A Case Study of using WikiWinWin into Bug Negotiation

Peng Wan[1, 2], Juan Li[1] and Yin Li[1]

[1] Institute of Software, Chinese Academy of Sciences, 100190 Beijing, China
[2] Graduate University of Chinese Academy of Sciences, 100190 Beijing, China

Abstract. In order to make well-considered solutions to software bugs, it is necessary for stakeholders to negotiate collaboratively. In traditional ways of making bug decisions, it not only lacks efficient tool to support the negotiation process which might lead to inadequate negotiation, but also cannot preserve the negotiation records as the experience for future development. In this paper, the WikiWinWin was applied into industrial bug negotiation as a case study to get better solutions. The comparison would be conducted between solutions after being negotiated and the original solutions to prove the improvement brought by WikiWinWin.

1 Introduction

Bug negotiation is a way to get bug solutions collaboratively for project stakeholders. The solutions they propose should solve bugs well and not at the cost of sacrificing stakeholders' expectation to the largest extent. The traditional way to negotiate bug solutions in software development is like meetings, emails and instant communication tools [9]. For a software project with many distributed stakeholders, it costs a lot to attend meetings together for all stakeholders. The quality of video conference might be affected by the network smoothness. Moreover, the scattered records in emails and other communication tools are not easy to be preserved and managed. After the investigation, we found that there were few tools to support the bug negotiation.

WikiWinWin which is a requirement negotiation tool based on the Theory W which is apt to reach a mutual agreement [4]. It not only designs a sequence of steps and instructions to guide the negotiation, but also owns the features of wiki which is suitable for the collaborative environment [6] and is able to preserve the revision history. In the previous work, WikiWinWin is usually used in requirement negotiation. Due to its collaborative features, we consider it can still improve the process in making bug decisions. Moreover, with the features of wiki, it can preserve the negotiation records which can be concluded as experience for future development.

NFS (www.nfschina.com) which is a software company uses Mantis as the bug tracker to record and process bugs from users and testers. Usually, product managers (PM) administrate the Mantis. Due to the stakeholders located in different places, it is impractical to hold meetings for every bug with stakeholders. Without involving all stakeholders into negotiation, sometimes, it is prone to get misunderstanding and even get wrong solutions in solving bugs. In this paper, we selected some of the wrong solutions and arranged three students to conduct a case study. In the case study,

Wan P., Li J. and Li Y..
A Case Study of using WikiWinWin into Bug Negotiation.
DOI: 10.5220/0003579600610067
In *Proceeding of the 1st International Workshop on Evidential Assessment of Software Technologies* (EAST-2011), pages 61-67
ISBN: 978-989-8425-58-4
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

students negotiated the initial bugs of the wrong solutions with WikiWinWin. Then we compared the negotiation results with the original ones. The comparison result revealed the improvement brought by WikiWinWin in bug negotiation.

The rest of paper will be arranged in the several parts. We discuss the related work in section 2. Then introduce the operating principle of WikiWinWin in the Section 3. Section 4 contains the analysis towards the software version evolution problems in NFS and our analysis. The case study will be carried out in Section 5. Finally, we conclude the pros and cons of this paper, as well as future work in the last section.

2 Related Work

Recently, in the bug prevention and removal area, there are a number of researches focused on defect prediction and automated test technique [10]. They focused on proposing new theories to predict defects or designing tools to test bugs automatically, in order to reduce the expensive rework later time. However, there are few tools raised to assist the negotiation in solving bugs.

EasyWinWin is the early support tool based on WinWin requirement negotiation. But it is not easy for clients to update content, preserve revision history and share other requirements related information [2]. Later, WikiWinWin was developed by employing wiki framework [2]. We also did some improvements to WikiWinWin, like balancing the shaper's work load, improving the notification and simplifying the operations [1]. However, all the WinWin related tools are applied in the requirement negotiation. Because of the collaborative features, we assume the WikiWinWin can still improve the bug decision making. The detailed internal logic of WikiWinWin will be described in the next section.

