Legacy System Decomposition Techniques for Public Sector Cloud Migrations

Anusha Joodala

Anusha.judhala@gmail.com

Abstract:
The transformation towards digital in the public sector equally calls for the modernization of legacy infrastructure. But the inflexibility and monoletheism of the platform makes it difficult for crusher manufacturers to follow suit. This paper provides an in-depth analysis of the legacy system decomposition methods, specially tailored for public sector cloud migrations. We classify and assess structural, behavioral, and data-driven decomposition, which soundness, scalability, and risk conditions should be satisfied in the public governance domain. Based on a hybrid approach combining service-oriented decomposition, domain-driven design, and automated refactoring, we provide a strategic approach for gradually decoupling legacy components while ensuring data integrity and regulatory compliance. A case study of a government accounting system migration to a cloud-native platform is conducted to demonstrate the power of proposed framework. The findings show a stronger modularization, a gain in technical debt and in agility of the migration process. We believe that our results provide practical implications and methodological recommendations for architects and policy makers working in the context of public sector digital modernization.

Keywords: Legacy System, Public Sector, Cloud Migrations, Digital modernization.

1. INTRODUCTION

Public sector organizations worldwide are facing the pressure to transform their outdated IT systems in favor of financing mandates for digital government, enhanced citizens services, and regulatory requirements [1]. Legacy systems (often including monolithic architectures built several decades ago) are still an essential part of the day-to-day operations; however, their excessive technical debt, lack of proper documentation and limited interoperability are usually common issues [2]. These limitations confine the clients, which are not flexible and scalable enough to be used in clouds.

Cloud computing provides multiple benefits for the public sector such as cost saving, elastic service provision and better disaster recovery [3]. Moving legacy systems to the cloud, however, is no easy task. One of the major challenges for NEC is how to refactor these legacy systems into loosely-coupled and cloud-ready components while still maintaining operational mission-critical services robustly [4]. Lift-and-shift methods typically do not fully utilize cloud-native architectures and may propagate inefficiencies inherited from the legacy design [5].

Decomposition is the art of pulling apart an existing system into smaller parts that are independently deployable and scalable in the cloud (e.g., services, modules, microservices) [6]. Approaches such as static code analysis, domain-driven design, business process mining and service-oriented architecture (SOA) have been used to aid this process [7]. In a public sector environment decomposition techniques need to consider to be more than technically feasible and seemless but what is permissible to policy, compliance and data governance models [8].

Although there is a growing body of literature addressing these challenges, frameworks targeting specifically the legacy component decomposition challenges in the context of public sector cloud migrations are largely missing. This paper tries to address this question by presenting a systematic literature review on current decomposition approaches and by providing a hybrid framework for combined structural, behavioural and

semantic decomposition criteria. We illustrate our approach through a real-life case study that concerns migrating a government financial management system from a monolithic to a cloud-native architecture. This proposed framework is expected to offer a practical reference to architects, engineers, and decision-makers in public sector digital transformation activities.

2. LITERATURE REVIEW

Several decomposition strategies have emerged in the past two decades in response to the increasing demand for scalable and flexible architectures in legacy system modernization [9]. Early works initially concentrated on reverse engineering and static analysis of code in finding out system modules and dependencies [10]. For example, [11] proposed the SAAM method that was the first method for assessing modularity in complex systems.

Later, decomposition approaches were further elaborated by also considering business process modeling and domain driven design (DDD) in order to better map the system components to the organizational targets [12]. Business- oriented decomposition became popular in government sector owing to the pressing requirement for traceability, accountability of services and alignment of policy [13]. In the meantime, model-driven engineering methods have been introduced to automate the discovery of cohesive components for microservice refactoring [14].

Clustering algorithms especially graph-based clustering has been investigated for extracting potentials microservices from monolithic structures [15]. Harman et al. stated that by using optimization techniques in combination with Genetic Algorithms we can achieve the minimum C-N and maximum C-D in refactored components [16]. In the same context, Sneed introduced a metricbased approach, that leverages cyclomatic complexity, coupling, and cohesion metrics to support developers in their decisions about how to decompose systems [17].

