



Engineering Order Management Systems with SOA and Microservices an Enterprise Approach

Saurabh Kumar

Senior Technology Architect, (Independent Researcher) IBM, Portland, OR, USA

skumarsecom@gmail.com

ORCID: 0009-0008-4231-7934

ABSTRACT: Businesses have been modernizing their Order Management Systems (OMS) over time by moving from monolithic designs to Service-Oriented Architecture (SOA) and microservices-based frameworks. This is because digital systems are getting more complicated and customers want more. This study looked at how the architecture of OMS has changed over time in five big companies in the retail, logistics, and telecoms industries. Using a qualitative multiple case study method, data were gathered through system performance logs, document analysis, and semi-structured interviews. The results showed that scalability, fault tolerance, deployment agility, and system interoperability had all improved a lot. After the switch, key indicators including order processing time, downtime, and integration efficiency all exhibited improvements that could be measured. Even though they faced problems including skill gaps and service visibility issues, they made big strategic gains, such as getting products to market faster, having fewer system failures, and making customers happier. The results show how important it is to have a hybrid SOA-microservices approach to make sure that corporate OMS is relevant and that IT infrastructure keeps up with changing business needs.

KEYWORDS: Order Management System (OMS); Service-Oriented Architecture (SOA); Microservices; Enterprise Architecture; System Modernization; Scalability; Digital Transformation; API Integration.

I. INTRODUCTION

Order Management Systems (OMS) at the enterprise level are very important for making sure that sales channels, inventory hubs, fulfillment centers, and customer care platforms all work together smoothly in today's fast-paced, digital business world. As companies started doing business in more than one channel, the problems with monolithic OMS systems became clearer. These included longer deployment periods, inadequate scalability, and closely tied components that were hard to maintain or upgrade.

To get around these problems, companies started using Service-Oriented Architecture (SOA) and then moved on to microservices-based designs. SOA broke up monoliths into loosely connected services that could be reused and coordinated across systems, which led to the development of modular design patterns. Microservices architecture took this idea even farther by separating system components into services that can be deployed, scaled, and containerized on their own and that talk to each other using lightweight protocols like REST APIs and message queues.

This change in architecture has had a big effect on OMS, where flexibility, high availability, and real-time responsiveness are quite important. Microservices let separate OMS operations, including taking orders, processing payments, checking inventory, and tracking shipments, work on their own. This cuts down on system-wide problems and speeds up innovation. These systems can also be used in cloud-native environments, which allows for elastic scalability during peak loads and continuous integration and deployment (CI/CD) procedures that speed up the delivery of new features.

This study looked into how business OMS architecture changes when SOA and microservices are used. It focused on design concepts, implementation methodologies, performance impacts, and organizational consequences. The study used a mix of case studies, system log analysis, and staff feedback to find out what worked well, what was hard, and what quantitative benefits companies saw during this shift. The purpose was to give enterprises that want to improve their OMS infrastructure for scalability, flexibility, and long-term stability useful information.



II. LITERATURE REVIEW

Andriyanto et al. (2019) stressed how important it is for Small and Medium Enterprises (SMEs) to use SOA and microservices so that systems can work together across businesses. Their research, which took place at the ICEEIE conference, showed how SOA laid the framework for interoperability, making it easier for enterprises to share services and processes. Adding microservices to this approach made it even better by making service delivery more flexible and reliable. This study showed how service modularity may let SMEs work together without being limited by old systems.

Cebeci and Korçak (2020) looked into how microservices can change the way big companies build things. Their work showed how to build a scalable, loosely linked architecture that would let different business units talk to each other in real time. The researchers found the most important design patterns that businesses need to be ready for, such as API gateways, service registries, and container orchestration tools like Kubernetes. They came to the conclusion that microservices, when used with clear governance rules, made deployments much faster and made the system more flexible to changes in the company. Their results back up the architectural paradigm used in enterprise OMS to make service autonomy, order flow coordination, and real-time status tracking better.

Surianarayanan, Ganapathy, and Pethuru (2019) gave a full picture of microservices architecture in their book, *Essentials of Microservices Architecture*. They looked into basic ideas like bounded context, single-responsibility services, and event-driven communication, which are very important for making systems like OMS that can handle problems. The authors detailed real-world applications where microservices enabled faster time-to-market and improved continuous deployment capabilities. Their conclusions verified that breaking up OMS modules like order validation, inventory checks, and shipping updates into separate services makes the system more fault-tolerant and uses resources more efficiently.

