*ICIC Express Letters*
Volume **17**, Number **4**, April **2023**

 ISSN 1881-803X
pp. **463–470**

# AN INNOVATIVE METHODOLOGY FOR TRANSITIONING FROM MONOLITH TO MICROSERVICES

Youngpyo Hong and Dongsoo Kim*

Department of Industrial and Information Systems Engineering
Soongsil University
369 Sangdo-Ro, Dongjak-Gu, Seoul 06978, Korea
ypyohong@soongsil.ac.kr; *Corresponding author: dskim@ssu.ac.kr

Received August 2022; accepted October 2022

Abstract. *This paper presents a practical and trustworthy microservices transition methodology. Recently, a lot of organizations are striving to achieve business agility through the transition from monolithic to microservices architecture. However, proceeding without a validated methodology can fail due to the inherent complexity of microservices. In particular, in large-scale projects such as next-generation systems or in the financial sector where stability is a core competency, a reliable and effective methodology is a key success factor for building microservices. This paper proposes an improved approach to microservices transition methodology through candidate microservices evaluation, microservices simulation, and microservices process mining. The proposed methodology is expected to be an approach that can reduce the risk of microservices transition and achieve stable and successful microservices development.*

**Keywords:** Microservices methodology, Microservices transition, Microservices evaluation, Microservices simulation, Microservices process mining

1. **Introduction.** In the era of digital transformation, VUCA – Volatility, Uncertainty, Complexity and Ambiguity are accelerated at a much faster pace. Modern enterprises are attempting to transit their enterprise architecture from monolith to microservices in order to respond with agility to the rapidly changing business landscape. However, the transition to microservices requires major changes in the entire existing enterprise architecture, so the right methodology is a critical success factor in the transition to microservices. Therefore, the transition to microservices is generally driven in stages, as shown in Figure 1.

Based on the general microservices transition methodology shown in Figure 1, this paper proposes an innovative and improved approach for the practical microservices transition methodolog. The proposed methodology enables effective and reliable transitions to microservices even in large-scale projects such as next-generation systems or in financial companies where stability is a core competency.

This paper is structured as follows. Section 2 reviews related work. Section 3 describes an improved approach for microservices transition methodology composed of candidate microservices evaluation for verification of candidate microservices, microservices simulation for predicting the expected effect of microservices and understanding the impact of change, and microservices process mining for continuous improvement for microservices based on operation logs from the integrated log system. Finally, Section 4 offers conclusions of this research.

2. **Related Work.** Microservices are an architectural approach to building applications. As an architectural framework, microservices are distributed and loosely coupled, so one team's changes will not break the entire application. The benefit of using microservices is