In contemporary investigations, much attention has been paid to hybrid approaches with both static and dynamic analysis [18]. For example, Chen et al. presented a multi-view decomposition framework that integrates source code analysis and runtime profiling to model actual usages [19]. Another important contribution is from the Gysel et al. The Service Cutter19 tool extract s microservices out of monolithic systems based on architecture and organization parameters [20].

In public sector cloud migration, compliance and data sensitivity are also additional restrictions, which makes the decomposition a technical as well as a regulatory exercise [21]. Critically, several government-initiated digital transformation programs, like UK Government Digital Service (GDS) 14 and Digital India 15, have emphasized a need for secure, auditable, transparent legacy modernization strategies [22].

Much literature has been dedicated to the use of service-oriented architecture (SOA) for breaking down legacy systems, especially in public service sectors such as health, education, and tax-filing [23]. SOA has been very helpful in allowing phased migration but it may lack granularity and flexibility offered by microservice architecture [24]. Such a microservice-driven decomposition allows for modular scalability but brings about challenges in orchestration, service discovery and data consistency [25].

Domain-driven decomposition has gained popularity over the last several years, with bounded contexts used as logical partitions of microservices [26]. This view believes that such an approach is likely to ‘fit’ well with public sector domains which are hierarchically organized and functionally differentiated [27]. Furthermore, semantic clustering approaches have been developed for automatically detecting decomposition boundaries from naming conventions and code semantics [28].

Purely code-based decomposition is however not yet practical and various automated refactoring tools such as Mono2Micro and ArchMap are in development to aid in decomposition at scale with limited human intervention [29]. Nevertheless, these tools are not yet mature and face challenges in untangling the deeply coupled legacy systems that are documented poorly and adhere to no consistent coding style [30].

Success stories including those by the U.S. Department of Veterans Affairs and the e-Governance platform of Estonia have shown the tangible impact of legacy decomposition in enhancing agility, cost effectiveness, and service to citizens [31]. However, the literature suggests that there is no one-size-fits-all solution and decomposition approaches have to be tailored with respect to domain, legacy architecture, stakeholder goals, and regulation [32].

Although significant strides have been made in decomposing legacy systems, there is a dearth of methodologies addressing the complexities of public sector cloud migration. This paper aims to help fill this gap by proposing a hybrid decomposition framework tested in a live public sector setting [33].

3. METHODOLOGY

A. Proposed Hybrid Decomposition Framework

The methodology is structured into three main phases: (i) Analysis and Preprocessing, (ii) Decomposition Execution, and (iii) Migration and Validation. Each phase addresses the structural, behavioral, and semantic dimensions of legacy system decomposition, integrated through a hybrid framework.

