Investigating Systems Modernisation: Approaches, Challenges and Risks

Gareth Hogan[1], Patricija Shalkauskaite[1], Mengte Zhu[1], Martin Derwin[1], Murat Yilmaz[2] [0000-0002-2446-3224], Andrew McCarren[1,3] [0000-0002-7297-0984], and Paul M. Clarke[1,4] [0000-0002-4487-627X].

[1] School of Computing, Dublin City University, Ireland
{gareth.hogan22, patricija.shalkauskaite2, mengte.zhu2, martin.derwin2}@mail.dcu.ie
[2] Department of Computer Engineering, Gazi University, Ankara, Turkey
my@gazi.edu.tr
[3] Insight, the Science Foundation Ireland Research Centre for Data Analytics
andrew.mccarren@dcu.ie
[4] Lero, the Science Foundation Ireland Research Centre for Software
paul.m.clarke@dcu.ie

Abstract.
Findings indicate that software system modernisation projects are often more complex and costly than initially thought, and that they risk failure as a result. Layered in business processes, workflows, historical implementation and evolution decisions, legacy systems have remained in use because they have been useful and not easily replaced. Modernizing these systems can involve partial functional migration or entire re-implementation, the extent to which is utilized requires deep upfront planning and investigation.

In this paper, we provide a multivocal literature review (MLR) of Software System Modernisation, examining its ongoing relevance amidst continuous technological advancements. Our research provides an analysis on the role of a legacy system, the need and benefits of legacy system modernisation, various strategies and approaches employed, challenges encountered throughout the modernisation process and the risks and costs that shape modernisation endeavours. This systematic investigation of the topic of modernisation can highlight the various considerations for researchers and practitioners tasked with modernisation evaluations.

Keywords: System Modernisation, Software Engineering, Legacy System Modernisation, System Migration, System Transformation.

1 Introduction

The need for software system modernisation resonates across industries and challenges organisations to adapt in the ever-evolving digital landscape. Systems are perpetually susceptible to obsolescence, and the need for system modernisation increases as compatibility and security support decreases [1]. Through applying a multivocal literature review, this paper aims to delve into the numerous dimensions of system modernisation that organizations need to consider when undertaking such an endeavour. The following research questions are asked:

1. RQ1: Why is system modernisation necessary?
2. RQ2: What approaches are used in legacy system modernisation?
3. RQ3: What are the risks and costs of system modernisation?
4. RQ4: What are the challenges associated with system modernisation?

Section 2 of this paper presents the Multivocal Literature Review (MLR) methodology, followed by a detailed analysis in Section 3. Research limitations and future work are outlined in Section 4, with Section 5 presenting a conclusion.

2 Research Methodology

2.1 Methodology

This research paper was created as part of an MLR. We adopted an MLR approach for searching and reviewing both peer-reviewed (white) and some non-peer reviewed (grey) literature. For each of the four research questions we defined search strings using relevant keywords and made use of Google Scholar to retrieve and review literature on the subject of this paper.

2.2 Search Strings

Initial research focused on the topic using general search strings like “System Modernisation” and some synonyms such as “Architectural Transformation” and “System Migration”. This facilitated the acquisition of a general sense of the subject, later breaking it down into four research foci, following which an individual researcher examined a single sub-topic through the creation of more specific search strings. Searches included “Software System Modernisation”, “Costs of System Modernisation”, “Risks of System Migration”, “Cloud Migration Automation”, “Dangers of System Modernisation”, and “Challenges of System Modernisation”. Logical operators such as “AND” and “OR” were further utilized to bring increased focus to the searches, and to obtain more accurate results. Searches also evolved over time in different ways as the research progressed, for example the term “Software” was added to searches to focus the literature on software systems, we also added “Modernization” as a second spelling to include papers using the American spelling convention.

2.3 Inclusion/ Exclusion Criteria

When reviewing the results of searches, initial results were filtered using the following criteria:

- Only papers written in English.
- Published in the last five years (2019-2024)
- Present in the top twenty results of the search.

Each identified paper in the filtered search results was briefly reviewed to check if they were relevant to the given research question, and if they were, the research would be added them to the collection for in-depth review.

Not every paper that was relevant was included in this paper, some proved not relevant to the specific research question or did not provide much valuable information upon in-depth reading. In total, 76 papers were reviewed, with 39 of the most informative included in this paper. Table 1 presents the most prevalent sources for white literature.

Table 1. Top Sources of Research Documents.

Source	Number Reviewed	Number Cited
IEEE	22	9
Science Direct	12	9
ACM	8	4
Springer	6	5
ArXiv	4	3

3 Analysis

3.1 RQ1: Why is System Modernisation Necessary?

Software modernisation?

Software modernisation is the re-architecture of existing software from legacy infrastructure to modern architecture. and the process of applying advanced technologies and methodologies, including cloud, agile, and containerisation, to an organisation's traditional IT infrastructure, architecture, and products to maximise resiliency, efficiency, agility, and speed [2] . Modernisation is a technical process that moves the logic of existing applications into a modern, state-of-the-art environment, always with the goal of reducing costs [2]. Therefore, software modernisation is the process of evolving existing software systems by replacing, redeveloping, reusing, or migrating software components and platforms when traditional maintenance practices are no longer able to achieve expected results [2].

Software modernisation can cover various aspects, such as migrating to a new platform, updating programming languages, or improving the user interface. Software modernisation becomes even more important when dealing with legacy systems.