DOI: 10.24507/icicel.17.04.463

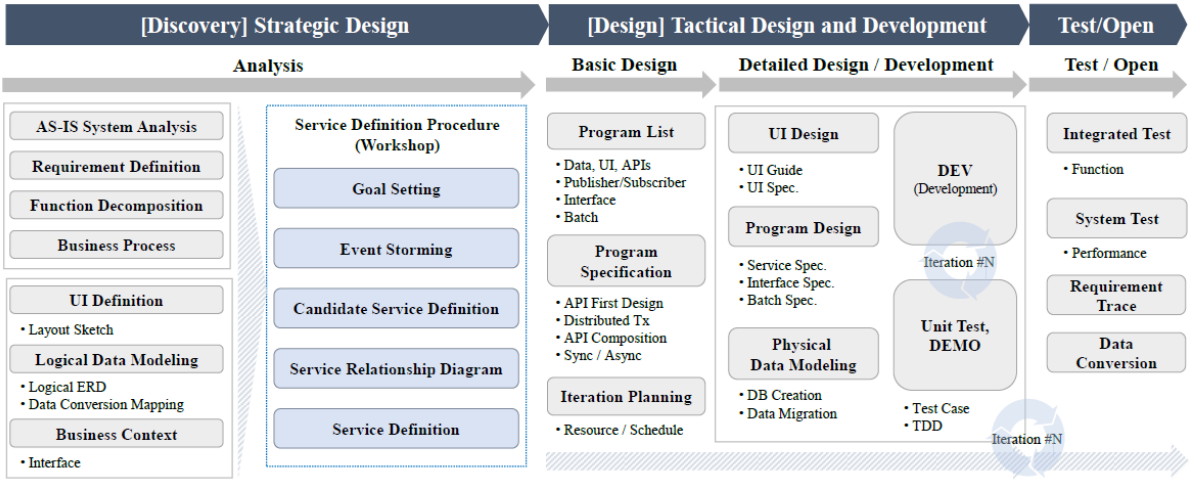



FIGURE 1. General microservices transition methodology

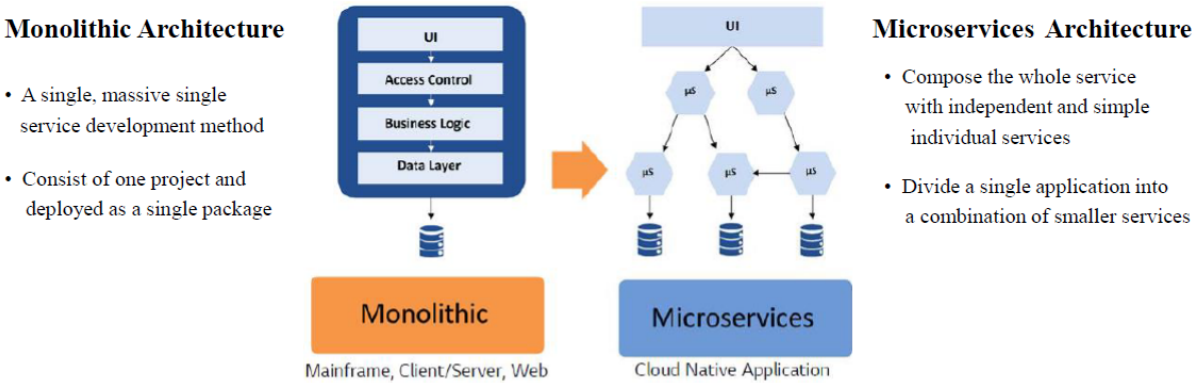



FIGURE 2. Monolithic vs. microservices architecture

that development teams can quickly build new components of applications to meet changing business requirements [16]. Microservices are organized as a suite of small services where each one runs in its own process and communicates with lightweight mechanisms. These services are built around business capabilities and are independently deployed [11]. The microservices architecture enables the rapid, frequent, and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack [9,10]. Figure 2 shows the difference between monolithic and microservices architecture.

A robust transition to microservices requires a comprehensive roadmap [11,13]. In addition, an evaluation framework can be utilized to verify the decomposition approach [5], simulation can also be used to assess the change impact on transition to microservices [15], and process mining helps to validate the process of microservices [1,7]. By combining the bottom-up strategy of evaluation, simulation, and process mining with the top-down strategy of transition roadmap, this paper proposes a practical and trustworthy methodology for microservices transition.

## 3. Proposed Microservices Transition Methodology.

The right methodology is essential for a successful transition to a microservices architecture [4]. This is because the decomposition and composition into microservices affect not only the enterprise architecture, but also the way organizations work. Therefore, after defining candidate microservices, rational and objective evaluation indicators for candidate microservices are needed. Then, simulations are performed on the derived candidate microservices to predict the

effect of transition to the microservices. Through microservices simulations, we can proactively respond to problems that may occur during the development stage and minimize risks. Finally, after the microservices transition is completed, in the operational phase, process mining for microservices is performed based on the integrated operation logs to derive improvements.

3.1. **Candidate microservices evaluation.** To derive microservices candidate tasks, a microservices workshop is conducted. Based on DDD (Domain Driven Design) concepts [2], the workshop is conducted in pairs between the person in charge of business and IT developers [8]. The steps of the workshop consist of goal setting, event storming, candidate service definition, service relationship diagram, and service definition. After defining the microservices, it is possible to evaluate the candidate microservices. Quantitative indicators are established for objective evaluation of candidate microservices, and the evaluation results are quantified to determine priorities. For this purpose, categories are divided into service identification criteria, transition feasibility, and correction factors. Then, the evaluation items are subdivided and evaluated for each item [6]. Table 1 shows specific microservices evaluation indicators.

