BULLETIN of Cherkasy State Technological University

Journal homepage: https://bulletin-chstu.com.ua/en

Vol. 29 No. 4. 2024

UDC 004.75
DOI: 10.62660/bcstu/4.2024.32

Microservices architecture for ERP systems

Serhii Slivka[*]
Postgraduate Student
Odesa Polytechnic National University
65044, 1 Shevchenko Ave., Odesa, Ukraine
https://orcid.org/0009-0003-1704-1415

Abstract. Traditional enterprise resource planning (ERP) systems built on the principle of monolithic architecture are becoming less and less effective in a modern business environment. The growing complexity of production processes, the increase in data volumes and the need to adapt quickly to changes pose significant challenges for such systems. The study aimed to analyse and develop methods for transitioning from a monolithic architecture to a microservice architecture in ERP systems to ensure their scalability, flexibility and security. A comprehensive analysis of the client-server architecture of ERP systems and their main components, including relational databases and structured query language (SQL) queries, was conducted. The principles of functions and procedures, interaction between tables, and query creation in systems with many users were investigated. The study also addressed modern strategies for decomposing monolithic systems into microservices, including methods of data synchronisation and component management. Security aspects and risks of transition to microservice architecture were emphasised. Approaches to the transformation of ERP systems based on microservice architecture were developed, which reduces management complexity, increases productivity and reduces risks in business processes. Mechanisms for effective synchronised data exchange between components and ensuring their reliability were proposed. The study determined that microservice architecture enables independent scaling of individual system elements, increasing fault tolerance and performance under high loads. Common issues, including incorrect balances and errors in reports, were investigated and resolved. The developed approaches can be used to modernise ERP systems of large enterprises, especially in the manufacturing sector, where high performance and accuracy of data processing are important

Keywords: client-server; scalability; stakeholder; critical path; critical event; automation; cloud technologies

INTRODUCTION

The rapid development of information technology and the growing need for efficient business process management are creating new challenges for businesses of all sizes. Traditional monolithic enterprise resource planning (ERP) systems are becoming less effective due to the increasing complexity of business processes, growing data volumes and the need to quickly adapt to market changes. This problem is particularly acute in the context of ensuring flexibility, scalability and efficiency of corporate systems.

The research relevance of implementing microservice architecture in ERP systems is determined by several factors. There is a need to modernise existing corporate systems to increase their adaptability and efficiency. In the context of digital business transformation, it is critical to ensure the ability to quickly scale and update individual system components. The globalisation of business processes requires the implementation of modern architectural solutions that meet international software development standards.

Article's History: Received: 05.08.2024; Revised: 29.11.2024; Accepted: 16.12.2024.

Suggested Citation:
Slivka, S. (2024). Microservices architecture for ERP systems. *Bulletin of Cherkasy State Technological University*, 29(4), 32-42. doi: 10.62660/bcstu/4.2024.32.

[*]Corresponding author

Microservices architecture enables the decomposition of complex systems into independent components and their isolated deployment, which makes it particularly beneficial for use in ERP systems. In the context of corporate systems, microservices open great opportunities for improving business processes. The use of microservices allows for independent scaling of individual components, simplifies the development and testing process, allows for the use of different technologies for different services, and guarantees high system availability due to fault isolation.

The study by M. Söylemez *et al*. (2022) addressed the key issues related to the implementation of microservice architecture. Particular attention was devoted to the issues of service coordination, data consistency in distributed systems, and challenges in transaction management. The microservice architecture was noted to significantly increase the scalability and adaptability of systems but requires the introduction of clear standards for interaction between services to avoid performance degradation.

A step-by-step approach to modernising outdated information systems through the integration of microservice architecture was presented by D. Wolfart *et al*. (2021). The authors proposed methods for decomposing a monolithic structure into independent components that support the use of application programming interface (API) gateways to facilitate interaction. It is noted that this approach helps to increase flexibility, reduce support costs and ensure faster adaptation to changes in business processes.

The benefits of moving to a cloud ERP system based on a microservice architecture were analysed by C. Lee *et al*. (2024). The main emphasis was devoted to ensuring the sustainability and adaptability of such systems, which is achieved by dividing the functionality into independent services. The possibilities of using microservices to reduce risks and improve productivity in large enterprises were also investigated.

A.S. Abdelfattah & T. Cerný (2022) highlighted the security challenges of microservice architecture, including API security, authentication, and access control. The authors proposed multi-level solutions to prevent attacks on microservices, emphasising the importance of integrating modern security protocols such as OAuth 2.0. The study emphasised that the security of distributed systems requires careful monitoring of each component.

The issue of improving the efficiency of business processes in service-oriented architectures was considered by T. Górski & A.P. Wozniak (2021). The authors analysed process automation techniques, including reducing delays and increasing the accuracy of transaction processing. The study emphasised that optimisation of such processes contributes to the performance of ERP systems.

The features of the implementation of microservice architecture in cloud applications are analysed in the study by O.C. Oyeniran *et al*. (2024). The authors analysed design templates that help reduce latency, increase scalability, and ensure the stability of systems in high-load environments. The study emphasised that the use of microservices in cloud ERP solutions allows for efficient resource allocation and system stability.

