Legacy Digital Transformation: TCO and ROI Analysis

Review Paper

Ravi Kiran Mallidi
School of Computer Applications
Lovely Professional University
Punjab, India
ravikiran.mallidi@gmail.com

Manmohan Sharma
School of Computer Applications
Lovely Professional University
Punjab, India
manmohan.sharma71@gmail.com

Jagjit Singh
Mittal School of Business
Lovely Professional University
Punjab, India
jagjit.singh@gmail.com

Abstract *– Legacy Digital Transformation is modernizing or migrating systems from non-digital or older digital technology to newer digital technologies. Digitalization is essential for information reading, processing, transforming, and storing. Social media, Cloud, and analytics are the major technologies in today's digital world. Digitalization (business process) and Digital Transformation (the effect) are the core elements of newer global policies and processes. Recent COVID pandemic situation, Organizations are willing to digitalize their environment without losing business. Digital technologies help to improve their capabilities to transform processes that intern promote new business models. Applications cannot remain static and should modernize to meet the evolving business and technology needs. Business needs time to market, Agility, and reduce technical debt. Technology needs consist of APIs, better Security, Portability, Scalability, Cloud support, Deployment, Automation, and Integration. This paper elaborates different transformation/ modernization approaches for Legacy systems written in very long or End of Life (EOL) systems to newer digital technologies to serve the business needs. EOL impacts application production, supportability, compliance, and security. Organizations spend money and resources on Digital Transformation for considering Investment versus Return on Investment, Agility of the System, and improved business processes. Migration and Modernization are critical for any Legacy Digital Transformation. Management takes decisions to proceed with Digital Transformation for considering Total Cost Ownership (TCO) and Return on Investment (ROI) of the program. The paper also includes a TCO-ROI calculator for Transformation from Legacy / Monolithic to new architectures like Microservices.*

Keywords: *Legacy, Digital Transformation, Automation, Migration, Modernization, Monolithic, Microservices*

1. INTRODUCTION

Organizations are currently investing money and resources in transforming their existing systems into feature-ready digital technologies. Each has a different business strategy to grow. They are adopting Digital Transformation by planning and implementing new digital systems for business changes in the Organization with employees' participation. Transformations help the Organization lower the operating cost considerably by adopting automation, innovation, and creativity rather than traditional methods. The best example of Digital Transformation is cloud computing in the Organization, which reduces the owned hardware cost and maintenance and increases the trust in subscription-based models. Digitalization provides opportunities to transform business models, consumption, legal and policy measures, and cultural barriers. Renovating and optimizing business processes is the success of digital Transformation. Organizations must factor in the cultural changes concerning workers, and organizational leaders adjust to adopting and relying on unfamiliar technologies while planning for the Digital Transformation strategy. The revenues, profits, and opportunities increase upside after Transformation

completes. Lack of innovation impacts the business's long-term competitiveness and profits. Automation also plays a critical role in increasing Business efficiency and Agility up to 50-60%. EOL also a significant objective for migrating the applications to a new technology stack. The majority of the software EOL ranges from 10-15 years. Major domains involved in Digital Transformation were Hospital Management, E-Commerce, Banking, Insurance, Training, Education, and Health care.

Legacy-based applications constantly face challenges like high maintenance and upgrades cost, unsupported hardware, resource cost, flexibility, scalability, and integration to the external system. Legacy application is an IT-critical day-to-day operations system build on outdated technologies. Several applications are running for the last 20-30 years in the world with legacy technologies. The applications use mainframe and other 3GL languages for core business operations. Many companies continuing legacy systems regardless of the age and technology of the application used. Such legacy applications' running cost is very high in maintaining and support—time to modernize such applications with proper planning and strategy. Legacy modernization is also called software or platform modernization. Legacy Modernization approach consists of two major phases called the Assessment phase and Implementation phase. Organizations are migrating their existing legacy applications to modern technologies with the help of microservices architecture patterns [28] [24] [22] [18] [15]. Microservices architecture promises high maintainability for software modernization [30], including decomposing the application into microservices-based on experience. Concluded due to higher efforts, the process is viable for large migrations and complex systems with high business value.

