GEORGE BROWN PRESS
JOURNALS, BOOKS & PROCEEDINGS

Journal of Medicine and Life Sciences

Article

Optimization of Medical Device Software Lifecycle Management Based on DevOps

Minkang Zhang [1,*]

[1] Clinical Diagnostics Group (CDG) Software Team, Bio-Rad Laboratories, Hercules, CA, 94547, USA
* Correspondence: Minkang Zhang, Clinical Diagnostics Group (CDG) Software Team, Bio-Rad Laboratories, Hercules, CA, 94547, USA

Abstract: As more and more technological advances affect medical equipment, the importance of software for medical equipment is becoming more and more significant, and it is crucial to effectively manage and control the life cycle of software. This paper proposes a software improvement scheme with DevOps as the core, including adopting agile development mode, refining test management and optimizing operation and maintenance process, so as to improve the efficiency of software development, test and operation and maintenance, shorten the delivery cycle and reduce compliance risks. The core philosophy of DevOps is continuous integration, continuous delivery and cross-team collaboration, providing a new approach to the development and management of medical device software, the implementation of this strategy to achieve the goal of improving software quality, improving system reliability, and meeting stringent industry standards.

Keywords: DevOps; medical device software; life cycle management

1. Introduction

With the development of medical device technology driven by the development of technology, its built-in software has become more important. However, software built into medical devices requires stringent compliance and complex operations. Traditional life cycle management methods are no longer able to meet the ever-changing needs, for which better management methods need to be sought. DevOps is a way to achieve development, test, and operational efficiency through continuous integration thinking, continuous delivery, and automated testing tools, which can reduce product release cycles while ensuring software quality and compliance. This paper introduces how to improve the life cycle management of medical equipment software by DevOps optimization method, and improve the work efficiency and quality.

Received: 07 April 2025
Revised: 12 April 2025
Accepted: 25 April 2025
Published: 28 April 2025

2. Basic Concepts of DevOps

2.1. Core Concepts of DevOps

DevOps is the culture and practice of integrating development and operations teams. First of all, automation is the core of DevOps, including automated build, testing, and deployment. Ensure efficient and consistent code structure with automated build tools to prevent errors or delays caused by human manipulation. Use automated testing tools such as Selenium and JUnit to ensure good quality at all stages of the software. Also leverage automated deployment to deliver timely and stable releases online to meet customer needs. Second, continuous integration, continuous delivery (CI/CD) is critical to DevOps. Frequent, small-batch release patterns combined with rapid feedback will help the team respond to market changes while avoiding large changes over a long period of time so that the product always meets consumer needs [1]. When a problem is found, the version rollback mechanism can help the team immediately fall back to a stable version to

protect the smooth running of the system. Finally, DevOps emphasizes teamwork and culture. Close contact between teams in various fields, information sharing, and continuous improvement habits are the keys to a successful DevOps implementation. Development, operation and testing teams are involved in each step to ensure a smooth transformation of software from development to operation. Through continuous review, improvement, continuous optimization of development, operation and maintenance process, improve efficiency, and finally achieve continuous delivery of efficient and more stable software services [2]. Combining these practices with ideas, DevOps is being introduced into current software development practices to help organizations improve software quality, responsiveness, and competitiveness (see Figure 1).

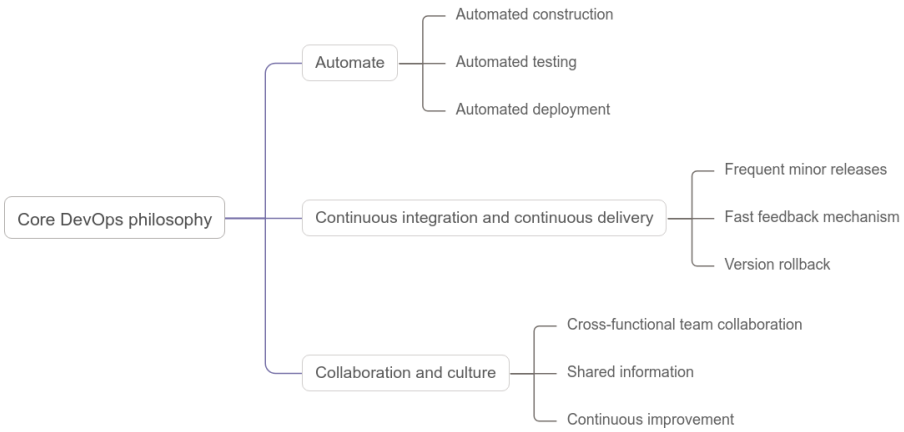

Figure 1. Basic Framework of Core DevOps Concepts.

