2009 International Conference on Machine Learning and Computing
IPCSIT vol.3 (2011) © (2011) IACSIT Press, Singapore

Software Risk Management and Avoidance Strategy

Hassan I. Mathkour [+], Basit Shahzad, Sami Al-Wakeel

Department of Computer Science, King Saud University, Riyadh, Saudi Arabia

Abstract. Software development is the art of developing the software in an appropriate manner by using the software development life cycle. The development is a dynamic activity and requires a lot of rational thinking during this process. As the development of software is becoming more systematic and tool-driven, the risks are increasing and the attention to risk management is not increasing with the same pace. Therefore the academic and industrial community is worried to consider that how the risks can be handled to minimize the losses and to increase the profits and reputation in the market this paper focuses on the aspect of suggesting the techniques to handle or manage the software risks. Taking into consideration the eighteen most prominent risk factors that effect the software quality and software process, the handling or avoidance strategy has been proposed. This paper suggests addressing the risk factors to be treated not only by the technology but by using intuition as well.

Keywords: Software risk management, software risk handling, risk mitigation

1. Introduction

Software risk management has been a very hot area of research since last three decades. Recently, the research community looks seriously interested to identify not only the risk factors but also the causes of the appearance of the risk factors in software development life cycle and how these risks can either be handled or avoided. A recent survey of 600 firms indicated that almost 35% of them had at least one ‘runaway’ software project [1]. In another study, conducted on almost 13,000 projects, it was investigated that almost 25% of the projects were either delayed or faced a failure. It has been observed that most problems in the software industry are faced just because of the poor software risk handling mechanisms or due to the absence of any such mechanism at all. In this regard it is important to note that currently strong emphasis is being given on this domain to identify more and more risk factors. Pressman [3] has made an effort to identify the software risks, and has provided the ten broader risk factors. Bohme, in his work has also provided a list of top ten risk categories [4]. Shahzad at el. [5] identify a list of software risk factors and also identify the relative impact of each risk factor. In [6], the risk factors have been prioritized according to their frequency of occurrences and the impact that they possess and thus a list of eighteen risk factors with respect to their total impact was prepared. The risk factors that are identified in the list [5, 6] are expected to cover a border range of the risks that may come into the software development process.

It is strongly believed that the risk identification, particularly, is an ongoing process, and apparently the risk factors keep on increasing with emerging technologies, people, environment, management and the circumstances. Thus, a claim of exhaustive identification of all risk factors in the entire software process may not be realistic. The ordered list of available risk factors, by calculating the overall impact and frequency of each available risk factor is also available in [6].

2. Handling and Avoidance Mechanism

[+] Corresponding author.
E-mail address: binmathkour@yahoo.com.

Following the order in [6], the risk factor that has the highest priority can be observed. Thus, such a risk factor should be addressed first. As the risks are either to be handled or avoided, it is necessary that a strategy is proposed for each risk factor. Subsections 2.1to 2.14 discuss the handling and avoidance strategies against each risk factors.

2.1. Requirements are not properly stated

Multiple requirement acquisition approaches must be used; this includes the questionnaires, interviews and direct communication. The team deployed on the requirement acquisition should be capable enough to extract the accurate/valuable information from the information lot coming from different sources.

The customer must allow the development team to have a flexible schedule if the requirements are expected to change dynamically. Only minor changes, which do not have the impact on the architecture of the software, can be changed dynamically.

The development team must be familiar with the Enhanced Information Deployment [7] technique to attend the default requirements that are not explicitly mentioned by the customer.

2.2. Low estimation of time and cost

It is appropriate that the management acquires multiple estimates from different sources, and suggest a flexible schedule in terms of time and cost.

It has been observed that if the funding and time are not flexible, the incremental model [4] of development may be a solution. As it grows in increments, if the funding or time collapses, at least there is something presentable to the customer, rather than having nothing at all.

The development team must try to find the maximum amount of reusable code, the availability of reusable code will have three dimensional positive effect. It will decrease the development time, cost, and also the resources required to test the component.

The team of experienced developers and management may decide, in consultation with the customer, that if there are any scrubable requirements that may not harm the overall working of the software. Such requirements may be eliminated to save time and cost.

Clean room engineering may not be implemented in the projects that have tight time and cost schedule.

2.3. More stress of users than expected

The developer must always expect and consider that the customer is not capable of describing all the requirements. The developer, if possible, must design and implement the system in a way that it can tolerate with the extra burden as well.