Legacy systems

A legacy system is a system that no longer fully functions as originally intended, mainly related to the performance of the system and associated adaptation inertia. Because technology becomes obsolete over time, even solutions developed in a short period of time can become legacy issues. When such a system cannot keep up with business goals, it needs to be maintained, thereby degrading the user experience, and making it difficult for users to benefit from the system due to its inability to satisfy their needs, which in some cases is related to increasing data volumes. When deficiencies in legacy systems hamper users' business performance, legacy software modernisation can make a huge difference [3]. Key areas where legacy systems can become disruptive are as follows:

- *Maintenance and support*: For systems that are launched within a short period of time, maintenance costs often exceed development costs [3].
- *Integration and Compliance*: Most regulations are updated annually, and failing to meet the requirements issued by regulators can cost you millions of dollars in penalties [3].
- *Security*: Outdated security protocols are a major concern for most businesses. For enterprises, real-time security updates are necessary. Outdated security protocols will cause data leaks and cause huge economic losses to the enterprise [4].
- *Lost Opportunities*: A lack of innovation can impact the long-term profitability of a software business, and it will be unable to adapt to changing market standards (customer needs, safety regulations, quality standards), resulting in lost business opportunities [4].
- *Lack of agility and efficiency*: Outdated software often fails to meet the needs of modern development and becomes inefficient. Modernisation and automation can improve the efficiency of most processes and significantly increase the efficiency growth rate [3].
- *Brand reputation*: Outdated software is often accompanied by suboptimal performance and potential crashes. Poorly performing software can also damage the software's brand reputation [4].

Modernisation is reported to deliver the following benefits:

Modernising legacy software can significantly improve performance, as outdated systems often suffer from slow response times, frequent crashes, and inefficiencies. By adopting modern technologies and frameworks, businesses can optimise their software to improve speed, reliability, and resource utilisation. This increases productivity, reduces downtime, and enhances user experience [2].

Scalability and flexibility: Legacy systems are often rigid and difficult to adapt to changing business needs. Software modernisation introduces scalability and flexibility by leveraging modern architectures and platforms. Cloud-native technologies, microservices, and containerisation enable enterprises to efficiently scale applications, respond quickly to market demands, and handle increased workloads. With modern software, organisations can easily add or remove features, integrate with third-party services, and expand their offerings.[5]

Cost optimisation: Legacy systems can become a financial burden due to high maintenance costs, licensing fees, and inefficient resource utilisation. Software modernisation helps organisations optimise costs by eliminating redundant processes, reducing infrastructure expenses, and streamlining operations. By leveraging cloud computing, businesses can move from capital expenditures to operational expenditures, paying only for the resources they use. Additionally, modern software requires less maintenance and support, freeing up resources for strategic initiatives [5].

Enhance security and compliance: Legacy software is often susceptible to security vulnerabilities and lacks the robust security features offered by modern systems. Software modernisation enhances security by implementing the latest security protocols, encryption technologies, and access controls. It ensures compliance with industry regulations and protects sensitive data from unauthorised access. Modern software also allows for timely security patches and updates, reducing the risk of security incidents and data breaches [6].

Competitive advantage: In today's rapidly evolving digital environment, maintaining a competitive advantage is critical. Software modernisation enables businesses to innovate, differentiate and gain a competitive advantage. By leveraging modern technology, organisations can deliver new features and functionality faster, improve customer experience and adapt to market trends. Modern software enables businesses to embrace digital transformation, unlock new opportunities and respond effectively to changing customer needs [7].

3.2 RQ2: What Approaches are used in Legacy System Modernisation?

Understanding the intricacies of legacy system modernisation is crucial for organisations seeking to optimise their IT infrastructure, improve operational efficiency, and remain competitive in rapidly evolving markets. The intricate nature of legacy systems offers a range of approaches, each possessing unique strengths, limitations, and applicability. From migration to re-implementation, organisations must choose the most suitable approach to achieve their modernisation objectives effectively.

Legacy System Modernisation?

Various definitions for a legacy system have been given by different authors and organisations. Common terms used to describe a legacy system are *old*, *critical* and

outdated as illustrated by [2, Sec. 3A]. A legacy system is not necessarily "bad", they are high dependency systems that contain important business logic and data collected over several years [2]. [8, Ch. 1 "].

Modernisation of legacy systems is not a simple process; the systems consist of interrelated components deeply ingrained within the organisation's culture and environment. Therefore, the system modernisation process must intricately consider these organisational culture and environment factors to provide unified and systematic guidance to the organisation [2, Sec. 4], [9] . Legacy systems are not all similar. Some are tightly coupled, highly monolithic "*monstrosities*" [10, Sec. 1]. Others use layered architecture which can be updated using incremental implementations. There are several approaches an organisation can take to system modernisation. Careful planning and implementation are necessary.

The complexity and diversity of legacy systems means that researchers struggle to find industry-wide applicable methodologies and solutions in an academic setting. The sociotechnical aspect of legacy system modernisation cannot be easily simulated for experimentation. Existing legacy systems have been in place for decades, undergoing relatively little technological evolution. Naturally, there were fundamental changes like replacing data storage mechanisms by relational databases or text based terminal interfaces with user interfaces, but at the core business, functionality was provided in similar ways over several years. In today's technological landscape, life expectancy of a software system has shortened significantly due to the speed of technological evolution [10, Sec. 1].

According to M. Bellotti, "*each legacy modernisation project starts off feeling easy. After all, a working system did exist at one point… All the modernising team should need to do is simply repeat that process using better technology, the benefit of hindsight, and improved tooling. It should be easy*". In reality, modernisation projects will take months, if not years to complete and maintaining motivation is a difficult challenge [11, Ch. 4]. Legacy systems are often too complex to easily replace and at the same time too valuable to the business to be removed [10, Sec. 1], [9], [12].