This paper organized as follows: Section II we describe Literature Review, Section III we describe Modernization Approaches, Section IV describes TCO_ROI Analysis, and Section V Concludes the paper

2. LITERATURE REVIEW

This section describes the literature review of the work done in this area. The research carried out [1] digital transformation capability maturity model on staged way using science research approach. Decomposition method [2] for migrating monolithic to microservices applications. Identify similar semantically operations and cluster together for a microservice. Evaluation metrics used to identify the right microservices candidate. Legacy system challenges [3] highlight along with solutions. The proposed legacy modernization uses Rehosting, Replacing, Mitigation, Retargeting, Revamping, Wrapping, and Program translation. Systematic [4] review was conducted on legacy system modernization papers and concluded the importance of integrating quality. Explains different modernization types as Complex migration, increased migration, and partial migration and compared the modernization strategies like partial migration, complete migration, and wrapping. System Migration Life Cycle [5] proposed for step-by-step migration strategy from legacy to cloud platforms. Framework includes three stages – pre-migration, migration, and post-migration. Studied [6] the cultural readiness of the executives is important. Studied [7] different papers on legacy migrations and identified the factors – Process Aspects, Human Aspects, and Organization Aspects. Proposed 5 phased migration approach called Plan for Modernization, System Requirement Gathering, Design & Development, Testing, and Execution. Explains [9] what next in the FinTech industry after Digitalization using AI, Blockchain, Quantum computing, IoT, and Smart contracts. Explains disruptive innovations in trading, crypto, and monetary in FinTech systems. Proposed different process to [8] [12] [13] identifying the microservices from monolithic applications and uses of blue-green [10] strategy for faster deployment.

Conducted survey [11] with specialists to identify valuable criteria for identifying microservices from legacy systems. Use of Architectural Trade-off Analysis Method [14] for Architecture evaluation of old legacy system. The proposed [15] architecture strategy around business functionality concepts comprises 5 phases – Functional analysis, Business functionalities identification, Business functionalities analysis, Business functionalities assignment, and Microservice creation. [16] [17] [21] [29] case study and survey conducted for migrating monolithic to microservices for large-scale industries to identify issues, solutions, challenges, strategies, and risks. VUCA [20] explains how the digitally immature organizations in the COVID pandemic. VUCA, acronym of Volatility, Uncertainty, Complexity, and Ambiguity to identify the unpredictable external environments. Concludes that COVID made digital transformation for all businesses and all sectors. Explains [22] three challenges for legacy migration to microservices are Multitenancy, Stateful, and Data Consistency. The three challenges addressed in microservices to develop stateful systems, implement multitenancy systems, and solving data consistency issues Proposed [23] reference model for Digital Transformation (DT) applications using Business process Management Contextual Factors and DT. Presented [25] lessons learned during migration - functional approach, norms and standards, microservices granularity, and integration outcomes are critical aspects described in the paper. Serverless compute architecture [26] [32] was proposed for migrating monolithic applications with benefits. Suggested boundary context approach [31] that extends static and dynamic analysis for decompositions of microservices. Recommended algorithm approach [33] for extracting microservices migration.

Introduced generic model [40] allows incorporating the characteristics of relevant dynamics that instantiated for specific characterizations. Studied [35] 20 migration techniques from literature, and results show that DB migration is the challenge. Address [37] migration issue by adding two questions – the cost of decomposition and

domain entries. Conducted survey [38] on motivational drivers for migration and concluded maintainability and long-term returns are primary factors. Reported [42] open-source research on modularization and modernization process. The process decompositions of web as well as technical components. Adopted CORAL (Collaborative Reengineering and Modularization Approach) for modernization process from legacy to microservices. Presented [36] high available application use case for ticket booking application modernization to microservice architecture and identified several benefits – choose preferred technology, no dependency on hardware, and new feature enablement using DevOps. Presented real-world financial case study [18] to demonstrate scalability by migrating monolithic to microservices architecture. Technical problems addressed using repeatable migration process adoption in Danske Bank legacy application to microservices architecture. The reengineering approach discussed reduced complexity, lower coupling, higher cohesion, and a simplified integration. Technical lessons [24] learned discussed as part migration process in MGDIA SA company spending around 3 years with 17,300 person-days. Demonstrated the Return on Investment for the migration with an Increase in Revenues, Replacement facilitated, Performance increase, and lower level of support. The proposed [28] candidates to show the relationship between extracted and whole structure. Analyze the relationships between program groups and data for preparing microservices.

