AJMAS
https://journal.utripoli.edu.ly/index.php/Alqalam/index eISSN 2707-7179

Original article

Realization of Quality Factors of Microservices Architecture

Ashraf Bourawy

Department of Computer Science, Omar AL-Mukhtar University, Albayda, Libya

ARTICLE INFO

Corresponding Email. *abourawy@omu.edu.ly*
Received: *18-10-2022* ***Accepted:****26-11-2022* ***Published:****23-12-2022*
Keywords: *Microservices, Monolith, Quality factors, Software Architecture.*

ABSTRACT

Microservices architecture has gained an increasing popularity due to its beneficial characteristics and advantages. It supports building independent, loosely coupled and single-task services while keeping track of improving availability, scalability, and fault tolerance. Many research studies have been conducted to address migration from monolith to microservices and to contribute to quality of these services. In addition, most studies' aim is to overcome challenges and issues faced by microservices development and deployment. However, there is a gap in the literature where systematic realization of quality factors related to microservices architecture is not thoroughly investigated. The aim of this study was to address and understand the quality factors of microservices-based systems. To accomplish our objective, we conducted a systematic mapping study to identify and construct a state-of-the-art study based on the retrieved relevant studies. Based on the results of the extracted data from 85 relevant studies, ten quality factors were identified and further discussed. These recognized factors are: scalability, performance, monitorability, availability, testability, reliability, security, maintainability, fault tolerance and reusability. Moreover, different challenges were identified and provided recommendations to address these issues. Although different quality factors were addressed, we concluded that more attention should be given to some factors such as security, availability, reliability, and reusability. In addition, trade-off between some factors such as scalability and performance should be addressed to optimize the overall system performance and to avoid service degradation.

Cite this article. *Bourawy A. Realization of Quality Factors of Microservices Architecture. Alq J Med App Sci. 2022;5(2):611-623. https://doi.org/10.5281/zenodo.74823812*

INTRODUCTION

Since their emergence in the past decade, microservices architecture (MSA) has led many enterprises to migrate their existing monolithic applications to MSA style. Unlike the service-oriented architecture (SOA), MSA requires that each microservice be implemented to address only one single task or functionality [1]. Following this manner, a monolithic application can be decomposed into a set of lightweight, loosely coupled and independent microservices. Consequently, each of these microservices has consistent business and functionality boundaries [1], [2]. The popularity of MSA has also drawn the awareness of the industry triggering the world's leading Internet enterprises such as Amazon, eBay and Netflix to migrate their applications to microservices technology.

Although adapting MSA style is beneficial by taking an advantage of its characteristics, it is essential to count for quality assurance when migrating monolithic applications to microservices [1]. Several challenges may face the process of converting and moving to MSA. In order for microservices to communicate, they can use synchronous or asynchronous communications with the help of some protocols such as Restful API or message-based streams. This inter-service communication entails a great deal of overheads that may lead to system performance degradation [3]. In addition, when a client sends a request, service discovery mechanism is invoked to locate the requested microservice in the service registry, which entails some overheads as well.

Quality assurance is a very crucial aspect when developing and deploying microservices. *Scalability* of services based on user's demand must be considered taking into account its impact on the system *performance*. Since the MSA is mainly related to the cloud, *security* also plays an important role and has to be addressed. On the other hand, *testability* and *maintainability* of the system have to be made easy accounting for systems in operation [2].

The aim of this paper is to investigate the quality factors of microservices architecture through conducting a systematic mapping study. Our contribution can be summarized into the following points: 1) Unlike other systematic mapping studies, this study identifies all quality factors addressed by 85 relevant studies considering a wider timespan. 2) Identifying quality factors of microservices that need to be considered more efficiently. 3) Indicating challenges related to quality assurance when developing and deploying microservices.

The remainder of this paper is organized as follows. The next section presents an overview of the related work. Research methodology is then presented and explained. Following, the results are presented with answers to the research questions.

Discussion section on the obtained results is provided. Finally, last section presents our conclusions of the conducted systematic mapping study.

Related work

Researchers and enterprises are interested in MSA due to its features of lightweight, loosely coupled, and independent nature that can be easily maintained. Since the main target of enterprises is to move from monoliths to microservices, most of the existing studies focus on the migration issues [2], [4]-[9]. Extracting microservices from monoliths using business functionality interface is presented by Agarwal et al. [2]. Christoforou et al. [4] relied on decomposing monolithic applications using layered component-based software architecture. A practical tool developed at IBM is proposed by Kalia et al. [5] to partition the application utilizing the technique of spatio-temporal decomposition. Laigner et al. [6] provided details of another mechanism to migrate from monolithic application to microservices targeting big data systems. A knowledge-graph approach is utilized by Shang et al. [7] to extract candidate microservices to be used in constructing a microservice architecture. Volybsky et al. [8] presented a tool named "Architect" to facilitate migration to microservices with preserving quality factors.

Additionally, due to the attractiveness of microservices, several systematic literature reviews (SLR) and systematic mapping studies (SMS) concerning microservices architecture have been proposed in literature. A systematic mapping study was carried out by Phal et al. [9] concerning techniques and challenges of microservices. The related studies were 21 covering the period of 2013-2015, where their focus was mainly on classification and taxonomy of existing studies. Despite the effort made to extract keywords and terms related to quality, the paper lacks any specific findings about the quality factors with regard to microservices. Another SMS was conducted by Alshuqayran et al. [10], which essentially aimed at identifying and reporting architectural challenges faced by microservices and reviewing related quality attributes. The timespan of their study was confined to the years 2014-2016, where 33 relevant studies were selected. Quality factors/attributes were only listed in this study and the discussion was mainly concerned with challenges faced when architecting microservices systems. Francesco et al. [11] also conducted an SMS on architecting microservices considering the trends, focus, and potential paths for MSA. A collection of 71 relevant papers published in any year up to 2016 was obtained and constituted their final pool for the study. The systematic study presented in [11] can be useful to researchers and practitioners as it contains a broad review of trends and research gaps regarding MSA. The study also covers quality attributes addressed by the reviewed studies, which is similar to our work. However, our work differs by considering a wider timespan, till 2022 and also by considering other aspects such as challenges faced by microservices. Li et al. [12] presented a thorough SLR on understanding quality attributes of MSA. They restricted their focus on six quality factors addressed in the 72 reviewed studies covering the period of 2015 to June 2018. Their work also considered the tactics along with quality attributes for migrating and implementing microservices while maintaining high quality assurance. Our study is similar to this study. However, our work differs from the work conducted in [12] as being a systematic mapping study and we also consider all quality factors addressed in reviewed studies between 2017 and 2022.

