Focusing Extreme Programming on Usability

Ralf Carbon[1], Jörg Dörr[2], Marcus Trapp[1]

[1] Technische Universität Kaiserslautern, Erwin Schrödinger Straße,
D-67653 Kaiserslautern, Germany
{ralf.carbon, marcus.trapp}@informatik.uni-kl.de
[2] Fraunhofer Institut Experimentelles Software Engineering, Sauerwiesen 6,
D- 67661 Kaiserslautern, Germany
joerg.doerr@iese.fhg.de

Abstract. Agile methods such as extreme programming are becoming increasingly important for the rapid development of software applications. They proofed to be applicable and beneficial for a variety of domains. Nevertheless, due to the iterative nature of these approaches, software developers focus on functional aspects and tend to neglect nonfunctional characteristics, like usability. This paper addresses this issue by providing an extension suggestion for extreme programming that helps the on-site customer and development team to take into account usability. Usability criteria are attached to user stories, and a new type of user stories, called usability stories are introduced. This also influences the way on-site customers perform the acceptance test. We present first results of a case study where small changes to extreme programming have been introduced. The benefits were an increased perception of usability due to the integration into the user stories and an early evaluation of usability criteria in the acceptance test. Eventually, these measures lead to a high usability of the software system.

1 Introduction

Agile methods such as extreme programming (XP) [1], are becoming increasingly important for the rapid development of software applications. One objective, enterprises try to accomplish with the application of agile methods, is a reduced time to market of their software systems. Approaches like XP, for instance, aim at the completion of first releases of software systems, within a few weeks or months. Thus, enterprises developing software for highly dynamic markets like mobile applications or Internet software can gain competitiveness by using agile methods. But today, putting into production just executable software is not enough. The usability of software systems is also a crucial factor for the competitiveness of an enterprise [2]. In a small development project using XP at the University of Kaiserslautern in 2003 we observed considerable usability problems of the resulting software product developed with XP [3]. The students produced releases of the software system with a remarkable amount of functionality in a short period of time, but the usability of the software system was rated as very poor. For example, without instructions it was very difficult for new users to perform any task,

because they were confused about the tool's dazzling navigation. Furthermore, it was impossible to undo a wrong action. The only chance to prohibit a persistent defect was to exit the software system.
Asking the involved students, we found out that they were totally focused on implementing the functional features requested by the customer through the user stories and did not spend enough time considering the usability of the system. The on-site customer was, as usual, not a usability expert and let the user stories pass the acceptance tests if the requested functionality was implemented, despite the bad usability of the system. Other XP projects like [4] had similar observations. Thus, our goal was to focus XP more on usability.

To focus XP on usability we adapted the XP process as follows: Before writing the user stories, the on-site customer gets advice by a usability expert on how to implement concrete usability goals, i.e., they derive usability criteria, the software system should fulfill. These usability criteria are integrated into existing and sometimes new user stories. The new user stories are called usability stories, as they represent pure usability features. The developers implementing an annotated user story or usability story are always aware of the usability criteria, because the usability criteria are documented on the story cards. In this case, the dissemination of usability goals does not rely exclusively on the communication with the on-site customer. But the key to achieve a compliance of the resulting software system with the usability criteria is the acceptance test. The on-site customer has to make sure that user stories can only pass the acceptance tests if the integrated usability criteria are fulfilled. Thus, we must write acceptance test cases to a user story checking the attached usability criteria. Therefore it is important to strengthen the customer's usability perception to allow usability evaluation during the acceptance test. To get first experiences with this new approach, we performed a small development project at the University of Kaiserslautern in 2004. The results are encouraging.

The remainder of the paper is structured as follows. In the following section, we give a brief introduction into the XP project flow with a focus on the aspects that are most relevant for our extension. In Section 3 we present our approach to focus XP on usability. We discuss our approach in Section 4 and show first results of our case study. Section 5 gives a short summary.

2 The Project Flow of a classical XP Project

XP provides a collection of twelve practices which, in combination, aim at the rapid development of high quality software. But XP is not just a collection of twelve practices but prescribes a specific flow of a classical XP project [7]. The four practices most important for our approach are: on-site customer, planning game, testing, and small releases. In this section we introduce the concept of user stories, the four practices mentioned above, and their interaction in more detail. For more information on the other eight practices we refer to Beck [1].

