EUROPEAN INSTITUTE OF INNOVATION, ENTREPRENEURSHIP AND TECHNOLOGY

DEPARTMENT OF BUSINESS ADMINISTRATION

LEGACY MODERNISATION, CLOUD ARCHITECTURE, AND ENTERPRISE AGILITY: STRATEGIC AND OPERATIONAL APPROACHES TO DIGITAL TRANSFORMATION

Digital Transformation and Innovation

Y25-27

submitted by

Hubert Nahimana

Matriculation ID: HMI26011

EIIET, 17th October, 2026

Lecture: Eng. Dr Balachanda Pattanaik

Table of Contents

1. Abstract……..1
2. Introduction……2
3. Literature Review……..3
    3.1 Digital Transformation and Enterprise Agility……..3
    3.2 Structural Barriers to Legacy Modernisation…….4
    3.3 Monolithic and Cloud-Native Architectures……..4
    3.4 Governance and Organisational Change…….....5
4. Methodology…….....5
5. Findings…….....6
    5.1 Challenges of Legacy Enterprise Systems…….6
    5.2 Cloud Architecture and Modernisation Strategies…….7
    5.3 Successful Transformation: Netflix…….....7
    5.4 Migration Failure: TSB Bank……....8
    5.5 Governance and Change Management Practices……...…9
6. Discussion……..9
    6.1 Strategic Recommendations…….…10
    6.2 Proposed Implementation Roadmap……....11
7. Conclusion……11
8. References……12
9. Appendices……...13

1. Abstract

In today's digital age, companies want to improve how they work, come up with new ideas, and stay ahead of the competition. But many older businesses are held back by outdated computer systems, old technology problems, and strict organizational rules. These things make it hard for them to keep up with what customers want, new technology, and competition. This report looks at the big challenges that come with updating old systems and how new technologies like cloud computing, microservices, APIs, and serverless technologies can help businesses be more flexible and adapt quickly. Companies can benefit from moving to the cloud, but only if they do it right. Netflix is a great example of a company that did it slowly and carefully, changing its architecture and engineering practices over time. On the other hand, TSB Bank's attempt to move its IT systems in 2018 was a disaster, mainly because they didn't test enough and didn't manage the change well. This shows that just using new technology isn't enough - companies need to have good governance, be able to manage risks, and be willing to change their culture. To get it right, companies should take a step-by-step approach, looking at each application and changing the architecture a little at a time. They should also use APIs to integrate systems, adopt DevOps practices, and make sure executives are accountable for the results. The goal is to make the company more agile, but that doesn't happen just because you're using the cloud - it takes work to align technology, people, and strategy.

Keywords: Digital transformation, legacy modernisation, cloud computing, microservices, enterprise agility, technical debt, governance, organisational change.

2. Introduction

Digital transformation is about using technology to change how businesses work and create value. It's not just about investing in new IT systems, but also about changing the way companies operate, make decisions, and interact with customers. This process can lead to significant changes in how businesses run and create value. In other words, digital transformation is a strategic change that affects the entire organization, not just a technical update. It requires businesses to rethink their models, structures, and relationships with customers. By doing so, companies can stay competitive and create new opportunities for growth. As one expert notes, digital transformation is a process that uses technology to trigger major changes in how businesses work and create value. This means that modernizing a business is not just about adopting new technology, but about making fundamental changes to how the organization operates. (Fowler & Lewis, 2024)

Many companies rely on old systems that have been around for a long time. These systems are used for important things like handling money, managing supplies, keeping track of customers, processing insurance, and running factories. While these systems still work, they can be costly to maintain and hard to connect with newer digital tools. This is because they have old code, unclear dependencies, and scattered databases, which slows down the company's ability to launch new products and keep up with changes in the market. As a result, companies struggle to innovate and respond quickly to new opportunities. They have to deal with technical debt, which is like having old loans that need to be paid off, and outdated programming languages that few people understand. All these issues make it tough for companies to move forward and stay competitive. (Lenarduzzi et al., 2020)