METHODS

The systematic mapping principles provided in [13] are applied here in this study as the research method. This section describes related steps, namely, planning for the study and defining the research questions, conducting search, screening and study selection, data extraction, data analysis and classification and reporting.

Developing the research method

Initially, the research method applied in this SMS starts with defining research questions, as illustrated in Fig. 1. Using specific search keywords, the search process is conducted to retrieve corresponding journal and conference papers. After obtaining the initial pool of papers, screening and filtration is accomplished by applying inclusion and exclusion criteria. Consequently, the final pool of relevant papers is constituted and ready for classification and data extraction as well as data analysis. The next step is to perform classification of papers based on identified attributes to map and answer the research questions.

Research questions

Defining the research questions is a crucial step to investigate and obtain a systematic understanding of the studied research topic. In order to accomplish this goal, the following research questions are defined in the pursuit of realization and addressing the quality factors of microservices architecture:

RQ1. What are the publication trends in literature concerning microservices architecture?

RQ2. What quality factors of microservices architecture are mostly addressed in the literature?
RQ3. What are the application domains of microservices?
RQ4. Which methods are used to study quality factors in microservices systems?
RQ5. What challenges are faced by microservices in development and deployment processes?

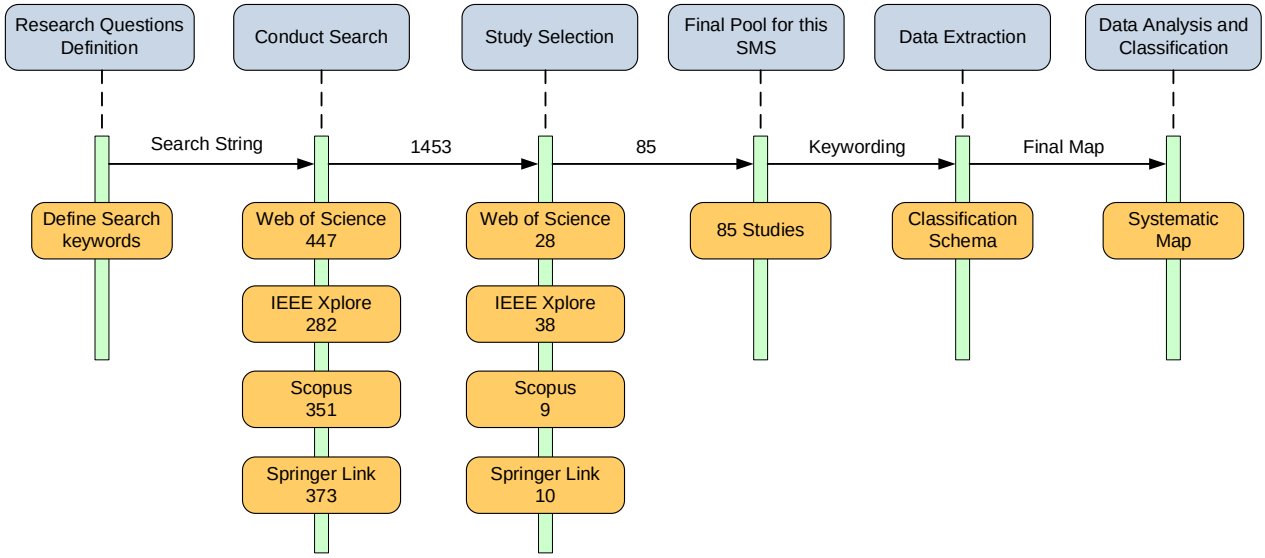

Figure 1. Overview of research method

Conducting search

The objective of conducting search is to obtain a significant collection of research studies pertaining to quality factors of MSA. Thus, it is essential to have a suitable balance between the scope of the existing studies and a reasonably managed number of papers to be selected for further analysis. Based on research questions and the topic of quality factors of MSA, the search string is designed in a broad manner to capture as many relevant studies as possible. The search string is formed as follows:

TITLE-ABSTR-KEY (microservice OR micro-service*) AND (quality)*

As clearly noted, the search string is made very generic in order to preclude missing any relevant studies. Four well-known databases were targeted to retrieve related conference and journal papers, namely, Web of Science, IEEE Xplore, Scopus, and Springer Link. The timespan of our study was restricted to the period from 2017 to 2022. The obtained studies from Web of Science (447), IEEE Xplore (282), Scopus (351) and Springer Link (373) were combined to constitute the initial pool including 1453 studies as shown in Fig. 1.

Study selection

Upon combining the raw set of studies, we applied the inclusion and exclusion criteria (as shown in Table 1) based on reading the title, abstract and sometimes full-text in case the abstract was unclear. Consequently, this resulted in a collection of 85 studies constituting the final repository of relevant candidate studies, as depicted in Fig. 1. The assessment procedure utilized the selection criteria to evaluate each study to ensure that it complied with the inclusion and exclusion criteria. In order to be selected, a paper should include discussion on quality factors, models, and/or challenges related to microservices architecture domain. Only primary studies published in the period 2017-2022 were included. However, primary studies that discuss the microservices in a low-level technical detail were excluded.

Data extraction and synthesis

The process of extracting data is accomplished by the help of the constructed data extraction form shown in Table 2. The keys F1 and F2 are used mainly for referencing purposes. The other data fields are used to extract data pertaining to the research questions stated above in subsection 2. Using this data extraction form along with the help of the excel spreadsheet; a matrix was implemented to extract different quality factors occurrences in the relevant studies. In addition, challenges and application domains were also identified and extracted. Due to heterogeneity of the relevant studies where most of the extracted data is qualitative, the most appropriate analysis method is the thematic analysis. We applied the thematic analysis procedure for data synthesis where results to research questions were obtained and presented in the results section.

Table 1. Selection Criteria

I/E	Inclusion/Exclusion Criteria
I1	Studies that validate the effect of quality factors of MSA
I2	Studies proposing solution to tackle issues related to quality factors or evaluating quality factors in MSA domain
I3	Only journals and conference papers published between 2017 and 2022
E1	Papers are not written in English
E2	Papers are not available in full-text
E3	Secondary Studies, e.g.: literature reviews, etc.
E4	Studies mentioning microservices as an example or in keywords only, or in low-level technical detail

Table 2. Data Extraction Form

No.	Field	RQ	Comments
F1	Title	Q1	N/A
F2	Year	Q1	N/A
F3	Venue Type	Q1	Conference/Journal
F4	Publisher	Q1	Elsevier, IEEE, ACM
F5	Quality factors	Q2	List of Quality Factors
F6	App. Domain	Q3	Domains, e.g., Cloud
F7	Methods	Q4	Research methods, e.g., Case study, Experimental
F8	Challenges	Q5	Challenges, e.g., scalability, migration

RESULTS

The results of this systematic mapping study are presented below.

