.200,15*	*6.200,15*	N/A	N/A
Virtualization	*9.031,80*	*9.031,80*	N/A	N/A
Database	*28.889,10*	*28.889,10*	N/A	N/A
Data Center	8.577,60	8.577,60	0	0
Networking	19.980,40	20.757,42	803,82	803,82
Storage	1.190,95	1.190,95	7.028,60	6.049,60
IT Labor	10.878,40	10.878,40	10.878,40	0

Total	**130.679,23**	**135.927,30**	**41.154.83**	**62.784,80**

As can be seen, there is a clear anomaly in the TCO calculator, regarding the on-premises adjusted. Azure calculator evaluates a 2-core CPU as being two 1-core processors, whereas it evaluates a 4-core CPU as being 1 processor, resulting in an increased hardware cost.

3.4. Migrating Serverless and static web pages.

For this approach, based on the computations we can summarize that:

Table 7. Overall serverless costs (* the Database part was estimated using the Azure PaaS solution)

Service Name	**Cost/year**	**Cost / 5 years**
Azure Static Web Apps	3.281,16 EUR	16.405,8 EUR
Azure Functions	1.844,04 EUR	9.220,2 EUR
Azure SQL*	6.716,96 EUR	13.634,78 EUR
Total	11.842,16 EUR	39.260,78 EUR

As we can see, the highest cost reduction is done by the backend improvements, by using Serverless with Azure Functions. As mentioned in the previous sections, these calculations are done using the consumption plan which comes with some drawbacks, like performance penalties when encountering "cold starts". Depending on the type of application, this might be an acceptable trade-off or not, but in our case, for the Hermit portal, adding performance penalties is not an option, even if it ends up being the cheapest solution. One point of attention is regarding the Azure Static Web Apps which end up having a higher cost than the backend processing. The main cost driver for the Static Web Apps is bandwidth usage, which we've estimated at 1.5 TB per month. Also, what we've observed is that the Azure Functions bandwidth costs per GB are 44% of the price of Azure Static Web Apps bandwidth costs, so projects that take this path should invest time in optimizing the bandwidth usage from the Static Web Apps perspective, to reduce the overall infrastructure costs.

4. Conclusions

4.1. Outcomes

This research reveals that hosting a web application in the cloud offers multiple options, each with its own variations.

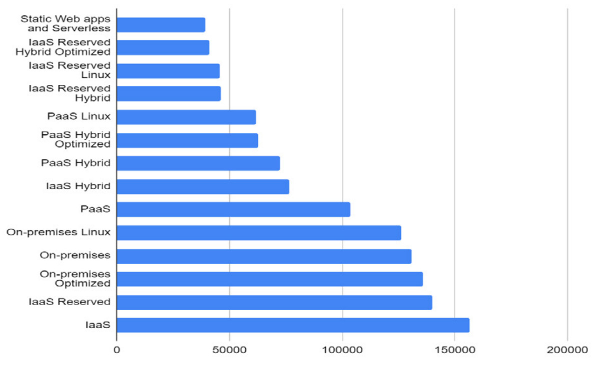

Figure 3: Summary of estimated costs comparison over 5 years using various migration strategies.

Based on Figure 3, as a consequence of our experiments, we can notice that IaaS surpasses the cost of an on-premises environment with an identical production configuration. On the other end of the spectrum, Static Web Apps and Serverless offer the lowest cloud infrastructure costs. However, the development perspective should be considered as well, as some methods may require additional expenses to make them deployable. The architecture of the Hermit Portal allows for flexibility in choosing any of the researched options. Transitioning to Serverless incurs some development costs but it can be achieved with reduced expenses if the architecture supports the necessary refactoring. For licensing, existing Windows or SQL Server licenses can be repurposed through Azure Hybrid, and significant cost optimizations can be achieved through resource reservation. Unfortunately, there is no one-size-fits-all solution for cost optimization, and trade-offs must be considered. Static Web Apps and Serverless are cost-effective in terms of infrastructure, offering pay-as-you-go models. However, trade-offs include potential performance issues with serverless functions and additional development time for adjusting to the new working method. This is more suitable for new projects willing to accept performance compromises. For applications with heavy loads, such as large social media platforms, Serverless may not be the best fit due to pricing models and bandwidth costs. The second-best option in terms of infrastructure cost is IaaS Reserved Hybrid Optimized. Migrating to IaaS incurs no extra development costs but requires managing all infrastructure concerns above the operating system. The main risk with this option is being locked into the chosen resources for the entire reservation period (1 or 3 years), limiting flexibility to downscale or decommission VMs earlier for cost reduction." To minimize risks and optimize costs for the Hermit Portal, we suggest a two-step approach. Initially, adopt IaaS Reserved Hybrid Optimized for easy migration and quick time to market, leveraging existing licenses. In the mid-term, consider migrating to PaaS for cost-effective infrastructure and reduced maintenance, enabling the development team to focus solely on application development while benefiting from scalability and elasticity.