2.2. DevOps Toolchain and Technology Stack

To achieve a successful DevOps practice, you need to rely on a strong and complete tool support system, including CI/CD tools, automated testing tools, configuration management systems, and monitoring and logging systems. Among them, common CI/CD tools such as Jenkins and GitLabCI can ensure rapid release automation under the premise of frequent code consolidation. Common automated test tools such as Selenium, JUnit, TestNG, etc., can automatically perform detection tasks after each code update to ensure software quality. Common configuration management systems, such as Ansible, Puppet, and Chef, help enterprises automatically manage environment Settings and software deployment to ensure the unity of various environments. Common monitoring and log management tools, such as Prometheus, Grafana, and ELK Stack. The health of applications and their infrastructure can be monitored in real time, so that possible problems can be identified early and measures can be taken to resolve them [3].

3. Life Cycle Management Status of Medical Device Software

3.1. Slow Response to Development Cycle

The development cycle of medical device software is generally long, mainly because the requirements are complex and strictly controlled by laws and industry norms. The development of these software often needs to go through multiple review and inspection processes, including functional, performance, safety and other requirements, so in addition to technical constraints, but also need to comply with regulatory standards such as FDA, ISO13485, so the overall process may take a long time. Especially when there are changes in demand and changes in laws and regulations, the general old work process is not easy to adjust the response in time, resulting in a lag in response [4]. In addition, development, verification and maintenance teams are often different units, resulting in insufficient cooperation between each other, low information transmission and feedback efficiency, so the research and development progress is slow, but if the lack of feedback

mechanism and efficient cooperation mode, the process of product development from demand to delivery is too long, it will affect the current market's rapidly changing needs. It has a negative impact on the competitiveness of products and the speed of responding to the market.

3.2. Compliance Verification Is Complex

In order to ensure the safety, quality and performance of medical equipment software, it must be implemented in accordance with strict industry norms, which may cover a large number of processes, such as functional verification, performance testing, documentation, and each step needs to be manually operated and a large number of manual records, which not only increases the difficulty of inspection work. It also greatly increases the likelihood of mistakes. In traditional methods, due to the complexity of documentation and inspection, compliance inspection is often inefficient and time-consuming. Especially for continuously updated products, repeated manual reviews will make the release time lag, unable to respond to market changes and regulatory requirements in a timely manner and delay the time to market. At the same time, complicated compliance verification can increase the burden on the development team and make it prone to omissions or errors, which can affect the software's compliance and go-to-market process [5].

3.3. Insufficient Test Efficiency

For medical device software testing, it is necessary not only to achieve functional verification in breadth, but also to carry out rigorous testing across multiple hardware platforms, operating systems and network configurations. Traditional single-point testing cannot complete the above certification work in a short time and in a full range, resulting in low test efficiency and poor test results. Often, manual audit can only check some functions or scenarios, and cannot find all possible vulnerabilities and problems, especially when there is a large number of parallel operations between servers, long-term continuous operation of the case, the overall back testing, performance analysis is difficult. In addition, in the absence of unified control and tracking, the degree of standardization of test cases is low and the reuse rate is not high, resulting in accurate and comprehensive test results. In the test work, the lack of support of automated testing means makes it difficult for the test team to complete all the quality assurance work as agreed in advance.

3.4. Lack of Automation in Operation and Maintenance

For the management and maintenance of medical equipment software, it is generally carried out by manual means, and there is insufficient automation support. For software installation, update, configuration and debugging problems encountered in the process, operation and maintenance personnel need to participate in a lot of work, not only increase their work burden, but also there will be human subjective errors. Due to the lack of automated operation and maintenance process, once the system fails, it is often very slow to judge and solve the problem, resulting in the suspension of the entire system for a long time, which has a continuous impact on the functional operation of the equipment and the medical services provided. In addition, the traditional operation and maintenance scheme lacks real-time monitoring and alarm, cannot find the latent system failure or performance bottleneck in time, will also prolong the problem discovery time, and have a negative impact on the reliability and stability of the system.