The on-site customer is responsible for writing the user stories, which describe the high level requirements of the software system, similar to very short use cases. The detailed requirements are verbally communicated by the on-site customer on request. XP does not prescribe a way to deal with nonfunctional requirements like usability. The degree they are dealt with totally depends on the expertise of the on-site customer. In the

release planning game, the developers estimate the effort they need to realize each of the user stories and the on-site customer selects user stories for the next release. The on-site customer and the development team have to set an end date of the first release. According to the practice small releases, they should agree to complete a first release in a few months or weeks to put a small system into production quickly. A release is broken down into iterations of several days or weeks.

During acceptance testing, the on-site customer performs acceptance tests for every user story. If the acceptance tests derived from a user story are performed successfully, the user story should be accepted, otherwise it is refused. But the customer is allowed to refuse a user story, even though there is a successful acceptance test for that story, for instance, if a feature is not usable.

Summarized, the XP project flow can be characterized as follows (Figure 1 without extensions in italics): The customer writes user stories and partitions them into small releases, based on effort and risk estimations and his view on how beneficial a user story is for his/her company. Every release is split into several iterations. At the end of each iteration an acceptance test is performed.

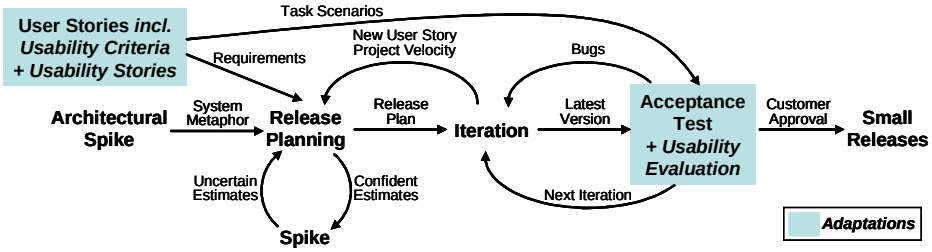

Figure 1: Adapted XP project flow

3 An Approach to focus XP on Usability

Based on the XP project flow described in Section 2, we identified different possibilities to influence the resulting usability of the software system. Our goal was to integrate usability criteria as early as possible into the development process to guarantee their realization from the beginning. This is especially important, as it is very difficult and costly to establish usability late in software development [2].

Figure 1 shows the adapted XP project flow based on [7]. Basically, there are four adaptations we made to XP:

- Integrate usability criteria to classic user stories.
- Introduce usability stories.
- Derive acceptance tests according to the integrated usability criteria and the usability stories.
- Perform usability evaluation during acceptance testing.

The first step towards our adaptations was to provide the on-site customer with a usability expert before writing the user stories. We assume that the on-site customer is not a usability expert and needs advice to derive precise usability goals. Starting with a list of usability heuristics [6] the usability expert and the on-site customer select the

heuristics most valuable to the customer and derive concrete usability criteria for the current project.
Let's assume two of these derived usability criteria are:

1. Every action of the user can be canceled (without consequences).
2. All error messages must be written in natural language and have to contain a possible solution.

The next step is to integrate these usability criteria into the user stories. We identified two possibilities to deal with these usability criteria. If a usability criteria is only related to a subset of all the user stories, we attach it directly to these user stories. If a usability criterion is related to all the user stories of a system, i.e., the whole system should fulfill this usability criteria, we write a usability story.

In the example given above, we decided to integrate criteria 1 in all user stories where the user modifies data in the system. Criteria 2 leads to a separate user story (usability story), because error messages can appear using any feature of the system.

In the release planning game all stories are estimated by the developers. The influence of the integrated usability criteria on the effort must be taken into account.

The development is performed as usual using the XP development practices. Together, the on-site customer and a tester write the acceptance test cases. They have to include test cases checking the functionality as well as the integrated usability criteria on the story cards. Acceptance tests are also derived from the usability stories. Thus, acceptance testing does not only mean to validate the features of the software system, but also the usability criteria. We cannot speak of formal usability testing here [5], because the on-site customer performing the tests is not necessarily the user of the system in operation. For the usability criteria 1 from our example above, we added at least one test case to every affected user story. In these test cases, data manipulation actions of the user are canceled. In the case of usability criteria 2, we must write test cases that produce error messages. Then the customer hast to check for each error message, if it is written in natural language and contains a solution. This example shows that usability criteria cannot always be checked automatically.