RQ1. What are the publication trends in literature concerning microservices architecture?

Most of the studied publications in this systematic study were published in IEEE conferences and journals, about 45%, followed by ACM with 19%, as demonstrated in Fig. 2. This is can be supported by Fig. 3 which shows that IEEE Xplore database contained the majority of retrieved studies, 38 papers, concerning the research topic of quality factors of microservices architecture.

The yearly distribution of reviewed studies (Fig. 4) showed that publications presented in conference proceedings have contributed more than journal articles in this systematic mapping study. Fig. 4, however, demonstrates an increasing interest in the research on the quality factors of microservices. The relevant studies in 2017 were 9 studies which increased to about threefold in 2022 with 24 publications.

RQ2. What quality factors of microservices architecture are mostly addressed in the literature?

Based on the 85 examined studies, ten quality factors have been identified in this SMS, which are related to microservices. These quality factors are: *scalability*, *performance*, *monitorability*, *availability*, *testability*, *reliability*, *security*, *maintainability*, *fault tolerance* and *reusability*.

The results about the frequency of occurrences of quality factors are illustrated in Fig. 5, which demonstrates the emphasis of quality factors of microservices addressed by the reviewed papers. From the figure and the counting process, we can see that *scalability* (19 papers) and *Performance* (16 papers) are the most concerned quality factors of microservices. *Monitorability* (15 papers) is placed next in importance along with *testability* (12). Following, other factors come as *availability* (10 papers), *security* (9 papers), *maintainability* (8 papers), and *reliability* (7 papers). Finally, the least addressed factor was *reusability* with only two papers. In addition, about 21 papers of the reviewed studies addressed more than one quality factor at the same time.

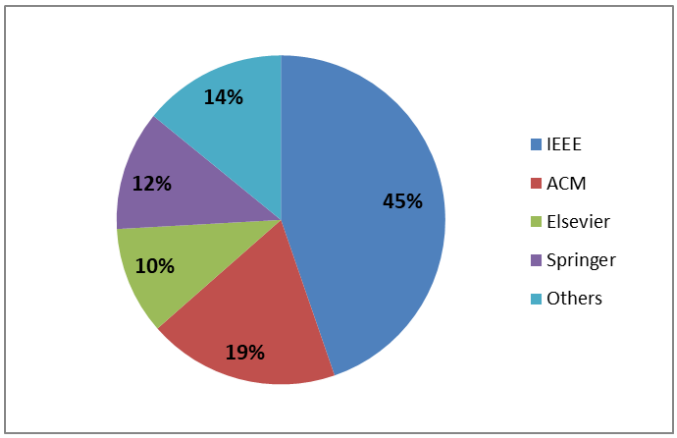

Figure 2. Distribution by publishers

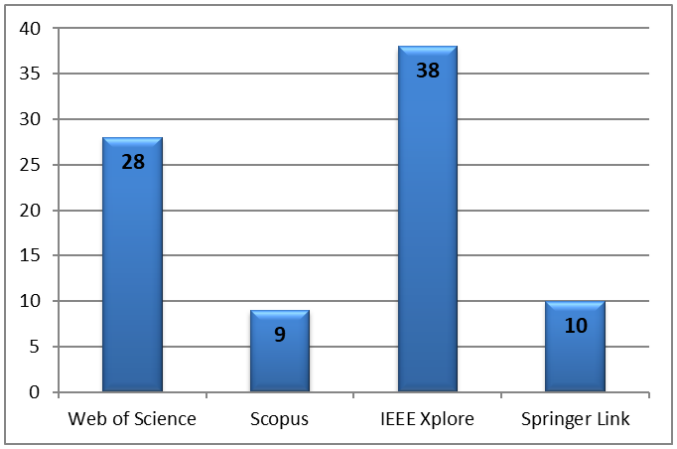

Figure 3. Distribution by searched databases

RQ3. What are the application domains of microservices?

Application domains can give an indication to researchers about the applicability of microservices in order to consider other issues concerning these domains. When reviewing the selected relevant studies, we observed that about 67% of these studies had clearly specified their application domains. Fig. 6 illustrates the application domains targeted by the studied and reviewed publications. As shown in this figure, cloud computing domain is the most targeted domain for microservices with 30 papers. In addition, IoT cloud and Edge computing are also considered with 6 papers and 9 papers, respectively. Other domains are e-commerce (2 papers), distributed systems (3 papers), big data (2 papers), and other domains (5 papers). About 33% of the studied papers, however, have not specified their targeted domains. Fig. 6 clearly expresses the apparent relationship between microservices and cloud computing domain.

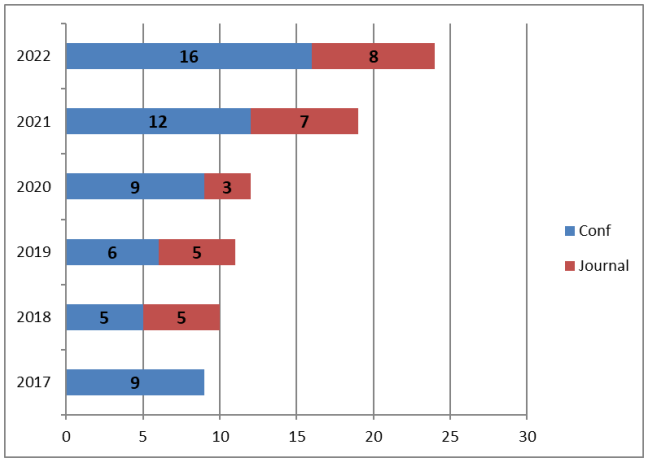

Figure 4. Distribution of studies by year

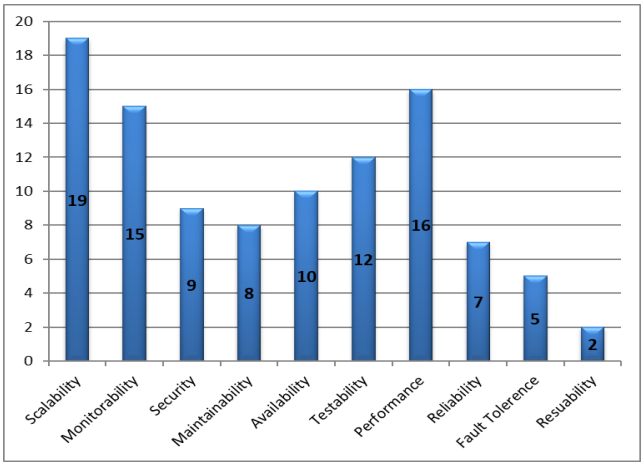

Figure 5. Quality factors addressed by studies

RQ4. Which methods are used to study quality factors in microservices systems?

