


Review

A Systematic Mapping Study on the Modernization of Legacy Systems to Microservice Architecture

Lucas Fernando Fávero, Nathalia Rodrigues de Almeida  and Frank José Affonso * 

Department of Statistics, Applied Mathematics and Computation (DEMAC), São Paulo State University (UNESP), 24A Avenue, 1515, Rio Claro 13506-900, SP, Brazil; lucas.favero@unesp.br (L.F.F.); nr.almeida@unesp.br (N.R.d.A.)

* Correspondence: f.affonso@unesp.br

Abstract

Microservice architecture (MSA) has garnered attention in various software communities because of its significant advantages. Organizations have also prioritized migrating their legacy systems to MSA, seeking to gather the intrinsic advantages of this architectural style. Despite the importance of this architectural style, there is a lack of comprehensive studies in the literature on the modernization of legacy systems to MSA. Thus, the principal objective of this article is to present a comprehensive overview of this research theme through a mixed-method investigation composed of a systematic mapping study based on 43 studies and an empirical evaluation by industry practitioners. From these, a taxonomy for the initiatives identified in the literature is established, along with the application domain for which such initiatives were designed, the methods used to evaluate these initiatives, the main quality attributes identified in our investigation, and the main activities employed in the design of such initiatives. As a result, this article delineates a process of modernization based on six macro-activities, designed to facilitate the transition from legacy systems to microservice-based ones. Finally, this article presents a discussion of the results based on the evidence gathered during our investigation, which may serve as a source of inspiration for the design of new initiatives to support software modernization.



Academic Editor: Luca Mainetti

Received: 31 March 2025

Revised: 4 June 2025

Accepted: 11 June 2025

Published: 20 June 2025

Citation: Fávero, L.F.; Almeida, N.R.d.; Affonso, F.J. A Systematic Mapping Study on the Modernization of Legacy Systems to Microservice Architecture. *Appl. Syst. Innov.* **2025**, *8*, 86. <https://doi.org/10.3390/asi8040086>

Copyright: © 2025 by the authors. Published by MDPI on behalf of the International Institute of Knowledge Innovation and Invention. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: legacy system; microservice architecture; modernization process

1. Introduction

Microservice Architecture (MSA) has acquired considerable traction in recent years as a means of developing complex applications [1,2]. This architectural style enables software to be organized into small, modular, and independently deployed services, where each service runs in its own process and communicates through lightweight, well-defined mechanisms to serve business objectives [3]. Because of this organization, cost reduction, improved quality in the final product, delivery agility, decentralized governance, independence of technological stacks, and reduced time to market can be cited as driving factors of the architecture [4].

While numerous advantages have been identified in the literature, the adoption of MSA has been perceived as a significant challenge, both in developing new systems and modernizing existing ones [5]. Regarding modernization, the study conducted by Taibi et al. [1] revealed that organizations were reluctant to adopt MSA. This unwillingness was attributed to two factors, namely, (i) the architectural style being perceived as a mere “hype” and (ii) a lack of awareness regarding the modernization of their systems with the implementation of this architectural style. Furthermore, there was a noticeable

deficiency in understanding the complexities and advantages associated with the process of modernization. Evidence from the study by Hassan et al. [4] showed that the transition to microservices is not merely a technical decision; it requires alignment with the business objectives for which the system was originally designed. Consequently, precipitated technical decisions may lead to very aggressive decomposition that favors autonomy without considering the business impact, which can often be negative.

As this study aims to engage with initiatives related to the modernization of legacy systems based on MSA, it is imperative to gather evidence regarding the definition of such systems. As stated by Di Francesco et al. [5], legacy systems present technical and business challenges. Technical challenges include coupling and maintenance costs, while business challenges include release times and low productivity. These issues arise from the fact that legacy systems were developed with outdated computing resources, which are no longer suitable for the current computing landscape. As defined by Kalske et al. [6], legacy systems are characterized by a significant increase in complexity, making them challenging to modify, scale, and maintain. For instance, organizations with high traffic, a substantial number of developers, and a considerable code base were factors observed by the aforementioned authors that stimulated the interest in migrating legacy systems to MSA. In this direction, Almeida et al. [7] conducted a tertiary study on the modernization of legacy systems based on microservices based on 20 secondary studies. In addition to the interest in modernization (e.g., motivations, faced challenges before and after modernization, and patterns), the study revealed a lack of clarity regarding the distinction between legacy systems and monolithic systems. For instance, a legacy system may be a monolithic system, but the reverse is not necessarily true. Consequently, in this study, we have chosen not to limit legacy systems to a specific type of architecture (e.g., monolithic) or technology stack.

As previously stated, the main purpose of the investigation conducted by Almeida et al. [7] was to contribute to the research topic addressed in this article. In summary, the investigation yielded evidence on a number of pertinent issues, including (i) the motivation behind organizations' decision to modernize their systems with the implementation of MSA, (ii) the challenges encountered during the modernization process, (iii) the anticipated benefits of modernization, (iv) the patterns adopted for modernization, and (v) the identification of the main step in the modernization process. Regarding the last item, the investigation has revealed that there is still much to be understood about the transition process from legacy systems to MSA. Therefore, further investigation is required to gain a deeper understanding of this topic.

A comparison between the challenges and the modernization process reveals a dearth of methodological understanding and the requisite infrastructure to support the activities inherent in such a process [7]. In order to address this issue, we present in this article a comprehensive overview of the modernization of legacy systems based on MSA, focusing on the process activities. This overview is based on a mixed-method investigation composed of a systematic mapping study (SMS) and an empirical evaluation by industry practitioners. In short, the SMS is a technique that enables researchers to conduct a complete and fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology with a well-defined strategy [8]. In order to establish a comprehensive and state-of-the-art overview, a total of 43 studies were selected for examination in this SMS. The objective of our investigation is to provide insights into the application domains, evaluation strategies, maturity level, quality attributes, and key activities identified in the design of the initiatives concerning the modernization of legacy systems based on MSA. As a result, a modernization process comprising six macro-activities was formulated, along with a set of steps for each activity. In the subsequent stage of our study, we submitted the SMS-derived modernization process to software industry practitioners from two companies for evaluation

of the activities and steps associated with each activity. Next, we present a synthesis of the findings from our investigation for the benefit of stakeholders (i.e., researchers and practitioners):

- A detailed and solid panorama of initiatives related to the modernization of legacy systems based on MSA. This panorama enables us to understand the following aspects:
 - A taxonomy of the initiatives identified in the literature. In light of our investigation, this taxonomy compiles the principal evidence on the initiatives identified in the literature. In summary, this taxonomy can serve as a theoretical framework and a source of valuable insights, thereby facilitating the design of new initiatives.
 - The application domains that needed to modernize their systems based on MSA. The evidence presented here can provide insights related to the features that are required by such initiatives, which may vary depending on the domain or subdomain in question.
 - The correlation between the evaluation strategies employed by these initiatives and their level of maturity. Our study found that more rigorous evaluation tends to improve the credibility of an initiative and increase its adoption by the organizations.
 - The most common quality attributes. The results of our investigation revealed the advantages of implementing monitoring systems for modernized software, specifically in the context of microservices.
- The main activities related to modernization. The discovered process has the potential to serve as a guideline framework for guiding different stakeholders in the activity of modernizing a legacy system with the implementation of MSA, taking into account the diverse organizational structures and technological architectures that may be involved. Furthermore, our findings show that the proposed process has the potential to be customized and instantiated to serve other application domains. This is due to the comprehensive nature of the identified activities and steps, which can be reused or adapted to act on software systems of neighboring domains. In this regard, it is noteworthy to highlight Micro4Dephi [9], a process initiated under the guidelines established in our investigation (i.e., the SMS) with the active involvement of practitioners from the software industry who have expertise in the Delphi development environment.

The remainder of this article is organized as follows: Section 2 provides an introduction to the concepts of microservices and software modernization, as well as an overview of related work. Section 3 addresses the systematic process used in our investigation, and Section 4 reports the main findings of our SMS. Section 5 discusses the main activities related to software modernization that can be used to build new initiatives and the most commonly discovered steps for designing such initiatives. Finally, Section 6 concludes this article and describes some perspectives for future work.

2. Background and Related Work

This section provides an overview of the background and related work that contributed to the development of our study. Initially, we report on the concepts of microservices and software modernization. We then address related work on the modernization of legacy systems based on microservices.

According to Lewis and Fowler [10], the term “Microservice Architecture”, or MSA, has been used to refer to a particular way of designing software applications as suites of independently deployable services. In short, the aforementioned authors presented a definition that researchers and practitioners have used in the literature to describe this architectural style as “an approach to developing a single application as a suite of small services,

each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API (Application Programming Interface). These services are built around business capabilities and independently deployable by fully automated deployment machinery. There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies”.

Drawing a parallel with other existing definitions in the literature on microservices, Tserpes [11] mentioned that the term “microservices” refers to a software architecture concept associated with the decomposition of an application into smaller, independent functions that run in the infrastructure resources. Newman [12] reinforces this idea by describing that microservices should be small and focused on performing a task efficiently and autonomously. As defined by IBM [13], microservices represent an architectural style in which large and complex applications are composed of one or more microservices. Finally, as outlined by Amundsen and McLarty [14], microservices encompass a multitude of concepts that have emerged from software engineering best practices and the agile developer community and are strongly associated with the DevOps approach (Development and Operations). Based on the concepts reported in this section, it can be said that microservice architectures facilitate scalability and boost the development of solutions, thereby fostering innovation and reducing the time required to introduce new features into the market.

Modernization can be defined as a transformation process in which the software system is adapted or refactored to meet new user requirements and adapt to new market technologies. In this direction, MSA offers several advantages, including a maintenance facility, faster time to market, and high scalability. These benefits are particularly sought after in systems that are considered legacy or monolithic, as evidenced by Newman [12]. Despite the existence of various approaches related to modernization, the feasibility of developing a microservice-based application from scratch remains uncertain, particularly due to the constraints of cost and time. While there are initiatives in the literature, the decomposition of a system into microservices remains a challenging task due to the lack of a comprehensive guide that covers the entire process. This is because of the necessity of several iterations to identify suitable microservice limits [15].

Almeida et al. [7] conducted a tertiary study on the modernization of legacy systems with the implementation of MSA. Of the 20 secondary studies selected for analysis in this tertiary study, none of them addressed the research interest reported in this article. Based on the key findings of the above tertiary study, it can be said that this research topic is still emerging and is of common interest in both academic and industry segments. Moreover, by synthesizing insights from various secondary studies, the aforementioned study provides a comprehensive overview of motivations, the main architectural patterns organized in a taxonomy, the faced challenges, the expected benefits, and the macro-steps to conduct modernization. With regard to the modernization of legacy systems based on MSA, the investigation revealed that there is still much to be understood about the process as a whole and not as isolated steps. Therefore, the authors reported that further investigation is required to gather further evidence for a deeper understanding of this research topic.

As far as we are aware, no secondary study has addressed the modernization of legacy systems based on MSA with a research interest focused on the activities of the process as a whole. To ensure the originality of our investigation, we combined the terms “secondary study” and “microservices” with their synonyms and executed the search string in the same publication databases used in our study. As presented in Section 3, the primary studies available in the literature address different aspects of the investigation, leaving gaps related to the modernization process itself, its activities, and its organizational particularities.

3. Planning and Conducting the Mapping

In order to gather an overview of the modernization of legacy systems based on MSA, a mixed-method investigation was adopted. Initially, we conducted a mapping by using the SMS technique, strictly following the guidelines proposed by Petersen et al. [8]. Moreover, it should be highlighted that our research group has successfully applied these guidelines in previous investigations [7,9,16]. With regard to the period, our investigation was conducted from August 2023 to December 2024 by software engineering researchers and software development practitioners. The following sections, Sections 3.1–3.3, address details of our SMS.

3.1. Planning

This section outlines the research protocol for our mapping, detailing the objectives, research questions, selection criteria, and processes for extracting and synthesizing data.