4 Results from the first Evaluation of our Approach

We evaluated our approach in a small development project at the University of Kaiserslautern in 2004. Seven master students in the 3rd respectively the 4th year of their studies spent two weeks full time on the development of a time planning system for projects. The students were taught XP in a one day tutorial before the project. They had no practical experience with XP. Six of them were advanced Java developers. The on-site customer derived a set of nine usability criteria in cooperation with a usability expert as proposed in Section 3. These usability criteria got integrated into the story cards. The starting point for the project were 36 user stories including 7 usability stories. The goal was to achieve an improved usability of the resulting system in comparison to the development of a similar time planning system in 2003 which was built with a classical XP approach. At the end of the two weeks, the students produced 60 classes with a total number of approximately 9000 LOC. 26 of the 36 user stories were implemented and accepted by the on-site customer. Overall, we achieved an increased usability of the resulting software system, i.e., the usability criteria integrated into the user stories and

the usability stories were fulfilled at the end of the project. During the first iteration, the students nearly ignored the usability criteria and were only focused on functionality. But the first acceptance test with integrated usability evaluation had a great effect. In the first acceptance test 16 out of 18 user stories were refused, 10 of them for usability reasons, i.e., more than 55% of all the user stories of the first iteration were refused for usability reasons. The perception of usability increased dramatically after this acceptance test with usability evaluation. In the acceptance test at the end of the 2nd iteration all the user stories refused for usability reasons in the first acceptance test were accepted. It seems that we need at least two iterations to achieve sufficient usability of a feature. The number of questions regarding usability asked to the on-site customer increased. Three out of six new user stories were refused at the end of the 2nd iteration, two of them for usability reasons, i.e., we have a refusal rate for usability reasons of 33% regarding the new user stories at the end of the 2nd iteration. Overall, two out of 22 user stories were refused for usability reasons in the acceptance test after the 2nd iteration, i.e., about 10 %. This result shows the great effect of the acceptance tests with integrated usability evaluation on the perception of usability and the usability of the resulting system, because the refusal rate could be reduced during the 2nd iteration.

The effect of the explicit documentation of usability criteria on story cards remains unclear. According to the first iteration of our example project, we could assume that the documentation of the usability criteria nearly had no effect, because the usability of the system after the first iteration was quite bad as shown above. During the next iterations, the effect of the documentation is not clear. The perception of usability increased in the 2nd iteration, but we cannot separate the effect of the first acceptance test including usability evaluation from the effect of the explicit documentation of usability criteria during the 2nd iteration.

5 Summary and Future Work

In this paper we presented our approach to focus XP on usability. The main adaptations to classical XP projects were 1) to attach usability criteria to classic user stories, 2) to introduce usability stories, 3) to derive acceptance tests according to the usability criteria on the story cards, and 4) to perform usability evaluations during acceptance testing. We evaluated our approach in a students project. The acceptance tests with integrated usability evaluation showed a great effect on the students perception of usability and the usability of the resulting software system. The effect of the explicit documentation of the usability criteria remains unclear. Our approach is a starting point for further research on focusing XP on usability. On the one hand, we need further evaluation of our approach, on the other hand we need to identify more possibilities to integrate usability engineering techniques into XP.

References

[1] Beck, K.: Extreme Programming Explained: Embrace Change. Addison Wesley, 1999.

[2] Bias, R. G., Mayhew, D. J. (editors): Cost-Justifying Usability, Academic Press, 1994.

[3] Bunse, C., Dörr, J., Feldmann, R. L.: Agile Methods in Software Engineering Education, 5th International Conference on Extreme Programming and Agile Processes in Software Engineering, XP Universe, 2004.

[4] Jokela, T. Abrahamsson, P.: Usability Assessment of an Extreme Programming Project: Close Co-operation with the Customer Does Not Equal to Good Usability, 5th International Conference on Product Focused Software Process Improvement, PROFES, 2004.
[5] Mayhew, D. J.: The Usability Engineering Lifecycle, Morgan Kaufmann, 1999.
[6] Nielsen, J., Mack, R. L.: Usability Inspection Methods, John Wiley & Sons, 1994.
[7] XP Flow Chart; http://www.extremeprogramming.org; last visited May, 24th, 2004.