Latest studies on legacy system modernisation tend to focus on migration and integration to the modern environment such as Cloud, SOA and Big Data. Researchers prefer to focus on the requirement phase of modernisation. Few papers tackle the management and strategic aspects of modernisation. The study by [2] examined several of these papers and concludes that there are six phases of legacy system modernisation. These are: "*planning*, *old and new requirements determination*, *design and development*, *testing and system implementation"*. The planning phase is less detailed while the old requirements determination is often emphasised because of the business value of legacy systems. Reverse engineering is highlighted as an important technique during the old requirements determination phase.

In summary, legacy system modernisation is a complex task with multiple phases. Modernisation projects are very time-consuming, they can take years to accomplish. It is challenging to replace a legacy system entirely, due to the importance of the system to the business. Several authors re-iterated the lack of emphasis on the management and strategic aspects of modernisation projects, and the fact that researchers face challenges in developing universally applicable methodologies due to system diversity and sociotechnical nuances [2], [11]. Current studies predominantly focus on migration and integration into modern environments. It is important for businesses to understand the significance of addressing both technical and organisational challenges to successfully navigate the modernisation process. Regrettably, many businesses only consider legacy system modernisation when it becomes imperative rather than proactively [11, Ch. 10]. Nonetheless, various approaches to modernisation exist that may be considered once the process begins.

Approaches

Unfortunately, one of the most common approaches to modernising a legacy system is to just ignore it. This is tempting in situations where there is a lack of documentation or system knowledge is lost. However, the risks of ignoring the system are potentially huge [8, Ch. 1].

The naïve approach to system modernisation is to redevelop the system entirely using the latest technology. System redevelopment is not always possible. It could be that the costs of redevelopment are too high, the time required is not acceptable or that organisations lack the experience needed to build large new business applications. Instead, they are often more adept at keeping their legacy assets alive. A cost effective solution might involve the utilisation of sophisticated tool support. However, this remains an area where greater tool capability is desirable.

Re-implementation

Re-implementation is when the original code is re-written (in another language or another platform) based on re-documentation of that code [12, 13]. Re-implementation is not an automated conversion, it requires human judgement. Developers must be familiar with both the old and new languages, understand the existing algorithms and be able to recreate them. The gap between the old and new language can be substantial, constructs that exist only in the old language may need to be simulated in the new language. In some cases the same constructs may have different semantics [13, Sec. 3]. System re-implementation involves several steps: measuring the size and complexity of the existing system. Then analysing, visualising, and documenting existing code, renaming data and procedures, recovering business logic, isolating the code to be re-implemented. And finally: re-writing code, testing code and integration [13, Sec. 5].

Migration

Migration is a system modernisation technique where the legacy system is moved to a more flexible platform without disturbing the original system’s data and rules [2, Sec.

3]. Migration moves the components of the system to a new technology rather than later versions of the current technology. This can be applied to any part of the system. There are many reasons why an organisation may want to migrate their legacy system. A new technology may have emerged, another vendor or product may offer better technical solutions, or the original vendor may have ceased trading entirely. Migration can result in significant cost savings. Only a fraction of migration projects are planned and staffed initially as migration projects. Typically, this happens in scenarios where a "*strict one-by-one replacement*" is required. However, in the context of large-scale enterprise modernisation projects, migration frequently happens as an afterthought. This negatively affects both the approaches taken in these scenarios as well as resulting solutions [10, Sec. 4.3.4].

One case study proposes a Topology for legacy system migration using reasoned generalisations [10]. Fig.1 shows groupings of those generalisations showing the relationships between them.

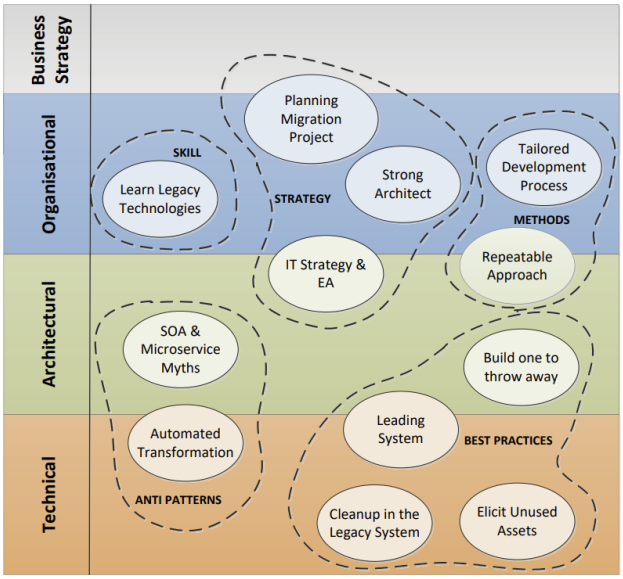

Figure 1: Legacy System Migration Topology: Grouping the generalisations into related topics to highlight relationships and create a thematic map [10]**.**

Migrations may be driven by technical or business requirements. There can be interpersonal conflicts that endanger migration projects. Problems may arise when trying to migrate data that has incompatible formats or required data is missing. Migration teams often need to enrich (make up) the missing, required data to get the migration process to work. Migration teams must also remember to update any documentation that is made irrelevant by the migration [2, Ch. 1]. The lack of an adequately skilled workforce to maintain legacy systems is one of the most commonly named reasons for legacy system migration. [10, Sec. 5].