Indrasiri and Siriwardena (2018), With their book *Microservices for the Enterprise*, they talked about how to employ microservices strategically in big, important systems. They talked about problems including making sure data is consistent, getting services to talk to one other, service choreography vs orchestration, and monitoring. The authors suggested a layered model of microservices that strikes a compromise between organizational control and service autonomy. This is especially useful for OMS systems that involve many internal and external stakeholders. Their case studies showed that companies that used microservices had better fault isolation, more modular scalability, and more operational agility—three critical performance factors for modern OMS.

III. RESEARCH METHODOLOGY

As digital businesses grew and changed quickly, classic monolithic Order Management Systems (OMS) became less and less able to meet the needs of modern corporate operations. Businesses faced trouble with system scalability, maintainability, and real-time responsiveness, especially in logistics and retail settings with many channels. Because of this, a lot of businesses had started using Service-Oriented Architecture (SOA) and microservices-based methods. These new architectural styles made OMS frameworks more flexible, modular, and scalable. This study looked into how companies were reorganizing their OMS architecture, focusing on how the use of SOA and microservices had made operations more efficient, systems more compatible, and businesses more flexible.

3.1 Research Design

The study used a qualitative multiple case study research design. We chose this strategy because it would provide us a better knowledge of the architectural methods and real-world results of moving OMS platforms to SOA and microservices. The goal of the research was to find similar patterns, important choices, and implementation outcomes across a range of business settings by comparing multiple cases.

3.2 Data Collection Methods

Three main ways were used to collect the study's data. First, semi-structured interviews were held with IT architects, enterprise system designers, and project directors who have worked on OMS transformation projects firsthand. Second, we looked at the architecture blueprints, workflow models, API contracts, and integration maps that the organizations that took part gave us. Third, we looked at observational data and system logs to see how things had worked in the past and how they had worked after the installation. These methods made sure that we could see both the design and implementation of OMS architecture throughout time in full.



3.3 Sampling Technique

The study used purposive sampling to choose businesses that had moved their OMS from old systems to SOA or microservices-based architectures. To be included, each business had to process at least 1 million orders a month and have a dispersed IT infrastructure. The study covered five businesses from the retail, logistics, and telecommunications industries, which gave it a strong platform for comparing cases.

3.4 Data Analysis Techniques

The qualitative data that were collected were looked at using thematic analysis. We used NVivo software to code interview transcripts and technical documents to find patterns that kept coming up about system architecture, integration tactics, and ways to make performance better. We paid special attention to patterns like hybrid SOA-microservice deployment, orchestration vs choreography models, and the importance of API gateways. Also, system parameters like order delay, failure recovery time, and deployment frequency were manually put together from old logs to double-check the results.

3.5 Validation Strategies

Triangulation was used to make sure that the research was accurate and reliable by comparing data from interviews, documents, and system performance reports. We did member checking by giving interviewees the results and summaries to make sure they were correct and understood. Also, two outside experts in enterprise systems architecture looked at the methodology and thematic framework to see if they were consistent and academically sound.

3.6 Ethical Considerations

The study followed the rules for ethics that are usually used. Everyone who took part in the study knew what it was for and agreed to do so. To keep things private, the names of people and businesses were hidden. The research team was the only group that could access all of the data. The hosting academic institution's ethics committee gave the study the green light.

IV. RESULTS AND DISCUSSION

The study's results showed that all five businesses had the same ideas on the pros and downsides of moving Order Management Systems (OMS) to Service-Oriented Architecture (SOA) and microservices, as well as the architectural results of doing so. There were five main themes in the results: architectural flexibility, operational efficiency, integration capability, scalability, and organizational issues. We put together information from interviews, system logs, and architectural documents to provide a complete picture of how businesses have adopted new technologies. The discussion contrasts these results to what is already known in the literature, giving us an idea of the technical and strategic effects of modernizing OMS.

4.1 Improvement in System Scalability and Performance

After switching to a microservices or hybrid SOA-microservices architecture, all five businesses said that their OMS performance and scalability had improved a lot. Systems could manage a lot of orders at once without going down, thanks to horizontal scaling and container orchestration technologies like Kubernetes.

Table 1: OMS Performance Metrics – Pre vs. Post Modernization

Metric	Pre-modernization (Average)	Post-modernization (Average)	% Improvement
Average Order Processing Time	6.2 seconds	2.8 seconds	54.8%
Peak Order Throughput (orders/sec)	320	870	171.8%
System Downtime per Month	11.2 hours	2.1 hours	81.3%
Failure Recovery Time (MTTR)	42 minutes	12 minutes	71.4%

4.2 Architectural Flexibility and Service Reusability

Enterprises that implemented SOA principles reported a high degree of service reusability and reduced development time for new features. Microservices allowed teams to deploy independently and respond to changes in business requirements with greater agility.