3. MODERNIZATION APPROCHES

Legacy Modernization approach consists of two major phases - The assessment phase and the Implementation phase. The assessment phase consists of IT portfolio Assessment, To-Be Architecture definition, modernization strategy, and business case justifications. The implementation phase considers modernization Application Transformation, Database Transformation, Infrastructure Transformation, and Operational Transformation. Below are the challenges considerations while modernizing the applications

- Lack of portfolios across applications
- Poor management of systems documentation
- Large application maintenance
- Cost and duration of modernization
- Legacy and new systems coexistence
- Commitment from stakeholders
- Adoption of new technology platforms
- Cultural change adoption
- Adopting a new way of working (E.g., Work from home options)
- Retain and enhance the application with business needs
- Business value and ROI

Application Modernizations is not a rewriting of code from one technology to another. Following principles has to take into consideration while doing modernizations.

- Clear Organization road map to get early ROI
- Developing new capabilities
- Adoption of Open standards rather than vendor lock-in
- Adoption of microservices design rather monolithic
- API version support for all service calls
- Use of integration layer for seamless external system integrations
- Identify new business channels and models to build new digital channels
- Balance between Business and Technology for time to market
- Effort and cost optimizations
- End-user OR customer delight
- Adoption of Agile and DevOps

Good architecture, design and coding, vendor-neutral, maintain open standards, and security compliance with measurable KPIs are the critical factors for successful application modernization. The assessment phase recommends the applications have to be Retain, Retire/Rationalize, Rehosting, Replace, Refactor, Re-architecture, and Rebuild / Rewrite (7-R's). Based on the assessment recommendation, the transformation strategy derives as per the Organization's goals. Legacy Applications categorization is based on the number of users, criticality, benefit, maintenance cost, risk, and feasibility. Reactor or Rewrite's legacy application decision is to study existing source code analysis and understand cloud-native compliance based on 12-factor anti-patterns and categorize applications. Adopt Agile and DevOps for a new way of working in the digital world. Table-1 describes each modernization approach with their benefit.

Application Modernization has several options from Legacy to new Digital. Below are the top modernizations options

- Legacy to Open platforms
- Legacy (Cobol / native languages) to Microservices
- Monolithic to Microservices
- Commercial to Open-Source

Application modernization/migration done manually and tool-based.

Tool-based migration approach - Conversion tool to automatically convert the existing use cases into the target technologies to whatever extent the tool can support and then hand-finish the code to adhere to the "as-is" functionalities as shown in Fig 1. Migration tool to convert the existing use cases into target platform, and this option typically has two-step processes:

- Tool based conversion (up to 60-80% conversion)
- Hand finishes the missing content by the tool

Modernization Approach	Scenarios to Fit	Benefit
Retain	Retaining the application as-is without adding any new features	No change in TCO and business value
Retire/Rationalize	Applications no longer in use. Merge the functionality to another application.	Reduce Total Cost of Ownership with rationalization of applications
Rehosting	Deploying the applications in a Virtual / Cloud environment without any change	Reduce migration cost and operations.
Replace	Eliminate existing application components and functions with new requirements.	Lower the operational cost with compared to Legacy application
Refactor	Standalone applications needs interfaces using modern technologies without modifying features and functions	Reduces Total Cost of Ownership and Improves maintainability
Re-architecture	Shift the application to new platform architecture with better capabilities	Improves the non-functional capabilities
Rebuild / Rewrite / Reengineering	Rewrite the application components with new technology stack without changing the scope and functionality	Flexibility to customize as per business needs

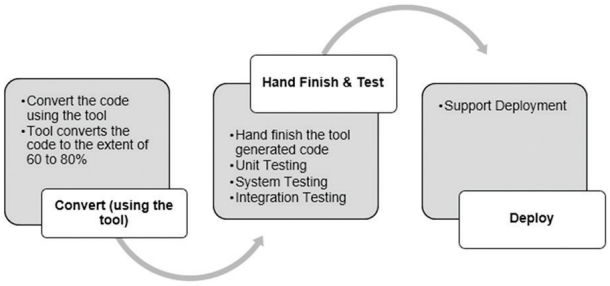

Fig. 1. Tool based migration approach

Manual migration approach – Consists of reverse engineer the existing technology stack requirements and rewrite manually into target technologies covering the entire SDLC phase of Design, Development, and Testing. Reverse engineering methodology is a two-step process called reverse engineering followed by forward engineering, as shown in Fig 2. Study the existing applications and collect the business rules, design models, and workflows and document as part of reverse engineering. After done the reverse engineering, build the new application on the target platform using the documentation.