3 WikiWinWin

The WikiWinWin creates a sequence of steps and instructions to guide the stakeholders working out mutually satisfactory requirements [2]. WikiWinWin is consisted by two parts: preparation and main WIOAs (win conditions, issues, options, and agreements). In short, stakeholders raise win condition first, find issues in or between win conditions, propose options to solve issues and reach to agreements finally. There are issues or conflicts among agreements, another iterative negotiation begins.

3.1 Set Up the WinWin Negotiation Context

As Figure 1 shows, the WikiWinWin negotiation process begins with setting up the negotiation context, proceeding to negotiate the WIOAs. According to the experience of EasyWinWin requirements negotiation in over 150 projects [6], we find that it is very important to set up a good context for effective WinWin requirements negotiation.

In setting up the negotiation context, it consists of identifying and engaging stakeholders, introduction to WikiWinWin, kick-off meeting and defining terminologies. More details about the preparations could refer to the previous work [2].

3.2 WIOAs Negotiation

WIOAs which are consisted by a set of activities, guide stakeholders to input win conditions, identify and solve issues and raise options to solve issues [2].

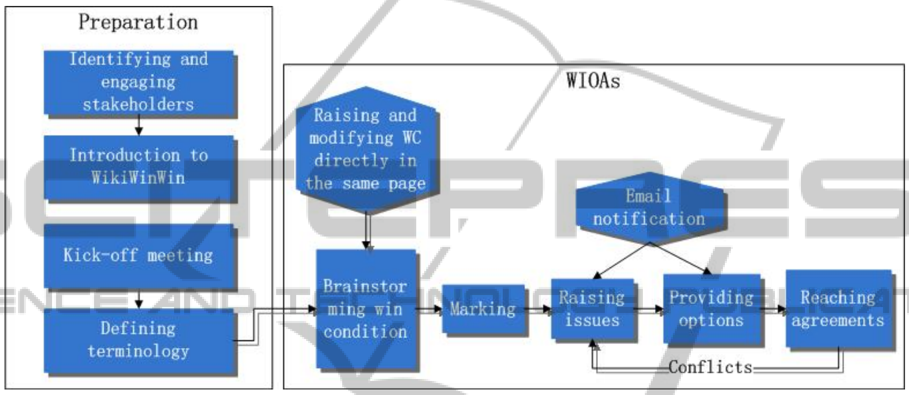
Preparation
Identifying and engaging stakeholders
Introduction to WikiWinWin
Kick-off meeting
Defining terminology
WIOAs
Raising and modifying WC directly in the same page
Email notification
Brainstorming win condition
Marking
Raising issues
Providing options
Reaching agreements
Conflicts

Fig. 1. WikiWinWin negotiation.

First of all, stakeholders can raise their goals, perspectives and expectations as the win conditions. The win conditions will be graded in importance by users or in difficulty by developers. The scores can be considered as development suggestions. Then, stakeholders could propose the potential conflicts and risks towards the win conditions in WikiWinWin. If there are no issues among win conditions, win conditions could be passed as agreements. In order to deal with these issues, we need to propose some options to deal with the issues. Certainly, we can propose more than one option to an issue, because it is possible to have more solutions to a problem. As long as the one of the options is adopted, the relevant issue is solved at that moment. At last, stakeholders could reach an agreement by accepting one option. When the conflicts still exist in agreements, more issues can be raised to settle the contradiction. Our goal is to resolve all the issues and convert win conditions into agreements.

4 Problems Analysis

Due to the distributed locations for stakeholders and large amount of bug reports every day, holding meetings for every bug is impossible. It is also not easy for other communication methods to record and manage negotiation notes. In this way, stakeholders lack the efficient way to negotiation bug solutions. Without deeply negotiation, plan maker like PM is apt to make misjudgments or mistaken solutions.