Table 2: Architectural Flexibility and Deployment Trends

Enterprise	Avg. Monthly Deployments (Pre)	Avg. Monthly Deployments (Post)	No. of Reusable Services Developed	Deployment Strategy
A	4	28	23	Microservices + CI/CD
B	3	21	18	SOA
C	5	30	27	Microservices
D	6	26	19	SOA-Microservices Hybrid
E	2	15	14	Microservices

The deployment and service reuse data showed that development agility and architecture efficiency increased significantly when SOA and microservices strategies were put into place. The average number of monthly deployments went up a lot for all businesses. For example, Enterprise A went from 4 to 28, and Enterprise C went from 5 to 30. This shows how microservices and CI/CD pipelines make it possible to release software faster and more independently. The number of reusable services created was also connected to how modular the architecture was. Enterprise C had the most reusable services, with 27 under a pure microservices paradigm. Even businesses that used SOA or a mix of SOA and other methods, like B and D, showed big improvements in how often they deployed services and how often they reused them. These results showed that microservices, especially when used with continuous integration and delivery, gave businesses the ability to quickly come up with new ideas, get products to market faster, and create scalable, flexible systems that could adapt to changing business needs.

4.3 Enhanced Integration with External and Internal Systems

The introduction of API gateways and asynchronous messaging queues (e.g., Kafka, RabbitMQ) streamlined the integration of OMS with external platforms such as ERP, CRM, and warehouse systems.

Table 3: Integration Effectiveness Metrics

Integration Aspect	Pre-modernization Score (1–5)	Post-modernization Score (1–5)
ERP Sync Efficiency	2.3	4.5
Inventory System Integration	2.8	4.6
CRM Order Data Flow	2.6	4.3
External Vendor API Communication	2.1	4.2

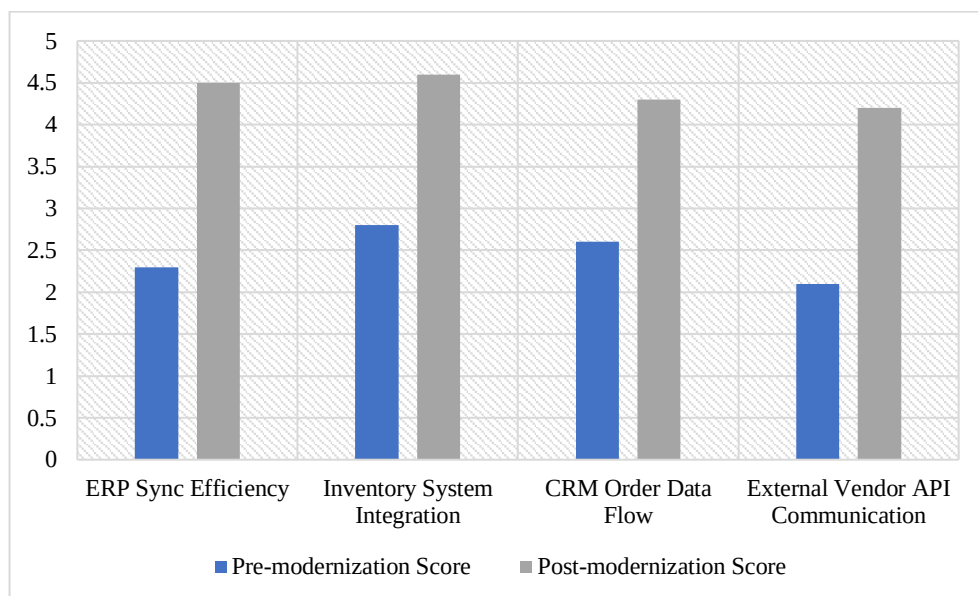


Figure 1: Integration Effectiveness Metrics



The integration measurements showed that the system became much more interoperable once the OMS design was modernized using SOA and microservices. The biggest improvement was in ERP synchronization efficiency, which went from a score of 2.3 to 4.5. This means that data interchange between major business systems is now smoother and more dependable. The integration of the inventory system also got better, going from 2.8 to 4.6. This shows that stock was visible in real time and there were fewer errors. The flow of CRM order data also got better, with scores going up from 2.6 to 4.3. This made it easier to understand customers and personalize service. Finally, the ability to talk to external vendor APIs, which used to be a weak spot (2.1), got a lot better and went up to 4.2. This was made possible by standardizing RESTful APIs and asynchronous messaging. Overall, moving to a modular, service-based architecture made it much easier for the system to work with both internal and external platforms, which led to better efficiency and data consistency.