Cloud computing and new cloud-based systems can help solve some problems by giving us more flexible and scalable infrastructure, automating deployment, and making it easier to integrate different applications. But, moving to these new systems also comes with its own set of risks, like interrupting services, inconsistent data, security issues, rising costs, and making operations more complicated. For example, TSB Bank's experience shows that even with a lot of preparation, a big technology change can still cause problems for customers. On the other hand, Netflix is a good example of how making gradual changes to the architecture and organization can help support big

digital operations. This can be seen in how Netflix has been able to handle large-scale digital operations over time, and how TSB Bank's migration issues can be learned from to avoid similar problems in the future. By looking at these examples, we can better understand the potential benefits and risks of moving to cloud computing and cloud-native architectures. (Netflix, 2016), (TSB Bank, 2019)

The central argument of this report is that successful legacy modernisation requires the coordinated transformation of enterprise architecture, operating models, governance, and organisational culture. Moving existing applications to the cloud without addressing their underlying limitations may simply reproduce legacy problems in a new environment.

The study looks at four main questions. First, what gets in the way of old systems being able to adapt to changing business needs? Then, how do traditional and cloud-based systems compare when it comes to driving innovation and daily operations? Next, what can we learn from companies that have successfully moved to the cloud and those that have tried but failed? Finally, what are the best ways to manage and lead big projects to update outdated systems?

3. Literature Review

3.1 Digital Transformation and Enterprise Agility

A company's ability to adapt quickly to changes is crucial in today's fast-paced business world. This is known as enterprise agility. It's about being able to spot changes, make smart decisions, and adjust resources rapidly without disrupting normal operations. In the digital market, agility plays a big role in how products are developed, how customers are treated, how well a company can handle tough situations, and how quickly it can respond to competitors.

According to Teece, back in 2007, he came up with a pretty useful idea - the dynamic capabilities framework. This framework is all about how companies can stay competitive in a changing world. It breaks down into three main parts: spotting opportunities and potential problems, taking advantage of those opportunities by making smart investments and decisions, and constantly updating your resources and processes to stay ahead of the game. When you modernize your legacy systems, it can really support these capabilities. For instance, it lets you try out new things faster, gives you better access to the information you need, and makes it easier to allocate your resources in a more flexible way.

But having the latest technology doesn't automatically make a company flexible and able to adapt quickly. A business can use the most advanced cloud systems, yet still have slow decision-making processes, unclear roles, and budgeting that's hard to change. In this case, just using new technology doesn't really improve how the company responds to things, it just makes the underlying systems better. Vial (2019) similarly emphasises that digital transformation involves organisational responses to technological disruption and the resulting changes in value creation.

This perspective suggests that architecture, governance, employee capabilities, and business strategy must be considered together.

3.2 Structural Barriers to Legacy Modernisation

A major obstacle is technical debt. Cunningham (1992) introduced the concept to describe the future costs associated with taking shortcuts in software development. In enterprise environments, technical debt can include duplicated code, obsolete dependencies, inadequate documentation, insecure components, and architectural decisions that make subsequent modifications expensive.

Maintaining old systems isn't always a sign of bad management. Some older applications have been around for decades and have a lot of business knowledge built into them, which helps keep important processes running smoothly. The problem is figuring out when the cost of keeping these old systems going and the risks of using them outweigh the benefits of sticking with what you already have. It's a tough call, but sometimes you have to weigh the pros and cons of holding on to outdated technology. Another problem is that different departments in a company often have their own separate databases for things like customer info, products, finances, and operations.

This can cause issues because these databases might not be consistent with each other, and not everyone in the company can access them. As a result, it's hard to get a clear picture of what's going on across the whole organization. This also makes it tough to use artificial intelligence and automated decision-making tools, because they need data that's easy to get to, reliable, and well-managed. When data is all over the place, it's hard to make these tools work well.

Companies can't innovate as much as they want because of rigid architecture. Older applications have parts that are closely linked, so changing one thing means testing the whole thing, which increases the risk of problems and slows down development. Some researchers, like Jamshidi and others, found in 2023 that the quality of the application and its architecture are big issues when moving to the cloud. So, when companies move their applications, they need to think about making them easy to maintain, reliable, and good quality, not just focus on getting them moved. This means looking at things like how well the application works, how reliable it is, and other important factors, to make sure the move is successful.

3.3 Monolithic and Cloud-Native Architectures

A simple application that does many things is often easier to put out and manage when it's small. This is because it's all in one piece, so it's not as complicated to get it working. But as a company grows and its processes get more complicated, this kind of setup can become hard to change. It's like trying to fix a big machine that's all connected - if one part breaks, it's tough to fix without affecting the whole thing.