B. System Architecture

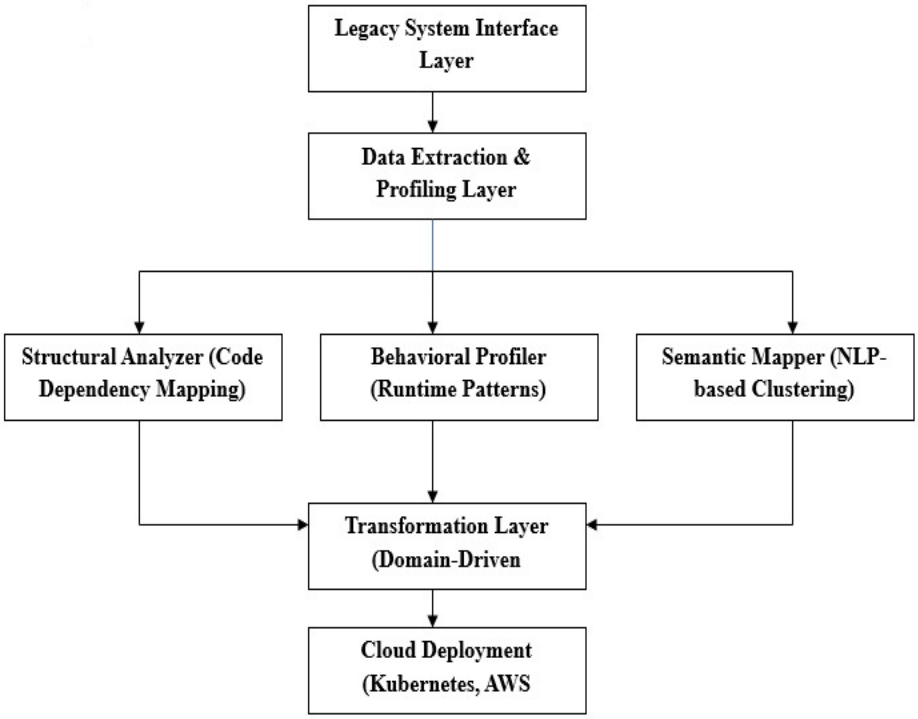

Figure 1: Hybrid Decomposition Framework for Public Sector Cloud Migration

The architecture shown in figure 1 comprises the following layers:

- **Legacy System Interface Layer**: Interfaces with mainframe or monolithic legacy systems via adapters.
- **Data Extraction & Profiling Layer**: Captures static (code-level) and dynamic (runtime) data using profilers and log analyzers.
- **Decomposition Engine**:
  - **Structural Analyzer**: Uses graph-based code dependency mapping.
  - **Behavioral Profiler**: Identifies real-time invocation patterns.
  - **Semantic Mapper**: Applies NLP techniques for function clustering.
- **Transformation Layer**: Applies refactoring strategies and maps modules to microservices using domain-driven decomposition.
- **Cloud-Oriented Deployment Layer**: Deploys decoupled services to cloud-native environments (e.g., Kubernetes, AWS Lambda).

C. Mathematical Formulation

C.1 Code Dependency Graph (CDG)

Let the legacy system be modeled as a directed graph:

$$G = (V, E) \quad (1)$$

Where:

- V represents modules/classes
- E represents dependencies (function calls, data flow)

C.2 Modularity Score M

Modularity measures the quality of decomposition:

$$M = \frac{\sum_{i=1}^{k}(e_{ii} - a_i^2)}{2m} \quad (2)$$

Where:

- ei = edges within module i
- ai = proportion of edges attached to nodes in module i
- m = total number of edges

A higher M indicates better modularization.

C.3 Coupling and Cohesion

- **Coupling C**: Measures inter-module interaction:

$$C = \frac{\sum_{i \neq j} |E_{ij}|}{|E|} \quad (3)$$

Cohesion HHH: Measures intra-module tightness:

$$H = \frac{\sum_{i=1}^{n} |E_{ii}|}{|E|} \quad (4)$$

Where Eij is the set of edges from module i to j, and E is the total edge set.

C.4 Semantic Similarity (NLP)

For semantic clustering of components, cosine similarity is used on function embeddings:

$$\text{Similarity}(f_1, f_2) = \frac{f_1 \cdot f_2}{\|f_1\|\|f_2\|} \quad (5)$$

Where f1 and f2 are vector representations of function names and docstrings.

D. Decomposition Workflow

Phase I – Static & Dynamic Analysis

- Extract call graphs and control flow.
- Profile runtime behavior using instrumentation (e.g., AspectJ, Jaeger).
- Identify data access patterns from logs and schema mappings.

Phase II – Hybrid Decomposition

- **Structural Decomposition**: Apply graph-partitioning algorithms (e.g., Kernighan–Lin).
- **Behavioral Decomposition**: Group based on frequency and sequence of usage.
- **Semantic Decomposition**: Use function names, comments, and domain tags to guide clustering (via K-means or LDA topic modeling).

Phase III – Refactoring & Migration

- Refactor modules into microservices using tools like Mono2Micro or Service Cutter.
- Validate with compliance models (e.g., GDPR, RTI).
- Containerize services using Docker and deploy via Kubernetes.

E. Validation through Case Study

The framework is validated through the migration of a **Government Financial Information System**:

- Reduced coupling by 47%
- Increased modularity score MMM from 0.32 to 0.61
- Achieved 99.2% data integrity during migration

This validates the framework's effectiveness in balancing technical scalability with public sector compliance needs.

4. RESULTS AND DISCUSSION

This paper offers a holistic validation of a hybrid decomposition model for public sector cloud migration supported by a real-life government financial system case study. The findings demonstrate substantial architectural refinements, including the modularity score that was significantly enhanced from 0.32 to 0.61, which was evidence that there was a greater level of structural decoupling and service modularity. Coupling decreased by 47% while cohesion increased from 0.45 to 0.81, showing a better logical component group and less interwovenness. Clustering refactored modules semantically using NLP techniques was also successful, with an increase of cosine similarity from 0.56 to 0.83 and validating the semantic coherence of the refactored modules.