M. Grambow *et al*. (2020) investigate methods for assessing the performance of microservice applications. The authors conducted a comparative analysis of approaches to testing systems in different load scenarios, emphasising the importance of choosing optimal strategies to avoid performance losses. The study noted that effective testing helps reduce risks in the process of system scaling.

The issue of interoperability of microservice architecture components was considered in the study by A. Bayramçavuş *et al*. (2021). The study determined that API standardisation and the implementation of transaction management strategies can achieve data consistency and ensure the stability of distributed systems. Particular attention is devoted to the development of interaction models that minimise the risks of data loss in the process of system scaling.

Existing research in the field of microservices architecture points to a number of unresolved issues that limit the effectiveness of traditional approaches to ERP system development. The main gaps are the complexity of managing distributed transactions, problems with data consistency between services, and challenges related to monitoring and diagnostics of distributed systems. In addition, existing approaches rarely address the specifics of business processes in particular industries, which creates difficulties in implementing and adapting systems.

The study aimed to analyse and optimise approaches to the transition from monolithic to microservice architecture of ERP systems to ensure their scalability, flexibility and security. Practical recommendations were developed for the design and implementation of microservice architecture in ERP systems. The research included the creation of a theoretical model of microservice architecture for ERP systems, exploring the possibilities of ensuring data consistency between services and developing methods for effective monitoring and management of a distributed system.

MATERIALS AND METHODS

The study is based on a theoretical analysis of modern approaches to the architecture of ERP systems, particularly the transition from a monolithic to a microservice structure. The main methodology used was a review of scientific papers, technical documentation and industry standards related to microservice architecture, client-server interactions, database management and security in distributed systems. The principles of distributing computing loads between client and server components of ERP systems, which are key to ensuring their efficiency, were investigated. Attention was devoted to the peculiarities of implementing structured query language (SQL) queries, transaction management and maintaining data consistency in a multi-user environment.

To model the transition from a monolithic architecture to a microservice architecture, the main approaches to system decomposition were analysed, including the use of API gateways, data synchronisation and transaction management methodologies. The theoretical aspects of mechanisms for integrating microservices with external modules, such as customer relationship management (CRM) systems or electronic document management systems, as well as architectural templates that ensure data consistency and minimise the risk of information loss when components interact were considered.

The theoretical analysis also included modelling ways to scale systems through independent module decomposition, optimisation of SQL queries, and the introduction of indexes to speed up the processing of large amounts of data. The use of cloud technologies as a means of improving the performance and stability of ERP systems, especially under high-load conditions, was investigated. In this context, the theoretical aspects of dynamic resource allocation and the use of automated monitoring tools to maintain system stability were considered.

Additionally, the theoretical study analysed modern approaches to the design of architectural solutions, including the use of programming patterns such as the Strangler Fig Pattern for the gradual transition from monolithic to microservice architecture. This approach was used to integrate new microservice components without interrupting the system operation and minimise the risks during the implementation of changes. The advantages of using standard API gateways to manage the interaction between system components, which is a key aspect for maintaining the scalability and reliability of ERP systems, are considered.

The client-server architecture of ERP systems has several advantages and disadvantages that affect its effectiveness in the corporate environment. To systematise this analysis, a SWOT analysis was employed to assess the strengths and weaknesses of the architecture, as well as identify opportunities for its modernisation and risks. Thus, the study covers a systematic analysis of the key theoretical aspects of the transition to microservice architecture, including its impact on the performance, scalability, security and adaptability of ERP systems, which creates the basis for the development of innovative solutions in this area.

RESULTS AND DISCUSSION

ERP systems are based on a client-server architecture, which enables efficient distribution of computing loads between the client and server sides. The server performs key functions of data processing, database management, and SQL query execution. The client part of the system, in turn, provides interaction with users and enters and displays data in a clear format.

One of the main advantages of this architecture is the ability to scale the system to meet the needs of the enterprise. New modules and users can be added without significant changes to the basic structure of the system. In addition, the versatility of the client-server architecture is ensured by the SQL standard, which is supported by various database management systems and allows the system to be transferred between platforms with minimal effort. However, there are also disadvantages. Significant computing resources are required to support large amounts of information that grow with the development of the enterprise. There are also difficulties in updating software on numerous client devices, which can slow down the process of implementing changes. Table 1 shows the key results of the SWOT analysis of the client-server architecture of ERP systems.

Table 1. SWOT analysis of the client-server architecture of ERP systems

Aspects	Parameters	Advantages	Disadvantages
Strengths	Scalability	Ease of adding new users and modules	Requires scaling of the server infrastructure as the number of users increases
	Productivity	Centralised data processing reduces the load on client devices	Dependence on network bandwidth and server speed
	Compatibility	Employment of SQL ensures versatility and cross-platform integration	High costs of integration with old or non-standard platforms
Weaknesses	Infrastructure	Significant server requirements for processing large amounts of data	Limitations in the rate of infrastructure upgrades
	Administration	The complexity of updating software on numerous client devices	Dependence on administrators' expertise to maintain the system
	Flexibility	Limitations to rapid innovation due to centralised structure	The possibility of a single server failure affecting the entire system
Opportunities	Integration of new technologies	Employment of cloud services to ensure stability and flexibility	The high initial cost of switching to new technologies
	Automation	Implementation of AI for data analysis and load forecasting	The need to adapt the existing system to new functions
	Expansion functionality	Integration of mobile platforms and external systems (CRM, logistics)	Potential conflicts between different modules during integration

Continued Table 1.