From the investigation, we found lots of bug solutions in Mantis were just simple version planning, to some product version later. These kinds of solutions have not been negotiated deeply and cannot express the exact information to developers. For example, user wanted to create new projects under a node of a tree. The PM thought it was a simple modification, so she planed the change to the next version, QONE 5.1. Actually, the modification involves the change to database which might influence other functions. It is better to change it in later version. Once they discussed it with developer or other experienced PM by using WikiWinWin, they would find the faults in the solution and correct it. Some of the similar solutions might happen in the comprehension of user's expectations.

We hope the WikiWinWin can mitigate the problems to some extent. WikiWinWin is a Browser-Sever structure system which supports the negotiation in different places and different time. So it can save people's time of attending numerous meetings. A sequence of steps and specific categories in it can guide the negotiation in an efficient and collaborative way. In the next section, we would conduct a case study to examine the effect of using WikiWinWin.

5 Case Study

In order to figure out whether WikiWinWin is helpful to get better bug solutions, we decide to apply the wrong bug solutions from industrial development to WikiWinWin.

5.1 Case Study Design

We arranged three students to undertake the case study. One of them had an internship as software engineer in NFS before. Another one did some similar product manager work in an internet company. They served as developer and PM respectively. The other one would act as user. In order to reduce the difference between the students and staffs in NFS, we were about to train the students according to their different roles. The developer should find the function points which were reported as bugs and be familiar with the technical framework documents. PM also needed to understand the operations of the products and the objectives of each main module. As for the user, we just introduced major functions of QONE on the basis of its instruction manual.

The staffs in NFS provide us 40 bug records, 19 of them have been recognized as incorrect ones. All the students did not know anything about the chosen records in advance. These records are the data to be negotiated in our case study. Moreover, it took 3 hours to introduce the main functions of WikiWinWin and the responsibility of each role and the project background to the students. Then they held a kick-off meeting to talk about the negotiation objectives. All the negotiation result would be evaluated by the professional staffs in NFS.

5.2 Negotiation in WIOAs

The three students began to negotiation the records one by one. User raised their objectives or bugs as win conditions. PM would summarize the win conditions and draft solutions. If nobody had an objection to a win condition, the win condition can be passed to be an agreement. If not, developer or user found there were conflicts or risks in win conditions, they could propose issues. At that moment, WikiWinWin would notify every stakeholder in the group about the progress. Then developer or PM can raise their solutions to the conflicts as options. As long as one option was adopted, the issue was resolved.

As for the record mentioned in Section 4, PM raised the nodes modification as win condition. Developer found the problem in the modification and proposed an issue that the nodes modification was related to the database modification which had not been planned in QONE 5.1. And he provided an option that it would be better to be modified in QONE 5.3 or later. The option was adopted, and detailed development plan was drafted. So this round of negotiation was over. Compared with the solution proposed by PM, the one after negotiation is more appropriate to the development.

5.3 Results and Evaluation

After having negotiated all the 40 records, we asked the staffs in NFS to evaluate our results. Till that moment, the bugs had already been handled. So the staffs knew the correct solutions to the records we chose and their judgments were relatively reliable.

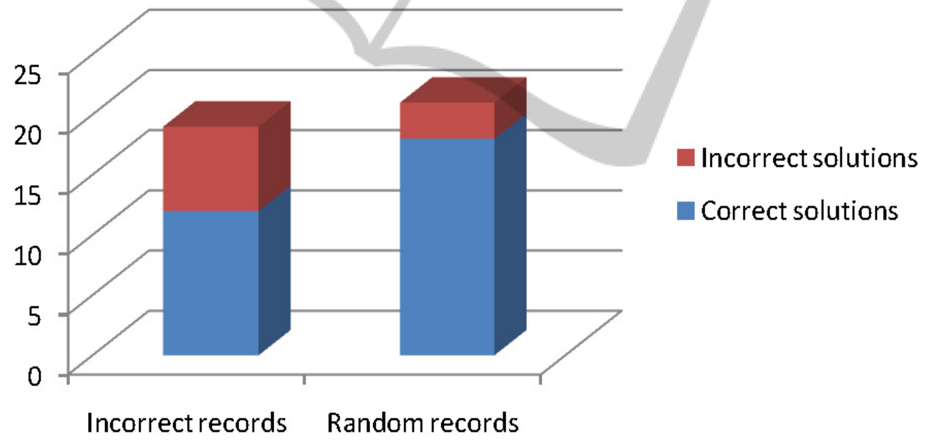

Fig. 2. Negotiation results.