When you break down a big application into smaller parts, it's called microservices architecture. Each part, or service, is like a small team that works on its own, but still talks to the other teams. They use special ways of communicating, like APIs and messaging systems, to make sure everything works together smoothly. This way, different teams can make changes and updates on their own, without affecting the whole system. And if one part of the system gets really busy, it can grow and handle more work without slowing down the rest of the application.

However, microservices introduce distributed-system challenges, including network failures, service discovery, latency, observability, data consistency, and more demanding security controls. Fowler and Lewis (2014) describe microservices as independently deployable services organised around business capabilities, but such independence requires appropriate engineering and organisational practices.

Cloud-native architecture extends beyond hosting applications in the cloud. It typically incorporates automation, elastic infrastructure, containerisation where appropriate, continuous integration and delivery, monitoring, and resilient application design.

Here's how companies can make their apps and services work together smoothly: they design the interfaces first, like a product, so that everything can communicate consistently. This way, different applications, partners, and digital channels can all talk to each other easily. Another approach is serverless computing, where companies don't have to manage the underlying servers to run their code or apps. This can simplify infrastructure management and handle sudden increases in workload.

However, it's essential to consider the potential downsides, such as unpredictable costs, limits on execution, relying too much on vendors, and the complexity of debugging.

3.4 Governance and Organisational Change

Enterprise modernisation requires governance arrangements that coordinate strategic objectives, technical decisions, regulatory obligations, and operational accountability. Excessive centralisation can slow innovation, whereas insufficient governance can produce inconsistent architectures, security weaknesses, and duplicated investment.

(Weill and Ross, 2004) demonstrate the importance of clearly defined IT decision rights and accountability. Their work supports governance models that distinguish enterprise-wide decisions from decisions that should remain close to individual business and engineering teams.

Organisational change theory also provides relevant insights. (Kotter, 1996) emphasises establishing urgency, building leadership coalitions, communicating a vision, and consolidating improvements.

Although the model offers a useful change management framework, it should not be treated as a rigid sequence for complex technology programmes. Large-scale transformations often require continuous feedback, iterative delivery, and adaptation.

DevOps practices provide a complementary operational approach by integrating software development and IT operations through automation, shared responsibility, monitoring, and continuous improvement. Effective implementation requires changes to incentives and team structures rather than simply introducing new tools.

4. Methodology

This report looks at how companies are modernizing by studying what other researchers have found and what companies have shared publicly. We're using this approach because we want to understand how companies are organized, the technology choices they make, how they govern themselves, and what we can learn from companies that have already gone through big changes.

We looked at three types of evidence. First, there are studies that give us a theoretical understanding of how digital transformation works, as well as concepts like dynamic capabilities, technical debt, software architecture, and IT governance. Then, there are reports from companies that show how they've actually put modernization programs into action and what results they've seen. Finally, we can learn from examples of when migrations have gone wrong, which can reveal weaknesses in how things are run and governed.

Netflix was selected as a successful modernisation example because its published account describes a prolonged transition towards cloud infrastructure, distributed services, and independent engineering teams. TSB Bank was selected as a contrasting case because its published independent review identifies technical and governance issues associated with the 2018 migration. (TSB Bank, 2019)

The study used a thematic approach to analyze the data. It grouped the findings into five main areas: outdated systems, updating technology, keeping things running smoothly, management, and changes within the organization. By looking at each case in terms of these areas, the researchers were able to pinpoint what makes projects succeed or fail. They identified common patterns that led to success or failure in each of these areas, which helped to shed light on what works and what doesn't.

The approach used here has its own set of limitations. When companies share their stories, they often focus on the good things that happened, while reports about failures might only show one side of the story. Also, just because something is publicly known doesn't mean we can say for sure that using a particular technology was the only reason a certain business result happened. So, this report separates what actually happened from what we think it might mean and what we suggest should be done. The results are based on analysis rather than new interviews, surveys, or number-crunching experiments. (Teece, D. J., 2007)

5. Findings

5.1 Challenges of Legacy Enterprise Systems

The analysis identifies five interconnected barriers to enterprise modernisation.