Different research methodologies were proposed and utilized by the relevant studies under investigation, as shown in Fig. 7. The most recognized methodology is proposing a solution to a given issue regarding microservices and using experimental methods to validate the suggested solution. This methodology, i.e. propose solution with experimental approach, is reported from 34 papers (about 40%). On the other hand, 23 studies propose solutions only with no method for verification or validation. Another methodology, followed by 22 studies, is to study different quality factors through empirical and experimental analyses. Case study methodology was reported by 4 relevant studies.

RQ5. What challenges are faced by microservices in development and deployment processes?

Challenges faced by microservices which influence the quality factors of microservices have been reported in several studies. Migration of monolithic applications to evolve as microservices on the cloud is considered a daunting task that many enterprises confront [S3], [S4], [S8]. The mission of migration comprises different steps including decomposing monolithic application components, defining module dependencies and defining independent functionality for each component [S3].

In addition, scaling services based on users' demands is a challenging issue due to performance considerations [S9]. Other measures have to be considered when decision is taken for scaling services, for instance, energy efficiency, resource provisioning, system performance and quality of service assurance [S9]-[S11]. Another challenge is the size of microservices. Determining the "optimal" size of an individual microservice in the process of decomposition a monolith is currently based on software engineers' experience [S6]. Depending on people's experience is a risk itself, which abandons the impact of hardware and other issues on the system's performance.

Inter-service communication is a considerable challenging issue due to its direct impact on system performance [S2], [S12]. The downside of inter-service communication is the number of overheads, which lead to an increase in the response time. This may result in degrading the performance of the system.

DISCUSSION

The systematic mapping study results regarding the quality factors of microservices are obtained and presented with figures and statistics in the results section.

The publications trend is clearly noticed with most of papers concerning quality factors of microservices are published in IEEE conferences and journals. The distribution of reviewed studies over years from 2017 to 2022 show an increasing growth, which can be interpreted as microservices architecture is still a hot research field that grabs researchers' attention.

Ten quality factors are extracted from the related papers which constitute the answer of main question of our study. From the results, the *scalability* factor is the most concern of the reviewed papers followed by *performance*. These two factors are correlated and in most studies a trade-off between these two factors is determined. Therefore, when designing and developing microservices, one should always consider the trade-off between *scalability* and *performance*.

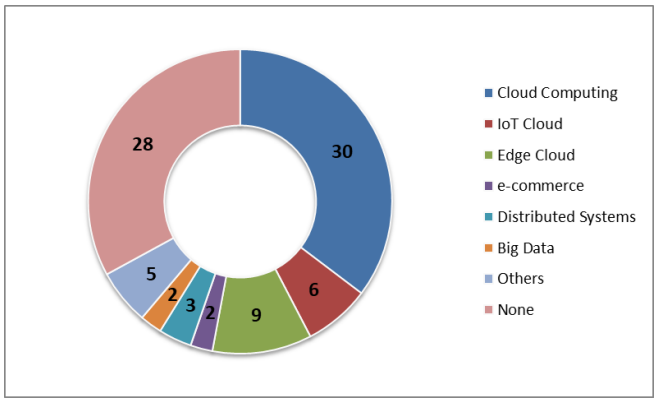

Figure 6. Application Domains targeted by studies

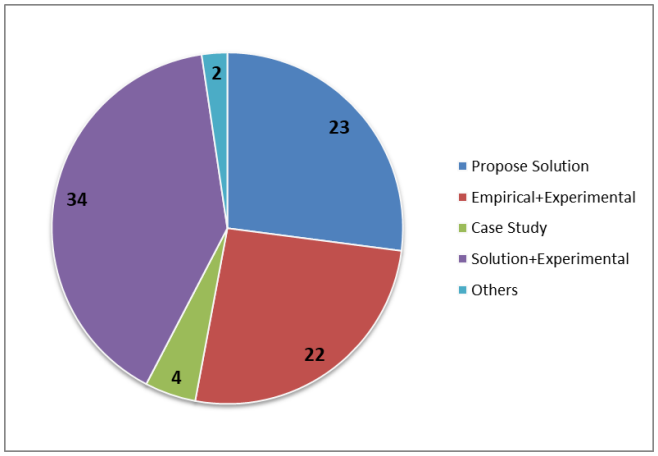

Figure 7. Methods applied by selected studies

Following these two factors, *monitorability* comes in the third place. This is can be understood as monitoring the system is very important issue to discover and report failures. *Testability* comes next due to its importance in providing and maintaining quality of the system. *Availability*, on the other hand, is very important bearing in mind that microservices are employed mainly in cloud computing domain. However, only 12% of the studies were concerned with *availability*. We believe that *availability* should be considered more because it touches the requirements of the customers and may lead to customers leaving the provided services. This argument applies also for the *reliability* factor, which gained only 10.5% of studies' recognition. We can generalize this argument for the other remaining factors as well due to their importance, such as *security*, *maintainability*, *fault tolerance*, and *reusability*. These concerns can be considered as research gaps where other researchers can start investigating these areas and cover the research gaps.

When trying to investigate the application domains of microservices, it revealed that cloud domain was the most related one amongst other domains. Because of their characteristics of being self-contained, lightweight, loosely coupled and independent units, it is obviously understood that microservices are mostly suitable for cloud computing. Other domains such as IoT and Edge Clouds are also related to cloud computing and can be considered parts of it.

Experimental methodology along with proposing solutions to different challenging issues of microservices are the two main methodologies in reviewed studies. This is expected as the microservices architecture is still an evolving technology and considered one of the topics that interest many researchers and organizations. Empirical and case study methodologies have also been considered.

Several challenges are reported in the selected studies, namely, migration from monolith to microservices, scalability of microservices, the size of individual microservice, and inter-service communication. In order to migrate to microservices, the monolithic application should be decomposed to loosely coupled units where each unit handles a single task. The decomposition process is very challenging and entails some issues that must be addressed. Monolithic software functional components can be replaced with adequate microservices by matching available components against an ontological set of specifications expressed in a formalized syntax [S3]. Another approach for decomposition of monolith to microservices is to use the Mono2Micro tool as proposed in [S4]. This tool starts with runtime trace collection, determining partition size, performing clustering and finally obtaining partitions.