4. Optimization Strategy of Medical Device Software Life Cycle Management Based on DevOps

4.1. Adopting Agile Development and Short Cycle Iteration

In the medical device software development process, adopting an agile development approach can help speed up the development process by putting the entire project into a series of iterations (generally two weeks to a month) of small but easily deliverable tasks,

which is more conducive to adapting to customer needs and market feedback. At the end of each iteration, the development team will provide a usable version of the feature for the customer to review, not only for prompt feedback, but also for early detection of problems. For medical device software, using agile methods not only makes product changes more flexible, but also ensures that software quality is constantly improved throughout the process. Because each iteration cycle is short, the team is better able to get feedback from customers and the market and make corrections and improvements in each iteration, preventing requirements errors caused by prolonged development. By adopting an agile approach to development, team members are able to work more intensively on the most important feature points to increase productivity, and are able to ensure a successful release within a clear deadline. In addition, Agile also has the characteristics of emphasizing communication and cooperation between internal departments, such as development, testing, operation and maintenance of all the relevant personnel involved, so as to promote effective communication and timely feedback, to ensure that the project can be completed on schedule.

4.2. Integrated Automated Compliance Check Tool

For the sake of safety and reliability, medical device software needs to comply with a variety of stringent regulations, such as FDA21CFRPart11, ISO13485, etc. This makes compliance checking an important part of the software life cycle of the device. However, traditional manual compliance checking causes time-consuming and high error rate. The application of DevOps automated compliance checking tools can significantly improve the efficiency and accuracy of compliance management (Table 1).

Table 1. Integrated Automated Compliance Checking Tools.

Procedure	Description
Integration of compliance checking tools	Incorporate compliance checking tools into the development and deployment process to ensure that every commit and release is automatically reviewed
Automated code scanning	Static analysis of code through automated tools to identify compliance issues and ensure code complies with relevant laws and regulations
Generate compliance reports	Compliance reports are automatically generated, and compliance issues with each release can be tracked in real time, allowing teams to fix and comply with the law in a timely manner
Integrate CI/CD processes	Integrate compliance checking tools into the CI/CD process to minimize human error by subjecting every commit and deployment to an automated compliance review

Firstly, static scanning is used for static compliance check during the whole process of code from development to production, including unit testing and integration testing, to achieve automatic code compliance detection. The ability to ensure that all changes, including new code, new versions, and changes in each release, are automatically checked for compliance, and to apply static analysis methods to find possible compliance violations to help developers write compliant code. Significantly improve the efficiency and quality of compliance checks by reducing the cost of code scanning for developers and errors for inspectors through automated code scanning technology, while also providing compliance reports to developers so that the development team can know the latest compliance status and quickly fix compliance issues by monitoring each release. Version compliance is also guaranteed at every release throughout the release. This mechanism enhances the development team's emergency response speed, and makes the development and release process more transparent, so that compliance issues can be quickly detected and resolved. Finally, automated compliance checks are integrated into continuous inte-

gration (CI) and continuous deployment (CD) processes, ensuring that an automated compliance check is performed before every change and release throughout the process. The compliance detection tool is integrated into the CI/CD pipeline, so that every released software goes through strict compliance inspection to reduce the human factor and ensure the compliance of the software version, and the pipeline between each step can be connected in series to improve the work efficiency and supervision efficiency of the entire software development process. It can be seen from the table that automation integrates compliance testing tools, which cannot only enhance the regulatory efficiency of software compliance, but also enable software products to ensure compliance with industry regulations and legal requirements, and promote the safety and marketing of medical device software.

4.3. Refined Test Case Management

The testing of medical device software is not only limited to realizing all the functions required, but also needs to ensure the stability and security of the software in various environments. Refined test case management is considered to be an important step to ensure the quality of medical device software, especially in complex controlled environments. Test case classification is the starting point of test case management. It divides test cases as finely as possible according to functions, risk levels, platforms and other factors, so as to ensure adequate testing of key functions and key areas. Through reasonable classification of test cases, the team can understand which functions have a greater impact on system stability and system security, and more tests can be conducted under the condition of limited resources. Priority design is conducive to improving test efficiency (see Table 2).

Table 2. Refined Test Case Management.