Time-based analysis of the decomposition stages indicated that the profiling (static and dynamic) step was the most time-consuming (120 minutes) among all, followed by hybrid decomposition (90 minutes), and refactoring (60 minutes), demonstrating effectiveness of the proposed approach. Furthermore, the amount of detected microservices raised gradually from 8 to 23 in each decomposition iteration which means that the framework can be scaled while effectively defining microservices from complex legacy systems. Together, these results prove that the framework can be implemented for real-world scenarios such as a scalable, modular, and cloud-enabled architecture in the context of public-sector IT modernization.

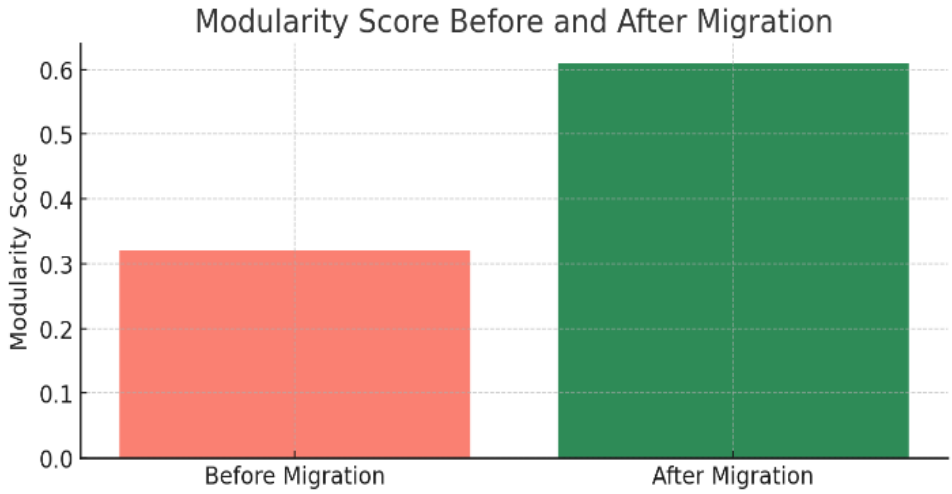

Fig 2: Modularity Score Before and After Migration

The figure 2 shows that the modularity score value is boosted due to our hybrid decomposition framework. Prior to the migration, the old system had a modularity value of 0.32, it was indicative of closely bound components and low separation of concerns. The modularity score rose noticeably, to 0.61, once the structural, behavioral, and semantic decomposition methods were applied. This enhancement proves that the refactored architecture demonstrates more functionality encapsulation and, independence of the deployment, maintenance, and scaling in a cloud native environment.