Table 1. Microservices evaluation indicators

| Category | Evaluation items | Descriptions |
|---|---|---|
| **Service identification criteria** | **(1) Business capability** | Evaluate whether a microservice organized around business capabilities is well partitioned according to the single responsibility principle. |
| | **(2) Data ownership** | Evaluate whether a microservice stores and owns all data that belongs to the capability the microservice implements. |
| | **(3) Scalability** | Evaluate whether a microservice can be independently scalable by decomposing system functions. |
| | **(4) Fault isolation** | Evaluate whether a bug or error with one microservice does not adversely affect the rest of the application. |
| **Transition feasibility** | **(5) Coupling between services** | Evaluate whether the business processes or call relationships of microservices are independent. |
| **Correction factors** | **(6) Service request increase/decrease rate** | Evaluate whether the number of service change requests increases or decreases over a period of time. |
| | **(7) Service sensitivity** | Evaluate whether a service is critical to delays or errors. |

In the following, we present an example of evaluating candidate microservices and prioritizing suitable business units for transition to microservices. Detailed scoring methods are shown in Table 2.

The transition score, which is the final score, is obtained by dividing each item's score by the maximum item value, multiplying the result by 10, and rounding off to the first decimal place. The overall score table, including the transition score, is shown in Figure 3. In this example, based on the transition score, it can be determined that BU 2, BU 1, and BU 4 are the business units suitable for transition to microservices in that order.

TABLE 2. How to score business units by evaluating candidate microservices

| Category | Evaluation items | Descriptions |
|---|---|---|
| **Service identification criteria** | **(1) Business capability** | **[STEP 1]** Score from 0 point (lowest) to 10 points (highest) for each candidate service. **[STEP 2]** Calculate the average value by dividing the evaluation score by the number of candidate services. |
| | **(2) Data ownership** | |
| | **(3) Scalability** | |
| | **(4) Fault isolation** | |
| **Transition feasibility** | **(5) Coupling between services** | **[STEP 1]** Calculate the coupling point by dividing the number of synchronizations by the number of candidate services. **[STEP 2]** Subtract each coupling point from the maximum value of coupling points. |
| **Correction factors** | **(6) Service request increase/decrease rate** | Rate of increase/decrease in the number of service change requests over the last year. |
| | **(7) Service sensitivity** | Score from 0 point (lowest) to 10 points (highest) for each business unit. |

| Business Unit | Number of Candidate Services | Transition Score | | | | | | | | Service Identification Criteria | | | | Transition Feasibility | | | | | | Correction Factors |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Service Identification Criteria | | | | Transition Feasibility | Correction Factors | | Score | | | | Number of Couplings | | | | Score | | Score |
| | | Total Score | (1) Business Capability | (2) Data Ownership | (3) Scalability | (4) Fault Isolation | (5) Coupling between Services | (6) Service Request Increase/Decrease Rate | (7) Service Sensitivity | (1) Business Capability | (2) Data Ownership | (3) Scalability | (4) Fault Isolation | Sum | Synchronization | Asynchronization | Others | Coupling Point | (5) Coupling between Services | (6) Service Request Increase/Decrease Rate |
| | | 70 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | | | | | | | | | | | |
| BU 1 | 6 | 43 | 8 | 7 | 3 | 8 | 6 | 10 | 1 | 6.7 | 3.3 | 1.7 | 3.3 | 31 | **14** | 16 | 1 | 2.3 | 2.2 | 0.939 |
| BU 2 | 14 | 49 | 9 | 1 | 6 | 10 | 8 | 10 | 5 | 7.1 | 0.7 | 3.6 | **4.3** | 51 | **20** | 26 | 5 | 1.4 | 3.1 | **0.942** |
| BU 3 | 12 | 30 | 7 | 3 | 3 | 8 | 9 | 0 | 0 | 5.8 | 1.7 | 1.7 | 3.3 | 42 | **13** | 28 | 1 | 1.1 | 3.4 | (0.030) |
| BU 4 | 6 | 33 | 10 | 0 | 6 | 8 | 9 | 0 | 0 | **8.3** | 0.0 | 3.3 | 3.3 | 31 | **8** | 23 | 0 | 1.3 | 3.2 | 0.000 |
| BU 5 | 13 | 26 | 9 | 3 | 4 | 0 | 10 | 0 | 0 | 7.7 | 1.5 | 2.3 | 0.0 | 52 | **11** | 39 | 2 | 0.8 | **3.7** | (0.030) |
| BU 6 | 5 | 31 | 10 | 0 | 10 | 5 | 4 | 2 | 0 | 8.0 | 0.0 | **6.0** | 2.0 | 37 | **16** | 15 | 6 | 3.2 | 1.3 | 0.211 |
| BU 7 | 2 | 24 | 6 | 10 | 8 | 0 | 0 | 0 | 0 | 5.0 | **5.0** | 5.0 | 0.0 | 25 | **9** | 16 | 0 | **4.50** | 0.0 | (0.162) |