Technical debt and maintenance costs. Legacy applications often require specialist knowledge, outdated development environments, and extensive regression testing. As maintenance consumes a growing proportion of IT capacity, fewer resources remain available for innovation. Replacing a stable system without adequate planning, however, can create greater financial and operational risks than continued maintenance. (Teece, 2007)

Data fragmentation. Separate databases and inconsistent data definitions undermine reporting, analytics, and customer visibility. Integration middleware and common data standards can reduce fragmentation, but centralised data platforms must preserve ownership, privacy, access controls, and data quality.

Architectural coupling. Tightly integrated applications make releases difficult to isolate. A minor change can affect multiple business functions, increasing testing requirements and the likelihood of disruption. Modernisation should therefore prioritise the most restrictive dependencies rather than automatically decomposing every application.

Changing to cloud-native engineering can be tough for companies. It needs a lot of skills like automating tasks, managing infrastructure, keeping an eye on things, securing systems, and handling distributed systems. But the problem is, employees might resist this change when it affects their current roles or if they don't get the training they need. Also, if companies don't have the right skills in-house, they have to rely on outside suppliers, which can be a challenge. This can lead to a lack of control and flexibility, making it harder for businesses to adapt to new technologies and stay competitive. (Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., & Tilkov, S., 2023).

Companies that deal with money, health, and other sensitive areas have to follow strict rules. They need to keep track of everything, keep things secret, make sure everything works smoothly, and follow the law. Just because they use a cloud service, it doesn't mean they can pass on all the responsibility for security and following rules to that service. These companies still have to be in charge of managing who can access what, handling data, recovering from problems, and keeping an eye on the people they work with. They can't just shift all the blame to the cloud service if something goes wrong.

These obstacles work together to make things harder. For instance, when old systems aren't well-documented, it's tougher to test them when you're trying to move to a new system. And when lots of different teams own different parts of the system, it can be slow to fix problems. So, updating old systems needs to be handled as a big, coordinated project that involves the whole company. This means teams need to work together to make it happen. (Weill, P., & Ross, J. W., 2004).

5.2 Cloud Architecture and Modernisation Strategies

Organisations should select modernisation strategies according to application criticality, business value, technical condition, and risk.

Rehosting is when you move an app to the cloud without making many changes. This can speed up the process of getting out of a data center, but it might not fix underlying issues with how the app is set up.

Replatforming: Making limited changes to use managed cloud services while retaining most application logic.

Refactoring or re-architecting: Redesigning components to improve scalability, maintainability, resilience, or deployment independence.

Replacing: Adopting a commercial software product or a newly developed application when the legacy solution no longer meets business needs.

Retaining or retiring: Keeping systems that remain appropriate or removing applications that no longer provide sufficient value.

These methods work well together. A financial company might move a simple reporting app to the cloud, update a customer service to make it better, replace an old human resources system, and keep a stable system for following rules until a safer option is available. This way, they can use the best approach for each part of their business.

A good modern architecture should be able to do several things. It should let old systems share information with each other in a controlled way, without needing to be replaced right away. This is done through API-led integration. It should also be able to use event-driven messaging to separate processes and reduce how much different applications rely on each other. Additionally, using managed cloud services and serverless computing can make things easier to run for certain types of workloads. And finally, automating tests, setting up deployment pipelines, and treating infrastructure like code can help make things more consistent and reliable. This way, businesses can keep up with the latest technology without getting rid of their old systems all at once. By combining these capabilities, companies can create a modern architecture that is flexible, efficient, and easy to manage.

Security must be integrated throughout the process. Identity and access management, encryption, vulnerability management, audit logging, and automated policy enforcement should be established before workloads are migrated. Organisations should also design recovery mechanisms, test backup restoration, and validate failover procedures. (TSB Bank, 2019)

5.3 Successful Transformation: Netflix

Netflix illustrates the strategic potential of sustained architectural and organisational modernisation. The company's cloud journey began following a major database failure in 2008 that prevented DVD shipments for three days. Netflix subsequently pursued a transition from vertically scaled data-centre systems towards cloud infrastructure and distributed services.

Netflix finally finished moving all its streaming services to the cloud in January 2016, after working on it for about seven years. They wrote about how they moved their customer-facing services to Amazon Web Services, created systems that work across many machines, and changed from one big application to hundreds of smaller services. This was a big change, but it's not the only reason why more people started using Netflix and watching more content during this time. The company saw a big jump in the number of people streaming and the amount of time they spent watching, but it's hard to say how much of that growth was directly because of the move to the cloud.