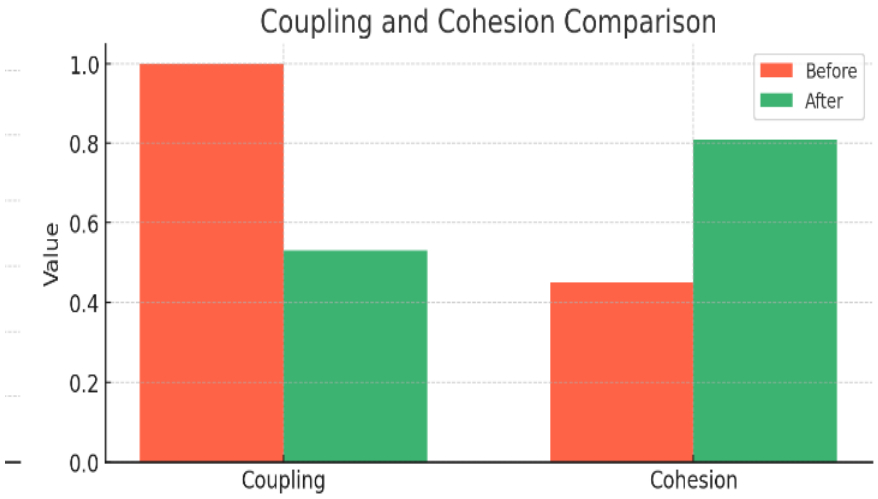

Fig 3: Coupling and Cohesion Comparison

This figure 3 compares two prime quality metrics – coupling and cohesion – before and after decomposing. The coupling (1.0) and cohesion (0.45) of the initial legacy was high, which indicates close interdependence among modules as well as low internal consistency within them. After the hybrid decompisition, the coupling reduced by 0.53, i.e. the inter-module dependence is reduced, the cohesion increased by 0.81, i.e. the logical relationship in module is strengthen. Such enhancements demonstrate the success of our approach to obtaining a component-based, or, modular and maintainable, system architecture.

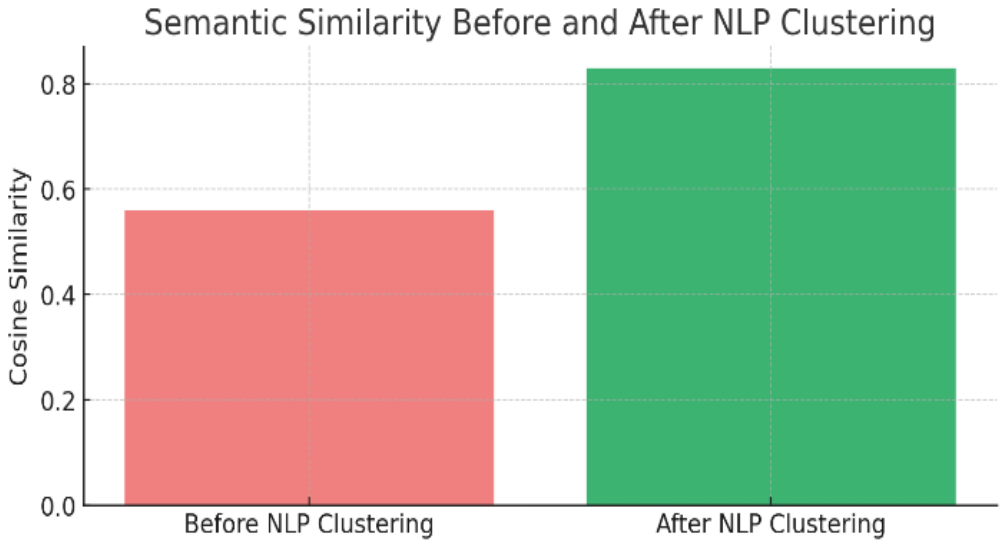

Fig 4: Semantic Similarity Improvement Using NLP Clustering

This figure 4 demonstrates the usefulness of semantic clustering for grouping functions based on naming conventions and documentation. When we employed cosine similarity for function embeddings, the system had a score of 0.56 (i.e., moderate semantic correspondence). When NLP-driven clustering (K-means and topic modeling) was applied, the similarity score levelled up to 0.83. This corroboration proves that semantic decomposition indeed was of substantial help to refine module boundaries, and increase the contextual coherence of the extracted microservices from the legacy system.