Emerging cloud technology has gained a lot of attention in recent years. Cloud migration is a process employed which involves transferring data from legacy database systems to typically more cost-effective and efficient cloud-based databases without necessitating a complete overhaul of the existing infrastructure. A Cloud migration often has concept variants and multiple methods of instantiation. "*This process itself* (cloud
migration) *is contingent on existing organisational structures and characteristics of a legacy system*" [14, Sec. 2.1].

Automation in other approaches to modernisation may be difficult due to the diversity of the legacy system [13]. Automated cloud migration tools (ACMTs) have been developed and are evolving to streamline the migration process. Their benefits are significant in terms of cost, time, and business innovation [15]. Migration is also commonly used to transition from monolithic architectures to Service-Oriented Architectures (SOAs) or microservices architectures (MSAs) [12], [16]. Re-engineering is an important step here, as it can reduce complexity, lower coupling, higher cohesion and simplify integration [16, Sec. 8].

There has been a surge in research aimed at generalising legacy system modernisation strategies and automating the modernisation process [2], [5], [10], [11], [12], [13], [16]. When considering using automation, the question is "*if the automation fails, will it be clear what has gone wrong*?" [11]. Automation, while beneficial in streamlining processes, raises concerns about engineers potentially losing familiarity with the inner workings of the legacy system.

3.3 RQ3: What are the Risks and Costs of System Modernisation?

Modernisation of software systems is an inherently risky business, the pressure to efficiently modernise systems to meet new requirements and mitigate growing risks is always building [10]. Systems that are the target of modernisation efforts have grown and evolved over years and even decades into millions of lines of code, they suffer from issues of maintainability, scalability and supportability, and they struggle to adapt to the latest developments in hardware and software [17]. Although recognising that these systems are problematic, companies nevertheless have continued with them as they are too complex to easily replace and at the same time much too valuable to be removed [10]. Then migration process is known to be complex and risky, and is it not unusual for these projects to fail as a result [17].

Modernisation happens at a point in a system's lifespan where risk is unavoidable, leaving a system as-is presents several risks while embarking on a modernisation effort also presents high levels of risk. Legacy systems are known to significantly increase security risks at a company because they are not designed to address contemporary cyber security risks [18], this was demonstrated in the case of the US federal government by a study examining the argument that systems are "secure-by-antiquity",

which believed that older systems with little documentation are more difficult to attack. However, research has found that antiquity does not make legacy systems more secure, but in fact significantly increases the frequency of security incidents [18]. In a separate study focused on legacy software in medical devices, it was found that devices often lacked basic security features and run legacy operating systems with publicly known vulnerabilities [19]. Breaches and security incidents in systems of this nature can cost the industry billions, endanger patient privacy, and enable large scale identity theft [19]. This example illustrates just how risky and costly security breaches in legacy systems can be if they are not prevented.

Owing to years of usage, development, and improvement, a legacy application's size and complexity increases, making the disadvantages of legacy systems outweigh their advantages [20]. Legacy systems can accumulate a lot of problems as they age, they are commonly running on obsolete hardware, use old programming languages and are subject to limited support and irreplaceable parts [20]. When the disadvantages and costs stack up, that is when firms look towards modernising.

Some research suggests that most migration projects fail because of poor planning and risk analysis [21], [22]. The most important factor to success is to carefully plan large migration projects and consider all the options presented before rushing into the project [21]. Modernisation projects are often critical projects vital to a company's competitiveness and success, they should be treated accordingly and systematically evaluated to identify trade-offs and risks [23]. A lack of understanding and preparedness from companies underpin a lot of the failures seen in these types of projects [22]. For example, one study [21] reported that generating new code accounts for the least cost in software migration. Testing was by far the highest cost driver of the project being studied, accounting for over 70% of project time [21], this is just one of many common misconceptions and pitfalls businesses face. Another common trap is being lured into rushed migrations by "fashionable architectural decisions" [23] and the hype of new technology [22]. Committing to a costly modern solution should be avoided by carefully considering all alternatives before rushing into them [21], particularly analysing the reasons for modernisation in the first place.

Alongside the analysis of modernisation opportunities and risk factors, projects are often bound by cost [21], companies need to establish what options are available to them within budget. However, this is not an easy task, modernisation is not a set-in-stone process with clear guidelines and costs, many papers acknowledge the facts that there are very few pieces of research that consider the cost of modernisation [21], [22]. There may also be many more cost drivers than first appear when planning a project, such as the pricing of servers and cloud services being complex, with varying pricing models and discounts across different regions [22], however they can be estimated. On the other hand, one of the largest cost drivers is the cost of acquiring the expertise and knowledge required to complete a large modernisation project [24], estimates on acquiring the skills and knowledge is mostly explorative, and a lot of it comes from experience with other cases [22], but each modernisation project is highly unique. The

true total cost can only accurately be known after the migration is complete [22], so careful and cautious analysis must be completed to account for setbacks, errors and additional costs that could not be accounted for before a project is started.

Modernisation projects are inherently risky, often performed on critical systems that suffer from poor security, maintainability, and adaptability. There are often just as many risks associated with leaving a system as-is as there are with modernising it. To increase the chances of project success, companies need to carefully consider all the options before them, and evaluate all potential risks and costs involved. Each project is highly unique and there is no universal guide to modernisation, but considering the major risk sources and costs is crucial to planning a successful project.

3.4 RQ4: What are the Challenges Associated with System Modernisation?

Modernising a system poses a formidable challenge. Organisations can face numerous hurdles in their decision to transition the legacy system and this may deter them from going through with their project. The cost of modernising may be too great, and those who weren't discouraged in some cases may find their migration process has not achieved the organisation's goals [24, p. 3].