What we can learn from Netflix's experience. First, they didn't just move their stuff to the cloud - they completely changed how their systems worked. They rebuilt everything and thought about how

they were using data in a new way. This helped them get around the problems they were having with their old setup. Second, they changed how they worked as a team. They gave their engineers more freedom to work on their own, set up systems that could be used by anyone, and made it so they could get new updates out quickly.

This meant they didn't have to rely on a central team to coordinate everything and wait for new equipment to be set up. Third, the move to the cloud wasn't just about moving some computers - it was about changing how the whole company operated.

Netflix didn't get rid of all its old systems right away. It took some time to deal with sensitive things like billing and customer data. This shows that it's crucial to make changes in the right order, based on how risky they are and what the business needs. They had to be careful with certain areas, and that's why it was a gradual process.

The thing is, this approach has its limits. For one, Netflix works in a world that's all about digital stuff and it has a lot of tech expertise, plus its business model really thrives on the internet. So, you can't just copy and paste its way of doing things to every other company that has to follow rules. Using microservices also makes things more complicated to run, and to make it work, you need to have really good automation, monitoring, and engineering practices in place. (Netflix, 2016)

5.4 Migration Failure: TSB Bank

TSB Bank's 2018 IT migration demonstrates how major modernisation programmes can fail operationally despite extensive preparation. The migration sought to move customer data from a legacy platform inherited during its separation from Lloyds Banking Group to a new technology platform.

After the big move in April 2018, TSB's online banking services were seriously disrupted, causing problems for customers. To figure out what went wrong, TSB hired a separate company, Slaughter and May, to investigate. The report they made public pointed out some major issues - the bank didn't keep a close enough eye on the companies it worked with, and there were technical problems like different systems not working well together, coding mistakes, and not enough capacity. On top of that, the bank admitted that the surge in customer calls and visits to branches put a lot of extra pressure on their services.

This example shows that just moving data to a new system isn't enough to call a migration a success. You also need to think about whether the system is actually working, if customers can use it, if transactions are happening correctly, and if you can recover data if something goes wrong. All these things are just as important as getting the data moved over.

First, testing must reproduce realistic operational conditions. Successful rehearsals do not necessarily establish readiness if production configurations differ or capacity tests fail to reflect actual demand. Configuration drift can invalidate otherwise comprehensive testing.

Second, governance must be evidence-based. Executive confidence and supplier assurances should not substitute for independently verified readiness criteria. Senior leaders must understand unresolved defects, operational dependencies, and rollback feasibility.

When making plans for unexpected events, companies need to think about the whole organization, not just one part. For example, when something went wrong, customers started calling and visiting branches, which couldn't handle the extra workload. To keep things running smoothly, businesses should plan for these kinds of secondary effects, like making sure they have enough support staff, can communicate with customers, and have a backup plan for when systems are down. This way, they can handle the extra demand and keep customers happy.

Fourth, accountability cannot be outsourced. Suppliers can provide specialist expertise, but the organisation remains responsible for ensuring that systems are fit for purpose and that migration decisions protect customers.

The TSB situation doesn't mean that updating systems is a bad idea. In fact, a new platform can offer better digital features once it's working smoothly. What it really shows is that the good things we expect from transformation can be ruined by poor technical checks and programme management. (TSB Bank, 2019, November 19).

5.5 Governance and Change Management Practices

The cases suggest that effective transformation requires a governance model combining enterprise-wide standards with delegated execution. To make sure a company's projects go smoothly, a top team should set big goals, decide on a budget, watch out for problems, and solve disagreements between different departments. A special office should help different teams work together, keep track of progress, and measure the benefits of the project. The company's tech team should come up with common rules, security standards, and approved ways of doing things to make sure everything works well together. This way, the company can make sure its projects are on track and meeting its goals. The tech team's rules and standards are important to keep the company's technology running smoothly and securely.

Teams that focus on products should be able to make changes on their own. Each team should be in charge of a specific business area, including the software, data, reliability, and how it all works together. This way, they can take care of everything related to that area without needing to check with others. It's like having a clear owner for each part of the business, making it easier to get things done. (Kotter, 1996)

A practical governance structure includes four levels: executive governance for strategic objectives and investment; architecture and security governance for standards and controls; product and engineering governance for delivery and service ownership; and operational assurance for reliability, incident management, and recovery testing.