Procedure	Description
Use case classification	Test cases are carefully divided according to function, risk, platform and other factors to ensure adequate coverage of important functions and high-risk areas
Priority ordering	According to the level and importance of risk in the test cases, priority is arranged to ensure that high-risk areas are tested first and test efficiency is improved
Automated execution	Use automated execution for regression tests and test cases with high repeatability to reduce manual involvement and increase execution rate
Test coverage monitoring	Monitor the execution of test cases with coverage tools to ensure that tests cover all critical functionality and areas of potential defect

Test cases are prioritized according to their risk level and importance. High-risk areas should be tested first. This ensures that testing resources are focused on the most important resources and that testing time or manpower is not wasted on smaller use cases. Prioritization helps the test team to carry out the test work in an orderly manner, shorten the test cycle, and improve the overall test process efficiency. Automated execution is an effective means to improve the effectiveness of repeated tests. For backtesting and frequently executed test cases, automated execution technology reduces manual intervention in test execution and improves execution efficiency and accuracy. Automated test tools run at the moment of every code update to find errors and correct them in time, ensuring high software quality and lasting stability. Test measurement is the most important measure for software coverage testing. Use of coverage measurement tools to track test case usage. Make the team confident that all important features and potential defects have been covered in testing. Applying coverage measurements can help identify undetected parts, assess whether current tests are adequate and complete, and determine if missing tests need to be supplemented.

4.4. Optimizing the O&M Process and Role Assignment

The O&M management and maintenance of medical equipment software is an important part to ensure the long-term stable operation of the system. By optimizing the O&M process and effectively assigning O&M roles, the efficiency can be greatly improved and the time for fault repair can be shortened. The traditional approach to operations tends to be that operations personnel perform a lot of manual configuration and maintenance work, with no process or standard working methods. You can use a DevOps approach to improve operations with standardized processes, automated tools, and real-time monitoring. For example, use automated deployment tools (such as Ansible, Puppet, and Chef) to quickly install and upgrade medical device software, avoiding unnecessary manual setup errors and delays. In addition, IaC technology helps operations personnel adopt code management infrastructure, simplifies configuration management, ensures environment consistency, and improves operational flexibility. In addition, appropriate division of responsibilities is also an important measure to improve operation and maintenance efficiency. DevOps emphasizes close cooperation between development, operations, and test teams, so that operations personnel are no longer just a system operation worker, but actively participate in the software development cycle, identifying problems at an early stage and solving problems as soon as possible. Standardized operation process and role arrangement cannot only reduce the probability of system failure, but also shorten the repair time after system problems, which plays an important role in ensuring the efficiency of medical equipment software and stable system.

5. Conclusion

With the increasing complexity of medical device software, traditional methods can no longer meet the stringent requirements of the current medical industry. Therefore, an improvement scheme of medical device software life cycle management based on DevOps is proposed. Through the use of agile development, automated compliance verification, refined testing and optimized operation and maintenance processes, the software quality of medical devices can be improved, the software development cycle can be shortened, and the time to market of products can be reduced. At the same time, the work collaboration mechanism based on DevOps can further enhance the close cooperation between development, testing and optimized operation and maintenance processes. Reduce error rates, risks and trial and error costs, improve the progress of medical device software development and improve the reliability of medical devices, and strengthen and protect the health of patients.

References

1. S. Das, N. Deb, N. Chaki, et al., "Minimising conflicts among run-time non-functional requirements within DevOps," *Syst. Eng.*, 2024, no. 1, pp. 27, doi: 10.1002/sys.21715.
2. L. E. Lwakatare, et al., "DevOps in practice: A multiple case study of five companies," *Inf. Softw. Technol.*, vol. 114, pp. 217–230, 2019, doi: 10.1016/j.infsof.2019.06.010.
3. K. Kikutani, T. Shimatani, A. Kawaguchi, T. Ikeyama, D. Yamaguchi, O. Nishida, and S. Ohshimo, "Medical equipment that improve safety and outcomes of inter-facility transportation of critically ill patients: A systematic review," *Medicine*, vol. 102, no. 22, p. e33865, Jun. 02, 2023, doi: 10.1097/MD.0000000000033865.
4. A. Hotz, E. Sprecher, L. Bastianelli, et al., "Categorization of a universal coding system to distinguish use of durable medical equipment and supplies in pediatric patients," *JAMA Netw. Open*, vol. 6, no. 10, p. e2339449, 2023, doi: 10.1001/jamanetworkopen.2023.39449.
5. A. Bhan, C. V. Green, L. Liang, L. Philpotts, et al., "Educational interventions to improve compliance with disinfection practices of noncritical portable medical equipment: A systematic review," *Infect. Control Hosp. Epidemiol.*, vol. 45, no. 3, pp. 360–366, 2024, doi: 10.1017/ice.2023.234.