In the context of cloud migration, organisations must decide on an appropriate migration strategy. However, organisations often fall short in fully identifying their circumstance to successfully execute the complex changes necessary for their legacy systems to take advantage of cloud services [24, p. 1]. Some may decide to go through with cloud migration because there is genuine technical and financial incentive to do so, whilst others may simply go through with it as it is a "fashionable" technical trend. Either way, it is crucial to realise the impact and scope of changes cloud migration brings. Prior to the COVID-19 pandemic, there was a lack of academic literature exploring what a generic and application agnostic cloud transition may look like [24, p. 1]. Frameworks have been proposed, but complete solutions are missing [25, p. 3]. With the recent increase in cloud adoption in the post-pandemic world [26] [27], there has been an increased effort in academia to provide methodologies on how to re-engineer legacy systems to be cloud-enabled [28] [29] [14] [30]. However, due to a lack of standardisation and consistency it can be difficult to choose the right strategy. Sufficient time has to be allocated for the detailed planning and design of a legacy system transition to avoid failure [31, p. 1].

Not only must a technical strategy be determined, but the business model and processes must be improved and aligned with the modernisation efforts if necessary. "*... [T]he software implemented to replace legacy systems is developed or configured to largely mimic their features and functionality in order to minimise the disruption to organisational operations that accompanies the introduction of new technology*", [32, p. 1]. However, doing so can lead to the carry-over of legacy features and the business processes that accommodate them. This is especially true for public agencies who start

a legacy replacement project [33, p. 1]. Due to the negligence of systematic adjustments and bureaucratised decision-making processes, this oversight leads to the "legacy problem" [33] and fails to take advantage of new technologies the legacy system is being transitioned to. It is therefore crucial that organisations identify if there may be a need to not only modernise their system technically, but the business model and culture as well.

Once the planning is complete, the subsequent hurdle lies in adhering to its implementation in the time allocated. Organisations often underestimate the time it can take for such projects to be completed [34, p. 2]. Delays can occur in the most unexpected of places, and organisations should be considerate of this to avoid halting their project. Compatibility issues between the current infrastructure and the cloud platforms may surface, and unforeseen dependencies between other systems can result in cascading issues. Extensive adjustments in order to resolve incompatibilities and ensure a smooth transition may be required and this may delay the modernisation project.

Other challenges include system requirement changes that introduce scope creep. Although it can be advantageous to use the modern services that cloud providers offer, it is worth considering if it is actually required, as sometimes this may cause technical debt due to too excess different technologies utilised in the communication between services [35]. Additional integrations and customisations can expand the original scope of the project, and this can increase the cost of the project as well as delay the project delivery. It is important to have a clear, defined scope in the beginning of the project to better identify if requirements are starting to expand. If constant changes are necessitated, it can be worthwhile adopting an agile methodology to help with the iterative development [34, p. 4]. If requirements expand or resource allocation wasn't performed adequately, resources can become constrained. Personnel and expertise limitations can arise due to the scarcity of people who know both the legacy system as well as the modern technology it is being transitioned to. Upskilling and training the workforce can be resource intensive, and potential skill gaps within the team can occur. On top of the day-to-day operational responsibilities, motivation challenges may arise as staff encounter the difficult task of adapting to new, unfamiliar technologies and overcoming learning curves. Knowledge transfer and documentation is important to reduce the effect of information loss, and as the number and variety of tools and technologies increase, weak knowledge management and lack of standardisation may emerge [35].

System modernisation can bring about security and regulatory concerns as well as uncertainty. As organisations go through with their cloud migration projects, they may have to grapple with compliance hazards and challenges regarding data protection and privacy [36]. Therefore, it becomes imperative to conduct a thorough risk analysis before migrating to the cloud. Legalities and regulations can be difficult to uphold if the cloud service provider is not transparent with their services. Due to the "*absence of visibility into the security of the platform"* [37, p. 2], some organisations are hesitant to

adopt cloud services. Ambiguities and misunderstandings can arise if there are no clearly laid out service level agreements regarding expectation management [38, p. 2]. Contractual obligations that were set out when the legacy system was created should also be reviewed to see if they may be affected and whether any agreements may be unintentionally broken. For example, licensing issues of legacy systems can surface if the transitioned system scales automatically which creates multiple instances of that system [14, p. 14].

For some stakeholders, the physical presence of servers and hardware instils a sense of control and security. Edward Snowden's leaks about the NSA's espionage programs creates doubt and hesitancy about storing critical, confidential information in the cloud [38, p. 5]. Additionally, organisations may be reluctant to rely on third-party entities for critical aspects of their operations in fear of outages and may feel vulnerable to attack since big data centres can be targets of hackers [38, p. 5]. Due to these difficulties, "*some high-privacy and safety critical systems such as military, aviation, and aerospace systems might not be able to directly take advantage of cloud services"* [24, p. 4]. It may be worthwhile for organisations to consider investing partially rather than fully into cloud services, creating a hybrid on-premises and cloud infrastructure.

Despite the plethora of challenges and setbacks associated with system modernisation, the potential benefits can significantly outweigh the risks. It can open the opportunity for enhanced efficiency and scalability, and benefits include streamlined processes, cost savings, and increased competitiveness. The long-term advantages of a modernised system can position organisations for prolonged success in a world where change is rapid. It is through thoughtful, considerate planning and design and a strategic mindset that organisations can navigate the challenges and ultimately reap the rewards of system modernisation.