3.1.1. Objectives and Research Questions

The objective of this study is to provide a comprehensive understanding of the modernization of legacy systems based on MSA. To address the aforementioned objective, a set of six distinct research questions (RQs) were formulated. Each RQ is associated with a specific objective and a well-established set of performance metrics. These efforts are directed towards the organization and synthesis of the knowledge dispersed throughout the extant literature, with a particular focus on existing initiatives. Next, the questions of our mapping are presented:

- RQ1:** What initiatives have been proposed to support the modernization of legacy systems based on MSA? The purpose of this question is to gather evidence regarding the modernization initiatives proposed in the primary studies selected in our SMS. As a result, this question aims to present an overview concerning the classification of these initiatives (e.g., approaches, processes, or methodologies), so that new initiatives can be proposed.
- RQ2:** What application domains have benefited from having this initiative type? This question provides evidence of application domains that have benefited from such initiatives. The existence of such evidence can facilitate a deeper comprehension of the necessity for these applications and the features they require, which can vary according to each domain or subdomain.
- RQ3:** How were these initiatives evaluated? The objective of this question is to provide evidence of the validation procedures adopted in these initiatives, such as proof of concepts, case studies, and experiments. Thus, a comprehensive overview of the rigor and efficacy of these initiatives can be obtained.
- RQ4:** What evidence motivates the adoption of these solutions? This question summarizes evidence of the maturity of these initiatives. The information from this question can provide insights concerning the nature of these initiatives (e.g., academic, industrial, or mixed). Thus, an overview of the maturity of these initiatives among practitioners and researchers can be outlined.
- RQ5:** What are the quality attributes addressed by these initiatives? The goal of this question is to identify which quality attributes have been used in a microservice-based application during modernization.
- RQ6:** What activities are conducted by these initiatives? This question aims to provide evidence of activities used to modernize a legacy system based on MSA.

To address the RQs that were formulated in this section, a search string was elaborated based on two key terms: modernization and microservice. These terms were linked to their synonyms by using the Boolean operator OR to form the main terms, which were combined by the Boolean operator AND. Specialists from different areas (i.e., systematic review, software engineering, and service-oriented systems) evaluated our search string in order to increase its reliability. Table 1 presents the search string utilized in our SMS.

Table 1. Search string.

("decomposing" OR "decoupling" OR "migrating" OR "migration" OR "modernization" OR "modernizing" OR "re engineering" OR "re-engineering" OR "reengineering" OR "reverse engineering" OR "transition") AND ("micro service" OR "micro services" OR "micro-service" OR "micro-services" OR "microservice" OR "microservices")

Regarding the publication databases, we followed particular criteria proposed by Dyba et al. [17], Kitchenham et al. [18], and Petersen et al. [8], namely, (i) content update (i.e., how frequently publications are updated), (ii) the availability of full-text articles, (iii) the quality of results (i.e., the accuracy of the results returned by the search), and (iv) versatility to export the results (i.e., since a search typically results in a sizable volume of data, a mechanism to export the results is required). The ACM Digital Library (<https://dl.acm.org>, accessed on 11 August 2023), IEEE Xplore (<https://ieeexplore.ieee.org>, accessed on 11 August 2023), ScienceDirect (<https://sciencedirect.com>, accessed on 11 August 2023), and Scopus (<https://scopus.com>, accessed on 11 August 2023) represent an efficient set of databases, namely, the most used in computer science and widely used in the context of software engineering.

3.1.2. Selection Criteria

To ensure the quality and relevance of the studies included in our SMS, a set of selection criteria was elaborated. For a study to be considered for inclusion, it must meet one inclusion criterion (IC); in short, the study presents some type of initiative related to the modernization of legacy systems based on MSA. Nevertheless, six exclusion criteria (EC) were defined as follows:

- EC1: The study did not present any initiative that can support the modernization of legacy systems with the implementation of MSA;
- EC2: The study was an editorial, position paper, keynote speech, opinion, tutorial, poster, or panel;
- EC3: The study was not written in English;
- EC4: The study only provided a summary or was not available in full length;
- EC5: The study was similar to previous work developed by the same author. In this case, only the most recent study or the most complete version was considered;
- EC6: The study was a secondary study.

3.1.3. Data Extraction and Synthesis Strategy

In order to capture the key aspects of the studies included in our SMS, we created a form based on the template adapted from Petersen et al. [8], as shown in Table 2. This form aims to gather the information from the studies into a single location, thus facilitating the subsequent data synthesis phase. Consequently, most of the extracted information pertained to one or more research questions (see Section 3.1.1), thereby enabling us to evaluate the extent to which each study addressed the RQs proposed in this mapping. To support this process, the following tools were used: (i) JabRef (<http://www.jabref.org>,

accessed on 11 August 2023) and (ii) spreadsheets from the LibreOffice (<https://www.libreoffice.org>, accessed on 11 August 2023) suite.

Table 2. Data extraction form.

Data Item	Value	RQ
Study ID	Integer	–
Article title	Name of the article	–
Author name	Set of names of the authors	–
Venue	Name of publication venue	–
Year	Calendar year	–
Initiative name	Approach, methodology, process, etc.	RQ1
Application domain	Domain in which the initiative was applied	RQ2
Evaluation method	Proof of concepts, case studies, experiments, etc.	RQ3
Adoption evidence	Maturity level (industrial, academic, or mixed)	RQ4
Quality attributes	List of reported quality attributes	RQ5
Activities	Steps for modernization	RQ6

3.2. Search Process

This section outlines the methodology employed in conducting our SMS, which is based on the protocol described in Section 3.1. As shown in Table 3, Step 1 represents the number of studies resulting from each publication database, amounting to 1.037 studies. Step 2 involves unifying the studies into a single file and removing any duplicates and studies deemed non-primary, which yielded 639 studies. In Step 3, the titles and abstracts of the studies were read and evaluated according to the selection criteria established in Section 3.1.2, amounting to 98 studies. In cases where it was deemed appropriate, the introduction and conclusion were also read. Finally, Step 4 represents the final number of studies in our SMS.

Table 3. Study steps.

Publication Database	Step 1	Step 2	Step 3	Step 4
ACM Digital Library	83			
IEEE Xplore	268			
ScienceDirect	58	639	98	43
Scopus	628			
Total	1.037			

In order to select the most relevant studies, we applied quality criteria to the final selection of studies (Step 4). Table 4 shows quality criteria and the respective scores based on criteria established by Kitchenham and Charters [19] and Kitchenham et al. [20]. The final list of studies was composed of all studies with a percentage above 80% of the total score, resulting in 43 studies. The spreadsheet containing a preliminary list of studies with the application of the quality criteria and the ranking of the final list can be accessed in Fávoro et al. [21].

Table 4. Quality criteria.

ID	Description	Score
C1	Are research objectives clearly presented?	0: No; 1: Partially; 2: Yes.
C2	Are study limitations discussed?	0: No; 1: Partially; 2: Yes.

Table 4. *Cont.*

ID	Description	Score
C3	Are findings clearly presented?	0: No; 1: Partially; 2: Yes.
C4	Is there evidence that the results of the study can be used by other researchers/practitioners?	0: No; 1: Partially; 2: Yes.
C5	Does the content of the study fully address the questions set out in the abstract?	0: No; 1: Partially; 2: Yes.
C6	Are the proposals presented empirically evaluated?	0: No; 1: Partially; 2: Yes.
C7	How was the study evaluated?	0: Not evaluated; 0.5: Illustrative example; 1: Case study; 1.5: Assessment by researchers and specialists; 2: Application in industry.
C8	Are ideas for future research presented?	0: No; 1: Partially; 2: Yes.

3.3. Threats to Validity

In this section, the threats to the validity of this SMS are presented and discussed. Moreover, we also present the measures we adopted to mitigate (or minimize) such threats, considering the previous experience of our research group [7,9,16]. Threats in the context of this SMS can be classified as the omission of papers, the number of reviewers and reliability, the selection of primary studies, data extraction, and replicability.

Omission of papers. Despite our best efforts to ensure the reliability of this SMS, it is still possible that some primary studies may have been overlooked and not included in the final report. Problems with search strings (e.g., ineffective search strings) and digital libraries (e.g., an automatic engine) are threats related to study selection for studies of this nature (i.e., SMSs). Despite the implementation of evaluation procedures to calibrate and ensure the efficiency of our search string, the reliability of our study is contingent upon the search engines utilized by each publication database. To mitigate potential issues during the search phase, we tailored the general search string to align with the constraints and limitations of each database.

Number of reviewers and reliability. As a single researcher was responsible for conducting this mapping, literature review, software architecture, and MSA specialists provided support to this researcher during the mapping process in order to reduce the potential for bias, besides boosting the impartiality of the results.

Selection of primary studies. The research questions defined in this mapping may not cover the entire area of modernization. Thus, in order to overcome this threat, we carefully selected a set of keywords that align with our RQs. We carefully selected keywords to ensure that the search string aligned perfectly with our research questions. It is noteworthy that our selection criteria were detailed in Section 3.1.2 to provide comprehensive information regarding the data selection and extraction processes, thereby reducing the probability of biased evaluation.

Data extraction. First, as this type of study is conducted by humans, it is hard to ensure that all the relevant primary studies were selected for this SMS or that the selected studies

were adequately analyzed. Second, the extraction process entailed human judgment because of the ambiguity of some data and the analysis or interpretation of multiple studies to address the RQs. When discrepancies arose among the reviewers, we conducted discussions to resolve them, as recommended by Wohlin et al. [22]. Additional sources of information (i.e., technical reports, websites, and manuals) were analyzed to ensure the accuracy of this SMS. To mitigate the impact of this potential threat, we developed a data extraction template (see Table 2) to optimize and standardize the data extraction process.

Replicability. To ensure the replicability of our study, we provided a comprehensive description of the methodology, as detailed in Section 3.1. It is important to note that all data were collected from electronic databases that are constantly updated. Thus, the results of a future search may differ from those presented here. Therefore, it can be argued that replicating this mapping is a challenging undertaking. However, periodic updating may prove beneficial in allowing stakeholders to monitor the evolution of the research area.

4. Results

This section presents the synthesis and analysis of our SMS, according to the research questions established in Section 3. Therefore, it is expected that the findings can provide researchers and practitioners with a comprehensive overview of the modernization of legacy systems with the implementation of MSA and of how this knowledge can be used in the transition to new systems. Table 5 presents a list of 43 studies selected in our mapping. The first column displays the unique identifier for each study, the second column provides the reference, and the third column indicates the year for each study. Each reference includes the author's name(s) in the author–year citation format and the title of the published work. It should be noted that the references provided in this article are available in their entirety in the reference list. Furthermore, to enhance accessibility, all titles have been linked to their corresponding DOI (Digital Object Identifier). The main findings are presented in Sections 4.1–4.6.

Table 5. List of primary studies analyzed in the mapping.

ID	Reference	Year
S1	Buchgeher et al. [23], Adopting Microservices for Industrial Control Systems: A Five Step Migration Path.	2021
S2	Lenarduzzi et al. [24], Does migrating a monolithic system to microservices decrease the technical debt?.	2020
S3	Trabelsi et al. [25], From legacy to microservices: A type-based approach for microservices identification using machine learning and semantic analysis.	2022
S4	Sellami et al. [26], Improving microservices extraction using evolutionary search.	2022
S5	Krause et al. [15], Microservice Decomposition via Static and Dynamic Analysis of the Monolith.	2020
S6	Mazzara et al. [27], Microservices: Migration of a Mission Critical System.	2021
S7	Freire et al. [28], Migrating production monolithic systems to microservices using aspect oriented programming.	2021
S8	Kalia et al. [29], Mono2Micro: A practical and effective tool for decomposing monolithic Java applications to microservices.	2021
S9	Wang et al. [30], Qualitative and quantitative comparison of Spring Cloud and Kubernetes in migrating from a monolithic to a microservice architecture.	2023
S10	Zaragoza et al. [31], Refactoring monolithic object-oriented source code to materialize microservice-oriented architecture.	2021

Table 5. Cont.

