ABSTRACT

Apart from the faster time-to-market demand, web application is becoming more complex and mission critical where it implements continual integration of new requirements for business logic. Such demands have increased the reputation of agile software processes such as Extreme Programming (XP) that allow flexibility and yet let the teams increase development productivity while maintaining software quality. But does XP principle that favours working software over comprehensive documentation is enough for web application development?

This paper describes the integration of User Experience (UX) Model in Planning Game Practice in XP to improve the communication and understanding of both developers and user. The enhancement helps to deliver the resulting product that will more likely satisfy end user’s expectations.

Keywords: Extreme Programming (XP), Unified Modeling Language (UML), User Experience (UX) Model, User Stories

1. INTRODUCTION

Trying to get the requirements right before the software is designed and implemented is problematic in Web application systems where requirements frequently change [1]. In recent years, Extreme Programming (XP), an agile methodology has been advocates as the suitable and popular method for the high-speed, volatile world of Internet and Web application development where it focuses on delivering executable code and close-customer collaboration as the main elements of software development[2].

Although XP has demonstrated to overcome the problems of requirements communication, XP was developed to resolve problems in software not Web development where there are a number of differences such as multidisciplinary team members, multi-user environment and other differences [1].

At the heart of any software modeling effort, is requirements gathering and the primary measure of success of a software system is the degree to which it meets the purpose for which it was intended [3]. Thus, Planning Game practice where user requirements are gathered is believed to be the most crucial part in systems development. This is because trying to get the requirements right before the software is designed and implemented is problematic in web based system where requirements are frequently change [4].

User requirements and estimation on the time duration of the development plan are discovered during planning game practice. The planning game and small releases depend on the User Stories which are written by the customers using Class-Responsibility-Collaboration (CRC) cards. User Stories is a short description of features that categorized the task to be performed in each release [24]. This practice replaces the traditional requirements specification in terms of the level of details captured and the focus of the user needs but provides enough detail to make a reasonably low risk estimation of how long the story will take to implement.

A collection of User Stories characterizes the full functionality of the system required by the customer, but only the identified User Stories with the highest priority are being implemented for the next release. New User Stories can be added to the collection at any time, therefore the requirement can be highly volatile, yet the volatility is managed by implementing only the identified stories for each release[5]. However, it leads to insufficient documentation that is identified as the major problem for projects with complex domains or large application systems, as CRC cards alone may not be sufficient as a discussion basis to capture
requirements for the planning game practice [6]. Wills [7] found that it is impractical to use program code and informal stories as the only means of discussing the requirements and the architecture of the system when it involves large team with varied experiences. Fowler [8] has combined both CRC cards and Unified Modeling Language (UML) in XP practices to improve requirements communication as the diagrams act to summarize and highlight important issues rather than textual CRC cards. User stories in CRC cards which are basically textual use-case descriptions are incapable to represent the static and dynamic content of web pages, navigational flow and relationships to business data in web application. Dennis [9] stated in his book that CRC cards are used to document the responsibilities and collaborations of a class only. Hence, it can represent neither web contents nor navigational paths.

User Experience (UX) Model is recommended by Jim Conallen for better Web application development modeling where it captures concept for representing engineering aspect of screen-based user interfaces which reflect the abstraction of the web pages and navigation routes of the system[10].

While maintaining the characteristic of XP, we suggest the enhancement of XP by integrating UX Model in Planning Game practice for web application development.

2. EXTREME PROGRAMMING (XP)

Extreme Programming (XP) is an agile software methodology developed by Kent Beck, Ron Jeffries and Ward Cunningham that usually targeted toward small to medium size teams building software in the face of vague and/or rapidly changing requirements as stated in Paul C.M [5]. XP is popular for the tight customer-developer collaboration in order to ensure any misunderstanding of user’s requirement are corrected and another a rigorous approach to software development is another characteristics of XP, where specifications are encoded carefully before the implementation and verified against the former functional (acceptance) tests to make sure the program does what it supposed to do. XP is a disciplined software development methodology demonstrated via four values: continual communication with the customer and within the team, simplicity by always focusing on the minimalist solution, rapid feedback via unit and functional testing (among other mechanisms) and the courage to deal with problems proactively [5] that consists of twelve major practices including Planning Game [5]. The original XP as illustrated in Figure 1 shows that client works closely with developers by incorporating all the twelve practices of XP.

![Figure 1: The Original XP](image)

Even though there are a lot of projects that adopt XP practices, applying all these practices as the literature suggests is sometimes difficult. Emery P. [11] reported in his paper that XP’s approach is fragile and can fail if the project environment changes or the people change. Many developers find that XP does not allow for modeling time, which is critical to ensure that a project meets its proposed requirements.