To make sure a company changes effectively, its leaders need to support the new structure. They should explain why updates are needed, find out who will be affected, and tell everyone how these changes will make things better for customers and employees. Any training should be related to the actual jobs people do, like working with cloud systems, managing products, keeping things secure, handling data, and running services. This way, everyone knows what to expect and can do their job well.

Resistance should be treated as useful information rather than automatically interpreted as unwillingness to change. Employees may identify operational risks that programme leaders have overlooked. Pilot projects, staff feedback, and iterative delivery provide opportunities to incorporate such knowledge.

6. Discussion

The findings reinforce the argument that legacy modernisation is a strategic capability rather than an isolated infrastructure project. The benefits of cloud adoption emerge through interactions among architecture, operating models, governance, and organisational learning. The dynamic capabilities framework helps explain this relationship. Cloud platforms can improve the ability to sense market opportunities by enabling faster access to operational data. (Teece, 2007)

Companies can quickly try out new products and services with modular services and automated deployment. This helps them grab new opportunities. To make the most of this, businesses often

need to reorganize, train their staff, and get rid of old systems that are no longer needed. By doing so, they can breathe new life into their existing resources and stay ahead in the game. Nevertheless, these capabilities are not guaranteed. An organisation that migrates applications without changing release processes may continue to experience lengthy delivery cycles. Similarly, a microservices environment without clear ownership can increase the number of dependencies that teams must coordinate. (Fowler & Lewis, 2014)

The comparison between Netflix and TSB highlights the importance of transformation design. Netflix's published experience emphasises sustained architectural redesign, independent engineering teams, and gradual migration. TSB's experience highlights the consequences of inconsistent configurations, inadequate assurance in critical areas, and insufficient resilience across customer-facing channels.

Although the organisations differ considerably, both demonstrate that migration outcomes depend on decisions made before, during, and after deployment. Here's the thing: when you're trying to balance being flexible and being in control, you've got to make some tough choices. If you've got a really strict, top-down approach, it can slow things down and create bottlenecks. But on the other hand, if you give everyone too much freedom, you might end up with security problems, different teams using different definitions for the same thing, and people duplicating work. One way to find a middle ground is to use a federated model. This means that the central teams set some guidelines and shared tools, but the teams working on specific products get to make the day-to-day decisions within those boundaries. (TSB Bank, 2019)

To stay ahead of the competition, a company needs more than just the latest technology. Its rivals can easily get the same cloud services, so it's not enough to just have the best tools. What really sets a company apart is how well it uses its unique strengths, like its own data, customer relationships, industry expertise, and special way of doing things. A researcher named Teece found that being able to adjust and change quickly is crucial for success in a fast-changing world. So, when looking at the benefits of modernizing, we should consider not just the cost of new infrastructure, but also how it can help us innovate and adapt to new challenges. This means thinking about how we can use technology to improve our processes, make new products, and respond to changes in the market. By doing this, we can create a lasting competitive advantage that sets us apart from our rivals.

6.1 Strategic Recommendations

1: Establish a business-led modernisation strategy. Create an inventory of applications, dependencies, costs, technical debt, and business criticality. Rank systems according to strategic importance and risk rather than technical age alone.

2: Modernise incrementally. Use pilots and bounded migration waves to test architecture, operating procedures, and recovery arrangements. Prioritise applications where measurable benefits can be achieved without exposing the organisation to unacceptable operational risk.

3: Adopt API-first integration and selective decomposition. Introduce stable interfaces around legacy systems and extract services where independent deployment or scaling provides demonstrable value. Retain monolithic designs where they remain simpler and economically appropriate.

4: Embed security and resilience. Apply security-by-design, automated configuration checks, realistic performance testing, independent readiness reviews, and tested rollback procedures.

5: Implement federated governance and DevOps. Establish central architecture, security, and data standards while empowering product teams to manage services within defined boundaries. Introduce automated delivery pipelines, shared operational accountability, and continuous monitoring. (Lenarduzzi et al., 2020)

6.2 Proposed Implementation Roadmap

1: Assess and align your business (this usually takes about three months). Get a top executive to back your project, look at all the apps you're using, figure out what's crucial, decide what you want to achieve, and make a list of what needs to change, in order of importance.