4.2. Summary

Our article focuses on two main areas: an overview of architectural design trends in business applications over the past 20 years, and a use case for migrating a SaaS application to Microsoft Azure using multi-tenancy and hexagonal architecture. We prioritize cost optimization and explore Azure options like IaaS, PaaS, and FaaS. Our approach involves the "Lift and shift" strategy. In [9] we encounter phases as general recommendations from Microsoft Azure migration strategy. As we depict in Figure 4, through the experiences performed and presented in this article, we added supplementary steps to these phases.

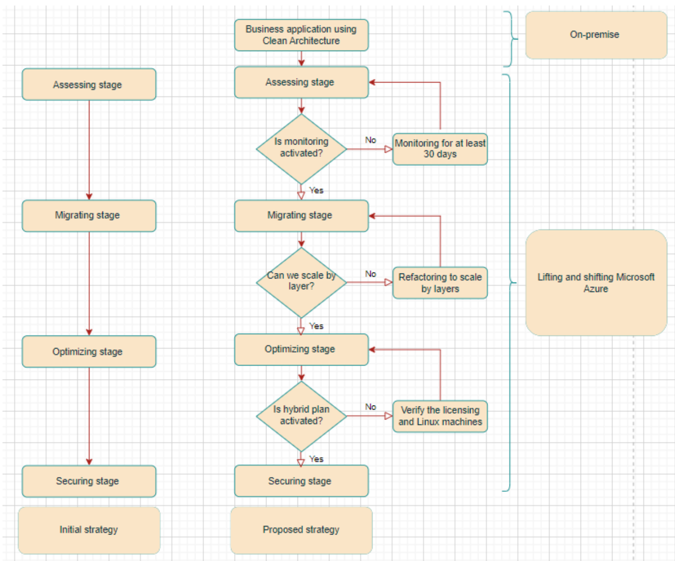

Figure 4: Initial strategy proposed by Microsoft vs improved strategy proposed by our paper.

We present enhanced phases and additional steps to Microsoft Azure's migration strategy, building upon our experiences and findings. As illustrated in Figure 4, we have identified key recommendations and expanded upon them to provide a more comprehensive approach. By incorporating these supplementary steps, we aim to improve the overall migration process and ensure a successful transition to the Azure platform. Figure 4 in our study illustrates the proposed migration strategy, which includes a range of tactics and patterns. Through a comprehensive analysis and comparison of these options, we have made significant advancements in our approach. One notable improvement is the introduction of intermediary checks, which play a crucial role in achieving better results throughout the migration process. Based on the insights derived from Figure 4 and our study's findings, we have identified two key aspects. First, for certain business applications, it is possible to choose a single alternative among IaaS, PaaS, or Serverless, depending on the specific requirements and constraints. This allows for a focused and streamlined migration approach tailored to the application's needs. Secondly, we have also recognized the effectiveness of an interactive strategy, specifically a two-step strategy, as demonstrated in the case of our Hermit Portal. This approach involves a sequential migration process that combines different alternatives based on the specific characteristics of the application. By leveraging this interactive strategy, we can optimize the migration process and achieve better outcomes. In summary, our enhanced approach, as depicted in Figure 4 and supported by our study's findings, emphasizes the importance of intermediary checks, the flexibility to choose between single alternatives or an interactive strategy, and the overall goal of achieving improved results in Azure migration.

Acknowledgements

Authors are thankful to Romanian Ministry of Research, Innovation and Digitization, within Program 1 – Development of the national RD system, Subprogram 1.2 – Institutional Performance – RDI excellence funding projects, Contract no.11PFE/30.12.2021, for financial support.

