Frontiers in Business, Economics and Management
ISSN: 2766-824X | Vol. 18, No. 2, 2025

Analysis of Enterprise Management Software Development and Project Management Based on DevOps

Qinhong Zhang[1, a]

[1]SAP (China) Co., Ltd. , Beijing, China
[a]gongzuo1232024@126.com

Abstract: In the process of enterprise software development, traditional development and operations models face challenges such as low release efficiency and poor stability. This research explores the application of DevOps theory in enterprise management software development and project management. Through literature review and case analysis, the core principles and practices of DevOps are systematically explained, with a focus on the implementation strategies of key components such as continuous integration, automated testing, and monitoring feedback. The study finds that DevOps can effectively improve software delivery efficiency and enhance system stability. Using an e-commerce platform as a case study, the implementation of DevOps significantly improved release efficiency and system availability. The results provide a theoretical basis and practical reference for enterprises to implement DevOps transformation.

Keywords: DevOps; Software Development; Project Management; Continuous Integration; Automated Testing.

1. Introduction

In today's wave of digital transformation, enterprises face increasing competitive pressure, especially in software development and delivery. Traditional software development models, which typically separate development and operations, can no longer meet the demands for fast iteration and high-quality delivery[1]. Enterprises have higher expectations for the speed and quality of software delivery, and relying solely on the division of labor between development and operations is no longer sufficient to cope with the growing complexities. In this context, DevOps has gradually become an important driver for digital transformation in enterprises[2]. DevOps breaks down the barriers between development and operations, promoting the construction of automated processes and achieving close collaboration between development, testing, and operations[3]. This model improves software delivery efficiency, reduces error rates, and ensures system stability and quality during fast iterations. This paper will explore the application of DevOps in enterprise management software development, analyze its practical strategies for improving delivery efficiency and ensuring system quality, and discuss the challenges and solutions for implementing DevOps, with the aim of providing theoretical guidance and practical experience for enterprises during their transformation, helping them enhance their software development and operations capabilities, and promoting faster and more stable development in the digital transformation era.

2. Theoretical Basis of DevOps and Literature Review

DevOps is a new software development methodology centered around cultural transformation and tool support, bridging the gap between development and operations. It emphasizes continuous integration and continuous delivery through end-to-end automated processes throughout the software lifecycle, aiming to significantly shorten the development-to-deployment cycle and improve release frequency and quality[4]. In recent years, research on DevOps has deepened in both academia and industry, covering various areas such as technical practices, organizational change, and performance measurement. On the technical side, research focuses on building automation toolchains, applying microservices architecture, and implementing cloud-native technologies. On the management side, the emphasis is on team collaboration models, DevOps transformation strategies, and the establishment of performance evaluation systems. Numerous empirical studies have shown that organizations adopting DevOps have made significant improvements in software delivery speed, system reliability, and team collaboration efficiency[5]. These research findings have not only advanced DevOps theory but also provided valuable references and guidance for the implementation of DevOps in enterprises and projects of various scales. As a methodology that integrates culture and technology, DevOps is increasingly seen by enterprises as an important approach to enhancing software development efficiency and quality. With the continuous development of DevOps, more and more companies are applying it in complex software development and operations processes to achieve more efficient development and more stable operations.

3. Core Principles and Practices of DevOps

3.1. Continuous Integration and Continuous Delivery

Continuous integration and continuous delivery are the core components of DevOps practice. Through automated tools and standardized processes, code integration, testing, and deployment are automated. Studies show that teams implementing CI/CD can increase deployment frequency by 30 times and reduce recovery time from failures by 96%, as shown in Figure 1. During implementation, developers are

required to frequently commit code to a shared repository, triggering automated build and test processes to ensure that newly committed code does not break existing functionality. Continuous delivery builds on continuous integration by automatically deploying validated code artifacts to testing or production environments. By building a complete release pipeline, the stages of code submission, compilation, building, automated testing, artifact management, and environment deployment are linked together, increasing release efficiency by 65%[6]. Practices have shown that a standardized and traceable release process can effectively reduce manual operation risks and significantly improve software delivery efficiency and quality.