Aspects	Parameters	Advantages	Disadvantages
Threats	Cybersecurity	Possible attacks on the server side due to its central role	Increased costs of data security
	Infrastructure risks	Dependence on network bandwidth and server stability	Equipment failure can lead to a shutdown of the entire system
	Technology development	Competition from microservice architectures and new systems	Risk of losing relevance due to lack of fast updates

Source: compiled by the author based on D. Wolfart *et al.* (2021), M. Söylemez *et al.* (2022), C. Lee *et al.* (2024)

The successful functioning of ERP systems depends on effective interaction between stakeholders. Internal customers, such as managers and employees, seek to optimise business processes and receive accurate reports, while system administrators are focused on ensuring stable operation, data protection and user service. The interests of these groups can often conflict, requiring careful communication management.

The support and influence matrix ensures systematisation of stakeholders' interests, identification of their influence on the project and development of individual interaction strategies. This is especially relevant when the system is scaled up when every change affects the work of both groups. For instance, internal customers value flexibility and quick implementation of changes, while administrators prefer stability and risk minimisation. The study devoted significant emphasis to the architecture of ERP systems, as it is the proper design and management of this architecture that ensures stable and efficient operation of corporate systems. One of the key aspects is the interaction between the main stakeholders, as each group has its requirements for the system's functioning. It is necessary to identify how different interests can influence the architectural decisions of ERP systems and what trade-offs need to be factored into their design and implementation. Table 2 provides a comparison of the interests of the main stakeholders of ERP systems, which will clarify how their needs affect the architectural aspects and technical decisions made during the development and implementation of such systems.

Table 2. Comparison of the interests of the main stakeholders of ERP systems regarding architecture and software solutions

Stakeholders	Main interests	Possible conflicts
Internal customers	1. Flexibility and scalability of functions. 2. Easy integration of new modules.	1. The need for constant updates can be at odds with the stability of the system. 2. Flexibility requirements can make it difficult to maintain and upgrade.
System administrators	Stable and reliable operation.	1. Risk of security breaches due to the integration of new modules. 2. Conflicts between stability and the need to introduce new features.
Software developers	1. Transparency and ease of development. 2. Ease of new technology implementation.	1. Complications due to the need to interact with existing components. 2. The use of new technologies may require significant changes to the system.
Users of the ERP system	1. Speed of operation and availability of functions. 2. Intuitive interface.	1. Problems with adapting the interface to the needs of users. 2. The introduction of new features may affect the ease of use of the system.

Source: compiled by the author based on A. Bayramçavuş *et al.* (2021), D. Wolfart *et al.* (2021), M. Söylemez *et al.* (2022)

Changes to ERP systems are an integral part of their operation. The need for changes can be caused by an increase in data volume, an expansion of the list of users, improvements in business processes, or technology development. For effective change management, it is necessary to utilise modern software development lifecycle methodologies, such as Agile or the spiral model. They can be used to integrate changes gradually, minimising risks and ensuring system stability.

Consideration of system changes should also include risk management. For instance, large-scale changes to the system core or databases have a high level of intervention and require thorough testing. Less critical changes, such as customising reports or creating new forms, can be implemented more quickly.

ERP systems actively use SQL to manage relational databases. The interaction between the client and the server is based on a clearly defined mechanism: the client creates a query; the server processes it and returns the results. This structure can be used to optimise the processing of large amounts of data and ensure the reliability of the system.

Figure 1 shows a diagram of SQL in a client-server ERP system. The client part, which runs on the personal computer of the user, generates queries that are transmitted to the server. The server processes these queries by interacting with the database and returns the results in the form of information ready for analysis. This interaction ensures efficient system operation, reduces network traffic, and adapts to the growth in the number of users or data volumes. The use of the SQL standard as the main database management tool facilitates cross-platform compatibility of the system. At the same time, the complexity of integrating new modules or scaling the architecture requires significant attention to change management.

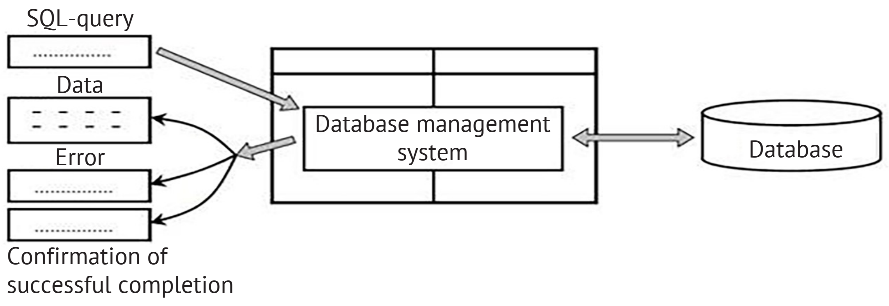
SQL-query
Data
Error
Confirmation of successful completion
Database management system
Database

Figure 1. The architecture of SQL queries interaction with the database
Source: compiled by the author based on E.F. Codd (1990), C.J. Date (1975)

ERP system scaling is a major challenge for large enterprises that regularly face an increase in the number of users, changes in business processes, and the need to integrate new modules. The flexibility of the microservices architecture can be used to adapt the system to new conditions without critical failures in its operation. For instance, the implementation of a modular approach to development ensures the independence of each component of the system, which allows making changes locally without affecting other parts. In addition, the use of modern spiral software development lifecycle models, such as Agile or the spiral model, facilitates the integration of updates, as each stage of change is tested and implemented gradually.