FIGURE 3. An example of calculating transition score based on Table 2

3.2. **Microservices simulation.** The transition to microservices inevitably comes with an increase in architectural complexity, so it is very difficult to modify or fix the service after it has been developed [14]. Therefore, it is necessary to recognize problems in advance through simulations from the perspective of the enterprise architecture, including the relationship between microservices before development. For impact prediction for derived candidate microservices, simulations can be performed for static analysis, dynamic analysis, and process analysis. Through simulation analysis, it is possible to predict the expected effect of microservices transition and understand the impact on change [3]. Figure 4 shows the analysis targets and detailed outputs for microservices simulation.

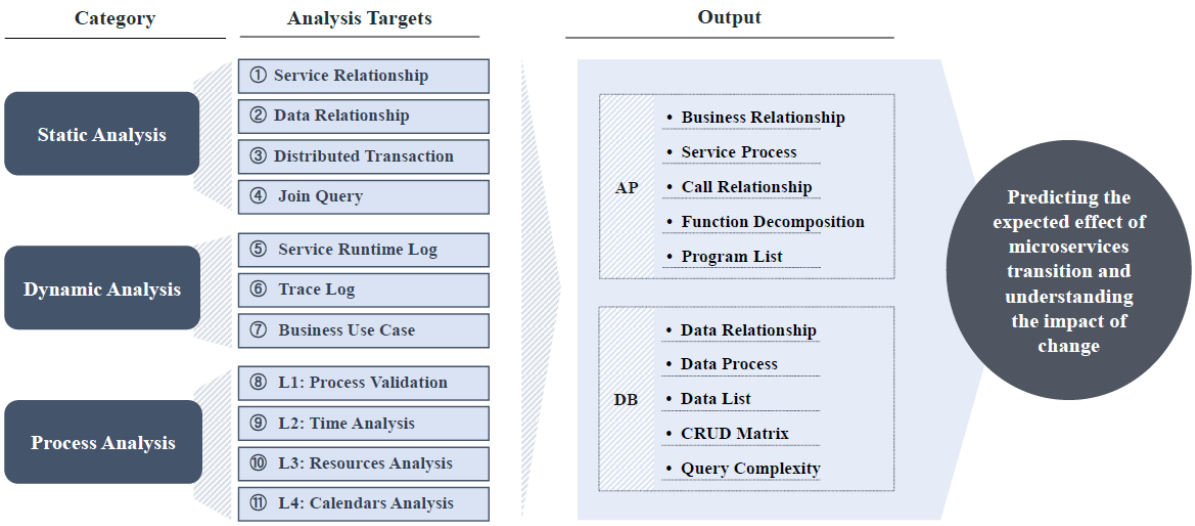
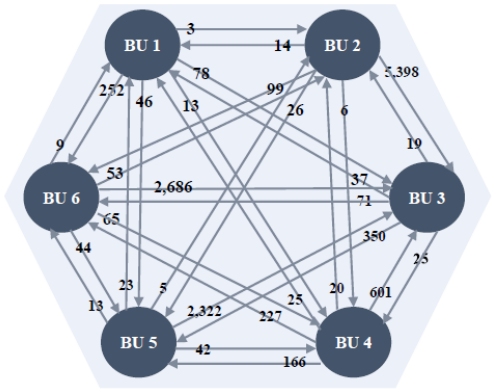