After the case study and evaluation, 12 of the 19 incorrect records reached to correct solutions and 18 of the 21 random records got correct solutions as Figure 1. So the total correct rate is (12+18)/40 = 75%. In the 10 incorrect solutions, 7 of them were resulted by the unfamiliarity to the products, including the relations of code class libraries and technical framework. There were also some outside reasons led to the incorrect solutions, like some complicated bugs are urgent for large customers, so that they cannot wait to later versions. The accidental and prior mission disturbed the later bug solutions.

Anyway, the negotiation results by using WikiWinWin improved the solutions to solve bugs. Compared with the average time to hold meetings, negotiating with WikiWinWin can also save a lot of time for most stakeholders. Except for the deviation caused by students, the negotiation results still have room to improve once the stakeholders can be more familiar with the products and techniques.

6 Conclusions and Future Work

In this paper, we employed WikiWinWin to negotiate the bug reports in an industrial software company. Due to the distributed locations and lots of bugs every day, it is impossible to hold meetings for every bug and traditional communication ways are either easy to lose records which can be significant references for software evolution or lack the structural discussion scheme. It is required to find an efficient and well-organized negotiation tool to fill in the blanks. WikiWinWin which combines the win-win spiral model with collaborative knowledge techniques mitigate the problem to some extent [10]. We also conducted a case study with 40 samples of the bug reports and compared negotiation result with the original ones. The case study result indicated the improvement to the handling of bug reports.

In future, we would keep on improving negotiation process based on WinWin Spiral Model. Also, we would contribute to improve the negotiation efficiency of WikiWinWin. WikiWinWin is planned to parse the win condition, issue and option automatically from the reply email. It would go a further step to facilitate the operation.

Acknowledgements

The authors are sponsored by the 973 Program No. 2007CB310802, 863 Program No. 2007AA010303, NSFC No. 90718042, 60803023, 60873072, 60903050 and 61073044, CASIP ISCAS2009-GR and National Science and Technology Major Project.

References

1. Peng Wan, Shengyu Huang, Juan Li, Da Yang, Ye Yang. Balancing Load of Shaper in WikiWinWin Requirements Negotiation Environment: An Empirical Evaluation. PROFES 2010.
2. Da Yang, Di Wu, Supannika Koolmanojwong, A. Winsor Brown, Barry W. Boehm, WikiWinWin: A Wiki Based System for Collaborative Requirements Negotiation, Proceedings of the 41st Hawai International Conference on System Sciences – 2008.
3. Barry Boehm, Alexander Egyed. Software Requirements Negotiation: Some Lessons Learned. Proceedings of the 20th international conference on Software engineering.
4. B. Boehm and R. Ross, "Theory-W Software Project Management Principles and Examples", IEEE Transactions on Software Engineering, Vol. 15, No. 7, July 1989, pp. 902 – 916.

5. B. Boehm, P. Bose, E. Horowitz and M. J. Lee, “Software Requirements Negotiation and Renegotiation Aids: A Theory-W Based Spiral Approach”, Proceedings of the 17th international conference on software engineering, April 1995.
6. Wikipedia: MediaWiki : www.mediawiki.org
7. C. Wagner and P. Prasarnphanich, “Innovating Collaborative Content Creation: The Role of Altruism and Wiki Technology”, HICSS, 2007.
8. Paul Grünbacher and Norbert Seyff. Requirements Negotiation, chapter 7.
9. A. Güne¸.S. Koru and Jeff Tian, Southern Methodist University. Defect Handling in Medium and Large Open Source Projects, IEEE SOFTWARE.
10. Eric Bean. Defect prevention and detection in software for automated test equipment, Autotestcon 2007, Page(s): 224 – 233.