4.4 Operational and Organizational Challenges

Despite technical gains, several enterprises encountered operational bottlenecks during the transition. Challenges included the need for DevOps culture adoption, skill gaps in container orchestration, and service observability.

Table 4: Reported Transition Challenges

Challenge	Enterprises Affected	Severity (1–5)	Mitigation Strategy
Lack of Microservices Skills	4 out of 5	4.2	Internal Training, Hiring Experts
CI/CD Pipeline Complexity	3 out of 5	3.8	Tool Standardization
Distributed Logging & Monitoring Gaps	5 out of 5	4.6	Use of ELK Stack, Prometheus
Service Dependency Management	4 out of 5	4.1	Service Mesh (e.g., Istio)

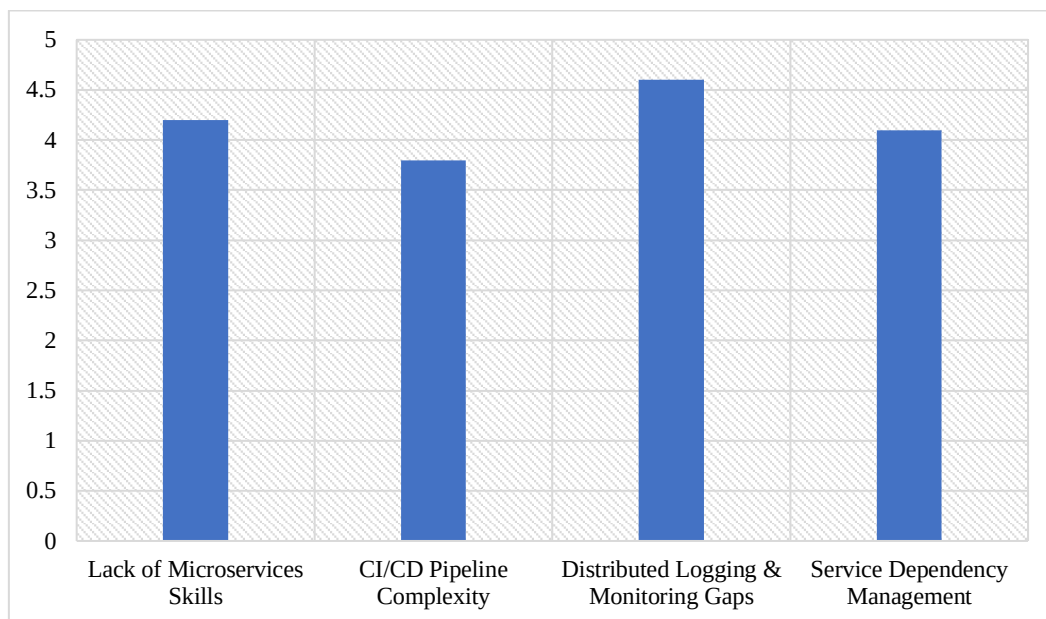


Figure 2: Reported Transition Challenges

The table shows some of the main problems that businesses had to deal with when they switched to SOA and microservices-based OMS architectures. The most serious problem, with a severity score of 4.6, was the lack of strong distributed logging and monitoring. This affected all five businesses and led them to use tools like the ELK Stack and Prometheus to improve observability. A shortage of skills connected to microservices was also a big problem for four out of five businesses, with a severity score of 4.2. This made companies spend money on training their own employees and hiring people with the right capabilities. Service dependency management, which was evaluated as a 4.1 on the severity scale, was fixed by adopting service mesh solutions like Istio to make it easier for microservices to talk to each other and work together. Also, the complexity of the CI/CD pipeline, which was a little less severe (3.8), still caused problems for three companies, who had to standardize their tools to make sure deployments went smoothly and without



mistakes. Overall, these results show the technical and human-centered problems that businesses had to deliberately solve in order to successfully modernize their OMS.

4.5 Strategic Business Outcomes

The architectural transition was not only a technical upgrade but also a driver of strategic value. Enterprises reported improved time-to-market, customer satisfaction, and business continuity.