FIGURE 4. Targets and outputs for microservices simulation

Figure 5 shows an example of simulation analysis of service relationship between each business unit described in Section 3.1. The left side of Figure 5 is a visualization of the call relationship between services of business units, and the right side is a tabular representation of the relationship between the caller and callee on the left side. According to Figure 5, it can be confirmed that the number of callee relationships is small in the order of BU 2, BU 1, and BU 4. These simulation results are consistent with the evaluation results of Section 3.1. For loosely coupled architecture, it can be determined that it is appropriate to transit to microservices in the order of BU 2, BU 1, and BU 4.




| Category | | CALLEE | | | | | |
|---|---|---|---|---|---|---|---|
| | | BU 1 | BU 2 | BU 3 | BU 4 | BU 5 | BU 6 |
| CALLER | BU 1 | | 3 | 78 | 13 | 46 | 252 |
| | BU 2 | 14 | | 5,398 | 6 | 26 | 99 |
| | BU 3 | 37 | 19 | | 25 | 350 | 71 |
| | BU 4 | 25 | 20 | 601 | | 166 | 227 |
| | BU 5 | 23 | 5 | 2,322 | 42 | | 13 |
| | BU 6 | 9 | 53 | 2,686 | 65 | 44 | |
| SUM | | 108 | 100 | 11,085 | 151 | 632 | 662 |

FIGURE 5. An example of simulation analysis for call relationship

3.3. **Microservices process mining.** Microservices are a powerful design paradigm that decomposes monolithic systems into smaller, independent services. Microservices can be built and deployed independently, but they are as complex as they are flexible. Therefore, it is very important to link the log generated by each microservice and trace them in order from the beginning to the end of the transaction. Consequently, it is necessary to establish log standards and integration plans from the beginning of the project to enable integrated monitoring and analysis after the microservices transition. After building an integrated log system, process mining can be applied using integrated operation logs to derive improvements in microservices [17,18]. Figure 6 shows microservices process mining based on log standardization and the integrated log system.

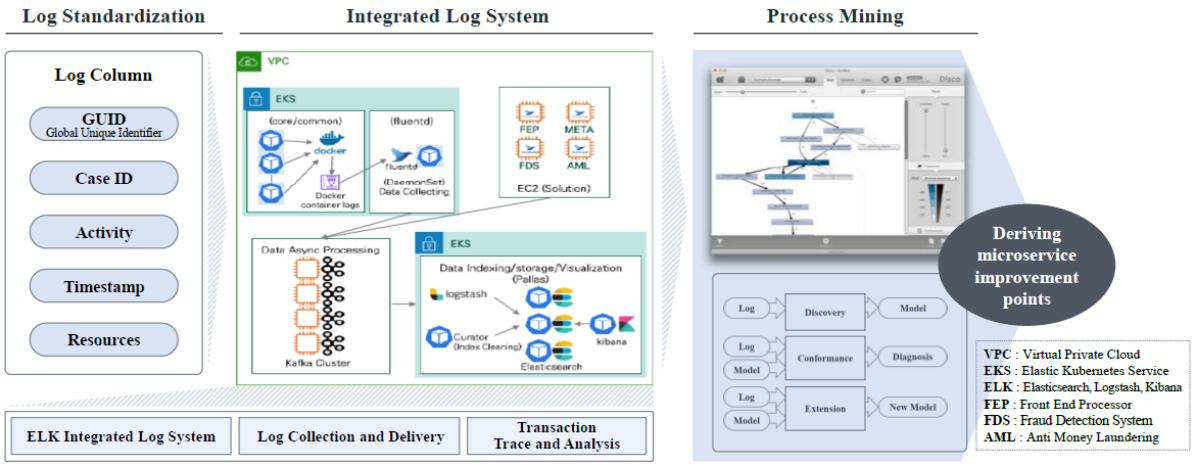



FIGURE 6. Conceptual diagram of microservices process mining

In summary, an overview of the microservices transition methodology improved by applying the approach described in Section 3, candidate microservices evaluation, microservices simulation, microservices process mining, is shown in Figure 7.