One of the key challenges in developing and implementing ERP systems is the need to integrate with other software solutions. To ensure comprehensive management of business processes, the system should combine functionality that covers not only the main but also auxiliary processes, such as logistics management, data analytics, e-commerce, etc. The transition from a monolithic architecture to a microservice architecture is complex and requires careful planning, as it involves significant changes in infrastructure, process design and organisation. One of the most effective methods of transition is to split the monolithic architecture into separate microservices in stages. This avoids the major risks associated with the simultaneous restructuring of the entire system. It is necessary to identify the functional areas of the monolith that can be divided into separate microservices, such as components such as order processing, inventory management, or financial accounting. These areas are then rewritten as separate microservices that can run independently of each other. Microservices should interact through well-defined APIs to ensure their independence and flexibility when changing or scaling.

One of the proven patterns for transition is to use the Strangler Fig Pattern strategy. This strategy consists of the gradual replacement of a monolithic system with microservices. At first, the old monolithic code remains in the system, and gradually its functions are replaced by new microservices. This allows for a safe transition without interrupting the entire system. In addition, API gateways are used for easy integration between old and new components, which minimises the complexity of integration. During the transition, it is also necessary to ensure that development and testing processes are automated to reduce the risk of errors. Microservices require well-defined unit tests that allow for quick detection of errors in the early stages of development. To optimise the testing and deployment of microservices, continuous integration and delivery tools should be used to reduce the risk of errors and improve the speed of deploying new versions.

There are certain risks involved in the transition. One of the main ones is the difficulty of integrating new microservices with the existing monolithic system. To minimise this risk, it is necessary to use a phased migration and the Strangler Fig strategy, as well as clearly define interfaces for interaction between components. Another important risk is data consistency between microservices. The absence of a clear data synchronisation strategy can lead to data discrepancies and errors, so approaches such as Event Sourcing or Command Query Responsibility Segregation should be used to ensure consistency. Another important challenge is the growing complexity of microservices management, which can increase the complexity of monitoring, updating and testing the system. To mitigate this risk, automated monitoring and orchestration tools such as Kubernetes or Docker Swarm should be used.

By migrating to a microservices architecture, companies can significantly increase the flexibility of their

ERP systems, reducing risks and support costs while improving stability and scalability. To achieve a successful outcome, it is necessary to carefully plan the transition, use a phased approach, apply migration templates, automate testing and monitoring processes, and minimise risks through a clear data synchronisation and security strategy. The microservices architecture simplifies the integration of external services and modules through standard APIs and data exchange protocols. For instance, modern ERP systems can interact with electronic document management services or CRM platforms. Such integrations provide end-to-end automation and simplify data transfer between different departments of the enterprise. Integration can be complicated by differences in data formats, external system interfaces, and security requirements. At the same time, the implementation of such changes often requires the cooperation of several teams, which makes it difficult to coordinate the work. To successfully overcome these challenges, it is critical to use unified data exchange standards and effectively manage stakeholder interactions.

Modern ERP systems are actively using data visualisation tools to improve analytics and management decision-making. Graphical reports, interactive dashboards and charts can quickly assess key performance indicators and identify problem areas. Visualisation also contributes to a deeper understanding of complex business processes. Automation of business processes in ERP systems significantly reduces the workload on staff by reducing the number of routine operations. For instance, automatic report generation, inventory management, and payroll functions reduce the time required to complete these tasks and reduce the risk of errors. The modular structure of ERP systems facilitates the adaptation of automation to the specifics of the enterprise. One of the key aspects of ERP systems is ensuring their stability, particularly as the company expands or business processes change. Stability depends on proper server configuration, database maintenance, and system performance monitoring. System administrators perform an important role in this process, as they are responsible for timely software updates, bug fixes, and ensuring that data is available to users.

Problems can arise due to insufficient hardware scalability, an increase in the number of database queries, or incorrect SQL query optimisation. To mitigate risks, system administrators use server and database performance monitoring to identify bottlenecks in the system, automated backups to prevent data loss, and query optimisation and index tuning to speed up database performance. The use of cloud technologies in the client-server architecture of ERP systems can significantly increase the stability and efficiency of such systems due to the ability to dynamically allocate computing resources. This allows ERP systems to adapt to changing business needs and ensure stable operation even under high loads. Cloud computing provides dynamic resource allocation, which can be used to automatically scale the system's capacity depending on the current load. In ERP systems, this is especially critical for companies that experience seasonal peaks in activity, such as retail and logistics, where computing resource requirements may vary depending on sales or the number of transactions.

One of the main advantages of cloud solutions is high availability and fault tolerance. By facilitating data backup in the cloud, which is carried out through geographically distributed data centres, the ERP system can remain available even in the event of hardware failures or local accidents. Cloud providers typically use clustering and load-balancing mechanisms to ensure minimal downtime and improve system stability. Scalability is another important advantage of cloud-based ERP solutions. Companies can start with a limited set of resources and gradually expand the system without significant capital expenditure on new server hardware. This makes it easy to integrate new modules or connect new branches or remote offices, which is important for businesses that are expanding or changing. In addition, cloud technologies can significantly reduce costs. The Pay-as-you-go model allows companies to pay only for the resources they use, making cloud solutions cost-effective. The absence of the need to maintain in-house server hardware also reduces operating costs.