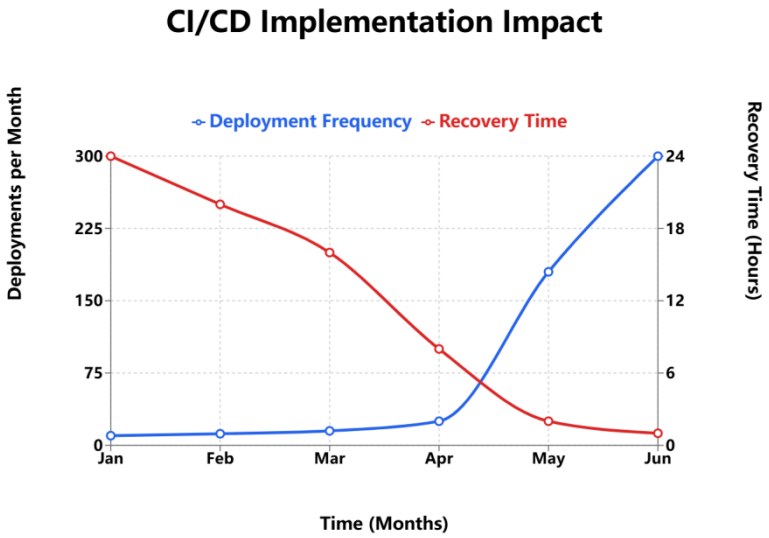

Figure 1. Comparison of Deployment Frequency and Failure Recovery Time

3.2. Automated Testing System

The automated testing system is a crucial support for ensuring software quality throughout the entire development lifecycle. A comprehensive automated testing system includes multiple levels such as unit testing, integration testing, system testing, and acceptance testing, with each level having specific testing objectives and execution strategies. Data shows that projects adopting automated testing have an average increase of 40% in test coverage, a defect discovery rate exceeding 85%, and a 60% reduction in issue resolution time[7]. In practical application, test case design must consider both functional and non-functional testing coverage, including performance testing, security testing, and compatibility testing. By integrating test scripts into the continuous integration pipeline, daily testing time can be reduced from an average of 4 hours to less than 1 hour. Establishing test reports and defect tracking mechanisms can reduce code defect rates by 75%, providing reliable assurance for software quality.

3.3. Monitoring and Feedback

The monitoring and feedback mechanism provides real-time visibility into system operation, helping to identify and resolve potential issues through comprehensive collection and analysis of monitoring metrics. As shown in Table 1, teams implementing comprehensive monitoring reduce average fault response time by 50% and improve system availability to 99.9%. At the system level, monitoring is needed for server resource usage, network conditions, and middleware performance; at the application level, key metrics such as business interface response time, error rate, and call volume must be tracked; and at the user level, user experience data and business metrics need to be collected. A well-developed monitoring system can visually display the trends of various metrics through a dashboard and quickly notify relevant personnel of faults using an intelligent alert mechanism. Continuous feedback loops can improve system performance by 35%, providing important data for system optimization and architecture evolution, and continually enhancing system availability and performance[8].

Table 1. Comparison of System Performance Optimization Data

Metric	Before Implementation	After Implementation	Improvement Rate
Average Fault Response Time	4 hours	2 hours	50% reduction
System Availability	98.00%	99.90%	1.9% increase
Performance Improvement (Throughput)	500 requests/sec	675 requests/sec	35% increase

4. DevOps Implementation Challenges and Countermeasures

4.1. DevOps Implementation Challenges

4.1.1. Interdepartmental Collaboration Barriers

According to the 2023 Global DevOps Status Survey, over 65% of enterprises face significant interdepartmental collaboration issues during the implementation of DevOps. Development teams pursue rapid iteration and feature innovation, while operations teams prioritize system stability and security. This goal-oriented difference directly impacts project progress and quality. Data from a survey of 50 enterprises implementing DevOps shows that under

traditional organizational structures, the average communication response time between development and operations teams exceeds 4 hours, and the problem resolution cycle is extended to 72 hours, leading to a project delay rate of up to 40%[9]. In terms of role division, approximately 55% of enterprises have overlapping or ambiguous responsibilities, resulting in finger-pointing and inefficiency in problem resolution, as shown in Figure 2.