The challenge of scaling services based on users' demands neglecting the system performance can be addressed by deploying a sufficient number of microservices while guaranteeing a certain limit of quality of service [S9]. Moreover, this approach can reduce and control energy consumption and resources usage. The size of an individual microservice is challenging as what size is considered optimal? Different approaches can be followed to determine the optimality of the size of a microservice. An approach based on knowledge-graph to extract microservice with suitable size is reported in [S6]. This approach starts with constructing a knowledge graph based on analyzing the system modules, functions and domain entities. The monolith is then converted into a graph structure. Last step is using a restricted Louvain algorithm to get a set of microservices candidates.

Inter-service communications can lead to degradation in system performance because of communication overhead. This is considered one of the most crucial challenges in MSA due to the increase in response time imposed by inter-service communication. Synchronous and asynchronous types of protocols used by the message brokers should be tested and experimented to decide which protocol and type of communication is suitable for each case of deployment. An evaluation of different communication protocols and types is presented in [S2]. In addition, a mechanism for supervising the interservice communication is needed.

CONCLUSION

In this study, we investigated quality factors of MSA through conducting a SMS. Data extraction and synthesis from a collection of 85 studies was accomplished and designated research questions were answered. The selected 85 studies addressed ten quality factors of MSA, where *scalability* and *performance* were dominant in frequency. However, more attention should be given to other factors such as *availability*, *security* and *maintainability*. Cloud computing was the most targeted application domain. In addition, several challenges had been encountered in the process of development and deployment of microservices. Migration from monolith to microservice, scaling services, inter-service communication, and size of individual microservice are examples of these challenges. Although MSA is favored to many enterprises, it is of great importance to count for other complexities when migrating from monolith to microservices. The future work of this paper is to conduct a systematic review study to investigate more on quality factors tactics. In addition, some experimental work can be carried out to evaluate and address some challenges presented in this study.

Conflict of Interest

There are no financial, personal, or professional conflicts of interest to declare.

APPENDIX

The following is a list of studies that were utilized for data extraction and synthesis in this SMS:

[S1] Agarwal S, Sinha R, Sridhara G, Das P, Desai U, Tamilselvam S, et al. Monolith to microservice candidates using business functionality inference. 2021 IEEE International Conference on Web Services (ICWS); 2021: IEEE.

[S2] Weerasinghe L, Perera I. Evaluating the inter-service communication on microservice architecture. 2022 7th International Conference on Information Technology Research (ICITR); 2022: IEEE.

[S3] Christoforou A, Odysseos L, Andreou A. Migration of Software Components to Microservices: Matching and Synthesis. Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE; 2019.

[S4] Kalia AK, Xiao J, Krishna R, Sinha S, Vukovic M, Banerjee D. Mono2micro: a practical and effective tool for decomposing monolithic java applications to microservices. Proceedings of the 29th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering; 2021.

[S5] Laigner R, Kalinowski M, Diniz P, Barros L, Cassino C, Lemos M, et al. From a monolithic big data system to a microservices event-driven architecture. 2020 46th Euromicro conference on software engineering and advanced applications (SEAA); 2020: IEEE.

[S6] Li Z, Shang C, Wu J, Li Y. Microservice extraction based on knowledge graph from monolithic applications. Information Software Technology. 2022;150:106992.

[S7] Volynsky E, Mehmed M, Krusche S. Architect: a framework for the migration to microservices. 2022 International Conference on Computing, Electronics & Communications Engineering (iCCECE); 2022: IEEE.

[S8] Busch A, Kammerer M. Network performance influences of software-defined networks on micro-service architectures. Proceedings of the ACM/SPEC International Conference on Performance Engineering; 2021.

[S9] Simon M, Spallina A, Dubocquet L, Araldo A. Parsimonious edge computing to reduce microservice resource usage. 2021 33th International Teletraffic Congress (ITC-33); 2021: IEEE.

[S10] Houmani Z, Balouek-Thomert D, Caron E, Parashar M. Enhancing microservices architectures using data-driven service discovery and QoS guarantees. 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID); 2020: IEEE.