Security and data protection are important aspects when implementing cloud ERP systems. Cloud providers typically offer advanced security features, including data encryption, real-time monitoring, and protection against denial-of-service attacks. Users can access the ERP system through secure channels, such as a virtual private network, which ensures a high level of data protection. Another important advantage is the global availability of cloud ERP systems. Such systems can be accessed from anywhere in the world, which is especially important for international companies or remote teams. Examples of such cloud solutions include Amazon Web Services, which offers services for dynamic resource allocation through EC2, S3 and RDS, and Microsoft Azure, which integrates with ERP systems such as Microsoft Dynamics 365 to ensure stable operation. Also worth mentioning is the Google Cloud Platform, which actively uses data analytics and machine learning technologies in the context of ERP systems.

To successfully implement cloud solutions in ERP systems, companies need to assess several key aspects. It is necessary to verify the compatibility of the existing ERP system with cloud infrastructures, assess the requirements for data security and confidentiality, and address the potential risks associated with dependence on a cloud provider. In summary, cloud technologies provide ERP systems with high stability, efficiency and flexibility, making them an important tool for businesses seeking to optimise their processes and adapt to a rapidly changing environment. In addition, the introduction of continuous monitoring systems can be used to respond quickly to potential problems, ensuring the smooth operation of the entire platform.

The process of editing ERP systems always involves certain risks, such as module failure, conflicts between old and new components, or configuration errors. To minimise the risks, it is necessary to carefully plan each stage of changes, test new components, and provide feedback from users. One of the most effective approaches is to use the spiral software development lifecycle model. This model can be used to iteratively implement changes, gradually testing each component and reducing the likelihood of major failures. In the case of less critical changes, it is advisable to use the Agile model, which provides flexibility and quick response to new requirements. Ranking risks according to the level of intervention allows administrators to allocate resources more efficiently. For instance, changes to the core of the system require maximum attention and thorough testing, while setting up new forms or reports can be done without significantly affecting other components. Adapting a system often requires upgrading technical equipment and implementing new technologies, such as microservice containers or distributed databases. At the same time, it is necessary to consider the interests of stakeholders to ensure that the adaptation meets the needs of users and does not disrupt the stability of the operation.

The use of a microservice approach can significantly improve the flexibility of the system, ensure its scalability and reduce the risks associated with expanding functionality. This is especially relevant for large enterprises that need to continuously improve processes and adapt to changes in the business environment. A key advantage of microservice architecture is the ability to independently scale individual system components. This ensures greater resilience to loads arising from the growth of data volumes or the number of users. The introduction of a modular approach also helps to reduce the complexity of system management and reduce maintenance costs.

The peculiarities of the transition from monolithic to microservice architecture using hybrid database design were considered by T. Ng *et al.* (2024). The authors analysed approaches to adapting databases for microservices, ensuring compatibility between old and new modules, and improving the efficiency of query processing in distributed systems. Secure microservices and a security framework for user interaction with microservice applications were described by M.I. Elkholy & M.A. Marzok (2022). The authors present methods of authentication, authorisation, and data protection in microservice ecosystems. The automation of business processes implemented as part of the study significantly reduces the time required to perform routine tasks and reduces the number of errors. Optimisation of SQL queries and database management improves system performance, providing fast access to data even under high load conditions. This creates the basis for improving the performance of ERP systems in a dynamic business environment. The study highlighted the integration of ERP systems with external modules and platforms. The use of standard APIs and protocols simplifies the process of data exchange between systems, providing a single information environment. This approach contributes to the creation of integrated business process management solutions that meet the needs of modern enterprises.

Another important aspect considered in the proposed methodology is ensuring the reliability of ERP systems in the process of implementing new modules and functionality. An important factor is to minimise downtime during upgrades and maintain stable system operation even under high load conditions. The use of testing methods in real-life scenarios and modelling of various operating conditions allows identifying potential bottlenecks before the implementation of changes, which significantly reduces the risk of failure. The use of microservice patterns for big data systems was covered by P. Ataei & D. Staegemann (2023). The authors considered ways to improve the processing of large amounts of data through modularity and distribution.

An additional result is the increased security of ERP systems due to the isolation of microservices from each other. This approach minimises the impact of failures in individual components and ensures high availability of the rest of the system. Achieving this level of security is critical for large enterprises that process large amounts of sensitive data. A systematic mapping study of microservices development based on domain-driven design is presented in J. Sangabriel-Alarcón *et al.* (2023). The concepts of domains, strategic modelling, and complexity management in large distributed systems are considered. A generalised description of the principles of microservice architecture is given in the study by V. Božić (2023). The basic concepts, advantages, disadvantages, and prospects for the development of this architecture are considered.

The business process automation implemented in the study not only increases the efficiency of tasks but also helps to improve the interaction between different departments of the enterprise. The integration of data visualisation tools, such as interactive dashboards, ensures quick decision-making for managers based on accurate and up-to-date performance indicators. This creates a transparent environment for resource management and operations planning. The peculiarities of decision-making at the stage of initiating corporate information systems projects were analysed by I. Barskaya *et al.* (2014). The specifics of management decision-making in the process of implementing corporate information systems, particularly the impact of initiation on further stages of the project life cycle, were considered. The main factors that contribute to the successful implementation of such projects are identified.

