10.18686/frim.v2i6.4670

Software Maintenance Cost Control Method in Software Engineering

Hongyu Zhao

Xihua University, Chengdu, Sichuan 610039

Abstract: This paper aims to explore the control method of software maintenance cost in the field of software engineering, combine theoretical knowledge and practical experience, analyze the factors affecting software maintenance cost in detail, and put forward the corresponding control strategy. Through literature review, case analysis and other methods, we reveal the importance of software maintenance cost control, and propose a series of concrete and feasible control measures, in order to provide reference for practitioners and students in the field of software engineering.

Keywords: Software engineering; Software maintenance cost; Control method

Introduction

With the rapid development of information technology, the software system is increasingly widely used in various fields. However, the maintenance and update cost of software often becomes an important challenge faced by enterprises and organizations. How to effectively control the maintenance cost of software and improve the sustainability and competitiveness of the software system has become an urgent problem to be solved in the field of software engineering.

1. Overview of the software maintenance costs

1.1 Definition and classification of software maintenance

Software maintenance is the modification work performed to correct errors, improve performance, or meet new requirements. According to the purpose and nature of maintenance, software maintenance can be divided into four types: adaptive maintenance, corrective maintenance, perfect maintenance and preventive maintenance. First, adaptive maintenance: modifications made to adapt to changes in the external environment (such as operating system upgrades, hardware environment changes, etc.). Second, corrective maintenance: modifications made to correct errors and defects in the software. Third, perfect maintenance: modifications to enhance software functions and improve software performance. Fourth, preventive maintenance: Modifications to prevent possible future problems, usually include code refactoring, performance optimization, etc[1].

1.2 Factors affecting the software maintenance cost

Software maintenance cost is affected by many factors. First, software quality, the higher the software quality, the lower the failure rate, and the maintenance cost is correspondingly reduced. Second, the project period, the shortened period will increase the direct cost, and the delay of the period may lead to the increase of the claim cost. Third, human resources. High-skilled and high-quality project team members can improve work efficiency and product quality, but the corresponding human resource cost is also high. Fourth, the price factor, the price fluctuation of market human resources, hardware and software directly affect the project cost. Fifth, technology selection, software design, programming language, development tools and other technical selection has a significant impact on the maintenance cost.

2. Software maintenance cost control method

2.1 Improve the software quality

As the core strategy of reducing the maintenance cost, the importance of improving the software quality is self-evident. Establish perfect quality management system is the first task, this not only includes the traditional unit testing, integration testing and system testing stage, also need to integrate into continuous integration (CI) and continuous deployment (CD) practice, ensure that every code submitted can pass automatic testing, timely detection and repair, to keep the steady improvement of software quality. In the coding phase, the promotion and adoption of industry-recognized coding specifications and best practices can significantly improve the readability and maintainability of the code. Regular code review can not only help team members learn from each other, but also detect and correct potential defects and bad programming habits in advance. In addition, in-depth analysis and optimization of performance bottlenecks, such as algorithm improvement, database query optimization, cache strategy adjustment and other means, to reduce resource consumption, improve software operation efficiency, is also an indispensable part of improving software quality. The improvement of the defect management mechanism is also important. Establish an efficient defect tracking system to ensure that every single defect found can be accurately recorded, timely allocated, tracked on the repair progress, and finally verified and closed. In this process, it is crucial to improve the overall stability and reliability of the software to pay attention to the defect analysis, find the causes of the problem and prevent the recurrence of similar defects.

2.2 Precise demand management

In the early stages of software development, communication is the first step to obtain precise requirements. The development team should actively listen to the customer's voice, understand their business scenarios, goals and potential pain points, and ensure that the demand definition meets the customer's expectations and technical feasibility. In order to systematically manage the needs, it is particularly necessary to establish a complete set of demand management mechanism, including a detailed classification, analysis and review process of the requirements. Through careful demand analysis, potential demand conflicts, omissions or ambiguity can be identified and corrected at an early stage, avoiding repeated modifications and additional costs in the subsequent development process. At the same time, regular demand review meetings are also an important means to ensure the accuracy of the requirements, which enables the project related parties to jointly review the demand documents, raise questions and suggestions, and promote the conclusion of consensus. In the software development process, changes in requirements are almost inevitable, but excessive changes will seriously affect the project schedule and cost. Therefore, it is necessary to strictly control the frequency and amplitude of the changes through the demand management mechanism to ensure the rationality and necessity of the change. For the changes that must be made, the impact on the project shall be assessed, with a detailed change plan formulated and the relevant parties notified to ensure the smooth implementation of the changes[2].