[S11] Sheshadri K, Lakshmi J. Qos aware faas for heterogeneous edge-cloud continuum. 2022 IEEE 15th International Conference on Cloud Computing (CLOUD); 2022: IEEE.
[S12] Al Maruf A, Bakhtin A, Cerny T, Taibi D. Using microservice telemetry data for system dynamic analysis. 2022 IEEE International Conference on Service-Oriented System Engineering (SOSE); 2022: IEEE.
[S13] Alipour H, Liu Y. Online machine learning for cloud resource provisioning of microservice backend systems. 2017 IEEE International Conference on Big Data (Big Data); 2017: IEEE.
[S14] Boeira C, Neves M, Ferreto T, Haque I. Characterizing network performance of single-node large-scale container deployments. 2021 IEEE 10th International Conference on Cloud Networking (CloudNet); 2021: IEEE.
[S15] Bogner J, Wagner S, Zimmermann A. Using architectural modifiability tactics to examine evolution qualities of Service-and Microservice-Based Systems: An approach based on principles and patterns. SICS Software-Intensive Cyber-Physical Systems. 2019;34(2):141-9.
[S16] Camilli M, Guerriero A, Janes A, Russo B, Russo S. Microservices integrated performance and reliability testing. Proceedings of the 3rd ACM/IEEE International Conference on Automation of Software Test; 2022.
[S17] Cardarelli M, Iovino L, Di Francesco P, Di Salle A, Malavolta I, Lago P. An extensible data-driven approach for evaluating the quality of microservice architectures. Proceedings of the 34th acm/sigapp symposium on applied computing; 2019.
[S18] Celesti A, Carnevale L, Galletta A, Fazio M, Villari M. A watchdog service making container-based micro-services reliable in IoT clouds. 2017 IEEE 5th international conference on future internet of Things and Cloud (fiCloud); 2017: IEEE.
[S19] Celesti A, Mulfari D, Galletta A, Fazio M, Carnevale L, Villari M. A study on container virtualization for guarantee quality of service in cloud-of-things. Future Generation Computer Systems. 2019;99:356-64.
[S20] Crecana C-C, Pop F. Monitoring-based auto-scalability across hybrid clouds. Proceedings of the 33rd Annual ACM Symposium on Applied Computing; 2018.
[S21] Fadda E, Plebani P, Vitali M. Monitoring-aware optimal deployment for applications based on microservices. IEEE Transactions on Services Computing. 2019;14(6):1849-63.
[S22] Fetzer C, Mazzeo G, Oliver J, Romano L, Verburg M. Integrating reactive cloud applications in sereca. Proceedings of the 12th International Conference on Availability, Reliability and Security; 2017.
[S23] Fukuzaki T, Liu S, Butler M. DevFemOps: enhancing maintainability based on microservices using formal engineering methods. Connection Science. 2022;34(1):2125-38.
[S24] Gan Y, Delimitrou C. The architectural implications of cloud microservices. IEEE Computer Architecture Letters. 2018;17(2):155-8.
[S25] Gan Y, Zhang Y, Cheng D, Shetty A, Rathi P, Katarki N, et al. An open-source benchmark suite for microservices and their hardware-software implications for cloud & edge systems. Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems; 2019.
[S26] Gias AU, van Hoorn A, Zhu L, Casale G, Düllmann TF, Wurster M. Performance engineering for microservices and serverless applications: The RADON approach. Companion of the ACM/SPEC International Conference on Performance Engineering; 2020.
[S27] Goli A, Mahmoudi N, Khazaei H, Ardakanian O. A Holistic Machine Learning-based Autoscaling Approach for Microservice Applications. CLOSER. 2021;1:190-8.
[S28] Gribaudo M, Iacono M, Manini D. Performance evaluation of replication policies in microservice based architectures. Electronic Notes in Theoretical Computer Science. 2018;337:45-65.
[S29] Guerrero C, Lera I, Juiz C. Genetic algorithm for multi-objective optimization of container allocation in cloud architecture. Journal of Grid Computing. 2018;16:113-35.
[S30] Guerrero C, Lera I, Juiz C. Resource optimization of container orchestration: a case study in multi-cloud microservices-based applications. The Journal of Supercomputing. 2018;74(7):2956-83.
[S31] Guo F, Tang B, Tang M, Liang W. Deep reinforcement learning-based microservice selection in mobile edge computing. Cluster Computing. 2022;26(2):1319-35.
[S32] Guo X, Peng X, Wang H, Li W, Jiang H, Ding D, et al. Graph-based trace analysis for microservice architecture understanding and problem diagnosis. Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering; 2020.
[S33] Hasselbring W, Steinacker G. Microservice architectures for scalability, agility and reliability in e-commerce. 2017 IEEE International Conference on Software Architecture Workshops (ICSAW); 2017: IEEE.
[S34] He X, Tu Z, Xu X, Wang Z. Programming framework and infrastructure for self-adaptation and optimized evolution method for microservice systems in cloud–edge environments. Future Generation Computer Systems. 2021;118:263-81.
[S35] Jander K, Braubach L, Pokahr A. Defense-in-depth and role authentication for microservice systems. Procedia computer science. 2018;130:456-63.
[S36] Jiang P, Shen Y, Dai Y. Efficient software test management system based on microservice architecture. 2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC); 2022: IEEE.
[S37] Kakivaya G, Xun L, Hasha R, Ahsan SB, Pfleiger T, Sinha R, et al. Service fabric: a distributed platform for building microservices in the cloud. Proceedings of the thirteenth EuroSys conference; 2018.
[S38] Karwowski W, Rusek M, Dwornicki G, Orłowski A. Swarm based system for management of containerized microservices in a cloud consisting of heterogeneous servers. International Conference on Information Systems Architecture and Technology; 2017: Springer.

[S39] Khaleq AA, Ra I. Development of QoS-aware agents with reinforcement learning for autoscaling of microservices on the cloud. 2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C); 2021: IEEE.
[S40] Khaleq AA, Ra I. Intelligent autoscaling of microservices in the cloud for real-time applications. IEEE Access. 2021;9:35464-76.
[S41] Kim Y, Park J, Yoon J, Kim J. Improved Q network auto-scaling in microservice architecture. Applied Sciences. 2022;12(3):1206.
[S42] Kitajima S, Matsuoka N. Inferring calling relationship based on external observation for microservice architecture. Service-Oriented Computing: 15th International Conference, ICSOC 2017, Malaga, Spain, November 13–16, 2017, Proceedings; 2017: Springer.
[S43] Lei C, Dai H. A heuristic services binding algorithm to improve fault-tolerance in microservice based edge computing architecture. 2020 IEEE World Congress on Services (SERVICES); 2020: IEEE.
[S44] Lin T, Leon-Garcia A. Towards a client-centric QoS auto-scaling system. NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium; 2020: IEEE.
[S45] Liu Z, Fan G, Yu H, Chen L. An approach to modeling and analyzing reliability for microservice-oriented cloud applications. Wireless Communications Mobile Computing. 2021;2021(1):5750646.
[S46] López MR, Spillner J. Towards quantifiable boundaries for elastic horizontal scaling of microservices. Companion Proceedings of the10th International Conference on Utility and Cloud Computing; 2017.
[S47] Ma S-P, Fan C-Y, Chuang Y, Lee W-T, Lee S-J, Hsueh N-L. Using service dependency graph to analyze and test microservices. 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC); 2018: IEEE.
[S48] Ma S-P, Fan C-Y, Chuang Y, Liu I-H, Lan C-W. Graph-based and scenario-driven microservice analysis, retrieval, and testing. Future Generation Computer Systems. 2019;100:724-35.
[S49] Ma SP, Liu IH, Chen CY, Wang YT. Version-based and risk-enabled testing, monitoring, and visualization of microservice systems. Journal of Software: Evolution Process. 2022;34(10):e2429.
[S50] El Malki A, Zdun U. Evaluation of api request bundling and its impact on performance of microservice architectures. 2021 IEEE International Conference on Services Computing (SCC); 2021: IEEE.
[S51] El Malki A, Zdun U, Pautasso C. Impact of api rate limit on reliability of microservices-based architectures. 2022 IEEE International Conference on Service-Oriented System Engineering (SOSE); 2022: IEEE.
[S52] Márquez G, Astudillo H. Identifying availability tactics to support security architectural design of microservice-based systems. Proceedings of the 13th European Conference on Software Architecture-Volume 2; 2019.
[S53] Mayer B, Weinreich R. A dashboard for microservice monitoring and management. 2017 IEEE International Conference on Software Architecture Workshops (ICSAW); 2017: IEEE.
[S54] Meiklejohn C, Stark L, Celozzi C, Ranney M, Miller H. Method overloading the circuit. Proceedings of the 13th Symposium on Cloud Computing; 2022.
[S55] Mendonca NC, Aderaldo CM, Cámara J, Garlan D. Model-based analysis of microservice resiliency patterns. 2020 IEEE International Conference on Software Architecture (ICSA); 2020: IEEE.
[S56] Merkouche S, Bouanaka C. A Hybrid approach for containerized Microservices auto-scaling. 2022 IEEE/ACS 19th International Conference on Computer Systems and Applications (AICCSA); 2022: IEEE.
[S57] Merkouche S, Haroun T, Bouanaka C, Smaali M. TERA-Scheduler for a Dependency-based Orchestration of Microservices. 2022 International Conference on Advanced Aspects of Software Engineering (ICAASE); 2022: IEEE.
[S58] Milić M, Makajić-Nikolić D. Development of a quality-based model for software architecture optimization: a case study of monolith and microservice architectures. Symmetry. 2022;14(9):1824.
[S59] Ortiz G, Boubeta-Puig J, Criado J, Corral-Plaza D, Garcia-de-Prado A, Medina-Bulo I, et al. A microservice architecture for real-time IoT data processing: A reusable Web of things approach for smart ports. Computer Standards Interfaces. 2022;81:103604.
[S60] Otterstad C, Yarygina T. Low-level exploitation mitigation by diverse microservices. Service-Oriented and Cloud Computing: 6th IFIP WG 214 European Conference, ESOCC 2017, Oslo, Norway, September 27-29, 2017, Proceedings 6; 2017: Springer.
[S61] Peuster M, Dröge C, Boos C, Karl H. Joint testing and profiling of microservice-based network services using TTCN-3. ICT Express. 2019;5(2):150-3.
[S62] Pulnil S, Senivongse T. A microservices quality model based on microservices anti-patterns. 2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE); 2022: IEEE.
[S63] Ranjitha K, Tammana P, Kannan PG, Naik P. A case for cross-domain observability to debug performance issues in microservices. 2022 IEEE 15th International Conference on Cloud Computing (CLOUD); 2022: IEEE.
[S64] Raharjo AB, Andyartha PK, Wijaya WH, Purwananto Y, Purwitasari D, Juniarta N. Reliability Evaluation of Microservices and Monolithic Architectures. 2022 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM); 2022: IEEE.
[S65] Almenares F. A Framework for Microservice Migration and Performance Assesment. Intelligent Environments 2020: Workshop Proceedings of the 16th International Conference on Intelligent Environments; 2020: IOS Press.
[S66] Samanta A, Tang J. Dyme: Dynamic microservice scheduling in edge computing enabled IoT. IEEE Internet of Things Journal. 2020;7(7):6164-74.