ID	Reference	Year
S11	Löhnertz and Oprescu [32], Steinmetz: Toward automatic decomposition of monolithic software into microservices.	2020
S12	Mishra et al. [33], Transition from Monolithic to Microservices Architecture: Need and proposed pipeline.	2022
S13	Dehghani et al. [34], Facilitating the migration to the microservice architecture via model-driven reverse engineering and reinforcement learning.	2022
S14	Osman et al. [35], From Monolith to Microservices: A Semi-Automated Approach for Legacy to Modern Architecture Transition using Static Analysis.	2022
S15	Colanzi et al. [36], Are we speaking the industry language? The practice and literature of modernizing legacy systems with microservices.	2021
S16	Auer et al. [37], From monolithic systems to Microservices: An assessment framework.	2021
S17	Zaragoza et al. [38], Materializing Microservice-oriented Architecture from Monolithic Object-oriented Source Code.	2022
S18	Kyryk et al. [39], Methods and process of service migration from monolithic architecture to microservices.	2022
S19	Stranner et al. [40], Microservice decomposition: A case study of a large industrial software migration in the automotive industry.	2020
S20	Ma et al. [41], Microservice Migration Using Strangler Fig Pattern and Domain-Driven Design.	2022
S21	Li et al. [42], Microservice Migration Using Strangler Fig Pattern: A Case Study on the Green Button System.	2020
S22	Ma et al. [43], Migrating Monoliths to Microservices based on the Analysis of Database Access Requests.	2022
S23	Salii et al. [44], Migrating to a microservice architecture: benefits and challenges.	2023
S24	Carvalho et al. [45], On the Performance and Adoption of Search-Based Microservice Identification with toMicroservices.	2020
S25	Gomes Barbosa and Maia [46], Towards Identifying Microservice Candidates from Business Rules Implemented in Stored Procedures.	2020
S26	Bamberger and Körber [47], Migrating Monoliths to Microservices Integrating Robotic Process Automation into the Migration Approach.	2022
S27	Parikh et al. [48], Monolithic to Microservices Architecture—A Framework for Design and Implementation.	2022
S28	Kuryazov et al. [49], Towards Decomposing Monolithic Applications into Microservices.	2020
S29	Freitas et al. [50], A methodology for refactoring ORM-based monolithic web applications into microservices.	2023
S30	Yang et al. [51], A Microservices Identification Approach based on Problem Frames.	2022
S31	Kazanavičius and Mažeika [52], An Approach to Migrate from Legacy Monolithic Application into Microservice Architecture.	2023
S32	Volynsky et al. [53], Architect: A Framework for the Migration to Microservices.	2022
S33	Laigner et al. [54], From a Monolithic Big Data System to a Microservices Event-Driven Architecture.	2020
S34	Santos and Paula [55], Microservice decomposition and evaluation using dependency graph and silhouette coefficient.	2021

Table 5. *Cont.*

ID	Reference	Year
S35	Prasandy et al. [56], Migrating application from monolith to microservices.	2020
S36	Haugeland et al. [57], Migrating Monoliths to Microservices-based Customizable Multi-tenant Cloud-native Apps.	2021
S37	Goncalves et al. [58], Monolith Modularization towards Microservices: Refactoring and Performance Trade-offs.	2021
S38	Preti et al. [59], Monolithic to Microservices Migration Strategy in Public Safety Secretariat of Mato Grosso.	2021
S39	Vera-Baquero et al. [60], Open Source Software as the Main Driver for Evolving Software Systems Toward a Distributed and Performant E-Commerce Platform: A Zalando Fashion Store Case Study.	2021
S40	Bajaj et al. [61], Partial Migration for Re-architecting a Cloud Native Monolithic Application into Microservices and FaaS.	2020
S41	Michael Ayas et al. [62], The Migration Journey Towards Microservices.	2021
S42	Batista et al. [63], Towards a Multi-Tenant Microservice Architecture: An Industrial Experience.	2022
S43	Bandara and Perera [64], Transforming monolithic systems to microservices—An analysis toolkit for legacy code evaluation.	2020

4.1. Overview of Primary Studies

This section presents results related to the distribution of studies by year and publication venues, as illustrated in Figure 1. Regarding the yearly distribution of the selected studies, the publication period spans from 2020 to 2023. This period encompasses the last three full years (i.e., 2020, 2021, and 2022) and the year 2023 up to the month in which the search string was executed (i.e., August). The data indicate that 12 studies were published in 2020 (27.91%), 13 studies in 2021 (30.23%), 14 studies in 2022 (32.56%), and 4 studies in 2023 (9.30%). The number of studies, excluding the incomplete year, demonstrates an upward trajectory, indicating a growing interest among researchers and practitioners in this research topic. Regarding the publication distribution, four distinct venues were identified: conference, journal, symposium, and workshop. Each of the symposium and workshop venues had only one study identified, which collectively accounted for 2.30% of the total number of studies. Thus, we observe that a significant proportion of the studies (95.40%) was published in conferences and journals, which may be due to the quality criteria applied to select the list of studies in this SMS (see Section 3.2). This concentration may be associated with two factors: (i) the study maturity, which is one of the quality criteria applied during the selection of studies aimed at classifying them according to the evaluation methods used, and (ii) the rigor of the publication venues, which required the presentation of a proposal to modernize systems, followed by some type of evaluation (e.g., a case study).

4.2. Taxonomy of Initiatives

In order to respond to RQ1, an analysis of the initiatives proposed by each primary study was conducted. Of the 43 studies selected for analysis, 27 (i.e., 62.79%) reported the development of approaches for supporting modernization (i.e., S2 to S10, S14, S17, S19 to S22, S24, S26 to S28, S30, S31, S34, S36, and S39 to S42). A total of 16.28% of the studies presented evidence on processes for modernization, including S1, S15, S25, S33, S35, S37, and S38. Additionally, three studies (6.98%) proposed methodologies (i.e., S11, S23, and S29) and frameworks (i.e., S13, S16, and S32). Two studies (4.65%) proposed a method to support software modernization based on MSA, and only one study (2.33%) directed its efforts toward proposing a pipeline for supporting modernization (i.e., S12). Figure 2

illustrates the distribution of these initiatives by type (on the X-axis) and year of publication (Y-axis). Next, a description of each initiative identified in our mapping is addressed [65].

- **Approach.** This initiative is characterized by its theoretical orientation, research methodology, and strategies employed to investigate, analyze, and present results. It offers a structured framework for the selection of suitable methods and provides a rationale for the execution of an activity.
- **Framework.** This initiative encompasses a reusable design, whether in the form of a model or code, that can be refined, specialized, and extended to provide a portion of the overall functionality of numerous applications.
- **Method.** This initiative represents a specific technique used to execute a process, activity, or task. This initiative represents a methodical series of steps to be undertaken in order to complete a process, activity, or task.
- **Methodology.** This initiative can be defined as a systematic approach that is centered on the orderly application of a specific collection of tools, techniques, and guidelines to the design of software.
- **Pipeline.** This initiative represents a software or hardware design technique in which the output of one process serves as the input to a second process, which in turn serves as the input to a third process, and so on. This information flow can frequently occur in parallel within a single cycle time.
- **Process.** This initiative encompasses a series of interrelated or interactive activities that are designed to transform the input(s) into one (or more) anticipated output(s). A process can have multiple starting points, a set of activities that can be a partially ordered sequence, and multiple endpoints, as a workflow specification.

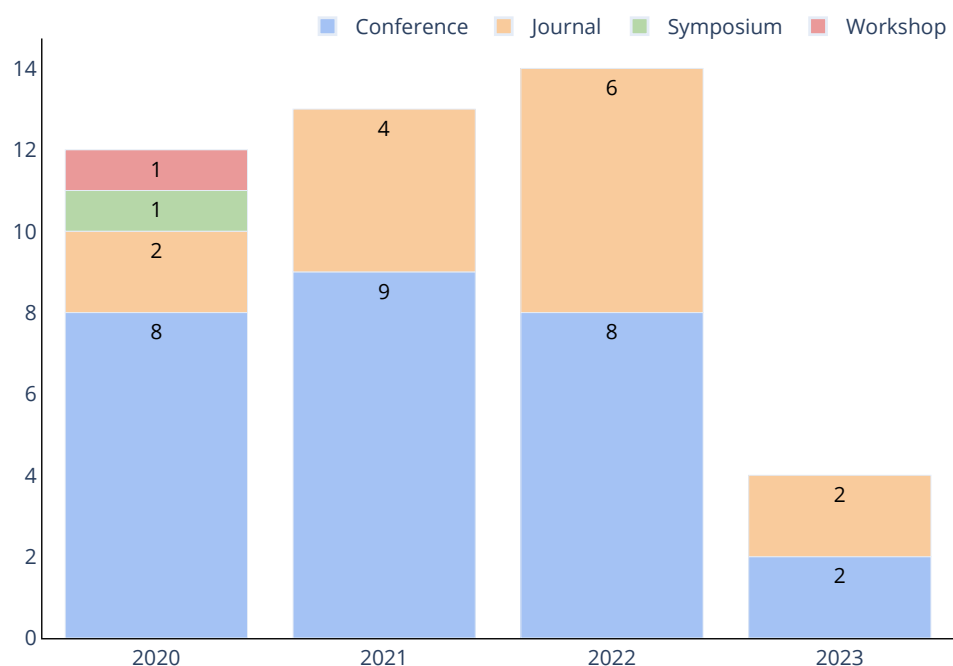


Figure 1. Yearly and venue distribution of studies.

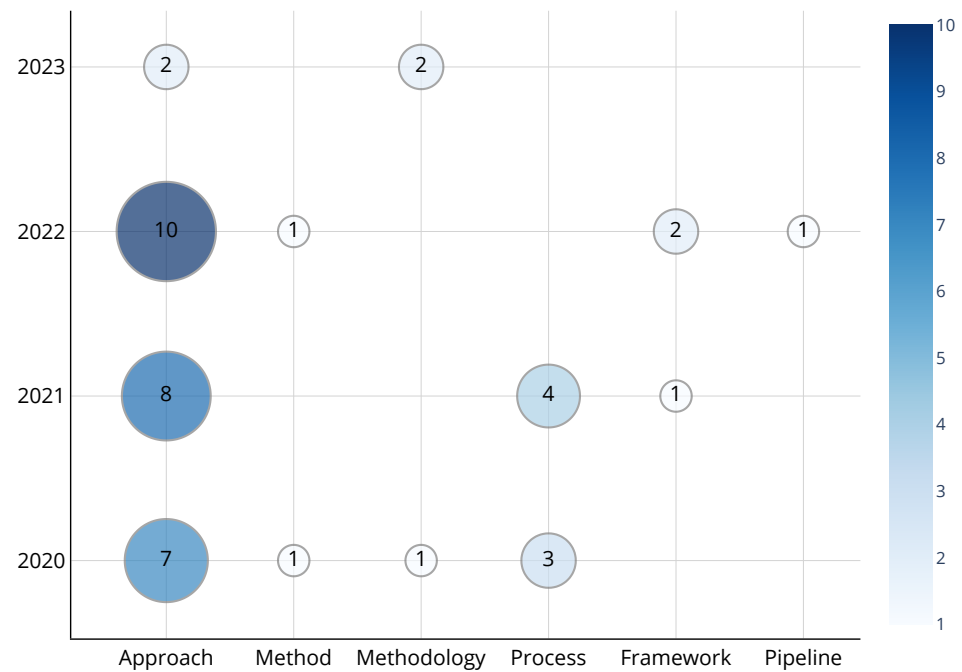


Figure 2. Initiatives' distribution.

By analyzing the distribution of initiative data related to this research question, it is evident that the proposals based on approaches had a significant increase in the years 2021 to 2022, standing out compared with other initiatives. In our investigation, we identified a piece of evidence that supports these statistics—the ambiguity of concepts observed in the studies. For instance, we found that some studies that proposed an approach also included initiatives to automate tasks, intending to boost the system modernization process. In addition, we also observed that many studies combined more than one of the initiatives presented in this section in their proposals. Based on this context, it is clear that the appropriate use of definitions in the development of new initiatives emerges as a viable alternative for their design and dissemination. Interested parties need to have access to accurate and appropriate information about the scope of application of these initiatives, which can promote their adoption by researchers and practitioners.

4.3. Application Domains

To address RQ2, we conducted a comparative analysis of the evaluation methods employed by the primary studies, focusing particularly on the case study approach and its application in different domains. Of the 43 studies selected in our mapping, 33 (i.e., 76.74%) underwent evaluation based on at least one application domain. In contrast, 10 studies (i.e., 23.26%) did not explicitly present an application domain, thereby precluding their classification in our mapping. This was performed in order to avoid introducing any bias (i.e., S15, S16, S18, S23, S26, S28, S31, S35, S41, and S43). As only one area (i.e., Information Systems) was identified in our mapping, Figure 3 illustrates the distribution of the studies in our mapping, considering the area, domain, application domain [65], and quantity.