A study of microservice architecture in the Edge Computing environment was conducted by N. Rathore *et al*. (2020). The study highlights the aspects of adapting microservices to the requirements of edge computing, including reducing latency and improving energy efficiency. The use of microservices and event-oriented architecture for processing large data streams was presented by S. Zhelev & A. Rozeva (2019). The advantages of distributed systems for processing streaming data in real-time were highlighted.

Of particular interest is the possibility of using the proposed approach in combination with modern artificial intelligence technologies. The integration of machine learning algorithms can significantly improve the processes of load forecasting, resource optimisation, and detection of anomalies in the system. Such solutions will allow ERP systems to automatically adapt to changes in the business environment, increasing their efficiency and reliability. The optimisation of enterprise infrastructure using microservices is considered by A.M. Abd-Elwahab *et al*. (2023). The authors analysed methods of adapting infrastructure to the growing needs of businesses and reducing costs.

Further development of the proposed methodology may include its adaptation to the specifics of various industries. For instance, in the manufacturing sector, ERP systems should address the complexity of supply chains, integration with Internet of Things devices to monitor equipment status, and automation of quality control processes. In trade and logistics, real-time inventory management and integration with e-commerce platforms are becoming increasingly important. The concept of creating a minimum viable product and design thinking in the management of IT teams was highlighted by I. Blyznyukova *et al*. (2021). The study focuses on the principles of building a minimum viable product, adapting design thinking to information technology projects, and developing a team management strategy to achieve the goals. Integration with smart enterprise concepts, where ERP systems are a central component of managing all processes, can be a separate area of development. In such conditions, not only microservice architecture plays an important role, but also the ability of ERP systems to support fast data exchange with automated production lines, robotic warehouse systems and other elements of a modern enterprise.

Integration of the proposed methodology with the latest technological approaches can enhance its effectiveness in solving complex problems of modern business. One of the most promising areas is the use of artificial intelligence to optimise management processes in ERP systems. For instance, machine learning algorithms can analyse large amounts of data, predict system failures, optimise resources, and automatically adjust workflows to increase productivity. Approaches to refactoring from monolith to microservices were classified in the study by J. Fritzsch *et al*. (2019). They show various strategies for reorganising systems to increase their flexibility and scalability.

It is also worth noting the importance of automating supply chain management, which is a key component of many ERP systems. Integration of demand forecasting, real-time inventory monitoring and automated logistics management will allow enterprises to reduce costs, minimise the risks of shortages or surpluses and ensure high accuracy of deliveries. Technologies of data integration in national information systems were studied by N. Shakhovska & D. Tarasov (2011). They analysed methods of integrating heterogeneous data sources, building a single database and using modern approaches to information exchange. Optimisation of microservices deployment costs using cluster autoscaling and the Spot pricing model is discussed by D. Edirisinghe *et al*. (2024). The authors present approaches to reducing costs while maintaining high availability. Peculiarities of integration of information systems of instrument-making enterprises were considered by O. Legotchenkov (2016). The study analyses the mechanisms of data integration, the principles of building relational tables and the means of ensuring communication between different information systems to optimise business processes.

Load balancing and service discovery in big data applications based on microservices and Docker Swarm were considered by N. Singh *et al*. (2023). The authors described methods for integrating Docker Swarm to improve data processing efficiency in distributed environments. The role of containerisation in transforming software development and deployment through microservice architecture was analysed by J. Mukaj (2023). The author described the use of Docker and Kubernetes to ensure flexibility, scalability, and reliability.

A separate focus area is integrating ERP systems with future technologies such as quantum computing. Although these technologies are only just beginning to find their way into the corporate environment, in the long run, they can provide unprecedented performance and security in data processing. Preparing an ERP architecture to support such technologies can be a strategic advantage for enterprises. The use of quantum computing in microservice architecture was analysed by S.K. Eddin *et al*. (2024). The potential of quantum microservices to revolutionise data processing and optimise complex computing is explored. Thus, the developed methodology opens wide opportunities for further improvement of ERP systems. Its adaptability, focus on innovation, and ability to integrate with current and future technologies render it a versatile solution that meets not only current but also future business challenges.

CONCLUSIONS

The study substantiates the feasibility of transitioning ERP systems from a monolithic architecture to a microservice architecture that meets modern business challenges, in particular, increased requirements for scalability, flexibility, performance and security. The results of the analysis showed that microservice architecture facilitates the adaptation of ERP systems to changes in the business environment due to modularity, independent deployment of components and simplified upgrade processes. The use of programming patterns, such as the Strangler Fig Pattern, minimises the risks of system upgrades by ensuring a phased transition to a new architecture.

The study demonstrated that the use of modern tools, such as API gateways, asynchronous interaction and caching technologies, helps optimise the performance of ERP systems in highly loaded conditions. The theoretical analysis has confirmed the effectiveness of using cloud technologies that provide dynamic resource allocation, increase stability and enable rapid scaling. In addition, approaches to the integration of ERP systems with external modules through standardised APIs are considered, which creates a single information space for the enterprise.