The developers must also do the extensive stress testing to ensure that the software is capable of handling the load and stress of the users. The development teams can stress test the software at component level, environment level, architecture level and end-to-end level.

2.4. Less reuse than expected

While estimating for the projects cost and resource requirement, the developers must know that what amount of software is available for re-use, this should be an rational decision as, if the reusable code is not available the effort to develop such code will be duplicated. As not only code is to be developed, but also the component is to be tested before integration with other components.

If the component is to be developed, it is necessary that a clean room engineering approach is applied is the development so that the time required for testing the component is minimized if not completely eliminated. The best developer, among the available lot, should be deployed to develop the components so that the expected time on development and testing is minimized.

2.5. Delivery deadline tightened or manager change circumstances

The managers somehow try changing the circumstances because of the deadline pressure or because of the orientation of new requirements. The absolute definition of requirements at the beginning ensures that circumstances remain constant and deadlines are not tightened.

The development team and management of the development firm must have the foreseeing capability, and should try adhering to the dynamic circumstances without disturbing the firm itself.

The FAST approach may be used to speed up the requirement acquisition, thus decreasing the negative impact of tightened deadlines.

2.6. Funding will be lost

In order to ensure that funding issues remain in order, the development team must first ensure that the software is developed within time, developing within time will not only help to improve the revenues and profits but would also ensure that the funding remains available throughout the software development lifecycle.

Its important that friendly relationship is maintained with the funding agency.Along with the cordial relationship with the funding agency, it is also important that the funding agency is kept updated regarding the progress of the software development process, and also any problem that is faced during the process. Being informed about the problems and achievements, the funding agency will be in a better place to help the development firm with the continuation of the funding.

2.7. Technology does not meet expectations

The decision about the choice of technology should be taken only after a very through consideration of the available tools and technologies and only by the experienced practitioners. The customer in some cases may allow the change in technology, but this change must not have any negative effect on the quality of the software.

If the change of tool, is agreed between the customer and the development team the development team must try to choose the best available tool in consultation with the customer.

2.8. Lack of training on tool or staff inexperience

The development firm can keep its employees updated by offering them training on the emerging tools.

The firm may hire the new graduates from the leading universities, having some knowledge of the current tools. The firm can train them and provide them small assignments to do, inorder to complete their training and making them a useful member of the firm.

It is important that the teams are made for each project. Developing the team structure will help in not only promoting the efficiency of the work but will also help in providing experience to new members. This will also help the new members to learn about the smooth flow and effective handling of the tedious work.

2.9. Staff turnover

Staff, and particularly the experienced staff is an asset to any firm, and firms generally do their best to retain such individuals. But this is very obvious that learned individuals still want to change the jobs, although this trend may not be eliminated yet it can be reduced. The employer should keep the honest estimations of the salaries available in the market for experienced people. Proven experience show that employee enjoy working with an employer who have more care for the families of the employee. The employer may offer the services like, free family medical, children school fee, car allowance, house rent, etc in order to keep the employee attracted.

The employer should provide other social gathering and meeting opportunities to the employees, inorder to help establish a family culture at the organization. This get-together is a good chance for the juniors to meet with the firms top management and listen to their views and vision about the future of the firm’s business strategy[10].

The employer must try to keep the employees updated and should provide the employees with chances to refresh their knowledge about the emerging tools and technologies. This can be done by arranging the courses at their own site, or by sending the employees to the specialized institutes for training.

The employer may introduce a loan scheme to help the needy individuals and the return may be in easy installments, without or at a minimal interest rate.

It is necessary that the employer try maintaining the respect and honor of the employees, and it is never compromised in any situation. The employer may introduce a bonus scheme to make the employees a part of the profit that the firm gains. This would give a sense of ownership to the employee and the employee will try to deliver according to the best of his capabilities.

2.10. Backup not taken & Actual document/data loss

Backup must be taken at multiple sites, so that in case of any physical or technical damage the backup itself remains intact.The management must try to introduce the paperless environment in the firm, this would help in maintaining the efficient, secure and traceable working environment.

The backup sites may be frequently updated and the updates should be inspected regularly.

The team strictures should be implemented in the development environment, this not only improves the working environment but also helps in decreasing the dependency on the individuals as the team members remain active and keep knowledge of the trends and patterns that someone uses in its development. This will not only help in introducing the harmony in the team members but would also increase the efficiency in the working environment.