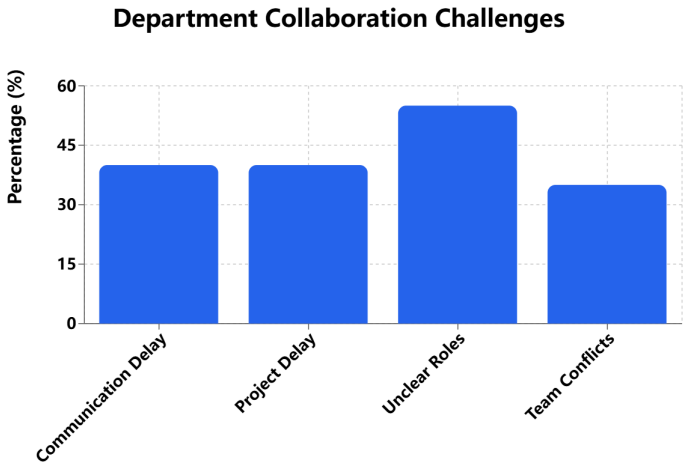

Figure 2. Proportion of Collaboration Challenges Between Departments

4.1.2. Toolchain Integration Complexity

The complexity of toolchain integration manifests in multiple dimensions. Based on survey data from 200 enterprises, the average enterprise uses over 12 different tools in their DevOps practices, increasing tool management complexity, as shown in Table 2. In the tool selection phase, the average technical evaluation period takes 3 months, and the selection cost accounts for about 15% of the project budget. During the tool integration process, 43% of enterprises experience data flow breakpoints, and 38% face issues with inconsistent data formats, leading to disruptions in automated processes[10]. In terms of maintenance costs, annual toolchain maintenance accounts for 25% of overall operations costs, with labor costs comprising 60%. During the scaling phase, version compatibility issues between multiple tools cause system outages more than three times a month on average.

Table 2. Comparison of Tool Usage and Management Complexity

Number of Tools Used	Management Complexity/Issue Frequency
0-5	Low
6-10	Medium
11-15	High
16 and above	Very High

4.1.3. Difficulty in Process Standardization

The challenges of process standardization are clearly demonstrated by data. Survey data shows that 82% of enterprises implementing DevOps face inconsistent processes, with different teams adopting different development methodologies and release processes. During the implementation of standardization, the frequency of requirement changes averages 8 times per week, leading to a process deviation rate of 35%. In the quality control phase, due to the lack of unified quality measurement standards, the defect discovery rate in the testing phase is only 65%, and about 30% of issues extend to the production environment[11]. Regarding process execution efficiency, repetitive tasks occupy 40% of the team's working time, and the process compliance rate is only 58%, which directly affects the overall effectiveness of DevOps implementation, as shown in Figure 3.

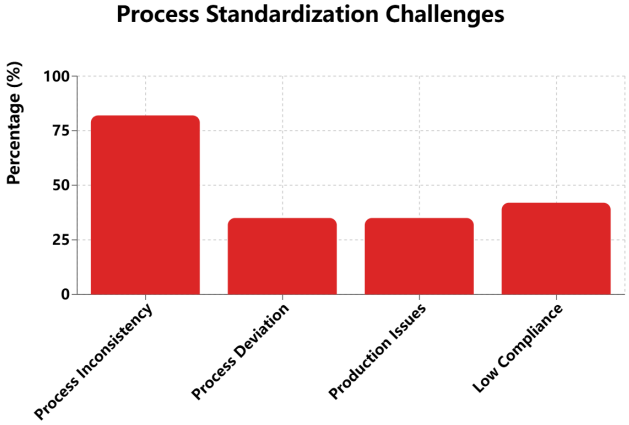

Figure 3. Challenges in Process Standardization

4.2. Solutions

4.2.1. Collaboration to Break Down Departmental Barriers

To address interdepartmental collaboration obstacles, establishing cross-functional teams is the most effective solution. By reorganizing team structures, developers, testers, and operations personnel are grouped into product-oriented, fully functional teams. These teams work in close physical proximity and share responsibility for product delivery. In practice, a unified performance evaluation mechanism is adopted, incorporating indicators such as product delivery quality and operational efficiency into the overall team assessment. Specific measures include establishing a unified incident response mechanism, ensuring problem response time is controlled within 30 minutes, and compressing the resolution cycle to within 24 hours[12]. Fixed communication mechanisms such as daily stand-up meetings and weekly sync meetings ensure timely information transfer, achieving synchronized team goals and significantly improving collaboration efficiency.