Recommendations for the implementation of business process automation solutions and the integration of artificial intelligence algorithms allow ERP systems to adapt to changing business conditions, increasing the efficiency of data and process management. The integration of modern technologies, including artificial intelligence and analytical platforms, creates prospects for automating complex business processes and increasing the competitiveness of enterprises. This research forms the basis for creating flexible, secure and scalable ERP systems that can meet the needs of the globalised market and ensure business stability in the face of dynamic change.

The obtained results created a comprehensive theoretical basis for further research and practical solutions in the field of ERP systems modernisation, in the context of the transition to microservice architecture. The proposed approaches can be the basis for developing innovative ERP solutions that can meet current and future business challenges. Further development of the theoretical foundations will allow these solutions to be adapted to the specific needs of various industries, such as manufacturing, logistics and e-commerce, ensuring their efficiency and sustainable development.

ACKNOWLEDGEMENTS

None.

CONFLICT OF INTEREST

None.

REFERENCES

[1] Abdelfattah, A.S., & Cerný, T. (2022). Microservices security challenges and approaches. In R.A. Buchmann, G.C. Silaghi, D. Bufnea, V. Niculescu, G. Czibula, C. Barry, M. Lang, H. Linger & C. Schneider (Eds.), *Information systems development: Artificial intelligence for information systems development and operations*. Cluj-Napoca: Risoprint. doi: 10.62036/ISD.2022.27.

[2] Abd-Elwahab, A.M., Mohamed, A.G., & Shaaban, E.M. (2023). MicroServices-driven enterprise architecture model for infrastructure optimization. *Future Business Journal*, 9, article number 90. doi: 10.1186/s43093-023-00268-3.

[3] Ataei, P., & Staegemann, D. (2023). Application of microservices patterns to big data systems. *Journal of Big Data*, 10(1), article number 56. doi: 10.1186/s40537-023-00733-4.

[4] Barskaya, I., Teslenko, P., & Denysenko, V. (2014). Peculiarities of decision-making at the stage of initiating projects for the creation of information information systems. *Project Management and Production Development*, 1, 32-39.

[5] Bayramçavuş, A., Kaya, M.Ç., & Dogru, A.H. (2021). Interoperability of microservice-based systems. In *Proceedings of the 13th international conference on electrical and electronics engineering* (pp. 594-598). Bursa: IEEE. doi: 10.23919/ELECO54474.2021.9677712.

[6] Blyznyukova, I., Teslenko, P., Danchenko, O., & Melenchuk, V. (2021). The concept of creating a minimum viable product and design-thinking in the IT-project team management. *Bulletin of the National Technical University "KhPI". Series: Strategic Management, Portfolio, Program and Project Management*, 2(4), 11-17. doi: 10.20998/2413-3000.2021.4.2.

[7] Božić, V. (2023). Microservices architecture. *ResearchGate*. doi: 10.13140/RG.2.2.21902.84802.

[8] Codd, E.F. (1990). *The relational model for database management (2nd ed.)*. Boston: Addison-Wesley Longman Publishing.

[9] Date, C.J. (1975). *An introduction to database systems*. Boston: Addison-Wesley Longman Publishing.

[10] Eddin, S.K., Salloum, H., Shahin, M.N., Mazzara, M., & Bahrami, M.R. (2024). Quantum microservices: Transforming software architecture with quantum computing. In L. Barolli (Ed.), *Advanced information networking and applications* (pp. 227-237). Cham: Springer. doi: 10.1007/978-3-031-57942-4_23.

[11] Edirisinghe, D., Rajapakse, K., Abeysinghe, P., & Rathnayake, S. (2024). Cost-optimal microservices deployment with cluster autoscaling and spot pricing. In *Proceedings of the 2024 IEEE international conference on cloud computing technology and science (CloudCom)* (pp. 87-94). Abu Dhabi: IEEE. doi: 10.1109/CloudCom62794.2024.00026.

[12] Elkholy, M.I., & Marzok, M.A. (2022). Trusted microservices: A security framework for users' interaction with microservices applications. *Journal of Information Security and Cybercrimes Research*, 5(2), 135-143. doi: 10.26735/QOPM9166.

[13] Fritzsch, J., Bogner, J., Zimmermann, A., & Wagner, S. (2019). From monolith to microservices: A classification of refactoring approaches. In J.-M. Bruel, M. Mazzara & B. Meyer (Eds.), *Software engineering aspects of continuous development and new paradigms of software production and deployment* (pp. 128-141). Cham: Springer. doi: 10.1007/978-3-030-06019-0_10.

[14] Górski, T., & Wozniak, A.P. (2021). Optimization of business process execution in services architecture: A systematic literature review. *IEEE Access*, 9, 111833-111852. doi: 10.1109/ACCESS.2021.3102668.

[15] Grambow, M., Wittern, E., & Bermbach, D. (2020). Benchmarking the performance of microservice applications. *ACM SIGAPP Applied Computing Review*, 20(3), 20-34. doi: 10.1145/3429204.3429206.

[16] Iegorchenkov, O. (2016). Information systems integration of instrument-making enterprise. *Managing the Development of Complex Systems*, 28, 124-129.

[17] Lee, C., Kim, H.F., & Lee, B.G. (2024). A systematic literature review on the strategic shift to cloud ERP: Leveraging microservice architecture and MSPs for resilience and agility. *Electronics*, 13(14), article number 2885. doi: 10.3390/electronics13142885.