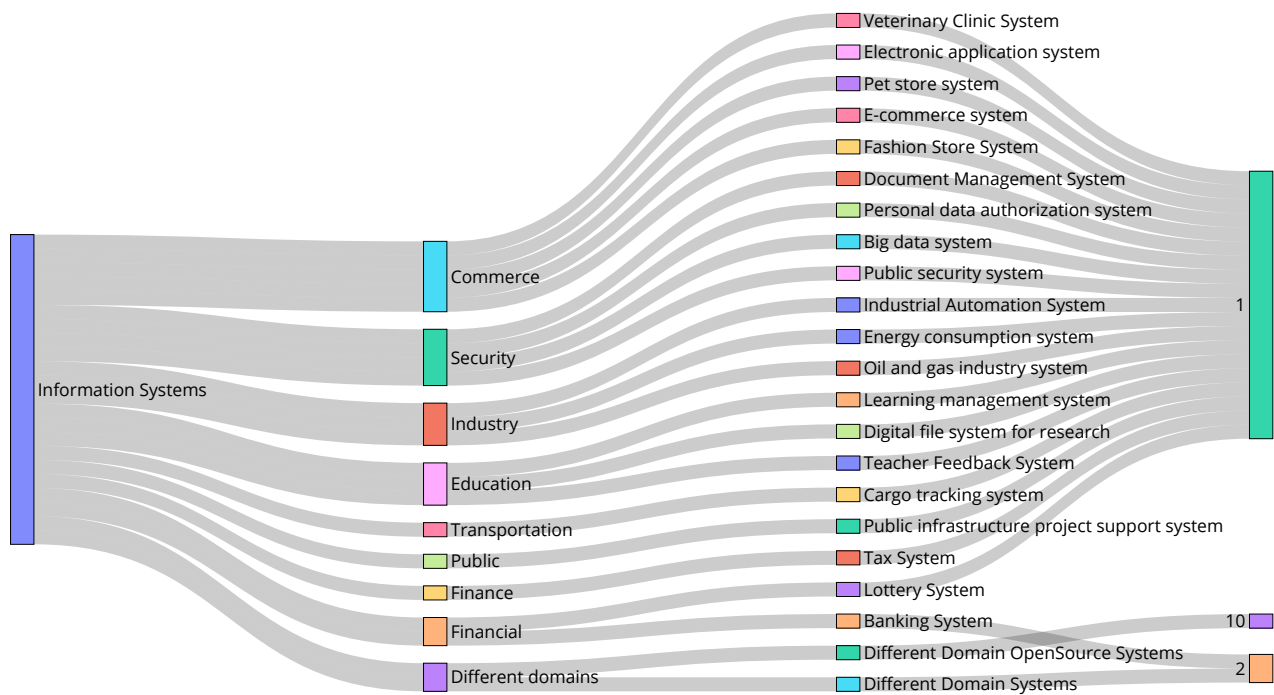


Figure 3. Sankey diagram for area, domain, application, and quantity.

The distribution, presented from left to right, illustrates the “Information Systems” area, the nine identified domains, the application domains, and the number of studies within each domain. The distribution reveals a higher representation of the “Different domains” domain with 12 studies. Among these, 10 studies were categorized as open-source systems (i.e., S3, S4, S10, S11, S14, S17, S19, S22, S29, and S34), while two studies belonged to owner systems (i.e., S7 and S20). We decided to form a group for this domain based on the application domains where accurately inferring the nature of the system used to evaluate the proposed initiative was impossible. In other words, the study provided an evaluation of the proposal that lacked clear contextualization of the system’s application domain. The “Financial” domain and banking systems were explored in two studies (i.e., S6 and S27), with the former being the most prominent real-world domain and the most in line with the study theme of this article. Nevertheless, the limited number of studies in this domain may be attributed to its nature, which values information confidentiality, making it challenging to disseminate results. Another noteworthy aspect of this distribution is the high use of open-source systems to assess the proposed initiatives. As previously stated, 10 studies fell into this category, underscoring the widespread interest among both the scientific communities and practitioners for this system type. Therefore, it can be said that this trend aligns with the principles of the open science movement, facilitating the establishment of transparent processes and fostering the dissemination of knowledge.

4.4. Evaluation Strategies and Maturity Level

The rate of adoption of any scientific innovation (e.g., a solution to support software modernization) by third parties is directly related to the level of maturity of the innovation in question. Among the potential indicators of maturity, our investigation revealed that an understanding of the evaluation process may be a viable parameter. In other words, the use of evaluation strategies facilitates the identification of maturity, performance, and significant features. Therefore, to establish a parallel between the evaluation strategies and the maturity level of the initiatives identified in this SMS, the data gathered from RQ3 and RQ4 was utilized. Of the 43 studies, only 2 did not present any evidence of

evaluation (i.e., S41 and S43). However, all studies were classified according to their respective maturity levels. The analysis of all the primary studies included in our mapping revealed the following evaluation strategies [22]:

- **Case study.** This strategy enables an initiative for software modernization to be analyzed in a scenario closer to a real-world application. Thus, it is possible to identify particularities of an application domain and gather more precise, exploratory, and qualitative data. On the other hand, some studies used a “toy example” as a case study, which can be considered insufficient because of their lack of real-life context;
- **Experiment.** This strategy involves a process with a well-defined sequence of steps based on an execution protocol that enables future replications. In short, experiments are elaborated when it is intended to control the situation and manipulate behavior directly, precisely, and systematically.
- **Proof of concept.** This strategy represents a test or demonstration that aims to define the feasibility of an initiative so that it can be explored usefully. Therefore, it can be said that when successful, this strategy represents an important step in the process of creating a truly operative prototype.

In order to ascertain the maturity level of the proposed initiatives, evidence was gathered regarding their applicability scenarios. The evaluation strategies were then analyzed, and three categories were established to classify a study according to its maturity level: academic, industrial, and mixed. It is noteworthy that this taxonomy has been previously utilized in the research conducted by [7,16]. Next, a brief description of this taxonomy is addressed:

- **Academic.** Studies categorized as academic focus on a theoretical and experimental presentation of the investigated issue. Therefore, when evaluating such studies, experiments, simulations, comparisons with other algorithms, and examples of case studies are considered.
- **Industrial.** Studies that adopt an industry-oriented approach primarily focus on operational aspects. They might not include an in-depth analysis of the adopted resource (e.g., algorithm or technique), as they mainly address the applicability in specific cases directed to industrial cases.
- **Mixed.** This category encompasses studies that combine the benefits derived from both industrial and academic approaches. For instance, a software modernization initiative was formulated considering both academic research and practical applicability in industry.

Figure 4 illustrates the relationship between the studies’ evaluation strategies and their maturity level. Of the 43 studies of this mapping, only 2 presented evidence of evaluation through proof of concepts, with one being conducted in the academic scope and the other being mixed. Three studies, one of which was conducted in an academic context and two of which were mixed, provided evidence of evaluation through experiments. Finally, 36 studies, 23 of which were conducted in the academic scope, 6 in the industrial scope, and 7 in the mixed scope, were evaluated through case studies. This distribution enables us to infer that all parties interested in modernizing legacy systems based on MSA have sought to conduct their activities collaboratively. This is evidenced by 16 studies that presented concrete evidence of the industry’s effective involvement across all evaluation strategies. Therefore, it can be said that this distribution reinforces our evidence of the strong relationship between the evaluation strategy and the level of maturity of an initiative. Therefore, as the robustness of an initiative undergoes the evaluation process, there is a heightened probability of adoption by practitioners in the software industry.

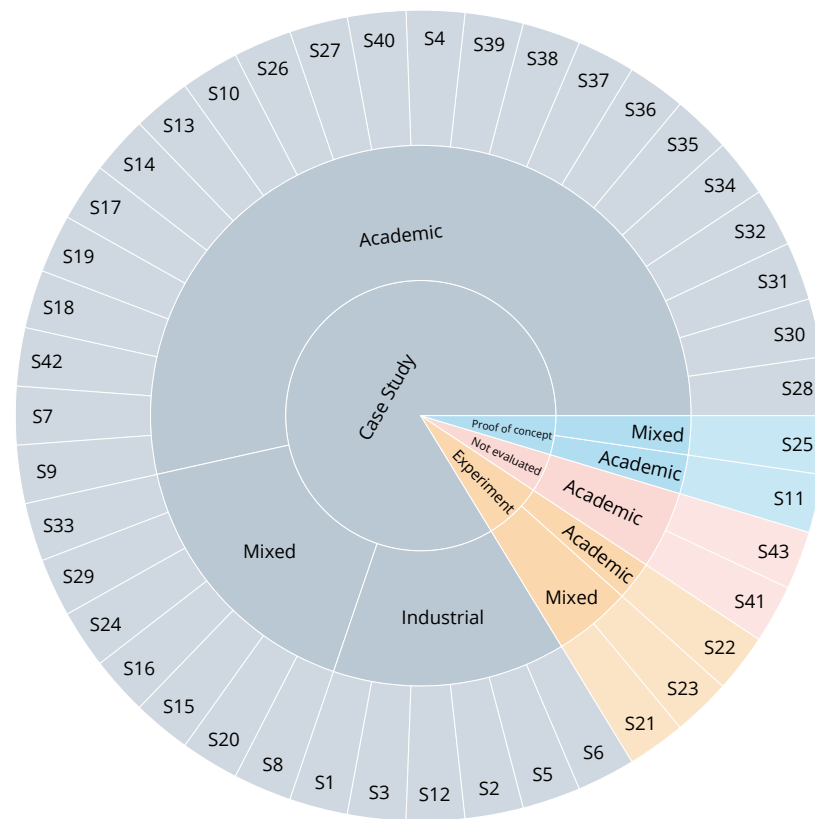


Figure 4. Evaluation strategies versus maturity levels.

4.5. Quality Attributes

Based on the empirical findings compiled by Passini et al. [16] and Almeida et al. [7], quality attributes are recognized as first-class elements in distributed applications. Therefore, the objective of RQ5 is to understand which QoS (Quality of Service) attributes are used in a microservice-based application during modernization. The definition used for this application type encompasses any service-oriented application that communicates by exchanging messages via APIs, such as SOAP (Simple Object Access Protocol) and REST (Representational State Transfer). Although the importance of quality attributes in this type of application is recognized, our investigation revealed that 19 (i.e., 44.19%) of the 43 studies in our SMS did not provide evidence of the adoption of any quality attribute in the proposed initiative (i.e., S2, S11, S12, S14, S17, S18, S22, S25, S26, S28, S29, S30, S31, S34, S35, S36, S37, S38, and S41). Figure 5 illustrates the relationship between the quality attributes identified in our investigation and the respective studies.

A correlation analysis of the studies that provided evidence on the incorporation of quality attributes in the proposed initiatives revealed that the attribute of availability was mentioned in nine studies (i.e., S1, S5, S7, S19, S24, S27, S39, S40, and S43). Our investigation revealed that the attributes of reliability and scalability were the most frequently referenced, occurring in the second and third positions, respectively. The attribute of reliability was identified in six studies (i.e., S20, S21, S32, S33, S39, and S42), while scalability was found in five studies (i.e., S6, S7, S9, S23, and S27). Furthermore, it is notable that several studies indicated the use of multiple quality attributes, which corroborates the initial evidence presented in this section (i.e., the distributed nature of the application following a modernization activity). Notwithstanding the aforementioned evidence, it should be highlighted that only one study (i.e., S7) explicitly stated the inclusion of three quality attributes in its proposed initiative. Finally, ten studies highlighted two attributes, and thirteen, only one quality attribute.