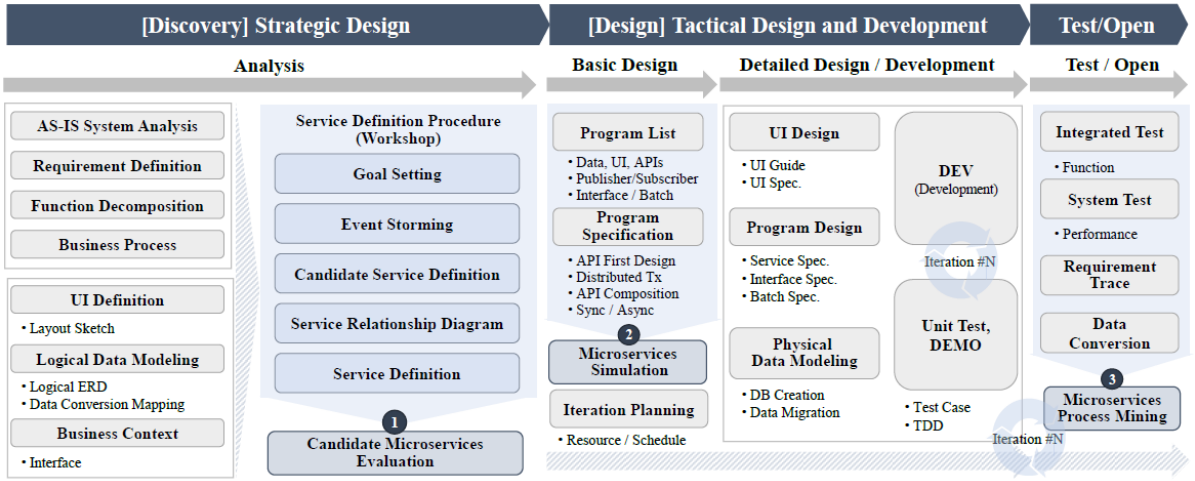



FIGURE 7. Proposed methodology for microservices transition

The three improvement approaches presented in Figure 7 can also be summarized in a Plan-Do-See value chain as shown in Table 3. Gap analysis between each step of the value chain helps uncover discrepance between microservices transition stages and mitigate the risk of transition.

TABLE 3. The proposed improvement approach of the Plan-Do-See perspective

| Category | Plan | Do | See |
|---|---|---|---|
| **Input** | Candidate services | Code, DB, and model | Log |
| **Approach** | Evaluation | Simulation | Process mining |
| **Output** | Verification and prioritization | Change impact assessment | Validation and improvement |

4. **Conclusions.** The proposed methodology for transition from monolith to microservices architecture overcomes the limitations of the theoretical methodology and provides a practical approach. In particular, the methodology is highly feasible because it reflects

lessons learned from experience of building microservices. In addition, it provides a systematic approach for the successful transition from monolithic to microservices architecture, not only in small organizations, but also in large-scale projects, or financial companies with conservative cultures.

Although a microservices architecture is a great way to improve business agility, it also increases the potential risk due to the inevitable increase in architectural complexity. Therefore, a right microservices transition methodology with a clear purpose of the transition is required. As a concrete action plan, this paper proposed an innovative approach to improve the microservices transition methodology through candidate microservices evaluation, microservices simulation, and microservices process mining.

The proposed methodology can be a successful guideline on a virtuous cycle of microservices transition. It is expected to quantitatively evaluate the candidate microservices and determine the scope of transition, predict the impact on microservices transition in advance, and constantly check processes for microservices with integrated operation logs. Moreover, the proposed Plan-Do-See value chain-based microservices transition methodology enables rapid response to problems and continuous improvement. The innovative methodology presented in this paper will be an effective and efficient procedure for transition to microservices to improve business agility while mitigating the risk of transition.