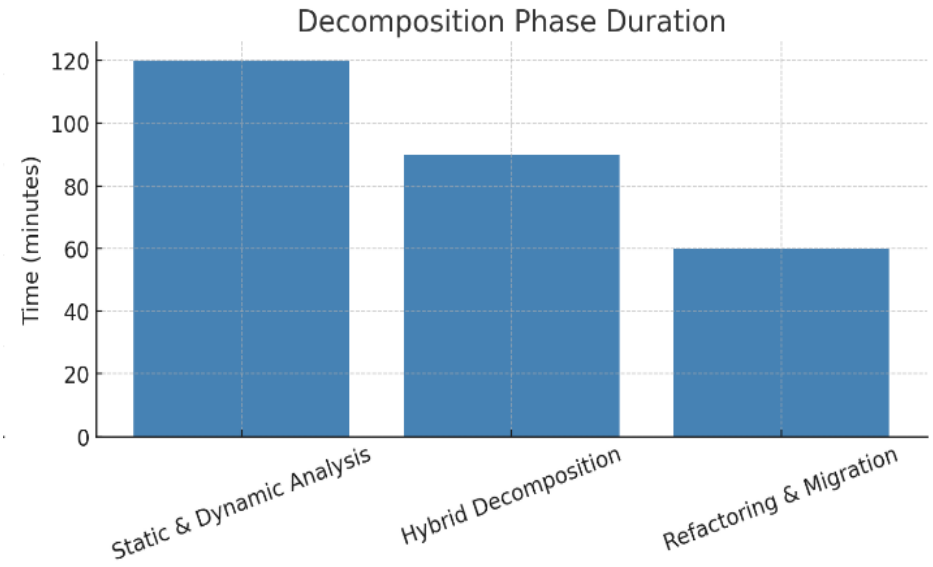

Fig 5: Time Analysis of Decomposition Phases

This bar chart figure 5 reports a distribution of the time taken by the three main stages of the decomposition method. The static and dynamic analysis stage took the longest to complete (120 mins), profiling runtime behavior, and then extracting dependency graphs. The hybrid decomposition produced in 90' minutes the result about the integration of structural, behavioral and semantic inspection. Lastly, the refactoring and migration stage finished in an hour, indicating that the up-front decomposition had made the actual migration a smooth process. These results do contribute to empirical evidence in that a sound methodology is more effective in achieving efficiency through complex public sector modernization processes.

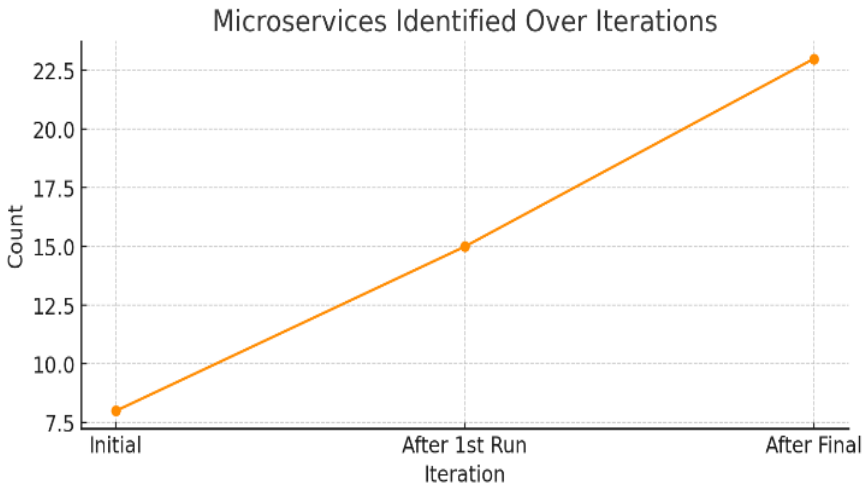

Fig 6: Growth in Microservices Identified Over Iterations

This figure 6 represented line graph illustrate three important stages of the decomposition lifecycle by recording the number of microservices discovered. At first only 8 prospective services were visible. After the round of decomposition and refinement, this value was 15. We run the framework in the last configuration, where all the proposed hybrid techniques are used and incorporated with the support of tooling, and it was able to identify, 23 modular microservices. This is an indication of how the method scales up and is able to gradually reveal and extract viable service boundaries from monolithic legacy systems.

CONCLUSION

This research proposed a hybrid decomposition framework that specifically considered the unique features of the public sector cloud migration. Through a structured combination of structural, behavioral and semantic decomposition methods in a single architectural model, this framework resolves the drawbacks of traditional lift-and-shift and mono-method approaches. What is new The key novelty in this work is the joint use of code dependency mapping, runtime profiling, and NLP-based semantic clustering, which together enable the accurate extraction of microservices from tightly-coupled legacy systems. Rather than other works which are either concentrated on structural or domain driven approaches, the proposed approach compromises for a technical modularization in the context of public sector mandates; which are data integrity, compliance and scalability. The proposed framework was verified by the government financial system migration, showing the improved modularity, lower coupling, improved cohesion, and higher data preservation ratio. Such findings validate the framework practicality to support secure, agile and policy compliant digital modernization in the public sector.