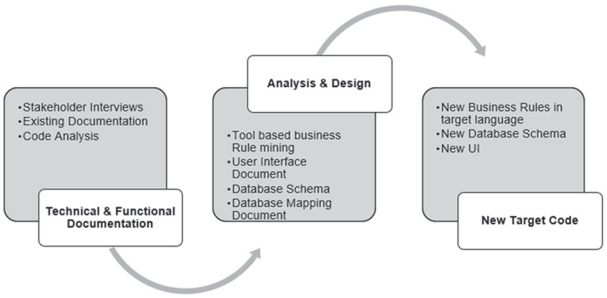

Fig. 2. Manual based migration approach

Table 2. Manual and Automation migration comparisons

Parameter	Manual Migration	Automation Migration
Cost	High	Low
Schedule / Time	Longer Development Cycle	Shorter Development Cycle
Code Quality	Sometime not clean and good	Clean and good
Flexibility	Maximum flexibility in design and modification	Very rigid, based on rules defined in the tool

Legacy modernization has many benefits in terms of reduced cost, enhanced flexibility, and vendor options. Nowadays, the modernizations are widespread, from Legacy to Cloud-ready, DevOps, Automation, AI and Big Data, and Mobile First approach application. Migrating legacy applications to Cloud for better scalability, performance, and flexibility. DevOps simplifies the process of modernization by shared responsibility and a collaborative approach. Manual processes are the barriers to the Legacy applications. Organizations can reduce operational costs and increase efficiency by moving manual to automation. Digital Transformation uses AI-based tools for Data-driven decisions. Mobile-first approach development leads the business to serve the customer in a faster and flexible manner.

4. TCO-ROI ANALYSIS

TCO and ROI analysis help the right investment decisions for digital transformations. TCO is a cost spent OR One time cost spent for the modernization of the systems. Return on Investment is the ratio between investment gains and TCO. Application is migrating to Cloud the cost distributed among several years for hardware and software. The below use case provides details of TCO-ROI calculation for commercial to open-source transformation.

Calculate the cost incurred for the current environment concerning application scopes described in Fig 3 – Hardware, Operating System, Web & Applications Servers, Database Servers, Application Maintenance, and Other Integration Systems. Fig 3 provides the information about current environment cost

Application Scope	Ontime Cost	Support		
		Year 1	Year 2	Year 3
Hardware Cost				
Operating System	$20,000.00			
Web & Application Servers (Commercial)		$20,000.00	$20,000.00	$20,000.00
DataBase Servers (Commercial)		$20,000.00	$20,000.00	$20,000.00
Applications Maintenace		$10,000.00	$10,000.00	$10,000.00
Overall Hardware Cost	**$20,000.00**			
Overall Software and Maintenance Cost	**$20,000.00**	**$50,000.00**	**$50,000.00**	**$50,000.00**
Overall Cost	**$40,000.00**	**$50,000.00**	**$50,000.00**	**$50,000.00**

Fig. 3. Current environment Cost

Calculate the one-time cost incurred for new environment concerning application scopes described in Fig 4 – Hardware, Operating System, Web & Applications Servers, Database Servers, Application Maintenance, Other Integration Systems, Migration Cost (Modernization efforts and time), Decommission cost, Parallel Run cost, and Maintenance. Fig 4 provides information about the new environment cost after migration.

Application Scope	Ontime Cost	Support		
		Year 1	Year 2	Year 3
Hardware Cost	$40,000.00			
Operating System (RedHat Linux)	$20,000.00			
Web & Application Servers (Open Systems)		$200.00	$200.00	$200.00
DataBase Servers (Open Systems)		$200.00	$200.00	$200.00
Migration Cost	$10,000.00			
Training Cost	$2,000.00			
Other productivity impact cost	$100.00			
Decommissioning cost	$100.00			
Parallel Run cost	$50.00			
Applications Maintenace		$8,000.00	$8,000.00	$8,000.00
Overall Hardware Cost	**$60,000.00**			
Overall Software and Maintenance Cost		**$8,400.00**	**$8,400.00**	**$8,400.00**
Ontime Migration Cost	**$12,250.00**			
Overall Cost	**$72,250.00**	**$8,400.00**	**$8,400.00**	**$8,400.00**

Fig 4. New environment Cost

ROI calculated based on the one-time investment shown in Fig 5.