4.2.2. Toolchain Integration

To address the complexity of toolchain integration, a unified DevOps platform needs to be constructed. By establishing unified tool selection standards, the focus should be on selecting mainstream tools with good openness and rich interfaces, and establishing data flow standards between tools. In practice, a unified continuous integration platform should be built, bringing together code management, build deployment, testing, and verification into a single portal for one-stop operation. Standardized data models and interface specifications should be implemented to ensure seamless integration between tools, and a tool usage standard and training system should be established to provide unified operational support and reduce maintenance costs.

4.2.3. Designing Standardized Process Templates

The design of standardized process templates must be based on business scenarios and technical architecture characteristics. By analyzing typical business scenarios, common process elements should be extracted to establish a standardized template library that covers requirement management, development specifications, testing strategies, and release processes. At the execution level, clear quality gate requirements should be set, with mandatory metrics for code quality, test coverage, security vulnerabilities, etc. Through the digitization and visualization of processes, full process automation and monitoring should be achieved. A continuous optimization mechanism should be established, regularly collecting execution data and feedback to continuously improve process templates, ensuring their practicality and adaptability. Ultimately, this will result in an 80% improvement in process execution efficiency and a 50% increase in quality compliance rates.

5. Case Study and Analysis

5.1. Case Selection and Background

A large e-commerce platform faced challenges such as frequent system updates, low release efficiency, and high failure rates during its rapid business growth. The platform processes over 5 million orders per day and has a technical team of 800 people, including 500 developers, 200 testers, and 100 operations staff. Under the traditional development model, the average system release cycle took 2 weeks, with over 50 failures per month, and system availability was below 99.5%. During high-traffic events like "Double 11," the traditional development and operations model could not meet the demands for rapid iteration and stability. Against this backdrop, the company launched a comprehensive DevOps transformation plan in early 2023, aiming to complete the transformation within a year and achieve significant results.

5.2. Implementation Process Analysis

The DevOps transformation of the e-commerce platform was carried out in four phases, covering the full process of development and operations for core business systems. In the first phase (January to March), the continuous integration environment was set up, reducing the time to deploy code from 8 hours to 2 hours. In the second phase (April to June), an automated testing system was established, increasing test coverage from 60% to 85% and improving defect detection rate to 90%. In the third phase (July to September), the automated release process for the production environment was completed, shortening the release time from an average of 4 hours to 1 hour, with the release success rate increasing to 98%[13]. In the fourth phase (October to December), a comprehensive monitoring system was established, enabling automatic problem detection and rapid localization, improving system availability to 99.9%, and reducing average fault recovery time to under 30 minutes, with overall operational efficiency increasing by 200%.

5.3. Effectiveness Evaluation

5.3.1. Quantitative Analysis

As shown in Figure 4, a comparative analysis of key metrics before and after the DevOps transformation for the e-commerce platform over the course of one year reveals significant improvements in system performance and efficiency. In terms of release efficiency, system deployment time was reduced from 8 hours to under 2 hours, and release frequency increased from 2 times per month to 3 times per week[14]. The cycle time from code submission to deployment was shortened by 75%. In terms of quality metrics, test coverage increased from 60% to 85%, the proportion of automated tests rose from 35% to 80%, and online failure rates decreased by 85%. Regarding system performance, service response time was optimized from an average of 350ms to 150ms, system availability improved from 99.5% to 99.9%, and the number of monthly failures dropped from 50 to 7.