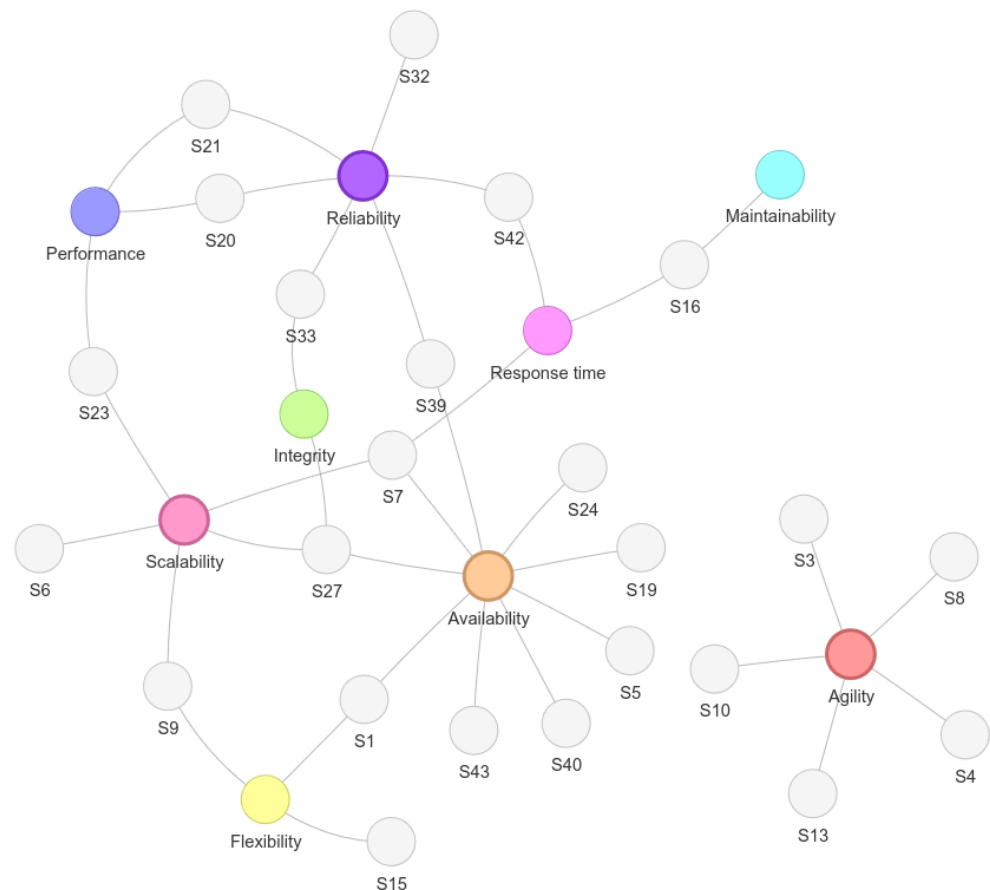


Figure 5. Cloud word for quality attributes.

As outlined in the overview provided by Passini et al. [16], agility is not a quality attribute pertinent to the distributed computing scenario. The use of this term within the context of this study enables us to present two interpretations. First, the term “agility” is being used as a method to meet new market demands rapidly, aligning seamlessly with development strategies centered on microservices. Second, the proponents have failed to utilize the term “agility” effectively, lacking a proper understanding of its concept.

4.6. Main Activities

In response to RQ6, this section provides an overview of the methodological process adopted specifically for this question. As reported in Section 1, our investigation was based on a mixed-method investigation composed of a mapping (SMS) and an empirical evaluation by industry practitioners. Next, a description of this process is addressed.

Regarding the mapping (SMS), evidence of modernization activities was identified in 40 (93.02%) of the 43 studies included in our SMS. In contrast, no evidence of such activities could be identified in three studies (6.98%), namely, S9, S39, and S42. The evidence extracted from these studies indicates that modernization activities are organized into a process comprising six activities, namely, planning, analysis, decomposition, development, integration, and monitoring. Moreover, this mapping also revealed that each activity consists of a set of steps that must be followed in order to achieve its objective.

Table 6 presents the distribution of the most significant studies for this research question, which included three or more activities in their proposed initiatives (see Section 4.2). The data from this distribution reveals the recurrence of the activities in the last row and the number of activities covered by each study in the last column. Notably, S21 is more representative of this distribution by addressing five activities related to the modernization

of legacy systems with the implementation of MSA. Concerning the extent to which studies adhered to the activities identified in our investigation, it is notable that none of the studies addressed all six activities. In this direction, only S21 covered five activities, four studies (i.e., S18, S20, S27, and S31) presented evidence of four activities, and ten studies (i.e., S6, S7, S13, S15, S16, S26, S28, S35, S41, and S34) reported addressing three activities. Therefore, it can be inferred that 25 studies included only one or two activities.

Table 6. Comparative distribution between the most recurrent studies and the activities identified in our investigation. The symbols (✓) and (–) show whether a study meets or does not meet an activity. PLA, ANA, DEC, DEV, INT, and MON are acronyms to the activities identified in our investigation (i.e., planning, analysis, decomposition, development, integration, and monitoring).

#	PLA	ANA	DEC	DEV	INT	MON	
S6	✓	✓	–	✓	–	–	3
S7	–	✓	–	✓	–	✓	3
S13	–	✓	–	✓	✓	–	3
S15	✓	–	–	✓	–	✓	3
S16	✓	✓	–	✓	–	–	3
S18	–	✓	✓	✓	✓	–	4
S20	–	✓	–	✓	✓	✓	4
S21	✓	✓	–	✓	✓	✓	5
S26	–	✓	–	✓	✓	–	3
S27	–	✓	✓	✓	✓	–	4
S28	–	✓	–	✓	✓	–	3
S31	–	✓	✓	✓	✓	–	4
S35	–	✓	–	✓	✓	–	3
S41	–	✓	–	✓	✓	–	3
S43	–	✓	–	✓	✓	–	3
	4	14	3	15	11	4	

The evidence of this distribution (see Table 6) also revealed that studies of this distribution concentrated their efforts on development (15 occurrences), analysis (14 occurrences), and integration (11 occurrences) activities, respectively. Of the 43 studies included in our mapping, 11 were exclusively dedicated to the analysis of the legacy system and the subsequent development and integration of microservices. Despite the fewer occurrences, activities like planning, decomposition, and monitoring were identified as crucial to the modernization process. These activities facilitate the effective transition of stakeholders, comprising software engineers and domain specialists, from a legacy system to MSA.

Based on the findings and insights provided by the scientific analysis, we proceeded to develop the proposed modernization process based on a preliminary step and three interactions with software industry practitioners. As a preliminary step, the first version of the process was formulated after the identification of the essential activities inherent in a modernization process. Each activity is delineated by a series of steps that vary in number and content, ensuring that its objectives are executed in an appropriate manner. For comprehensive guidance on the modernization process, all activities and steps were systematically documented. Regarding the conducting of the activity process, the following points should be highlighted:

- **Cyclical nature.** The cyclical nature of software development has been a subject of considerable interest in modern software engineering. According to the principles outlined by Pressman [66], incremental processes tend to be more effectively managed and monitored by development teams and stakeholders who have a vested interest in the outcome. Furthermore, the evidence gathered in this SMS reveals that the gradual advancement of modernization also favors the transition between the legacy system and the modernized system, allowing them to act concurrently.

- Starting point. Since the process aims to support the modernization of legacy systems based on MSA, the starting point for conducting the process can be (i) a binary system, (ii) a binary system and its respective source code repository, or (iii) the content of the second item with the input of documentation artifacts and manuals, among others. The time required to conduct the process can be significantly reduced, depending on the value of the legacy system and the availability of artifacts. The rationale underlying this approach is that such artifacts can be reused in the development of microservices.

First interaction. Concerning the collaboration of industry practitioners, the researchers presented the first version of the process to the participants from the first company, emphasizing each activity and its respective steps. The documented process was subsequently provided to the practitioners for review, accompanied by a questionnaire comprising 60 questions. This questionnaire was structured into three sections: the first focused on the identification of parameters related to the practitioners' profile (e.g., position, experience), the second centered on modernization interests, and the third focused on each process' activity.

After collecting the information, the researchers analyzed the responses related to the modernization process, and it was noted that 15 steps presented problems in terms of understanding. In order to solve such problems, the researchers updated the process documentation in two stages, namely, (i) refining the process documentation without changing the number of activities and steps and (ii) defining inputs and outputs for each activity and their respective steps. The evidence gathered suggests that the last update can be a significant differentiator for comprehending and applying the process in a real software development environment. Based on the exposed context, the literature survey revealed a set of activities and steps that are consistent with the daily professional routine of a software company. While the initial findings indicated the necessity for process modifications, specifically the removal or consolidation of steps, the researchers chose to maintain the process in its original form and replicate the procedure in a second company.

Second interaction. Following the refinement of the process documentation, the process was submitted for analysis to a second company, with the participation of four professionals with similar levels of knowledge. The same procedure described in the previous interaction was adopted, resulting in fewer questions or a lack of understanding regarding the steps (i.e., four) to be taken for the modernization activity to be carried out. After analyzing the responses and the evidence gathered, the researchers maintained the original number of steps in each activity. Several factors guided this decision, including reports of experience from academia, the size of the teams that participated in the interactions, and the level of knowledge and professional experience of the participating professionals.

Third interaction. After compiling the evidence provided by practitioners from both companies, the researchers made a decision to refine the modernization process. It is noteworthy that this refinement preserved the fundamental activities and steps identified through research in the extant literature. In this manner, while maintaining the integrity of the original process (first version), some steps were reorganized and/or grouped to optimize the work of the teams responsible for carrying out the modernization. Following this refinement, the process underwent a reduction in steps from 38 to 29, which remained distributed throughout the set of the identified activities, as illustrated in Figure 6. Next, a description of the modernization activities identified in our SMS is addressed, offering a synthesis of the steps within each activity.

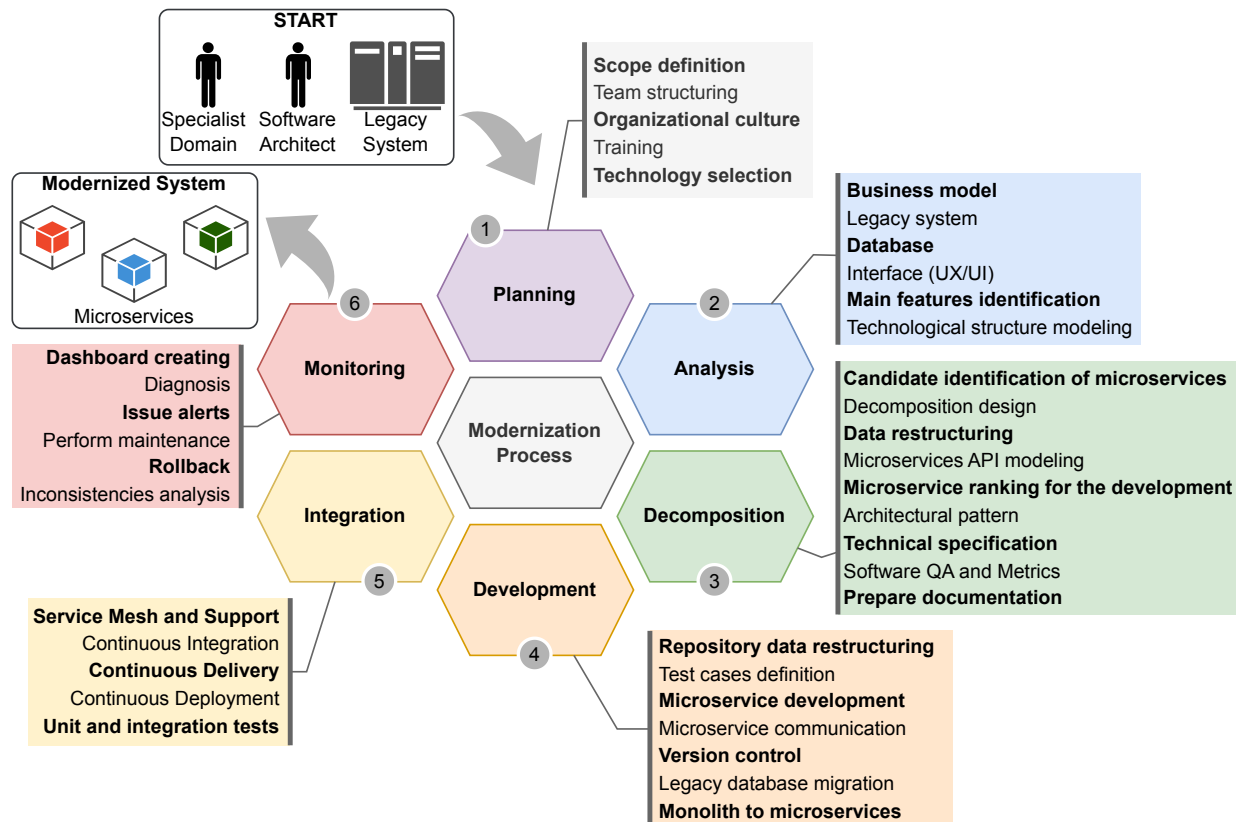


Figure 6. Modernization process.