[S67] Santana C, Andrade L, Delicato FC, Prazeres C. Increasing the availability of IoT applications with reactive microservices. Service Oriented Computing Applications. 2021;15(2):109-26.
[S68] Shi Y, Guo Y, Lv L, Zhang K. An Efficient Resource Scheduling Strategy for V2X Microservice Deployment in Edge Servers. Future Internet. 2020;12(10):172.
[S69] da Silva MAP, Times VC, de Araújo AMC, da Silva PC. A Microservice-based approach for increasing software reusability in health applications. 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA); 2019: IEEE.
[S70] Stang J, Walther D, Myrseth P. Data quality as a microservice: an ontology and rule based approach for quality assurance of sensor data in manufacturing machines. Proceedings of the 2nd International Workshop on Software Engineering and AI for Data Quality in Cyber-Physical Systems/Internet of Things; 2022.
[S71] Štefanič P, Kochovski P, Rana OF, Stankovski V. Quality of Service-aware matchmaking for adaptive microservice-based applications. Concurrency Computation: Practice Experience. 2021;33(19):e6120.
[S72] Stévant B, Pazat J-L, Blanc A. QoS-aware autonomic adaptation of microservices placement on edge devices. CLOSER 2020: 10th International Conference on Cloud Computing and Services Science; 2020.
[S73] Torkura KA, Sukmana MI, Meinel C. Integrating continuous security assessments in microservices and cloud native applications. Proceedings of the10th International Conference on Utility and Cloud Computing; 2017.
[S74] Vale G, Correia FF, Guerra EM, de Oliveira Rosa T, Fritzsch J, Bogner J. Designing microservice systems using patterns: an empirical study on quality trade-offs. 2022 IEEE 19th International Conference on Software Architecture (ICSA); 2022: IEEE.
[S75] Vassiliou-Gioles T. Quality assurance of micro-services-when to trust your micro-service test results? 2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C); 2021: IEEE.
[S76] Walker A, Das D, Cerny T. Automated code-smell detection in microservices through static analysis: A case study. Applied Sciences. 2020;10(21):7800.
[S77] Wang C, Jia B, Yu H, Li X, Wang X, Taleb T. Deep reinforcement learning for dependency-aware microservice deployment in edge computing. GLOBECOM 2022-2022 IEEE Global Communications Conference; 2022: IEEE.
[S78] Waseem M, Liang P, Shahin M, Ahmad A, Nassab AR. On the nature of issues in five open source microservices systems: an empirical study. Proceedings of the 25th International Conference on Evaluation and Assessment in Software Engineering; 2021.
[S79] Waseem M, Liang P, Shahin M, Di Salle A, Márquez G. Design, monitoring, and testing of microservices systems: The practitioners' perspective. Journal of Systems Software. 2021;182:111061.
[S80] Xu M, Song C, Ilager S, Gill SS, Zhao J, Ye K, et al. CoScal: Multifaceted scaling of microservices with reinforcement learning. IEEE Transactions on Network Service Management. 2022;19(4):3995-4009.
[S81] Yarygina T, Otterstad C. A game of microservices: Automated intrusion response. Distributed Applications and Interoperable Systems: 18th IFIP WG 61 International Conference, DAIS 2018, Held as Part of the 13th International Federated Conference on Distributed Computing Techniques, DisCoTec 2018, Madrid, Spain, June 18-21, 2018, Proceedings 18; 2018: Springer.
[S82] Yilmaz R, Buzluca F. A fuzzy quality model to measure the maintainability of microservice architectures. 2021 2nd International Informatics and Software Engineering Conference (IISEC); 2021: IEEE.
[S83] Yu G, Chen P, Zheng Z. Microscaler: Automatic scaling for microservices with an online learning approach. 2019 IEEE International Conference on Web Services (ICWS); 2019: IEEE.
[S84] Zhang Y, Hua W, Zhou Z, Suh GE, Delimitrou C. Sinan: ML-based and QoS-aware resource management for cloud microservices. Proceedings of the 26th ACM international conference on architectural support for programming languages and operating systems; 2021.
[S85] Zheng T, Zheng X, Zhang Y, Deng Y, Dong E, Zhang R, et al. SmartVM: a SLA-aware microservice deployment framework. World Wide Web. 2019;22(1):275-93.