2.11. Fire, flood and building loss

The firm must ensure that the working environment across the organization is not only conducive but also safe for the employees. Proper smoke detectors and fire alarms must be installed in the building to detect the fire and also the emergency exit should be provided in case of any emergency.

The organization must also ensure that the building codes have been followed and the structure is according to the prescribed standards. With the orientation of more earthquakes recently in the world, it is also important that the building structure is developed in a way that it can absolve the earthquake shocks of an adequate level.

2.12. Too many development error

Although testing techniques can help in identifying errors yet it is more appropriate to try enforcing the clean room engineering approach

For this purpose not only the development team must try working accurately but also the continuous inspections of the work being done by the developer must be reviewed by some senior colleague, so that the guideline may be provided early and correction are made without serious harm [9].

Along with the availability of the inspections, the developer must unit test the piece of software that he is developing and must ensure that the code is free of errors and also that it is meeting the requirements.

Sometimes there are so many errors identified in a piece of code that correction may not only cause the wastage of time but also the resources. In such circumstance, the re-development of that component may be easier than correcting the existing one.

It is also important that the testing process works fine, i.e. identification of too many errors can still be less harmful as compared to the ignoring errors, because the identified errors can somehow be tackled and addressed for correction, but an un-identified error may cause harm after all the bugs have been fixed.

2.13. Developer run away with code

At the time of appointment, the Human Resource (HR) department must ensure that the person they are hiring is adequately trustable and owes a good employment history. His credibility can be checked from the previous employer. The contact details provided by the employer must be verified before the employee is hired permanently.

Backup must be taken at multiple sites, so that in case of any physical or technical damage the backup itself remains intact. The backup sites may be frequently updated and the updates should be inspected.

2.14. Lack of intuition

It has been observed that the experienced individuals can help in estimating the cost, budget and manpower of any project by just using their intuition [11]. The guess provided by them is generally accurate, and thus causes a huge benefit for the organization. The organization must do adequate effort to retain such people and should continue befitting from their experience.

Talented individual must be attached to work with the experienced individuals so that they can learn that how the estimations can be made by using the previous knowledge and intuition.

3. Conclusion

Software development process is complex and requires efficient handling of the available resources. Poor planning invites risk factors, that are very difficult to deal with. The paper unleashes the possible strategies to avoid or overcome risk, once they have been identified in a software process. Although a complete list of software risk factors is impossible to produce, as the risk factors keep on growing with the new tools and technologies, yet a comprehensive list has been considered for providing knowledge about the handling and avoidance mechanism. The software houses that are developing the small and medium software can especially benefit by following the avoidance strategy.

4. Acknowledgements

This work is partially supported by the Research Center (RC) of the College of Computer & Information Science of King Saud University, Riyadh, Saudi Arabia.

5. References

[1] J. Rothfeder, “It’s Late, Costly, and incomplete-But Try Firing a Computer System”, *Business Week*, November 7, Coper Jones, “patterns of software success and failure”, 1996

[2] Roger S. Pressman, “Software engineering: a practitioner’s approach”, 5th ed, McGraw-hill, pp 151-159

[3] Barry W. Boehm, “software risk management: principles and practices”, pp 13

[4] Basit Shahzad, Tanvir Afzal, “Enhanced risk analysis and relative impact factorization”, *1st ICICT, IBA Karachi*, August 27-28, 2005 ,pp 290-295.

[5] Basit Shahzad, Javed Iqbal, “Software Risk Management – Prioritization of frequently occurring Risk in Software Development Phases. Using Relative Impact Risk Model”, *2nd International Conference on Information and Communication Technology (ICICT2007)*, December 16-17, 2007, IBA Karchi.

[6] Roger S. Pressman, “Software engineering: a practitioner’s approach”, 5th ed, *McGraw-hill*, pp 151-159

[7] Borland, the open alm company, A Load Testing Strategy, *white paper*, April 2006,pp6

[8] Jiantao Pan, Software Testing, Carnegie Mellon University, *Dependable Embedded Systems*, spring 1999, pp 1-14.

[9] Duport, “how to control and manage the staff turnover. http://www.duport.co.uk/guides/staff%20issues/Controlling%20and%20managing%20staff%20turnover.htm, May 2006.

[10] Magic intuition, “definition of intuition. http://www.magicintuition.com/intuition.html”, 2009