**Acknowledgment.** This work is supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A1B05029080).

## REFERENCES

[1] A. Khattar, *Process Mining for a Data-Driven Migration Strategy to Microservices*, https://medium.com/@anaskhattar/process-mining-for-a-data-driven-migration-strategy-to-microservices-2fc086d827a9, Accessed on August 10, 2022.

[2] B. Hippchen, P. Giessler, R. H. Steinegger, M. Schneider and S. Abeck, Designing microservice-based applications by using a domain-driven design approach, *International Journal on Advances in Software*, vol.10, nos.3-4, pp.432-445, 2017.

[3] C. Courageux-Sudan, A.-C. Orgerie and M. Quinson, Automated performance prediction of microservice applications using simulation, *Proc. of the 2021 29th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp.1-8, 2021.

[4] C. Schöer, S. Wittfoth and J. M. Gómez, A process model for microservices design and identification, *Proc. of the 2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)*, pp.1-8, 2021.

[5] C.-H. Im, Implementation and evaluation of an application framework based on micro service architecture, *Journal of Information Technology and Architecture*, vol.18, no.4, pp.309-318, 2021.

[6] D. Taibi and K. Systä, A decomposition and metric-based evaluation framework for microservices, *Proc. of the International Conference on Cloud Computing and Services Science*, pp.133-149, 2019.

[7] D. Taibi and K. Systä, From monolithic systems to microservices: A decomposition framework based on process mining, *Proc. of the International Conference on Cloud Computing and Services Science*, pp.153-164, 2019.

[8] D. Taibi, V. Lenarduzzi, C. Pahl and A. Janes, Microservices in agile software development: A workshop-based study into issues, advantages, and disadvantages, *Proc. of the XP2017 Scientific Workshops*, 2017.

[9] E. D. Giovanni and I. B. K. Manuaba, Event-driven approach in microservices architecture for flight booking simulation, *ICIC Express Letters*, vol.16, no.5, pp.545-553, 2022.

[10] H. Suhendinata and G. P. Kusuma, Serverless microservices architecture for indoor positioning system using Bluetooth low energy, *ICIC Express Letters*, vol.16, no.9, pp.1001-1009, 2022.

[11] K. Bozan, K. Lyytinen and G. M. Rose, How to transition incrementally to microservice architecture, *Communications of the ACM*, vol.64, no.1, pp.79-85, 2020.

[12] B. Ma, H. Ni, X. Zhu, Z. Wang and Z. Wang, A transformed salp swarm algorithm on container deployment problem, *International Journal of Innovative Computing, Information and Control*, vol.16, no.1, pp.283-299, 2020.
[13] M. L. Fanomezana, A. M. Rapatsalahy, N. R. Razafindrakoto and C. Bãdiã, Proposed methodology for designing a microservice architecture, *Proc. of the 2022 23rd International Carpathian Control Conference (ICCC)*, pp.303-308, 2022.
[14] M. Kalske, N. Mäkitalo and T. Mikkonen, Challenges when moving from monolith to microservice architecture, *Current Trends in Web Engineering*, pp.32-47, 2018.
[15] P. Bocciarelli, A. D'Ambrogio, E. Paglia and A. Giglio, A service-in-the-loop approach for business process simulation based on microservices, *Proc. of the 50th Computer Simulation Conference*, 2018.
[16] Red Hat, *What Are Microservices?*, https://www.redhat.com/en/topics/microservices/what-are-microservices, Accessed on August 10, 2022.
[17] S. Kim and D. Kim, Analyzing mobile application logs using process mining techniques: An application to online bookstores, *ICIC Express Letters, Part B: Applications*, vol.9, no.6, pp.607-614, 2018.
[18] V. Raj and G. P. Chander, Monitoring of microservices architecture based applications using process mining, *Proc. of the 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp.486-494, 2022.