4 Research Limitations and Future Work

Although this work was strictly time-limited, the methods used were considered academically rigorous. The search strings used are clearly identified. In addition, the main body of the paper is systematically sorted based on its citations and the logical correlation of each part of the paper. Although a high citation count does not necessarily guarantee better publication quality or greater relevance, a powerful paper selection mechanism emerges when combined with the rationale derived from the judgment of the research team.

As software modernisation continues to advance at a rapid pace, increasing business efficiency, user experience, and scalability, the importance of software modernisation services becomes even more apparent as organisations strive to stay ahead of the curve in a rapidly evolving digital environment. Adopting cutting-edge technologies such as AI-driven modernisation, microservices architecture, low-code/no-code platforms, DevOps practices, cloud-native paradigms, and innovative legacy modernisation

technologies enable enterprises to unlock increased efficiency, scalability, and agility. In the future we hope to visit current and emerging trends and further delve into the analysis of how they impact the future of software modernisation.

5 Conclusion

Taking full advantage of emerging infrastructure innovations such as serverless computing [39] (for example Function-as-a-Service [40]) can assist firms not only in devolving hardware concerns to reliable third-party providers, but also in incorporating technology to increase their speed of delivery of new software into existing systems [41, 42]. Unfortunately, however, system modernisation is a complex technical undertaking, especially where knowledge loss affects a product or team [43]. This study is therefore focused on examining the currently reported approaches, challenges and risks associated with modernization efforts through investigation of four research questions. In RQ1, we examined the necessity of system modernisation, finding that not modernising may severely affect your budget, security, and reputation and that there are many benefits to system modernisation. Through RQ2, we discussed the complexity of legacy systems and researched the various approaches to modernisation. We analysed their applicability and limitations, focusing on migration due to recent technological advancements. RQ3 addressed the risks and cost of modernisation, finding that a key factor to success is to rigorously analyse and plan any modernisation effort, it also found that few research papers focus on how to estimate the true cost of modernisation which makes accounting for budgets severely challenging. Finally, RQ4 researched the challenges in the modernisation process, observing challenges such as data protection during cloud migration and “scope creep”.

System modernisation is a critical process in software engineering. Its benefits to the system’s performance, scalability, cost, security, and competitive advantage outweigh the inherent risks and challenges faced during the modernisation process. Furthermore, the benefits of system modernisation can strategically position organisations for sustained success amidst continual technological evolution.

Our research underlines that planning and organisation are fundamental to successful modernisation. Modernisation projects are complex and often require varied solutions. Organisations often overlook the importance of planning, resulting in pitfalls, poor management, and diminished motivation. Some challenges encountered are not solely technical but stem from organisational conflicts. This dimension poses a significant obstacle to researchers, as it is difficult to address the sociotechnical aspect of system modernisation. However, through careful planning and consideration organisations can enjoy the fruits of modernisation.

Acknowledgements. This research is supported in part by SFI, Science Foundation Ireland (https://www.sfi.ie/) grant No SFI 13/RC/2094_P2 to Lero - the Science Foundation Ireland Research Centre for Software. It is also supported in part by SFI,

Science Foundation Ireland (https://www.sfi.ie/) grant No SFI 12/RC/2289_P2 to Insight - the Science Foundation Ireland Research Centre for Data Analytics.

References

[1] K. Hrisafov and N. Chivarov, 'Implementing Industry 4.0 solution with legacy informational systems', *Ind. 40*, vol. 5, no. 5, pp. 235–238, 2020.

[2] H. K. A. Bakar, R. Razali, and D. I. Jambari, 'Implementation Phases in Modernisation of Legacy Systems', in *2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS)*, Dec. 2019, pp. 1–6. doi: 10.1109/ICRIIS48246.2019.9073628.

[3] H. K. A. Bakar, R. Razali, and D. I. Jambari, 'A Guidance to Legacy Systems Modernization', *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 10, no. 3, Art. no. 3, Jun. 2020, doi: 10.18517/ijaseit.10.3.10265.

[4] P. L. Leon and F. E. A. Horita, 'On the modernization of systems for supporting digital transformation: A research agenda', in *Proceedings of the XVII Brazilian Symposium on Information Systems*, in SBSI '21. New York, NY, USA: Association for Computing Machinery, Jul. 2021, pp. 1–8. doi: 10.1145/3466933.3466976.

[5] M. Abdellatif *et al.*, 'A taxonomy of service identification approaches for legacy software systems modernization', *J. Syst. Softw.*, vol. 173, p. 110868, Mar. 2021, doi: 10.1016/j.jss.2020.110868.

[6] M. Khan *et al.*, 'Modernization Framework to Enhance the Security of Legacy Information Systems', *Intell. Autom. Soft Comput.*, vol. 32, no. 1, pp. 543–555, 2022, doi: 10.32604/iasc.2022.016120.

[7] Catherine, J. D. Trisaktyo, T. Ranas, M. Rasyiid, and M. R. Shihab, 'Embracing Agile Development Principles in an Organization using The Legacy System: The Case of Bank XYZ in Indonesia', in *2020 6th International Conference on Computing Engineering and Design (ICCED)*, Oct. 2020, pp. 1–5. doi: 10.1109/ICCED51276.2020.9415831.

[8] *Working with Legacy Systems*. Accessed: Feb. 02, 2024. [Online]. Available: https://learning.oreilly.com/library/view/working-with-legacy/9781838982560/

[9] H. Abu Bakar, R. Razali, and D. I. Jambari, 'A Qualitative Study of Legacy Systems Modernisation for Citizen-Centric Digital Government', *Sustain. Basel Switz.*, vol. 14, no. 17, pp. 10951-, 2022, doi: 10.3390/su141710951.