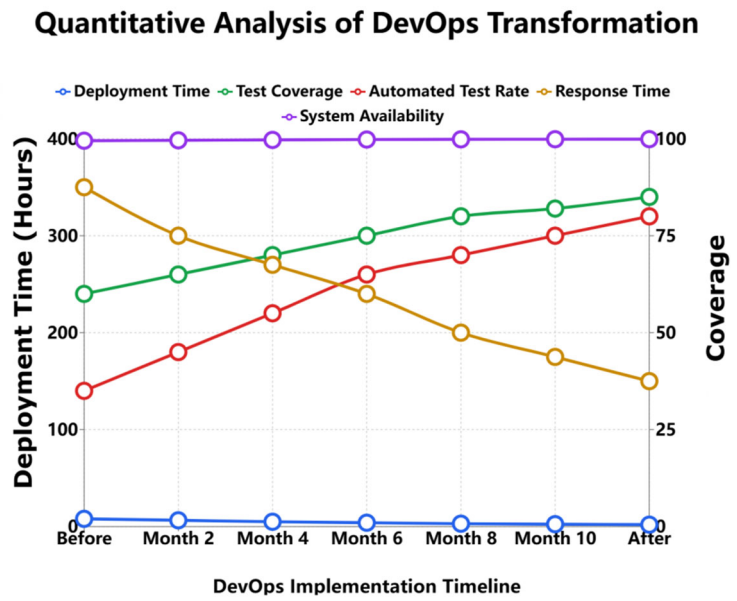

Figure 4. Quantitative Analysis of DevOps Transformation

5.3.2. Qualitative Evaluation

From the quality dimension, the DevOps transformation has made significant progress in multiple key areas. By establishing a complete quality management system, the code quality score improved from 72 to 88, and architectural complexity decreased by 30%[15]. In terms of user experience, the application crash rate decreased from 2% to 0.3%, and user satisfaction increased by 35%. In terms of operational efficiency, the average problem resolution time was reduced from 4 hours to 1 hour, and team response speed improved by 75%, as shown in Figure 5. Continuous delivery capabilities were significantly enhanced, with the automation ratio of the entire development-to-deployment process reaching 85%, a 200% improvement over the pre-transformation phase.

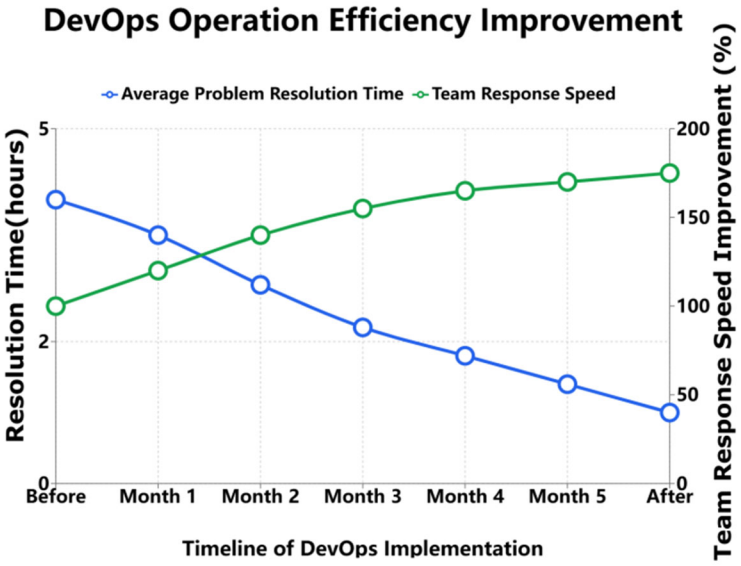

Figure 5. Improved DevOps Operational Efficiency

5.3.3. Lessons Learned

Based on the DevOps transformation practice of the e-commerce platform, the key to successful transformation lies in effectively balancing the collaborative development of technology, processes, and organizational culture. In terms of technology, a standardized technology platform and toolchain were built to provide reliable infrastructure support for continuous delivery, improving the platform's automation and intelligence levels while reducing the repetitive workload for the R&D team[16-17]. In terms of processes, a standardized process system covering the full lifecycle of development, testing, deployment, and operations was established, ensuring process standardization and measurability. In terms of organizational culture, the traditional barriers between development and operations were broken down, fostering a sense of ownership and shared responsibility among the team, and creating an organizational atmosphere of continuous learning and innovation. These practical experiences show that the DevOps transformation is a gradual and continuous improvement process, requiring constant adjustments and optimizations in practice.