- Planning.** This activity was identified in five studies included in our mapping (i.e., S6, S15, S16, S21, and S33). In essence, planning entails the definition of the objectives, strategies, and resources required to attain the desired outcome, which, in this case, is the modernization of the system. The evidence obtained from our investigation indicates that when planning modernization activities, organizations and development teams are more likely to succeed if they anticipate potential challenges and ensure that objectives are achieved within the specified timeframe and resources. From the perspective of the organization, the evidence suggests that a suitable methodology and appropriate infrastructure must be adopted to conduct modernization practices. With regard to development, the findings of our mapping recommend that teams have cultures around development (e.g., unit testing). In this sense, training can be an effective measure to align the interests of organizations with development practices and facilitate the development and modernization of applications based on microservices.
- Analysis.** Of the 43 studies included in our mapping, 35 (i.e., S2 to S8, S10 to S13, S16, S18 to S38, S40, S41, and S43) presented concrete evidence related to this activity. In other words, the analysis activity can be considered the initial phase of modernization, as it is during this process that the development team will gain insights into the artifacts of the legacy system, including the binary system, source code repository, and documentation, which can serve as a source of information. Consequently, it is anticipated that a sufficient quantity of information will be amassed to facilitate the formulation of an appropriate starting point for the decomposition process. This encompasses the identification of the disparate components and functionalities of the legacy system, an understanding of its business domain boundaries, and an analysis

of the dependencies between the aforementioned components. In summary, this step is of paramount importance to the successful migration of the legacy system to microservices, as it establishes the foundation for the design and implementation of this architectural style.

- **Decomposition.** This activity was addressed in five studies (i.e., S3 to S5, S18, S27, and S31) of our mapping. This activity plays a fundamental role in the modernization of legacy systems based on MSA, as it directly influences the decomposition of the legacy system into microservices, thereby giving rise to a novel architectural configuration for the system. From a design perspective, the evidence gathered from our investigation suggests that each microservice should be designed based on the organizational and business aspects that will be offered to users. Furthermore, the evidence indicates that the primary challenge associated with this activity is determining the optimal size for microservices. This is because the legacy system must be divided into smaller units that present low coupling and high cohesion. In other words, each microservice must be responsible for a delimited context, providing a cohesive set of functionalities with a well-defined scope to meet a specific business capability. Although (semi)automated techniques can be employed to assist with the decomposition into microservices, the primary recommendation is to initially prioritize features with minimal impact or risk for the transition from the legacy system to the modern system. In essence, this process entails the identification of prospective microservices that offer the greatest value and exhibit minimal external dependencies. While there is no singular methodology for microservice decomposition, Domain-Driven Design (DDD) is a prevalent approach utilized to facilitate the transformation of legacy systems into MSA.
- **Development.** This activity is the second most recurrent in our mapping, with 26 studies (i.e., S1, S6, S7, S12 to S18, S20, S21, S24, S26 to S29, S31, S32, S34 to S38, S41, and S43). Once the scope of the legacy system has been fully understood and a preliminary solution has been outlined, development teams can proceed with the development of microservices. In terms of development, two principal approaches to migrating a legacy system to a microservice-based architecture can be highlighted. The first of these is to rewrite the legacy system from scratch, while the second is to extract the microservices (i.e., functionalities) from the legacy source code. The evidence from our mapping suggests that the second option is the most recommended approach for legacy systems with high-value legacy code. In parallel, it is also crucial to highlight that the previous iteration of the legacy system must be terminated as the development of microservices progresses to prevent the emergence of maintenance issues stemming from the coexistence of functionalities in two distinct locations. Moreover, these findings also indicate that microservice development should be guided by the principle of one microservice per team to ensure that each microservice addresses a single, specific problem. Therefore, after this activity, the microservice-based system must be fully functional and ready for release.
- **Integration.** This activity was identified in 13 studies (i.e., S1, S13, S17, S18, S20, S21, S26 to S28, S31, S35, S41, and S43) of our mapping. In the integration activity, microservices must be combined and synchronized to form a functional and cohesive application. Throughout this activity, microservices establish connections via APIs or alternative communication mechanisms, ensuring the exchange of data and information essential to the overall functioning of the system. In addition, this activity may also involve the integration of the modernized system (i.e., microservices) with other external systems or components. The integration of microservices enables the verification of interactions between microservices and the acquisition of communication

parameters with external systems. At this stage, any incompatibilities or inconsistencies can be identified and corrected, ensuring that the integrated application is stable and reliable. Finally, evidence from our investigation suggests that integration can also entail the establishment of continuous deployment pipelines and process automation to facilitate the distribution of microservices across disparate environments, such as development, testing, and production. Upon completion of this activity, the modernized software is expected to be ready for deployment and available for use by end users.

- **Monitoring.** This activity is the least common in our mapping, as evidenced by four studies (i.e., S7, S15, S20, and S21). In the context of MSA, monitoring is a first-class task due to the distributed and complex nature of these systems. Our findings suggest that given the interactions among multiple microservices and their deployment in disparate environments, it is imperative to possess comprehensive visibility of the status of each component and its interactions with other microservices. Therefore, monitoring in microservices environments typically involves the collection and analysis of performance-related metrics, including response time, error rate, resource utilization (e.g., CPU, memory, and network), and availability. These metrics are monitored in real time and can be viewed through dashboards and reports, thereby providing valuable insights into the health of the system and facilitating the identification of bottlenecks, failures, and optimization opportunities. In addition, monitoring can include the detection and early warning of anomalies and potential problems, enabling operations and development teams to take corrective action to minimize the impact on end users. Finally, the findings of our study indicate that tools and platforms that specialize in collecting, storing, and analyzing monitoring data, as well as observability practices that aim to make the system more understandable and diagnosable in the event of problems, must be used to implement an effective monitoring system in microservice environments.

Upon the activities reported in this section, it can be posited that the proposed process has the potential to act as a framework of guidelines that can guide different stakeholders in the activity of modernizing a legacy system with the implementation of MSA. Furthermore, evidence gathered from our investigation suggests that this process brings together the best activities and steps, thereby enabling its adaptation to operate in disparate application domains and software areas. The proposition of Micro4Delphi [9] by our research group, a process based on modernization activities presented in this section, corroborates these insights. In summary, the proposed process aims to facilitate the modernization of legacy systems developed in Delphi based on MSA. The description of the aforementioned process was meticulously delineated for each activity (i.e., planning, analysis, decomposition, development, integration, and monitoring), along with their respective steps (see Figure 6). Furthermore, a case study (<https://github.com/LucasFFavero/Avance>, accessed on 26 March 2025) was presented in order to evaluate the applicability of Micro4Delphi, thereby providing a significant theoretical and practical contribution to the advancement of microservice application development using Delphi.

5. Discussion of Results

This section will present a comprehensive examination of the results of our investigation. The evidence gathered suggests that the modernization process is of crucial importance for both public and private organizations because it enables them to transition their legacy systems to MSA, an architectural style that enables them to address new business requirements. As previously stated in Section 4.4, the modernization of legacy systems with the implementation of a new architectural style is an industry-wide necessity that has

been addressed through collaborative initiatives between academic and industrial partners. Our SMS encompasses 43 studies, selected from a total of 1037 studies through a systematic process comprising several phases. Each study reported an initiative that was targeted at supporting the modernization process, thereby contributing to a more comprehensive analysis for identifying the genuine activities and steps associated with this process type. Next, we address the relevant topics of each RQ established for our investigation.

Initiatives. As identified in RQ1 (Figure 2), the initiative type named “Approach” was the most common in our SMS. At the same time, the findings of our investigation indicated a lack of clarity regarding the terminology employed to define these initiatives. By analyzing the initiatives, their level of maturity, and their adoption by the industry, it can be surmised that this confusion of concepts presents a significant obstacle for industry professionals to adopt such initiatives in their daily activities, largely due to a lack of comprehension of the activities involved. Although a significant number of studies have been industrial or with industry participation (see Figure 4), this number can be considered below expectations, as the subject of our investigation is directly related to the interest of the software industry. In this sense, the proposed process (see Section 4.6) can serve as a guideline framework for the creation of alternative software modernization processes, since it encompasses a comprehensive set of activities and steps that can be universally applied to any software system.

Application domains. Our analysis revealed that 33 of the 43 studies selected for review in our SMS provide concrete evidence and were classified in one of the nine identified domains (see Figure 3). Although these studies were classified as belonging to the Information Systems area, the evidence showed that researchers and practitioners were engaged in efforts to promote initiatives in a range of domains. It can thus be inferred that despite belonging to different domains, the systems in question exhibited common execution problems, including a lack of scalability, high response times, and an inability to meet new market demands. In essence, the systems identified in these studies exhibit a set of common features across nearly all application domains, facilitating information exchange between users and systems to meet user needs. Based on this scenario, it can be posited that the modernization of such systems with the implementation of a more contemporary architectural style enables the mitigation of adverse failures or QoS degradation, thereby preventing any adverse impact on users. The potential consequences of this scenario can be significantly exacerbated, contingent on the criticality of the application in question. To illustrate such adversities, one may consider the scenario in which a lottery system receives a considerable number of requests from its users on the last day of operation. This scenario could result in an overload of processing power, which, in turn, could lead to a significant slowdown or even a disruption of the service/system.

Evaluation vs. maturity. The act of conducting some type of evaluation may be used as an indicator of the maturity of the initiatives identified in our investigation. The majority of the studies focused on conducting case studies (i.e., 36 studies), which may indicate an interest by researchers and practitioners in evaluating the more practical aspects of these initiatives. This focus on practicality is evident in the choice of case studies, which often present situations similar to those that will be found in real-world scenarios. This type of evaluation is regarded as a positive aspect, as it demonstrates the willingness of the stakeholders to assess the efficacy of modernization initiatives in real contexts. Furthermore, it enables stakeholders, such as software architects and developers, to identify potential bottlenecks and areas for improvement in future iterations of the process. Notwithstanding the limited number of studies that did not undertake an evaluation (i.e., two studies), we have classified these studies as academic, indicating that they are in an early stage of development and present challenges for evaluation. On the other hand, it may be surmised

that studies/initiatives that have undergone evaluation are more readily accepted for future implementation, given the evident interest on the part of both researchers and practitioners in investigating initiatives designed to modernize legacy systems based on MSA.

Quality attributes. The evidence gathered in this study, along with the experiences reported by Passini et al. [16], indicates that quality attributes must be considered a primary element for service-oriented or microservice-based systems. We observed that monitoring, although less than the others, was an activity identified in our SMS as a follow-up measure for the modernized system (see Table 6). In short, monitoring is most often achieved through regular health checks performed on each microservice handled by the monitoring system. Therefore, it is necessary to specify a set of parameters that should be used to determine the health of each microservice in the modernized software according to the needs of each application domain. Among the interests that can be monitored in a modernized system (i.e., microservice-based), performance, reliability, response time, and availability were identified as the quality attributes (see Figure 5) that can be monitored via a dashboard. Therefore, by using a monitoring system to determine the quality level of each microservice, this system can rank each microservice and provide a list of potential microservices that best meet the needs of their users or that need to be scalable. In this direction, as reported in Camargo et al. [67], Grafana (<https://grafana.com>, accessed on 26 March 2025) and Wavefront (<https://github.com/wavefrontHQ/wavefront-spring-boot>, accessed on 26 March 2025) are feasible solutions to promote not only the monitoring of the aforementioned attributes but also the full observability of an application based on microservices. More agile problem detection, modernization monitoring, bottleneck identification, and release agility are some of the benefits that can be achieved with the observability of microservice-based applications using the above solutions.

Modernization process. The findings of our investigation suggest that there is a growing interest among researchers and practitioners in the modernization of legacy systems based on MSA. The investigation has identified a series of steps, which have been organized into six distinct activities, as illustrated in Figure 6. In this sense, it is worth highlighting the proposition of the process called Micro4Delphi [9] as an instance of the modernization process outlined in this article (see Section 4.6). Additionally, a case study was conducted to evaluate the behavior of the Micro4Delphi process concerning the established set of activities and steps. Therefore, it can be suggested that the activities and steps identified for modernization in this article proved to be of high value, as they contributed to the complete transition from a legacy system to a modernized system (i.e., from planning to monitoring).