[18] Mukaj, J. (2023). *Containerization: Revolutionizing software development and deployment through microservices architecture using Docker and Kubernetes*. (Bachelor's thesis, Epoka University, Tirana, Albania). doi: 10.13140/RG.2.2.23804.51841.

[19] Ng, T., Rawi, A.A.B., Sum, C.S., Tso, E., Yau, P.C., & Wong, D. (2024). Migrating from monolithic to microservices with hybrid database design architecture. In *Proceedings of the 9th international conference on intelligent information technology* (pp. 536-541). New York: Association for Computing Machinery. doi: 10.1145/3654522.3654602.

[20] Oyeniran, O.C., Adewusi, A.O., Adeleke, A.G., Akwawa, L.A., & Azubuko, C.F. (2024). Microservices architecture in cloud-native applications: Design patterns and scalability. *Computer Science & IT Research Journal*, 5(9), 2107-2124. doi: 10.51594/csitrj.v5i9.1554.

[21] Rathore, N., Rajavat, A., & Patel, M. (2020). Investigations of microservices architecture in edge computing environment. In R.K. Shukla, J. Agrawal, S. Sharma, N.S. Chaudhari & K.K. Shukla (Eds.), *Social networking and computational intelligence* (pp. 77-84). Singapore: Springer. doi: 10.1007/978-981-15-2071-6_7.

[22] Sangabriel-Alarcón, J., Ocharán-Hernández, J.O., Cortés-Verdín, K., & Limón, X. (2023). Domain-driven design for microservices architecture systems development: A systematic mapping study. In *Proceedings of the 11th international conference in software engineering research and innovation* (pp. 25-34). León: IEEE. doi: 10.1109/CONISOFT58849.2023.00014.

[23] Shakhovska, N., & Tarasov, D. (2011). Technologies of data integration of information systems of Lviv Polytechnic National University. *Bulletin of Lviv Polytechnic National Univeristy. Series of Information Technology in Higher Education*, 703, 9-20.

[24] Singh, N., Hamid, Y., Juneja, S., Srivastava, G., Dhiman, G., Gadekallu, T.R., & Shah, M.A. (2023). Load balancing and service discovery using Docker Swarm for microservice based big data applications. *Journal of Cloud Computing*, 12(1), article number 4. doi: 10.1186/s13677-022-00358-7.

[25] Söylemez, M., Tekinerdogan, B., & Tahran, A.K. (2022). Challenges and solution directions of microservice architectures: A systematic literature review. *Applied Sciences*, 12(11), article number 5507. doi: 10.3390/app12115507.

[26] Wolfart, D., Assunção, W.K.G., da Silva, I.F., Domingos, D.C.P., Schmeing, E., Villaca, G.L.D., & Paza, D.D.N. (2021). Modernizing legacy systems with microservices: A roadmap. In *Proceedings of the 25th international conference on evaluation and assessment in software engineering* (pp. 149-159). New York: Association for Computing Machinery. doi: 10.1145/3463274.3463334.

[27] Zhelev, S., & Rozeva, A. (2019). Using microservices and event driven architecture for big data stream processing. *AIP Conference Proceedings*, 2172, article number 090010. doi: 10.1063/1.5133587.

Архітектура мікросервісів для ERP-систем

Сергій Слівка

Аспірант
Національний університет «Одеська політехніка»
65044, просп. Шевченка, 1, м. Одеса, Україна
https://orcid.org/0009-0003-1704-1415

Анотація. Традиційні системи управління ресурсами підприємства (ERP), побудовані за принципом монолітної архітектури, стають все менш ефективними в сучасному бізнес-середовищі. Зростаюча складність виробничих процесів, збільшення обсягів даних та необхідність швидкої адаптації до змін створюють значні виклики для таких систем. Метою дослідження були аналіз та розробка методів переходу від монолітної архітектури до мікросервісної в ERP-системах для забезпечення їх масштабованості, гнучкості та безпеки. Проведено комплексний аналіз клієнт-серверної архітектури ERP-систем та їх основних компонентів, включаючи реляційні бази даних і запити на мові структурованих запитів (SQL). Досліджено принципи роботи функцій і процедур, взаємодії між таблицями та створення запитів у системах з великою кількістю користувачів. Також були розглянуті сучасні стратегії декомпозиції монолітних систем на мікросервіси, включаючи методи синхронізації даних та управління компонентами. Наголошено на аспектах безпеки та ризиках переходу до мікросервісної архітектури. Розроблено підходи до трансформації ERP-систем на основі мікросервісної архітектури, що дозволяє знизити складність управління, підвищити продуктивність і знизити ризики в бізнес-процесах. Запропоновано механізми ефективного синхронізованого обміну даними між компонентами та забезпечення їх надійності. Дослідження визначило, що мікросервісна архітектура дає можливість незалежного масштабування окремих елементів системи, підвищуючи відмовостійкість та продуктивність при високих навантаженнях. Були досліджені та вирішені поширені проблеми, включаючи неправильні баланси та помилки у звітах. Розроблені підходи можуть бути використані для модернізації ERP-систем великих підприємств, особливо у виробничому секторі, де важлива висока продуктивність і точність обробки даних

Ключові слова: клієнт-серверна; масштабованість; стейкхолдер; критичний шлях; критична подія; автоматизація; хмарні технології