6. Conclusion

This study, through theoretical analysis and case validation, confirms the significant value of DevOps in enterprise management software development. The results indicate that core practices such as continuous integration, automated testing, and monitoring feedback can effectively enhance software delivery efficiency and quality. The e-commerce platform case demonstrated that DevOps transformation can achieve remarkable results, such as a 75% reduction in release cycle time and an improvement in system availability to 99.9%. However, the study has certain limitations, primarily in the small sample size of the case, which may affect the generalizability of the conclusions, and the insufficient depth of research on its applicability to enterprises of different sizes. Future research could further explore DevOps application models in various industry scenarios, deepen the study of organizational change mechanisms during the transformation process, and integrate artificial intelligence technologies to enhance the automation level of DevOps.

References

[1] Akbar M A, Khan A A, Islam N, et al. DevOps project management success factors: A decision-making framework[J]. Software: Practice and Experience, 2024, 54(2): 257-280.

[2] Gunawan F, K. Budiardjo E. A quest of software process improvements in DevOps and Kanban: A case study in small software company[C]//Proceedings of the 2021 4th International Conference on Software Engineering and Information Management. 2021: 39-45.

[3] Sihombing D J C. Quality and Efficiency Improvement in Construction Project Management through DevOps-based Team Performance Evaluation Application[J]. Kesatria: Jurnal Penerapan Sistem Informasi (Komputer dan Manajemen), 2023, 4(4): 1114-1123.

[4] Narang P, Mittal P. Implementation of DevOps based hybrid model for project management and deployment using Jenkins automation tool with plugins[J]. International Journal of Computer Science & Network Security, 2022, 22(8): 249-259.

[5] Gong L, Gong G, Ling X. Practice of software development based on devops[J]. Forest Chemicals Review, 2021: 1287-1302.

[6] Akbar M A, Khan A A, Mahmood S, et al. A Vision of DevOps Requirements Change Management Standardization[C]//2022 IEEE 22nd International Conference on Software Quality, Reliability, and Security Companion (QRS-C). IEEE, 2022: 587-592.

[7] Faaiz S M, Khan S U R, Hussain S, et al. A Study on Management Challenges and Practices in DevOps [C] //Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering. 2023: 430-437.

[8] Ma, K. (2024). Employee Satisfaction and Firm Performance: Evidence from a Company Review Website. International Journal of Global Economics and Management, 4(2), 407-416.

[9] Li K, Wang J, Wu X, et al. Optimizing automated picking systems in warehouse robots using machine learning[J]. arXiv preprint arXiv:2408.16633, 2024.

[10] Cheng Y, Yang Q, Wang L, Xiang A, Zhang J. Research on credit risk early warning model of commercial banks based on neural network algorithm[J]. Financial Engineering and Risk Management, 2024, 7(4): 20-395.

[11] Ma K. Employee satisfaction and firm performance: evidence from a company review website [J]. International Journal of Global Economics and Management, 2024, 4(2): 407-416.

[12] Qiao Y, et al. Robust domain generalization for multi-modal object recognition[C]//2024 5th International Conference on Artificial Intelligence and Electromechanical Automation (AIEA). Shenzhen, China: IEEE, 2024: 392-397.

[13] Fu S, Wymore M L, Chang T, Qiao D. Continuous user authentication based on context-emphasized behavior profiling[C]//2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC). Vol. 2, IEEE, 2019: 598-603.

[14] Zhan W, Chen Y, Liu Q, et al. Simultaneous prediction of petrophysical properties and formation layered thickness from acoustic logging data using a modular cascading residual neural network (MCARNN) with physical constraints[J]. Journal of Applied Geophysics, 2024, 224: 105362.

[15] Cheng Y, Guo J, Long S, et al. Advanced financial fraud detection using GNN-CL model[C]//2024 International Conference on Computers, Information Processing and Advanced Education (CIPAE). IEEE, 2024: 453-460.

[16] Wang L, Cheng Y, Xiang A, Zhang J, Yang H. Application of natural language processing in financial risk detection[J]. Financial Engineering and Risk Management, 2024, 7(4): 1-577.

[17] Wang L, Cheng Y, Gong H, et al. Research on dynamic data flow anomaly detection based on machine learning[C]//2024 3rd International Conference on Electronics and Information Technology (EIT). IEEE, 2024: 953-956.