ROI Consolidated				
	One time Cost	Year 1	Year 2	Year 2
Overall Cost - Current Environemt	$40,000.00	$50,000.00	$50,000.00	$50,000.00
Overall Cost - New Environemt	$72,250.00	$8,400.00	$8,400.00	$8,400.00
Total Savings		$41,600.00	$41,600.00	$41,600.00
Total Investment	**$32,250.00**			
Return on Investment		**$9,350.00**		

Fig 5. ROI Calculation

Return on Investment (ROI) = Overall Cost (Current Environment) – Overall Cost (New Environment)

From the above Fig 5 the Total Investment (TI) = One-time New Environment Cost – Current Cost (Software + Environment)

Total Investment
= $ 72,250.00 - $ 40,000.00 = $ 32,250.00

ROI Year-1 = Current Environment Cost (before migration) – New Environment Cost (after migration)

ROI Yeat-1 = $ 60,000.00 - $ 8,400.00 = $ 41,600.00

If the value is positive in Fig 5, the new environment maintenance cost of the product is less compared to the Old environment. Compare the overall cost between current and new environment. In the above case study, the difference between the new environment's one-time cost and the current environment is $32,250.00. Investment made by the modernization of the application to a newer environment. From Year-1 onwards, the saving from New and Current environment is around $41,600.00. The Return on Investment made in the first year of the modernization.

Sometimes applications are digitalizing to new technology stack to support new business. Banking industry enabling of Mobile for the existing application to invest some amount to get ROI with the time frame. The ROI analysis helps the management team to identify the applications to be migrated first.

5. CONCLUSION

Legacy Digital Transformations are essential nowadays for Organizations to enabling the time-to-market in their business. While making any decisions on transformation or modernization, management has to calculate the TCO-ROI of the program. ROI calculation helps the Organization to spend the time and efforts in the right direction. Sometimes the ROI calculation is not direct and has to develop logic how much revenue growth performs for the new feature. The ROI has to return from 2-3 years, and then the TCO spend is perfectly utilized. The right R (Retain, Retire/Rationalize, Rehosting, Replace, Refactor, Re-architecture, and Rebuild / Rewrite) approach has to choose while performing Digital Transformation consideration Agility using DevOps. The modernization program has to complete as per business agility. This paper highlights the TCO-ROI factors for any transformation or modernization, which influences the modernization strategy. There is scope to identify new transformation approaches for the newer business needs and technology transformations.

6. REFERENCES:

[1] T. Aguiar, S. B. Gomes, P. R. da Cunha, M. M. da Silva, "Digital Transformation Capability Maturity Model Framework", Proceedings of the 23rd IEEE International Enterprise Distributed Object Computing Conference, Paris, France, 28-31 October 2019, pp. 51-57.

[2] O. Al-Debagy, P. Martinek, "Extracting Microservices' Candidates from Monolithic Applications: Interface Analysis and Evaluation Metrics Approach", Proceedings of the 15th IEEE International Conference of System of Systems Engineering, Budapest, Hungary, 2-4 June 2020, pp. 289-294.

[3] M. Ali, S. Hussain, M. Ashraf, M. K. Paracha, "Addressing Software Related Issues On Legacy Systems–A Review", International Journal of Scientific Technology Research, Vol. 9, No. 3, 2020, pp. 3738-3742.

[4] B. Althani, S. Khaddaj, "Systematic review of legacy system migration", Proceedings of the 16th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, Anyang, China, 13-16 October 2017 pp. 154-157.

[5] B. Althani, S. Khaddaj, B. Makoond, "A quality assured framework for cloud adaptation and modernization of enterprise applications", Proceedings of the IEEE International Conference on Computational Science and Engineering and IEEE International Conference on Embedded and Ubiquitous Computing and the 15th International Symposium on Distributed Computing and Applications for Business Engineering, Paris, France, 24-26 August 2016 pp. 634-637.

[6] S. J. Andriole, "The hard truth about soft digital Transformation", IT Professional, Vol. 22, No. 5, 2020, pp. 13-16.

[7] H. K. A. Bakar, R. Razali, D. I. A. Jambari, "Guidance to Legacy Systems Modernization"

[8] M. H. G. Barbosa, P. H. M. Maia, "Towards Identifying Microservice Candidates from Business Rules Implemented in Stored Procedures", Proceedings of the IEEE International Conference on Software Architecture Companion, Salvador, Brazil, 16-20 March 2020, pp. 41-48.