[10] S. Strobl, M. Bernhart, and T. Grechenig, 'Towards a Topology for Legacy System Migration', in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, Seoul Republic of Korea: ACM, Jun. 2020, pp. 586–594. doi: 10.1145/3387940.3391476.

[11] *Kill It with Fire*. Accessed: Feb. 02, 2024. [Online]. Available: https://learning.oreilly.com/library/view/kill-it-with/9781098128883/

[12] T. R. Vinay and A. A. Chikkamannur, 'A Novel Methodology to Restructure Legacy Application onto Micro-Service-Based Architecture System', in *Emerging Research in Computing, Information, Communication and Applications*, N. R. Shetty, L. M. Patnaik, H. C. Nagaraj, P. N. Hamsavath, and N. Nalini, Eds., in Lecture Notes in Electrical Engineering. Singapore: Springer, 2022, pp. 509–519. doi: 10.1007/978-981-16-1342-5_39.

[13] H. Sneed and C. Verhoef, 'Re-implementing a legacy system', *J. Syst. Softw.*, vol. 155, pp. 162–184, Sep. 2019, doi: 10.1016/j.jss.2019.05.012.

[14] M. Fahmideh, F. Daneshgar, F. Rabhi, and G. Beydoun, 'A generic cloud migration process model', *Eur. J. Inf. Syst.*, vol. 28, no. 3, pp. 233–255, May 2019, doi: 10.1080/0960085X.2018.1524417.

[15] 'Best Leadership Practices of Multinational Corporations in the use of Automated Migration Tools in Adoption of Commercial Cloud Computing Platforms: A Meta-Analysis - ProQuest'. Accessed: Feb. 16, 2024. [Online]. Available: https://www.proquest.com/openview/1b08e1a13c87f43a1d83783e5a2e858d/1?pq-origsite=gscholar&cbl=18750&diss=y

[16] M. Mazzara, N. Dragoni, A. Bucchiarone, A. Giaretta, S. T. Larsen, and S. Dustdar, 'Microservices: Migration of a Mission Critical System', *IEEE Trans. Serv. Comput.*, vol. 14, no. 5, pp. 1464–1477, Sep. 2021, doi: 10.1109/TSC.2018.2889087.

[17] H. Rambarassah and S. Khaddaj, 'The complexity attachment in modernization journey', in *2022 21st International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*, Oct. 2022, pp. 119–122. doi: 10.1109/DCABES57229.2022.00078.

[18] M.-S. Pang and H. Tanriverdi, 'Strategic roles of IT modernization and cloud migration in reducing cybersecurity risks of organizations: The case of U.S. federal government', *J. Strateg. Inf. Syst.*, vol. 31, no. 1, p. 101707, Mar. 2022, doi: 10.1016/j.jsis.2022.101707.

[19] T. Tervoort, M. T. De Oliveira, W. Pieters, P. Van Gelder, S. D. Olabarriaga, and H. Marquering, 'Solutions for Mitigating Cybersecurity Risks Caused by Legacy Software in Medical Devices: A Scoping Review', *IEEE Access*, vol. 8, pp. 84352–84361, 2020, doi: 10.1109/ACCESS.2020.2984376.

[20] M. H. Hasan, M. H. Osman, N. I. Admodisastro, and M. S. Muhammad, 'Legacy systems to cloud migration: A review from the architectural perspective', *J. Syst. Softw.*, vol. 202, p. 111702, Aug. 2023, doi: 10.1016/j.jss.2023.111702.

[21] H. M. Sneed and C. Verhoef, 'Cost-driven software migration: An experience report', *J. Softw. Evol. Process*, vol. 32, no. 7, p. e2236, 2020, doi: 10.1002/smr.2236.

[22] T. Talvitie, 'Estimating the migration cost to modern cloud: An exploratory case study', 2020, Accessed: Feb. 05, 2024. [Online]. Available: https://trepo.tuni.fi/handle/10024/119644

[23] P. Cruz, H. Astudillo, R. Hilliard, and M. Collado, 'Assessing Migration of a 20-Year-Old System to a Micro-Service Platform Using ATAM', in *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*, Mar. 2019, pp. 174–181. doi: 10.1109/ICSA-C.2019.00039.

[24] M. Fahmideh, F. Daneshgar, G. Beydoun, and F. Rabhi, 'Challenges in migrating legacy software systems to the cloud an empirical study'. arXiv, Apr. 16, 2020. doi: 10.48550/arXiv.2004.10724.

[25] M. Khan *et al.*, 'CMMI Compliant Modernization Framework to Transform Legacy Systems', *Intell. Autom. Soft Comput.*, vol. 27, no. 2, pp. 311–331, 2021, doi: 10.32604/iasc.2021.014280.

[26] M. Gokarna, 'Reasons behind growing adoption of Cloud after Covid-19 Pandemic and Challenges ahead'. arXiv, Feb. 27, 2021. doi: 10.48550/arXiv.2103.00176.

[27] Z. R. Alashhab, M. Anbar, M. M. Singh, Y.-B. Leau, Z. A. Al-Sai, and S. Abu Alhayja'a, 'Impact of coronavirus pandemic crisis on technologies and cloud computing

applications', *J. Electron. Sci. Technol.*, vol. 19, no. 1, p. 100059, Mar. 2021, doi: 10.1016/j.jnlest.2020.100059.