6. Conclusions and Future Work

The modernization of legacy systems based on MSA has revealed an important issue that requires careful planning to achieve the desired results. Given the need for familiarity with both legacy systems and MSA areas, it would be advantageous for stakeholders involved in this research topic to have a comprehensive and detailed understanding of these areas. Therefore, the main objective of the researchers and practitioners engaged in the modernization of legacy systems with the implementation of MSA is to develop solutions that can facilitate this type of activity. In order to address this research topic, we presented in this article a comprehensive overview of the modernization of legacy systems with the implementation of MSA, focusing on the process activities. This overview was elaborated based on a mixed-method investigation composed of a systematic mapping based on 43 primary studies and an empirical evaluation by industry practitioners. By synthesizing insights from these primary studies, we offered a comprehensive overview of taxonomy for modernization initiatives, application domains that benefited from such initiatives,

a panorama of maturity level and evaluation strategies, quality attributes adopted in such initiatives, and a process to support software modernization.

By analyzing initiatives, our investigation revealed that the topic of research is still emerging and is of common interest to both researchers and practitioners. Therefore, aiming to strengthen the collaboration between academia and industry while facilitating the advancement of novel initiatives, we recommend that more empirical experiments and industrial studies be conducted. Our suggestion is not limited to simply increasing the validity or maturity level of a proposal, but it also aims at ensuring that the industry's needs are met with respect to the use of this type of initiative.

The detailed analysis on the evaluation strategies and the level of maturity of the initiatives revealed a set of key indicators which were established that describe the current stage of the initiatives identified in our investigation. Based on these findings, it can be suggested that the studies that introduced these initiatives were seen as the most effective guidelines for modernization efforts. In parallel, an understanding of the application domains related to these initiatives also provided essential parameters for the steps that should be followed in each activity.

Regarding the modernization process itself, we provide a comprehensive process, delineating the principal activities and steps related to the modernization of legacy systems based on MSA. In general, the proposed process was designed independently in relation to the number of software artifacts that were available for modernization to be conducted. However, the steps within each activity may require an input artifact, which must be processed and transformed into a new artifact for the subsequent step. In short, this is performed to reduce the complexity of the process itself, as well as establishing the standardization of concepts and a minimum set of steps to ensure that any negative aspects of these systems are not involuntarily migrated. In other words, the process presented in this article has the potential to serve as a theoretical framework for researchers and practitioners alike, providing a solid and robust foundation of knowledge for addressing similar activities, including the design of new initiatives and the modification of existing ones. For instance, the modernization of a legacy system based on MSA offers advantages that can be leveraged at the organizational and product levels.

From another perspective, our findings can indeed be used by researchers to develop novel solutions, analyze and experiment with research implications, and establish future research dimensions. As can be observed in Section 4.6, our mapping summarizes the main evidence for the modernization of legacy systems based on MSA and presents a structured process comprising six activities. In this direction, we must highlight Micro4Delphi [9] as a process instantiated from the guidelines presented in this article (see Section 4.6). Thus, it is hoped that researchers and practitioners can also consider our contribution as a potential guideline for the design of future initiatives.

As future work, we intend to conduct at least two activities. Firstly, we intend to monitor the evolution of this research topic by updating this mapping. Despite the existence of initiatives (see Figure 2) identified in this mapping (as evidenced by the 43 studies referenced in Table 5), a comprehensive guide to the modernization of legacy systems to MSA that provides detailed guidance on the subject has not yet been investigated in depth. Secondly, we also intend to validate the proposed process (i.e., activities and steps) through instantiating case studies, which will simulate real scenarios for the modernization of legacy systems to MSA in different domains and different technology stacks. Based on the proposed content, it can be said that the objective of these activities is to convene the viewpoints of researchers and practitioners on this research topic. It is noteworthy that, although their scopes are related, these professionals have worked separately (see Figure 4).

Author Contributions: Conceptualization and Formal analysis—L.F.F.; Investigation and Manuscript preparation—L.F.F.; Support in writing the manuscript, N.R.d.A.; Supervision and Project administration—F.J.A. All authors have read and agreed to the published version of the manuscript.

Funding: This study was financed in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES)—Finance Code 001.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: All data relevant to this study are included within the article itself. Any additional data and materials can be obtained from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Taibi, D.; Lenarduzzi, V.; Pahl, C. Processes, Motivations, and Issues for Migrating to Microservices Architectures: An Empirical Investigation. *IEEE Cloud Comput.* **2017**, *4*, 22–32. [\[CrossRef\]](#)
2. Silva Filho, H.C.d.; Carneiro, G.d.F. Strategies Reported in the Literature to Migrate to Microservices Based Architecture. In Proceedings of the 16th International Conference on Information Technology—New Generations (ITNG 2019), Las Vegas, NV, USA, 1–3 April 2019; Latifi, S., Ed.; Springer International Publishing: Cham, Switzerland, 2019; pp. 575–580. [\[CrossRef\]](#)
3. Silva, H.H.S.d.; de F. Carneiro, G.; Monteiro, M.P. An Experience Report from the Migration of Legacy Software Systems to Microservice Based Architecture. In Proceedings of the 16th International Conference on Information Technology—New Generations (ITNG 2019), Las Vegas, NV, USA, 1–3 April 2019; Latifi, S., Ed.; Springer International Publishing: Cham, Switzerland, 2019; pp. 183–189. [\[CrossRef\]](#)
4. Hassan, S.; Bahsoon, R.; Kazman, R. Microservice transition and its granularity problem: A systematic mapping study. *Softw. Pract. Exp.* **2020**, *50*, 1651–1681. [\[CrossRef\]](#)
5. Di Francesco, P.; Lago, P.; Malavolta, I. Migrating Towards Microservice Architectures: An Industrial Survey. In Proceedings of the 15th International Conference on Software Architecture, ICSA 2018, Seattle, WA, USA, 30 April–4 May 2018; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2018; pp. 29–38. [\[CrossRef\]](#)
6. Kalske, M.; Mäkitalo, N.; Mikkonen, T. Challenges When Moving from Monolith to Microservice Architecture. In *Current Trends in Web Engineering*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2018; Volume 10544 LNCS, pp. 32–47. [\[CrossRef\]](#)
7. Almeida, N.R.; Campos, G.N.; Moraes, F.R.; Affonso, F.J. Modernization of Legacy Systems to Microservice Architecture: A Tertiary Study. In Proceedings of the 26th International Conference on Enterprise Information Systems—Volume 2: ICEIS, Angers, France, 28–30 April 2024; INSTICC: Lisboa, Portugal; SciTePress: Setúbal, Portugal, 2024; pp. 581–592. [\[CrossRef\]](#)
8. Petersen, K.; Vakkalanka, S.; Kuzniarz, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Inf. Softw. Technol.* **2015**, *64*, 1–18. [\[CrossRef\]](#)
9. Fávero, L.F.; Mário, G.S.; Affonso, F. Micro4Delphi: A Process for the Modernization of Legacy Systems in Delphi to Microservice Architecture. In Proceedings of the 27th International Conference on Enterprise Information Systems, Porto, Portugal, 4–6 April 2025; ICEIS: Setúbal, Portugal; INSTICC: Lisboa, Portugal; SciTePress: Setúbal, Portugal, 2025; Volume 2, pp. 328–339. [\[CrossRef\]](#)
10. Lewis, J.; Fowler, M. *Microservices Guide*. 2019. Available online: <https://martinfowler.com/microservices/> (accessed on 26 March 2025).
11. Tserpes, K. stream-MSA: A microservices’ methodology for the creation of short, fast-paced, stream processing pipelines. *ICT Express* **2019**, *5*, 146–149. [\[CrossRef\]](#)
12. Newman, S. *Building Microservices*, 2nd ed.; O’Reilly Media: Sebastopol, CA, USA, 2021.
13. IBM. What Is Service-Oriented Architecture (SOA)? 2024. Available online: <https://www.ibm.com/topics/soa> (accessed on 26 March 2025).
14. Amundsen, M.; McLarty, M. *Microservice Architecture*; O’Reilly Media: Sebastopol, CA, USA, 2016.
15. Krause, A.; Zirkelbach, C.; Hasselbring, W.; Lenga, S.; Kroger, D. Microservice Decomposition via Static and Dynamic Analysis of the Monolith. In Proceedings of the IEEE International Conference on Software Architecture Companion, ICSA-C 2020, Salvador, Brazil, 16–20 March 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 9–16. [\[CrossRef\]](#)
16. Passini, W.F.; Lana, C.A.; Pfeifer, V.; Affonso, F.J. Design of frameworks for self-adaptive service-oriented applications: A systematic analysis. *Softw. Pract. Exp.* **2022**, *52*, 5–38. [\[CrossRef\]](#)
17. Dyba, T.; Dingsoyr, T.; Hanssen, G.K. Applying Systematic Reviews to Diverse Study Types: An Experience Report. In Proceedings of the First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), Madrid, Spain, 20–21 September 2007; pp. 225–234. [\[CrossRef\]](#)

18. Kitchenham, B.; Pretorius, R.; Budgen, D.; Brereton, O.P.; Turner, M.; Niazi, M.; Linkman, S. Systematic literature reviews in software engineering—A tertiary study. *Inf. Softw. Technol.* **2010**, *52*, 792–805. [\[CrossRef\]](#)
19. Kitchenham, B.; Charters, S. *Guidelines for Performing Systematic Literature Reviews in Software Engineering*; Technical Report; Keele University and Durham University Joint Report; Keele University: Keele, UK; Durham University: Durham, UK, 2007.
20. Kitchenham, B.; Sjöberg, D.I.K.; Brereton, O.P.; Budgen, D.; Dybå, T.; Höst, M.; Pfahl, D.; Runeson, P. Can we evaluate the quality of software engineering experiments? In Proceedings of the 4th International Symposium on Empirical Software Engineering and Measurement (ESEM 2010), Bolzano, Italy, 16–17 September 2010; pp. 1–8.
21. Fávero, L.F.; Almeida, N.R.; Affonso, F.J. Quality Criteria Spreadsheet. 2024. Available online: https://docs.google.com/spreadsheets/d/1ZG-NywAOctA_MNb3aNL0s2Uan9TIHqb1P_32D1l6mQ8/edit?usp=sharing (accessed on 26 March 2025).
22. Wohlin, C.; Runeson, P.; Hst, M.; Ohlsson, M.C.; Regnell, B.; Wessln, A. *Experimentation in Software Engineering*; Springer Publishing Company, Incorporated: Berlin/Heidelberg, Germany, 2012.
23. Buchgeher, G.; Ramlerf, R.; Stummer, H.; Kaufmann, H. Adopting Microservices for Industrial Control Systems: A Five Step Migration Path. In Proceedings of the 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vasteras, Sweden, 7–10 September 2021; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2021; pp. 1–8. [\[CrossRef\]](#)
24. Lenarduzzi, V.; Lomio, F.; Saarimäki, N.; Taibi, D. Does migrating a monolithic system to microservices decrease the technical debt? *J. Syst. Softw.* **2020**, *169*, 110710. [\[CrossRef\]](#)
25. Trabelsi, I.; Abdellatif, M.; Abubaker, A.; Moha, N.; Mosser, S.; Ebrahimi-Kahou, S.; Guéhéneuc, Y.G. From legacy to microservices: A type-based approach for microservices identification using machine learning and semantic analysis. *J. Software: Evol. Process* **2022**, *35*, e2503. [\[CrossRef\]](#)
26. Sellami, K.; Ouni, A.; Saied, M.A.; Bouktif, S.; Mkaouer, M.W. Improving microservices extraction using evolutionary search. *Inf. Softw. Technol.* **2022**, *151*, 106996. [\[CrossRef\]](#)
27. Mazzara, M.; Dragoni, N.; Bucchiarone, A.; Giarretta, A.; Larsen, S.T.; Dustdar, S. Microservices: Migration of a Mission Critical System. *IEEE Trans. Serv. Comput.* **2021**, *14*, 1464–1477. [\[CrossRef\]](#)
28. Freire, A.F.A.A.; Sampaio, A.F.; Carvalho, L.H.L.; Medeiros, O.; Mendonça, N.C. Migrating production monolithic systems to microservices using aspect oriented programming. *Softw. Pract. Exp.* **2021**, *51*, 1280–1307. [\[CrossRef\]](#)
29. Kalia, A.K.; Xiao, J.; Krishna, R.; Sinha, S.; Vukovic, M.; Banerjee, D. Mono2Micro: A practical and effective tool for decomposing monolithic Java applications to microservices. In Proceedings of the 29th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece, 18 August 2021; Spinellis, D., Ed.; Association for Computing Machinery, Inc.: New York, NY, USA, 2021; pp. 1214–1224. [\[CrossRef\]](#)
30. Wang, Y.T.; Ma, S.P.; Lai, Y.J.; Liang, Y.C. Qualitative and quantitative comparison of Spring Cloud and Kubernetes in migrating from a monolithic to a microservice architecture. *Serv. Oriented Comput. Appl.* **2023**, *17*, 149–159. [\[CrossRef\]](#)
31. Zaragoza, P.; Seriai, A.D.; Seriai, A.; Bouziane, H.L.; Shatnawi, A.; Derras, M. Refactoring monolithic object-oriented source code to materialize microservice-oriented architecture. In Proceedings of the 16th International Conference on Software Technologies, ICSoft 2021, Virtual Event, 6–8 July 2021; SciTePress: Setúbal, Portugal, 2021; pp. 78–89. [\[CrossRef\]](#)
32. Löhnertz, J.; Oprescu, A.M. Steinmetz: Toward automatic decomposition of monolithic software into microservices. In Proceedings of the CEUR Workshop Proceedings, Uzbekistan, Tashkent, 7–9 October 2020; Constantinou, E., Ed.; CEUR-WS; Volume 2754, pp. 1–8.
33. Mishra, R.; Jaiswal, N.; Prakash, R.; Barwal, P.N. Transition from Monolithic to Microservices Architecture: Need and proposed pipeline. In Proceedings of the 2022 International Conference on Futuristic Technologies, INCOFT 2022, Belgaum, India, 25–27 November 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 1–6. [\[CrossRef\]](#)
34. Dehghani, M.; Kolahdouz-Rahimi, S.; Tisi, M.; Tamzalit, D. Facilitating the migration to the microservice architecture via model-driven reverse engineering and reinforcement learning. *Softw. Syst. Model.* **2022**, *21*, 1115–1133. [\[CrossRef\]](#)
35. Osman, M.H.; Saadbouh, C.; Sharif, K.Y.; Admodisastro, N.; Basri, M.H. From Monolith to Microservices: A Semi-Automated Approach for Legacy to Modern Architecture Transition using Static Analysis. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 907–916. [\[CrossRef\]](#)
36. Colanzi, T.; Amaral, A.; Assunção, W.; Zavadski, A.; Tanno, D.; Garcia, A.; Lucena, C. Are we speaking the industry language? The practice and literature of modernizing legacy systems with microservices. In Proceedings of the ACM International Conference Proceeding Series. Association for Computing Machinery, Joinville, Brazil, 5 October 2021; pp. 61–70. [\[CrossRef\]](#)
37. Auer, F.; Lenarduzzi, V.; Felderer, M.; Taibi, D. From monolithic systems to Microservices: An assessment framework. *Inf. Softw. Technol.* **2021**, *137*, 106600. [\[CrossRef\]](#)
38. Zaragoza, P.; Seriai, A.D.; Seriai, A.; Shatnawi, A.; Bouziane, H.L.; Derras, M. Materializing Microservice-oriented Architecture from Monolithic Object-oriented Source Code. *Commun. Comput. Inf. Sci.* **2022**, *1622*, 143–168. [\[CrossRef\]](#)