We also would like to express our sincere gratitude to Centric IT Solutions Romania for their invaluable support throughout this research endeavor.

References

[1] *Guidance on applying inherent safety in design: reducing process safety hazards whilst optimizing CAPEX and OPEX*, 2nd edition. London: Energy Institute, 2014.

[2] M. S. Dobson, *The triple constraints in project management*. in Project management essential library. Vienna, VA: Management Concepts, 2004.

[3] E. Verzuh, *The fast forward MBA in project management*, Fifth edition. in Fast forward MBA series. Hoboken, New Jersey: Wiley, 2016.

[4] H. Kerzner, *Project management: a systems approach to planning, scheduling, and controlling*, Eleventh edition. Hoboken, New Jersey: John Wiley & Sons, Inc, 2013.

[5] V. Vernon and T. Jaskuła, *Strategic monoliths and microservices: driving innovation using purposeful architecture*, First. in The Addison-Wesley signature series/a Vaughn Vernon signature book. Boston: Addison-Wesley, 2021.

[6] R. C. Martin, *Clean architecture: a craftsman's guide to software structure and design*. in Robert C. Martin series. London, England: Prentice Hall, 2018.

[7] "Flexera."

[8] "State of the cloud Report 2022," 2022. https://resources.flexera.com/web/pdf/Flexera-State-of-the-Cloud-Report-2022.pdf

[9] O. Microsoft, "Lift and Shift approach - Microsoft," 2021. https://docs.microsoft.com/en-us/dotnet/architecture/modernize-with-azure-containers/lift-and-shift-existing-apps-azure-iaas - last review 29.05.2023

[10] Microsoft, "Total cost of Ownership (TCO) calculator," 2022. https://azure.microsoft.com/en-us/pricing/tco/calculator/

[11] O. Microsoft, "Azure pricing," 2022. https://azure.microsoft.com/en-us/pricing/?&ef_id=CjwKCAjwmJeYBhAwEiwAXlg0Ab8SxMzbKswnREVMVQa-2aaHA33uWObFF-xRBWvX550cXbLuFs2FVxoCW-QQAvD_BwE:G:s&OCID=AIDcmmkvpj1ueg_SEM_CjwKCAjwmJeYBhAwEiwAXlg0Ab8SxMzbKswnREVMVQa-2aaHA33uWObFF-xRBWvX550cXbLuFs2FVxoCW-QQA - last review 29.05.2023

[12] "AWS-Azure-Google," *Cloud adoption*. https://appinventiv.com/blog/aws-vs-azure-vs-google-cloud/

[13] *Software architecture patterns*. Place of publication not identified: O'Reilly Media, 2015.

[14] A. Thackray and D. C. Brock, *Moore's law: the life of Gordon Moore, Silicon Valley's quiet revolutionary*. New York: Basic Books, a member of the Perseus Books Group, 2015.

[15] M. P. e Cunha, *Elgar introduction to designing organizations*. in Elgar introductions to management and organization theory. Cheltenham, UK ; Northampton, MA: Edward Elgar Publishing, 2022.

[16] J. Palermo, "Onion Architecture," 2008. https://jeffreypalermo.com/2008/07/the-onion-architecture-part-1/ - last review 29.05.2023

[17] A. Cockburn, "Hexagonal Architecture," *Hexagonal Architecture*, Sep. 04, 2010. https://alistair.cockburn.us/hexagonal-architecture/

[18] M. Fowler, "CQRS," 2011. https://martinfowler.com/bliki/CQRS.html - last review 29.05.2023

[19] "C4 Model," *C4 Model*, 2023. https://c4model.com/ - last review 29.05.2023

[20] BFS, “Business Formation Statistics,” *Business Formation Statistics*, 2021. https://www.census.gov/econ/bfs/index.html?
[21] “Benefits of cloud migration,” *Benefits of cloud migration*, Apr. 12, 2021. https://azure.microsoft.com/en-us/overview/cloud-migration-benefits-challenges - last review 29.05.2023
[22] IBM, “IBM, ‘Lift and Shift,’” *IBM, “Lift and Shift,”* May 12, 2021. https://www.ibm.com/cloud/learn/lift-and-shift
[23] “What is PaaS?,” *What is PaaS?*, Apr. 12, 2021. https://azure.microsoft.com/en-us/overview/what-is-paas/ - last review 29.05.2023