2.3 Scientific project management

At the beginning of the project, detailed and reasonable project planning should be formulated, requiring project managers to not only clarify the overall goals and vision of the project, but also set clear and quantifiable milestones and delivery nodes to effectively monitor the project progress. At the same time, according to the project needs and workload, the project tasks should be reasonably divided to ensure that each task has a clear responsible person, schedule and required resources, so as to realize the optimal allocation of resources. In the process of project execution, the tracking and analysis of progress and risk is an indispensable link. Through regular project review meetings, progress reports and risk assessment, project managers can keep abreast of the latest progress and potential problems of the project, and take corresponding measures to adjust and optimize. This helps to ensure the stability of the project schedule and avoid additional costs due to delays or overruns. In addition, the project team should maintain close contact with the customer and regularly communicate the project progress, demand changes and potential issues to ensure a common understanding of the project objectives and expectations. Through effective communication, potential differences and conflicts can be identified and resolved in a timely manner, so as to avoid additional costs and time waste due to demand changes or misunderstanding.

2.4 Modular design and programming

The application of modular design and programming strategies in the field of software development greatly promotes the maintainability, scalability and reusability of the software, thus effectively reducing the development and maintenance costs. Through modular design, complex software system is disassembled into a number of clear, relatively independent modules, each module internal high cohesion, low coupling between module, this design way not only improve the readability and comprehensibility of the software, also makes the subsequent software upgrade or function extension, can more flexible and efficient positioning and modify the related module, without a comprehensive reconstruction of the whole system. At the same time, choosing the appropriate programming language is also the key to realize the modular programming. Different programming languages have different characteristics and advantages. According to the specific requirements of the software system, choose a programming language that can meet the functional requirements and facilitate the development and maintenance, which can significantly improve the development efficiency and reduce the maintenance cost in the later stages. In addition, reusing code modules is another significant advantage of modular programming. In the software development process, by identifying and reuse existing, validated code modules, a lot of repeated programming work can be avoided, reducing potential errors and defects, and further improving the development efficiency and quality.

2.5 Establish a cost control system

At the early stage of the project, through detailed cost estimation and budgeting, the resources required by the project and their corresponding costs can be clearly defined, so as to provide a solid foundation for the subsequent cost control. This process needs not only to take into account the direct costs such as human and material resources, but also to include the potential risk costs and market change factors to ensure the comprehensiveness and accuracy of the budget. With the advance of the project, with the help of the advanced cost control system, the project manager can grasp the expenditure situation of various expenses in real time, find the signs of cost overspending in time, and quickly take adjustment measures, such as optimizing resource allocation and adjusting the project schedule, to ensure that the project cost is always kept within a controllable range.

3. Case Analysis

Take a maritime safety administration supervision command system project as an example, which took a number of measures to reduce the cost of software maintenance. The total investment of the project is 7.8 million yuan. After strict demand management, project management and quality management, the project was successfully completed and passed the acceptance inspection within a limited time. Specific measures include: First, in the cost estimation, refer to the company's knowledge base and the experience of similar projects to improve the accuracy of the estimate. Secondly, experts will be invited to review the functional points and the required amount of work to ensure the rationality of resource demand. Then, the company's internal digital nervous system is used to track and control the cost in real time to ensure that the project cost is within the budget. Finally, professional training should be provided to speed up the development speed, shorten the development cycle and reduce the development cost. Through the implementation of the above measures, the project has achieved remarkable results in cost control. On the one hand, the project cost is effectively controlled without overspending; on the other hand, the software quality is significantly improved, with the failure rate significantly reduced and the maintenance cost significantly reduced.

4. Conclusion

Software maintenance cost control is an important topic in the field of software engineering, by improving the quality of software, accurate demand management, scientific project management, modular design and programming and establish cost control system, can effectively reduce the cost of software maintenance, improve the sustainability and competitiveness of the software system. As students majoring in software engineering, they should fully master these methods and skills, and constantly accumulate experience and improve their ability in practice.

References

[1] Liu Weixue. The index system of software maintenance cost influencing factors is established [J]. Electronic test, 2014, (11): 34-35.

[2] Shen Xu, Liu Weixue. Gray association analysis of factors influencing software maintenance costs [J]. Journal of Bohai University (Natural Science Edition), 2014, 35 (01): 61-65.

[3] Wang Fan. Cost estimation and quality assurance techniques in software maintenance study [D]. Tutor: Chen Deren; Yang Xiaohu. Zhejiang University, 2011.