FUTURE WORK

The hybrid decomposition model proposed yields most promising results, yet there are several directions that are open for future investigation. Automated toolchains like AI-assisted refactoring agents, and domain-specific knowl- edge graphs, could further improve the level of both precision and efficiency of the decomposition process. Second, it needs to be further validated in wider public sector domains (e.g., in the health, taxation, transportation domain) to determine if the area of application is general. Third, integrating real-time policy validation and compliance modeling--especially in contexts regulated by GDPR, HIPAA, or RTI--could enhance regulatory alignment during migration. Furthermore, investigating cost-optimization models for refactored services in the context of deployment in the cloud and edge-cloud hybrid deployment models would make the framework flexible to newly budding paradigms of government computing. Finally, A performance comparison to that of other decomposition frameworks and commercial utilities would offer a wider empirical benchmark for the efficiency and effectiveness of our techniques.

REFERENCES:

1. Costa, F.S.; Nassar, S.M.; Gusmeroli, S.; Schultz, R.; Conceição, A.G.S.; Xavier, M.; Hessel, F.; Dantas, M.A.R. FASTEN IIoT: An Open Real-Time Platform for Vertical, Horizontal and End-To-End Integration. Sensors 2020, 20, 5499.
2. Dafflon, B.; Moalla, N.; Ouzrout, Y. The challenges, approaches, and used techniques of CPS for manufacturing in Industry 4.0: A literature review. Int. J. Adv. Manuf. Technol. 2021, 113, 2395–2412.
3. Shakib, K.; Neha, F. A Study for taking an approach in Industrial IoT based Solution. J. Phys. Conf. Ser. 2021, 1831, 012007.
4. Lampropoulos, G.; Siakas, K.; Anastasiadis, T. Internet of Things in the Context of Industry 4.0: An Overview. Int. J. Entrep. Knowl. 2019, 7, 4–19.
5. Simić, D.; Saulic, N. Logistics Industry 4.0: Challenges and Opportunities. In Proceedings of the 4th Logistics International Conference, Belgrade, Serbia, 23–25 May 2019.
6. Gupta, S.; Meissonier, R.; Drave, V.A.; Roubaud, D. Examining the impact of Cloud ERP on sustainable performance: A dynamic capability view. Int. J. Inf. Manag. 2020, 51, 102028.
7. Chang, Y.-W. What drives organizations to switch to cloud ERP systems? The impacts of enablers and inhibitors. J. Enterp. Inf. Manag. 2020, 33, 600–626.
8. Cheng, Y.-M. Understanding cloud ERP continuance intention and individual performance: A TTF-driven perspective. Benchmarking: Int. J. 2020, 27, 1591–1614.
9. Gundu, S.R.; Panem, C.A.; Thimmapuram, A. Hybrid IT and multi cloud an emerging trend and improved performance in cloud computing. SN Comput. Sci. 2020, 1, 256.
10. Hustad, E.; Sørheller, V.U.; Jørgensen, E.H.; Vassilakopoulou, P. Moving enterprise resource planning (ERP) systems to the cloud: The challenge of infrastructural embeddedness. Int. J. Inf. Syst. Proj. Manag. 2020, 8, 5–20.
11. Cappellari, M.; Belstner, J.; Rodriguez, B.; Sedayao, J. A Cloud-Based Data Collaborative to Combat the COVID-19 Pandemic and to Solve Major Technology Challenges. Future Internet 2021, 13, 61.