REFERENCES

1. Capuano R, Muccini H. A systematic literature review on migration to microservices: a quality attributes perspective. 2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C); 2022: IEEE.
2. Agarwal S, Sinha R, Sridhara G, Das P, Desai U, Tamilselvam S, et al. Monolith to microservice candidates using business functionality inference. 2021 IEEE International Conference on Web Services (ICWS); 2021: IEEE.
3. Weerasinghe L, Perera I. Evaluating the inter-service communication on microservice architecture. 2022 7th International Conference on Information Technology Research (ICITR); 2022: IEEE.
4. Christoforou A, Odysseos L, Andreou AS. Migration of software components to microservices: Matching and synthesis. 2019.
5. Kalia AK, Xiao J, Krishna R, Sinha S, Vukovic M, Banerjee D. Mono2micro: a practical and effective tool for decomposing monolithic java applications to microservices. Proceedings of the 29th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering; 2021.
6. Laigner R, Kalinowski M, Diniz P, Barros L, Cassino C, Lemos M, et al. From a monolithic big data system to a microservices event-driven architecture. 2020 46th Euromicro conference on software engineering and advanced applications (SEAA); 2020: IEEE.
7. Li Z, Shang C, Wu J, Li Y. Microservice extraction based on knowledge graph from monolithic applications. Information Software Technology. 2022;150:106992.
8. Volynsky E, Mehmed M, Krusche S. Architect: a framework for the migration to microservices. 2022 International Conference on Computing, Electronics & Communications Engineering (iCCECE); 2022: IEEE.
9. Pahl C, Jamshidi P. Microservices: A Systematic Mapping Study. CLOSER. 2016:137-46.
10. Alshuqayran N, Ali N, Evans R. A systematic mapping study in microservice architecture. 2016 IEEE 9th international conference on service-oriented computing and applications (SOCA); 2016: IEEE.
11. Di Francesco P, Malavolta I, Lago P. Research on architecting microservices: Trends, focus, and potential for industrial adoption. 2017 IEEE International conference on software architecture (ICSA); 2017: IEEE.
12. Li S, Zhang H, Jia Z, Zhong C, Zhang C, Shan Z, et al. Understanding and addressing quality attributes of microservices architecture: A Systematic literature review. Information software technology. 2021;131:106449.
13. Petersen K, Feldt R, Mujtaba S, Mattsson M. Systematic mapping studies in software engineering. 12th international conference on evaluation and assessment in software engineering (EASE); 2008: BCS Learning & Development.
14. Busch A, Kammerer M. Network performance influences of software-defined networks on micro-service architectures. Proceedings of the ACM/SPEC International Conference on Performance Engineering; 2021.
15. Simon M, Spallina A, Dubocquet L, Araldo A. Parsimonious edge computing to reduce microservice resource usage. 2021 33th International Teletraffic Congress (ITC-33); 2021: IEEE.
16. Houmani Z, Balouek-Thomert D, Caron E, Parashar M. Enhancing microservices architectures using data-driven service discovery and QoS guarantees. 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID); 2020: IEEE.
17. Sheshadri K, Lakshmi J. Qos aware faas for heterogeneous edge-cloud continuum. 2022 IEEE 15th International Conference on Cloud Computing (CLOUD); 2022: IEEE.
18. Al Maruf A, Bakhtin A, Cerny T, Taibi D. Using microservice telemetry data for system dynamic analysis. 2022 IEEE International Conference on Service-Oriented System Engineering (SOSE); 2022: IEEE.

AJMAS
https://journal.utripoli.edu.ly/index.php/Alqalam/index eISSN 2707-7179

فهم وتحقيق عوامل الجودة في هيكلية الخدمات المصغرة

أشرف علي بوراوي

قسم الحاسوب، كلية العلوم، جامعة عمر المختار، البيضاء، ليبيا

المستخلص

اكتسبت بنية الخدمات المصغرة شعبية متزايدة بسبب خصائصها ومزاياها المفيدة. فهي تدعم بناء خدمات مستقلة ومقترنة بشكل غير متماسك وذات مهمة واحدة مع تتبع تحسين التوافر وقابلية التوسع والتغلب على الأخطاء. وقد أُجريت العديد من الدراسات البحثية لمعالجة الانتقال من الخدمات الأحادية إلى الخدمات المصغرة والمساهمة في جودة هذه الخدمات. بالإضافة إلى ذلك، تهدف معظم الدراسات السابقة إلى التغلب على التحديات والمشاكل التي تواجه تطوير الخدمات المصغرة ونشرها. ومع ذلك، هناك فجوة في الأبحاث حيث لم يتم التحقيق بدقة في الإدراك المنهجي لعوامل الجودة المتعلقة ببنية الخدمات المصغرة. تهدف هذه الدراسة إلى معالجة وفهم عوامل الجودة في الأنظمة القائمة على الخدمات المصغرة. ولتحقيق هدفنا، أجرينا دراسة منهجية لتحديد وبناء دراسة حديثة بناءً على الدراسات السابقة ذات الصلة. استنادًا إلى نتائج البيانات المستخرجة من 85 دراسة ذات صلة، تم تحديد عشرة عوامل للجودة ومناقشتها بشكل أكبر. هذه العوامل التي تم التعرف عليها هي: قابلية التوسع، كفاءة الأداء، وقابلية المراقبة، والتوافر، وقابلية الاختبار، والموثوقية، والأمن، وقابلية الصيانة، والتسامح مع الأخطاء، وقابلية إعادة الاستخدام. علاوة على ذلك، تم تحديد التحديات المختلفة وتقديم توصيات لمعالجة هذه القضايا. على الرغم من تناول عوامل الجودة المختلفة، فقد خلصنا إلى أنه ينبغي إيلاء المزيد من الاهتمام لبعض العوامل مثل الأمان والتوافر والموثوقية وقابلية إعادة الاستخدام. بالإضافة إلى ذلك، ينبغي معالجة المفاضلة بين بعض العوامل مثل قابلية التوسع وكفاءة الأداء لتحسين الأداء العام للنظام وتجنب اضعاف الخدمة في الانظمة المعتمدة على بنية الخدمات المصغرة.

الكلمات الدالة. بنية الخدمات المصغرة، الخدمات الأحادية، عوامل الجودة، بنية البرمجيات.