39. Kyryk, M.; Tymchenko, O.; Pleskanka, N.; Pleskanka, M. Methods and process of service migration from monolithic architecture to microservices. In Proceedings of the 16th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2022, Lviv-Slavske, Ukraine, 22–26 February 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 553–558. [\[CrossRef\]](#)
40. Stranner, H.; Strobl, S.; Bernhart, M.; Grechenig, T. Microservice decomposition: A case study of a large industrial software migration in the automotive industry. In Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering, Online, 5–6 May 2020; SciTePress: Setúbal, Portugal, 2020; pp. 498–505.
41. Ma, S.P.; Li, C.Y.; Lee, W.T.; Lee, S.J. Microservice Migration Using Strangler Fig Pattern and Domain-Driven Design. *J. Inf. Sci. Eng.* **2022**, *38*, 1285–1303. [\[CrossRef\]](#)
42. Li, C.Y.; Ma, S.P.; Lu, T.W. Microservice Migration Using Strangler Fig Pattern: A Case Study on the Green Button System. In Proceedings of the 2020 International Computer Symposium, ICS 2020, Tainan, Taiwan, 17–19 December 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 519–524. [\[CrossRef\]](#)
43. Ma, S.P.; Lu, T.W.; Li, C.C. Migrating Monoliths to Microservices based on the Analysis of Database Access Requests. In Proceedings of the IEEE International Conference on Service-Oriented System Engineering, SOSE 2022, Newark, CA, USA, 15–18 August 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 11–18. [\[CrossRef\]](#)
44. Salii, S.; Ajdari, J.; Zenuni, X. Migrating to a microservice architecture: Benefits and challenges. In Proceedings of the ICT and Electronics Convention, MIPRO 2023, Opatija, Croatia, 22–26 May 2023; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2023; pp. 1670–1677. [\[CrossRef\]](#)
45. Carvalho, L.; Garcia, A.; Colanzi, T.E.; Assuncao, W.K.G.; Pereira, J.A.; Fonseca, B.; Ribeiro, M.; De Lima, M.J.; Lucena, C. On the Performance and Adoption of Search-Based Microservice Identification with toMicroservices. In Proceedings of the 2020 IEEE International Conference on Software Maintenance and Evolution, ICSME 2020, Adelaide, SA, Australia, 28 September–2 October 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 569–580. [\[CrossRef\]](#)
46. Gomes Barbosa, M.H.; Maia, P.H.M. Towards Identifying Microservice Candidates from Business Rules Implemented in Stored Procedures. In Proceedings of the 2020 IEEE International Conference on Software Architecture Companion, ICSCA-C 2020, Salvador, Brazil, 16–20 March 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 41–48. [\[CrossRef\]](#)
47. Bamberger, B.; Körber, B. Migrating Monoliths to Microservices Integrating Robotic Process Automation into the Migration Approach. *J. Autom. Mob. Robot. Intell. Syst.* **2022**, *2022*, 72–82. [\[CrossRef\]](#)
48. Parikh, A.; Kumar, P.; Gandhi, P.; Sisodia, J. Monolithic to Microservices Architecture—A Framework for Design and Implementation. In Proceedings of the International Conference on Computer, Power and Communications, ICCPC 2022, Chennai, India, 14–16 December 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 90–96. [\[CrossRef\]](#)
49. Kuryazov, D.; Jabborov, D.; Khujamuratov, B. Towards Decomposing Monolithic Applications into Microservices. In Proceedings of the IEEE International Conference on Application of Information and Communication Technologies, AICT 2020, Tashkent, Uzbekistan, 7–9 October 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 1–4. [\[CrossRef\]](#)
50. Freitas, F.; Ferreira, A.; Cunha, J. A methodology for refactoring ORM-based monolithic web applications into microservices. *J. Comput. Lang.* **2023**, *75*, 101205. [\[CrossRef\]](#)
51. Yang, Z.; Wu, S.; Zhang, C. A Microservices Identification Approach based on Problem Frames. In Proceedings of the IEEE International Conference on Software Engineering and Artificial Intelligence, SEAI 2022, Xiamen, China, 10–12 June 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 155–159. [\[CrossRef\]](#)
52. Kazanavičius, J.; Mažeika, D. An Approach to Migrate from Legacy Monolithic Application into Microservice Architecture. In Proceedings of the IEEE Open Conference of Electrical, Electronic and Information Sciences, eStream 2023, Vilnius, Lithuania, 27–27 April 2023; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2023; pp. 1–6. [\[CrossRef\]](#)
53. Volynsky, E.; Mehmed, M.; Krusche, S. Architect: A Framework for the Migration to Microservices. In Proceedings of the 2022 International Conference on Computing, Electronics and Communications Engineering, ICCECE 2022, Southend, UK, 17–18 August 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 71–76. [\[CrossRef\]](#)
54. Laigner, R.; Kalinowski, M.; Diniz, P.; Barros, L.; Cassino, C.; Lemos, M.; Arruda, D.; Lifschitz, S.; Zhou, Y. From a Monolithic Big Data System to a Microservices Event-Driven Architecture. In Proceedings of the 46th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2020, Portoroz, Slovenia, 26–28 August 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 213–220. [\[CrossRef\]](#)
55. Santos, A.; Paula, H. Microservice decomposition and evaluation using dependency graph and silhouette coefficient. In Proceedings of the ACM International Conference Proceeding Series, Joinville, Brazil, 27 September–1 October 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 51–60. [\[CrossRef\]](#)
56. Prasandy, T.; Titan.; Murad, D.F.; Darwis, T. Migrating application from monolith to microservices. In Proceedings of the 2020 International Conference on Information Management and Technology, ICIMTech 2020, Bandung, Indonesia, 13–14 August 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 726–731. [\[CrossRef\]](#)

57. Haugeland, S.G.; Nguyen, P.H.; Song, H.; Chauvel, F. Migrating Monoliths to Microservices-based Customizable Multi-tenant Cloud-native Apps. In Proceedings of the 2021 47th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2021, Palermo, Italy, 1–3 September 2021; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2021; pp. 170–177. [\[CrossRef\]](#)
58. Goncalves, N.; Faustino, D.; Silva, A.R.; Portela, M. Monolith Modularization towards Microservices: Refactoring and Performance Trade-offs. In Proceedings of the 2021 IEEE 18th International Conference on Software Architecture Companion, ICSCA-C 2021, Stuttgart, Germany, 22–26 March 2021; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2021; pp. 54–61. [\[CrossRef\]](#)
59. Preti, J.P.D.; Souza, A.N.A.; Freiburger, E.C.; De Almeida Lacerda, T. Monolithic to Microservices Migration Strategy in Public Safety Secretariat of Mato Grosso. In Proceedings of the 3rd International Conference on Electrical, Communication and Computer Engineering, ICECCE 2021, Kuala Lumpur, Malaysia, 12–13 June 2021; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2021; pp. 1–5. [\[CrossRef\]](#)
60. Vera-Baquero, A.; Phelan, O.; Slowinski, P.; Hannon, J. Open Source Software as the Main Driver for Evolving Software Systems Toward a Distributed and Performant E-Commerce Platform: A Zalando Fashion Store Case Study. *IT Prof.* **2021**, *23*, 34–41. [\[CrossRef\]](#)
61. Bajaj, D.; Bharti, U.; Goel, A.; Gupta, S. Partial Migration for Re-architecting a Cloud Native Monolithic Application into Microservices and FaaS. *Commun. Comput. Inf. Sci.* **2020**, *1170*, 111–124. [\[CrossRef\]](#)
62. Michael Ayas, H.; Leitner, P.; Hebig, R. The Migration Journey Towards Microservices. In *Product-Focused Software Process Improvement*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2021; Volume 13126, pp. 20–35. [\[CrossRef\]](#)
63. Batista, C.; Proença, B.; Cavalcante, E.; Batista, T.; Morais, F.; Medeiros, H. Towards a Multi-Tenant Microservice Architecture: An Industrial Experience. In Proceedings of the IEEE 46th Annual Computers, Software, and Applications Conference, COMPSAC 2022, Los Alamitos, CA, USA, 27 June–1 July 2022; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2022; pp. 553–562. [\[CrossRef\]](#)
64. Bandara, C.; Perera, I. Transforming monolithic systems to microservices—An analysis toolkit for legacy code evaluation. In Proceedings of the 20th International Conference on Advances in ICT for Emerging Regions, ICTer 2020, Colombo, Sri Lanka, 4–7 November 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020; pp. 95–100. [\[CrossRef\]](#)
65. SEVOCAB. IEEE Computer Society. Software and Systems Engineering Vocabulary. 2023. Available online: https://pascal.computer.org/sev_display/index.action (accessed on 26 March 2025).
66. Pressman, R.; Maxim, B. *Software Engineering: A Practitioner's Approach*, 9th ed.; McGraw-Hill Education: New York, NY, USA, 2019.
67. Camargo, M.P.d.O.; Pereira, G.d.S.; Almeida, D.; Bento, L.A.; Dorante, W.F.; Affonso, F.J. RA4Self-CPS: A Reference Architecture for Self-adaptive Cyber-Physical Systems. *IEEE Lat. Am. Trans.* **2024**, *22*, 113–125. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.