[28] M. Fahmideh, G. Low, G. Beydoun, and F. Daneshgar, 'Cloud Migration Process A Survey Evaluation Framework and Open Challenges'. arXiv, Apr. 16, 2020. doi: 10.48550/arXiv.2004.10725.

[29] M. Fahmideh, J. Grundy, G. Beydoun, D. Zowghi, W. Susilo, and D. Mougouei, 'A model-driven approach to reengineering processes in cloud computing', *Inf. Softw. Technol.*, vol. 144, p. 106795, Apr. 2022, doi: 10.1016/j.infsof.2021.106795.

[30] M. Fahmideh, G. Low, and G. Beydoun, 'Conceptualising Cloud Migration Lifecycle'. arXiv, Sep. 03, 2021. doi: 10.48550/arXiv.2109.01757.

[31] K. Ramchand, M. Baruwal Chhetri, and R. Kowalczyk, 'Enterprise adoption of cloud computing with application portfolio profiling and application portfolio assessment', *J. Cloud Comput.*, vol. 10, no. 1, p. 1, Jan. 2021, doi: 10.1186/s13677-020-00210-w.

[32] A. Alexandrova and L. Rapanotti, 'Requirements analysis gamification in legacy system replacement projects', *Requir. Eng.*, vol. 25, no. 2, pp. 131–151, Jun. 2020, doi: 10.1007/s00766-019-00311-2.

[33] A. Alexandrova, L. Rapanotti, and I. Horrocks, 'The legacy problem in government agencies: an exploratory study', in *Proceedings of the 16th Annual International Conference on Digital Government Research*, Phoenix Arizona: ACM, May 2015, pp. 150–159. doi: 10.1145/2757401.2757406.

[34] C. S. Ranganathan and R. Sampathrajan, 'Cloud Migration Meets Targeted Deadlines', in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Jul. 2023, pp. 672–676. doi: 10.1109/ICESC57686.2023.10193104.

[35] S. Soares de Toledo, A. Martini, A. Przybyszewska, and D. I. K. Sjøberg, 'Architectural Technical Debt in Microservices: A Case Study in a Large Company', in *2019 IEEE/ACM International Conference on Technical Debt (TechDebt)*, May 2019, pp. 78–87. doi: 10.1109/TechDebt.2019.00026.

[36] Maniah, B. Soewito, F. Lumban Gaol, and E. Abdurachman, 'A systematic literature Review: Risk analysis in cloud migration', *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 6, Part B, pp. 3111–3120, Jun. 2022, doi: 10.1016/j.jksuci.2021.01.008.

[37] R. Amin, S. Vadlamudi, and M. M. Rahaman, 'Opportunities and Challenges of Data Migration in Cloud', *Eng. Int.*, vol. 9, no. 1, Art. no. 1, Apr. 2021, doi: 10.18034/ei.v9i1.529.

[38] A. Iqbal and R. Colomo-Palacios, 'Key Opportunities and Challenges of Data Migration in Cloud: Results from a Multivocal Literature Review', *Procedia Comput. Sci.*, vol. 164, pp. 48–55, Jan. 2019, doi: 10.1016/j.procs.2019.12.153.

[39] McAleese, S., Conway-McLoughlin, J., Detyna, F., Murashev, A., Yilmaz, M., Clarke, P.M.: Serverless Software Engineering - And How To Get There. In: Proceedings of Systems, Software and Services Process Improvement, EuroSPI 2022, 30 August 2022 - 2 September 2022, Salzburg, Austria. Communications in Computer and Information Science (CCIS), vol 1646. Springer, Cham. pp. 75–90. Available: https://doi.org/10.1007/978-3-031-15559-8_6

[40] Grogan, J., Mulready, C., McDermott, J., Urbanavicius, M., Yilmaz, M., Abgaz, Y., McCarren, A., MacMahon, S.T., Garousi, V., Elger, P., Clarke, P.M.: A Multivocal Literature Review of Function-as-a-Service (FaaS) Infrastructures & Implications for Software Developers. In: Proceedings of the 27th European and Asian Conference on Systems, Software and Services

Process Improvement (EuroSPI 2020), Springer CCIS Vol. 1251, 9-11 September 2020, Dusseldorf, Germany. https://doi.org/10.1007/978-3-030-56441-4_5

[41] Lapuz, N., Clarke, P., Abgaz, Y.: Digital Transformation and the Role of Dynamic Tooling in Extracting Microservices from Existing Software Systems. In Proceedings of the 28th European and Asian Conference on Systems, Software and Services Process Improvement (EuroSPI 2021), Springer CCIS Vol. 1442, pp. 301-305, 1-3 September 2021, Krems, Austria.

[42] Clarke, P., Elger, P., O'Connor, R.V.: Technology-Enabled Continuous Software Development. In: Proceedings of the International Conference on Software Engineering (ICSE) Workshop on Continuous Software Evolution and Delivery (CSED) (2016)

[43] Rashid, M., Clarke, P., O'Connor, R.V. A Systematic Examination of Knowledge Loss in Open Source Software Projects. International Journal of Information Management (IJIM), Volume 46, June 2019, pp.104-123. Y. Abgaz, McCarren, A., Elger, P., Solan, D., Lapuz, N., Bivol, M., Jackson, G., Yilmaz, M., Buckley, J., and Clarke, P., "Decomposition of Monolith Applications Into Microservices Architectures: A Systematic Review," in IEEE Transactions on Software Engineering, vol. 49, no. 8, pp. 4213-4242, Aug. 2023, doi: 10.1109/TSE.2023.3287297.