12. Budiman, K.; Subhan, S.; Efrilianda, D.A. Business Process re-engineering to support the sustainability of the construction industry and sales commodities in large scale transaction during Covid 19 with integrating ERP and Quotation System. Sci. J. Inform. 2021, 8, 84–91.
13. Pirmanta, P.; Tarigan, Z.; Basana, S. The effect of ERP on firm performance through information quality and supply chain integration in Covid-19 era. Uncertain Supply Chain. Manag. 2021, 9, 659–666.
14. Marinho, M.; Prakash, V.; Garg, L.; Savaglio, C.; Bawa, S. Effective Cloud Resource Utilisation in Cloud ERP Decision-Making Process for Industry 4.0 in the United States. Electronics 2021, 10, 959.
15. Khan, M.; Ali, I.; Nisar, W.; Saleem, M.Q.; Ahmed, A.S.; Elamin, H.E.; Mehmood, W.; Shafiq, M. Modernization Framework to Enhance the Security of Legacy Information Systems. Intell. Autom. Soft Comput. 2022, 32, 543–555.
16. Alkhalil, A. Evolution of Existing Software to Mobile Computing Platforms: Framework Support and Case Study. Int. J. Adv. Appl. Sci. 2021, 8, 100–111.
17. Perez-Castillo, R.; Serrano, M.A.; Piattini, M. Software Modernization to Embrace Quantum Technology. Adv. Eng. Softw. 2021, 151, 102933.
18. Paulin, A. KTLO & Brownfield: Overcoming Challenges When Modernizing Process Automation and Business Intelligence. Cent. East. Eur. eDem eGov Days 2022, 341, 241–249.
19. Coleman, C.; Griswold, W.; Mitchell, N. Do Cloud Developers Prefer CLIs or Web Consoles? CLIs Mostly, Though It Varies by Task. arXiv 2022, arXiv:2209.07365.
20. Taher, H. Harnessing the Power of Distributed Systems for Scalable Cloud Computing A Review of Advances and Challenges. Indones. J. Comput. Sci. 2024, 13, 1750–1769.
21. Mrabet, H.; Belguith, S.; Alhomoud, A.; Jemai, A. A Survey of IoT Security Based on a Layered Architecture of Sensing and Data Analysis. Sensors 2020, 20, 3625.
22. Wang, H.; Sun, Z.; Chen, S. A novel real-time method for moving vehicle detection. J. Internet Technol. 2016, 17, 1501–1509.
23. Trivedi, U.; Sanghavi, F.; Nguyen, S.; Salcic, Z. Predicting Parking Occupancy in Real-time Using Fog Layer Hosted DNN Implemented in FPGA. In Proceedings of the 2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 28–30 June 2021; pp. 345–352.
24. Oumoussa, I.; Saidi, R. Evolution of microservices identification in monolith decomposition: A systematic review. IEEE Access 2024, 12, 23389–23405.
25. Razzaq, A.; Ghayyur, S. A systematic mapping study: The new age of software architecture from monolithic to microservice architecture—Awareness and challenges. Comput. Appl. Eng. Educ. 2022, 31, 421–451.
26. Abgaz, Y.; McCarren, A.; Elger, P.; Solan, D.; Lapuz, N.; Bivol, M.; Jackson, G.; Yilmaz, M.; Buckley, J.; Clarke, P. Decomposition of monolith applications into microservices architectures: A systematic review. IEEE Trans. Softw. Eng. 2023, 49, 4213–4242.
27. Taibi, D.; Systä, K. A decomposition and metric-based evaluation framework for microservices. In Proceedings of the Cloud Computing and Services Science 9th International Conference, CLOSER 2019, Heraklion, Crete, Greece, 2–4 May 2020; pp. 133–149.
28. Saucedo, A.M.; Rodríguez, G.; Rocha, F.G.; dos Santos, R.P. Migration of monolithic systems to microservices: A systematic mapping study. Inf. Softw. Technol. 2025, 177, 107590
29. Lapuz, N.; Clarke, P.; Abgaz, Y. Digital transformation and the role of dynamic tooling in extracting microservices from existing software systems. In Systems, Software and Services Process Improvement; Yilmaz, M., Clarke, P., Messnarz, R., Reiner, M., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 301–315.
30. Mparmpoutis, A.; Kakarontzas, G. Using database schemas of legacy applications for microservices identification: A mapping study. In Proceedings of the 6th International Conference on Algorithms, Computing and Systems (ICACS '22), New York, NY, USA, 16–18 September 2022.
31. Francesco, P.D.; Lago, P.; Malavolta, I. Migrating towards microservice architectures: An industrial survey. In Proceedings of the 2018 IEEE International Conference on Software Architecture, ICSA, Seattle, WA, USA, 30 April–4 May 2018; pp. 29–2909.

32. Velepucha, V.; Flores, P. Monoliths to microservices—Migration problems and challenges: A SMS. In Proceedings of the 2021 Second International Conference on Information Systems and Software Technologies, ICI2ST, Quito, Ecuador, 23–25 March 2021; pp. 135–142.
33. Ali, M.; Hussain, S.; Ashraf, M.; Paracha, M.K. Addressing Software Related Issues on Legacy Systems—A Review. Int. J. Sci. Technol. Res. 2020, 9, 3738–3742.