[9] T. Butler, "What's Next in the Digital Transformation of Financial Industry?" IEEE Computer Architecture Letters, Vol. 22, No. 1, 2020, pp. 29-33.

[10] A. Buzachis, A. Galletta, A. Celesti, L. Carnevale, M. Villari, "Towards osmotic computing: a blue-green strategy for the fast re-deployment of microservices", Proceedings of the IEEE Symposium on Computers and Communications, Barcelona, Spain, 29 June-3 July 2019, pp. 1-6.

[11] L. Carvalho, A. Garcia, W. K. Assunção, R. de Mello, M. J. de Lima, "Analysis of the criteria adopted in industry to extract microservices", Proceedings of the IEEE/ACM Joint 7th International Workshop on Conducting Empirical Studies in Industry and 6th International Workshop on Software Engineering Research and Industrial Practice, Montreal, QC, Canada, 28 May 2019, pp. 22-29.

[12] M. D. Cojocaru, A. Uta, A. M. Oprescu, "MicroValid: A Validation Framework for Automatically Decomposed Microservices", Proceedings of the IEEE International Conference on Cloud Computing Technology and Science, Sydney, NSW, Australia, 11-13 December 2019, pp. 78-86.

[13] M. D. Cojocaru, A. Uta, A. M. Oprescu, "Attributes assessing the quality of microservices automatically decomposed from monolithic applications", Proceedings of the 18th International Symposium on Parallel and Distributed Computing, Amsterdam, Netherlands, 3-7 June 2019, pp. 84-93

[14] P. Cruz, H. Astudillo, R. Hilliard, M. Collado, "Assessing migration of a 20-year-old system to a microservice platform using ATAM", Proceedings of the IEEE International Conference on Software Architecture Companion, Hamburg, Germany, 25-26 March 2019, pp. 174-181.

[15] L. De Lauretis, "From Monolithic Architecture to Microservices Architecture", Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops, Berlin, Germany, 27-30 October 2019, pp. 93-96.

[16] S. S. de Toledo, A. Martini, A. Przybyszewska, D. I. Sjøberg, "Architectural technical debt in microservices: a case study in a large company", Proceedings of the IEEE/ACM International Conference on Technical Debt, Montreal, QC, Canada, 26 May 2019, pp. 78-87.

[17] P. Di Francesco, P. Lago, I. Malavolta, "Migrating towards microservice architectures: an industrial survey", Proceedings of the IEEE International Conference on Software Architecture, Seattle, WA, USA, 30 April - 4 May 2018, pp. 29-2909.

[18] N. Dragoni, S. Dustdar, S. T. Larsen, M. Mazzara, "Microservices: Migration of a mission critical system", 2017, arXiv:1704.04173.

[19] C. Y. Fan, S. P. Ma, "Migrating monolithic mobile application to microservice architecture: An experiment report", Proceedings of the IEEE International Conference on AI & Mobile Services, Honolulu, HI, USA, 25-30 June 2017, pp. 109-112.

[20] G. Fletcher, M. Griffiths, "Digital Transformation during a lockdown", International Journal of Information Management, Vol. 55, 2020, 102185.

[21] J. Fritzsch, J. Bogner, S. Wagner, A. Zimmermann, "Microservices migration in industry: intentions, strategies, and challenges", Proceedings of the IEEE International Conference on Software Maintenance and Evolution, Cleveland, OH, USA, 29 September - 4 October 2019 pp. 481-490.

[22] A. Furda, C. Fidge, O. Zimmermann, W. Kelly, A. Barros, "Migrating enterprise legacy source code to microservices: on multitenancy, statefulness, and data consistency:, IEEE Software, Vol. 35, No. 3, 2017, pp. 63-72.

[23] S. B. Gomes, F. M. Santoro, M. M. Da Silva, M. E. Iacob, "A Reference Model for Digital Transformation and Innovation", Proceedings of the IEEE 23rd International Enterprise Distributed Object Computing Conference, Paris, France, 28-31 October 2019 pp. 21-30.

[24] J. P. Gouigoux, D. Tamzalit, "From monolith to microservices: Lessons learned on an industrial migration to a web oriented architecture", Proceedings of the IEEE International Conference on Software Architecture Workshops, Gothenburg, Sweden, 5-7 April 2017, pp. 62-65.