2: Foundations and pilots (months 4–6). Establish cloud landing zones, identity and security controls, cost management, observability, automated delivery pipelines, and a pilot migration with measurable acceptance criteria.

3: Incremental modernisation (months 7–12). Migrate suitable workloads, introduce API integration, refactor selected business capabilities, train product teams, and conduct operational resilience exercises.

4: Scaling and optimisation (month 13 onward). Expand successful patterns, retire redundant systems, optimise cloud consumption, review governance effectiveness, and continuously reassess business outcomes.

7. Conclusion

Many companies need to update their old systems to stay competitive and innovate faster. But just because a system is old doesn't mean it's not useful. These legacy systems often have important business rules, work reliably, and hold vital historical data. So, when updating them, companies must carefully weigh the business benefits, technical drawbacks, and operational risks. This balanced approach is key to a successful transformation.

Using cloud computing, microservices, and other new technologies can help businesses grow and innovate faster. These tools, like APIs, serverless computing, and DevOps, make it easier to scale and deploy new services. However, they also bring new challenges, such as extra costs and security concerns. To really benefit from these technologies, companies need to make smart architectural choices and have the right organizational capabilities in place to manage them effectively. For example, they need to be able to handle the added complexity and operational dependencies that come with these new technologies. If done right, though, the payoff can be significant, leading to faster innovation and better scalability. This way, they can stay competitive and respond quickly to changes in the market.

8. References

Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., & Tilkov, S. (2023). Legacy systems to cloud migration: A review from the architectural perspective. Journal of Systems and Software, 202, Article 111702. https://doi.org/10.1016/j.jss.2023.111702

Kotter, J. P. (1996). Leading change. Harvard Business School Press.

Lenarduzzi, V., Lomio, F., Saarimäki, N., & Taibi, D. (2020). Does migrating a monolithic system to microservices decrease the technical debt? Journal of Systems and Software, 169, Article 110710. https://doi.org/10.1016/j.jss.2020.110710

Netflix. (2016, February 12). Completing the Netflix cloud migration. About Netflix. https://about.netflix.com/en/news/completing-the-netflix-cloud-migration

Teece, D. J. (2007). Explicating dynamic capabilities: The nature and microfoundations of (sustainable) enterprise performance. Strategic Management Journal, 28(13), 1319–1350. https://doi.org/10.1002/smj.640

TSB Bank. (2019, November 19). TSB Board publishes independent review of 2018 IT migration. https://www.tsb.co.uk/news-releases/slaughter-and-may.html

Vial, G. (2019). Understanding digital transformation: A review and a research agenda. The Journal of Strategic Information Systems, 28(2), 118–144. https://doi.org/10.1016/j.jsis.2019.01.003

Weill, P., & Ross, J. W. (2004). IT governance: How top performers manage IT decision rights for superior results. Harvard Business School Press.

9. Appendices

Appendix A: Comparison of Legacy and Modern Enterprise Architectures

Dimension

Traditional monolithic environment.

Modern cloud-native environment.

Architecture

Tightly coupled application components.

Modular services where justified.

Deployment

Often coordinated, application-wide releases.

Automated and potentially independent releases.

Scalability

Frequently scales the application as a whole.

Can scale selected services independently.

Integration

Point-to-point connections and older interfaces.

APIs, messaging, and managed integration.

Appendix B: Proposed Transformation Performance Indicators

Performance indicator

Purpose

Measurement approach

Lead time for changes

Assess development speed

Time from committed change to production

Change failure rate

Assess release quality

Percentage of deployments requiring remediation

Mean time to recovery

Assess operational resilience

Average time to restore service after disruption

Cloud cost per transaction

Assess cost efficiency

Allocated cloud expenditure divided by business transactions

Customer satisfaction

Assess user outcomes

Consistent customer survey or satisfaction measure

Appendix C: Migration Readiness Checklist

We've finished making a list of all our applications and figured out which ones are crucial to our business.

Data ownership, classification, and migration requirements documented.

Application dependencies and interface contracts mapped.

Security, identity, and compliance controls validated.

We've finished testing how well the system performs and its capacity in conditions that are similar to real-life situations.

Production configuration consistency independently verified.

Backup restoration and disaster recovery tested.

Rollback criteria and decision authority documented.