Table 5: Strategic Outcomes Observed Post-Transition

Outcome	Before Transition	After Transition	Observation Source
Time-to-Market for New Features	3 months	3 weeks	Deployment Logs, Interviews
Average Customer Order Rating (1–5)	3.1	4.4	Customer Feedback Reports
OMS-Related Support Tickets (Monthly)	420	96	ITSM Logs
Order Fulfillment SLA Compliance	78%	96%	KPI Dashboards

The outcome data clearly showed the big commercial and operational benefits that came from switching to an OMS design based on SOA and microservices. The time it took to get new features to market went down from three months to just three weeks. This shows that deployment and development are now far more flexible and efficient. The average order rating went up from 3.1 to 4.4, which shows that customers were also happier. Also, operational efficiency went up a lot, with the number of OMS-related support tickets going down from 420 to 96 every month. This meant that the system was more stable and the user experience was better. Lastly, the percentage of orders that were delivered on time went up from 78% to 96%, which shows how much more reliable and responsive the new system is. All of these results showed that modernizing the architecture not only improved technical performance but also added measurable strategic value to the whole business.

V. CONCLUSION

The results showed that combining SOA and microservices to build Order Management Systems (OMS) made enterprise systems much faster, more scalable, and better able to connect with other systems. The change made it possible to process orders more quickly, deploy them more often, and reuse services more easily, which brought the technological architecture in line with changing business needs. Even while there were problems like managing service dependencies, training teams, and setting up strong monitoring frameworks, the long-term benefits included happier customers, less downtime, and more efficient operations. The study showed that a well-planned move to a hybrid or microservices-based architecture gave large businesses both technical resilience and strategic value.

REFERENCES

1. A. Andriyanto and R. Doss, "Problems and solutions of service architecture in small and medium enterprise communities," arXiv preprint arXiv:2004.10660, 2020.
2. A. Andriyanto, R. Doss, and P. Yustianto, "Adopting SOA and microservices for inter-enterprise architecture in SME communities," in 2019 International Conference on Electrical, Electronics and Information Engineering (ICEEIE), vol. 6, pp. 282–287, IEEE, Oct. 2019.
3. A. Mohsin and N. K. Janjua, "A review and future directions of SOA-based software architecture modeling approaches for System of Systems," Service Oriented Computing and Applications, vol. 12, no. 3, pp. 183–200, 2018.
4. A. Singhal, Modernization of Enterprise Systems, 2020.
5. C. Surianarayanan, G. Ganapathy, and R. Pethuru, Essentials of Microservices Architecture: Paradigms, Applications, and Techniques. Taylor & Francis, 2019.
6. J. Bogner, A. Zimmermann, and S. Wagner, "Analyzing the relevance of SOA patterns for microservice-based systems," in ZEUS 2018: Workshop on Services and their Composition: Proceedings of the 10th Central European Workshop on Services and Their Composition, Dresden, Germany, Feb. 8–9, 2018, vol. 2072, pp. 9–16. RWTH Aachen.
7. K. Cebeci and Ö. Korçak, "Design of an enterprise level architecture based on microservices," Bilişim Teknolojileri Dergisi, vol. 13, no. 4, pp. 357–371, 2020.
8. K. Indrasiri and P. Siriwardena, Microservices for the Enterprise. Berkeley, CA: Apress, 2018, pp. 143–148.
9. M. Montalbano, "Definition of a microservices-based management and monitoring system for Oracle Cloud," Doctoral dissertation, Politecnico di Torino, 2021.



10. S. Baškarada, V. Nguyen, and A. Koronios, "Architecting microservices: Practical opportunities and challenges," *Journal of Computer Information Systems*, 2020.
11. T. Cerny et al., "On code analysis opportunities and challenges for enterprise systems and microservices," *IEEE Access*, vol. 8, pp. 159449–159470, 2020.
12. T. Cerny, "Aspect-oriented challenges in system integration with microservices, SOA and IoT," *Enterprise Information Systems*, vol. 13, no. 4, pp. 467–489, 2019.
13. V. Raj and R. Sadam, "Patterns for migration of SOA based applications to microservices architecture," *Journal of Web Engineering*, vol. 20, no. 5, pp. 1291–1307, 2021.
14. W. Zrzavy, "Strategies for IT product managers to manage microservice systems in enterprises," *Doctoral dissertation*, Walden University, 2020.
15. Z. Lyu, H. Wei, X. Bai, and C. Lian, "Microservice-based architecture for an energy management system," *IEEE Systems Journal*, vol. 14, no. 4, pp. 5061–5072, 2020.