[25] J. P. Gouigoux, D. Tamzalit, "Functional-First Recommendations for Beneficial Microservices Migration and Integration Lessons Learned from an Industrial Experience", Proceedings of the IEEE International Conference on Software Architecture Companion, Hamburg, Germany, 25-26 March 2019, pp. 182-186.

[26] B. Jambunathan, K. Yoganathan, "Architecture Decision on using Microservices or Serverless Functions with Containers", Proceedings International Conference on Current Trends towards Converging Technologies, Coimbatore, India, 1-3 March 2018, pp. 1-7.

[27] A. Janes, B. Russo, "Automatic performance monitoring and regression testing during the transition from monolith to microservices", Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops, Berlin, Germany, 27-30 October 2019, pp. 163-168.

[28] M. Kamimura, K. Yano, T. Hatano, A. Matsuo, "Extracting Candidates of Microservices from Monolithic Application Code", Proceedings of the 25th Asia-Pacific Software Engineering Conference, Nara, Japan, 4-7 December 2018, pp. 571-580.

[29] J. Kazanavičius, D. Mažeika, "Migrating legacy software to microservices architecture", Proceedings of the Open Conference of Electrical, Electronic and Information Sciences, Vilnius, Lithuania, 25 April 2019, pp. 1-5.

[30] H. Knoche, W. Hasselbring, "Using microservices for legacy software modernization", IEEE Software, Vol. 35, No. 3, 2018, pp. 44-49.

[31] A. Krause, C. Zirkelbach, W. Hasselbring, S. Lenga, D. Kröger, "Microservice decomposition via static and dynamic analysis of the monolith", Proceedings of the IEEE International Conference on Software Architecture Companion, Salvador, Brazil, 16-20 March 2020, pp. 9-16.

[32] W. Lloyd, M. Vu, B. Zhang, O. David, G. Leavesley, "Improving application migration to serverless computing platforms: Latency mitigation with keep-alive workloads", Proceedings of the IEEE/ACM International Conference on Utility and Cloud Computing Companion, Zurich, Switzerland, 17-20 December 2018, pp. 195-200.

[33] G. Mazlami, J. Cito, P. Leitner, "Extraction of microservices from monolithic software architectures", Proceedings of the IEEE International Conference on Web Services, Honolulu, HI, USA, 25-30 June 2017, pp. 524-531.

[34] A. A. Pflaum, P. Gölzer, "The IoT and digital Transformation: toward the data-driven enterprise", IEEE Pervasive Computing, Vol. 17, No. 1, 2018, pp. 87-91.

[35] F. Ponce, G. Márquez, H. Astudillo, "Migrating from monolithic architecture to microservices: A Rapid Review", Proceedings of the 38th International Conference of the Chilean Computer Science Society, Concepcion, Chile, 4-9 November 2019, pp. 1-7.

[36] D. Richter, M. Konrad, K. Utecht, A. Polze, "Highly-available applications on unreliable infrastructure: Microservice architectures in practice", Proceedings of the IEEE International Conference on Software Quality, Reliability and Security Companion, Prague, Czech Republic, 25-29 July 2017, pp. 130-137.

[37] N. Santos, A. R. Silva, "A complexity metric for microservices architecture migration", Proceedings of the IEEE International Conference on Software Architecture, Salvador, Brazil, 16-20 March 2020, pp. 169-178.

[38] D. Taibi, V. Lenarduzzi, C. Pahl, "Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation", IEEE Cloud Computing, Vol. 4, No. 5, 2017, pp. 22-32.

[39] M. Waseem, P. Liang, "Microservices architecture in DevOps",. Proceedings of the 24th Asia-Pacific Software Engineering Conference Workshops, Nanjing, China, 4-8 December 2017, pp. 13-14.

[40] S. Yucel, "Modeling Digital Transformation Strategy", Proceedings of the International Conference on Computational Science and Computational Intelligence, Las Vegas, NV, USA, 12-14 December 2018, pp. 221-226.

[41] P. Yugopuspito, F. Panduwinata, S. Sutrisno, "Microservices architecture: Case on the migration of reservation-based parking system", Proceedings of the IEEE 17th International Conference on Communication Technology, Chengdu, China, 27-30 October 2017, pp. 1827-1831.

[42] C. Zirkelbach, A. Krause, W. Hasselbring, "The collaborative modularization and reengineering approach CORAL for open source research software", International Journal On Advances in Software, Vol. 13, No. 1&2, 2020, pp. 34-4