In this paper, we concentrate only on planning game practice for the enhancement.

2.1 Planning Game

Planning Game is the main planning process of XP. where it can be divided into two parts: Release planning to answer what will be accomplished by the due date and Iteration Planning for predicting, and determining what to do next[12].

Release planning is a practice where the Customer presents the desired features to the developers using User Stories that act as the substitute of a large requirement document. Customers will write the User Stories describing what they desired for the system by using CRC cards in the format of just few sentences. The sentences are written in the customer’s terminology without any technical jargons or techno-syntax. Details on specific technology, database lay out and algorithms are avoided in User Stories. These User Stories are never formalized; no relationships or
dependencies between the cards are identified [13]. Then, the developers create time estimates based on their difficulty for each releases of the system.

User stories should only provide enough details for the developers to make low risk estimation on the time for the implementation of the story. The ideal estimation time in XP practices that a developer should consider usually in a week to 3 weeks time for each release. User stories also drive the creation of acceptance tests in order to verify the User Story has been correctly implemented [14]

Release Plan is then used to create iteration plan for each individual iteration. It is the practice whereby the developers are given direction every couple of weeks. Every two or three weeks top, a successful release will be presented. During Iteration Planning, the customer presents the features desired for the next two weeks where it will be split down into smaller tasks, and estimate their cost (at a finer level of detail than in Release Planning) [15]. Based on the amount of work accomplished in the previous iteration, the developers sign up for what will be undertaken in the current iteration.

Being only textual documented, User Stories alone are not able to sufficiently show the desired requirements as the appropriate level of detail is still a concern when writing the story. There were problems associated with user stories: first, is when customers struggles with finding the level of detail and decomposition that ensures the programmer is able to engage and understand the purpose of each story while ensuring the customer does not make design decisions in how the functionality is decomposed. Secondly, one critical aspect of story writing that determine the quality of the product, and occasionally being neglected is the inclusion of exceptions and error conditions for each story [8].

Many developers find that XP does not allow for modeling time, which is critical to ensure that a project meets its proposed requirements. Being only textual documented, User Stories alone are not able to satisfactorily explain the desired requirements. A customer being the non-technical people may wish for the impossible that might be beyond what the developers and the current technologies are capable of. Models help in understanding the system by simplifying some of the details. A model is a representation of a system from particular viewpoints for better understanding of the system.

3. USER EXPERIENCE (UX) MODEL

The Rational Unified Process defines User Experience (UX) as the model, which describes the screens of the system, the dynamic content that appears on the screens, and how the user navigates through the screen to execute the system functionality [16].

UX Model is vital in modeling web application development as it can simplify the details. Conallen [17] stated in his book that the real values in models and modeling are the ability to look at a simplification of a system through a particular viewpoint where the system becomes easier to understand. If the models are as complex as what is being modelled, there is little point in modeling.

For systems like web applications system that has significant amount of user iteration, UX Model is essential to the developers and customer since it will provide clearer view of the web application. The user interface in web applications is almost like a set of Web pages, with each page containing static and dynamic content. Use case scenarios are typically executed across a number of web pages. The navigational routes through these pages become an architecturally significant element of the system.

The UML is appropriate for modeling systems ranging from enterprise information systems to distributed Web-based applications and even to hard real time embedded systems [18]. In comparison with the traditional UML, UX Model is the extension of UML notation with additional semantics and constraints that is Web Application Extension (WAE) to permit modeling of web specific architectural elements as part of the system’s model. By using UML alone, it would not be sufficient to capture the relevant semantics of a particular domain or architecture. Therefore by implementing UX Model, it captured screens and content description, storyboard scenarios and navigational paths through the screen of web system [19].

UX is used to describe the team and the activities of those specialists responsible for keeping the user interface consistent with current paradigm and most important for the context in which the system is expected to run. UX Model provides a complete view of the system from a particular viewpoint; the system through its screen that includes the main principles that is the
architecturally properties of the screens and their navigational relationships.

When screens of the web system are combined and ordered, they can describe use case scenario or mini-stories of the behavior in an application, expressed in terms of sample screens. Keeping and maintaining these scenarios throughout the development process is an excellent way for non-technical stakeholders to keep in touch with the system’s progress. It might also give these same stakeholders the sense of completion, since the storyboards are almost always completed long before the rest of the system is ready. Navigational Path is one of the most architectural important artifacts in UX Model. It expresses the structure of an application’s screens with their potential navigational pathways: a road map of the application’s screen. An important characteristic of this diagram is that it expresses all the legal and expected paths through the system [19]. Therefore, we believe UX Model is capable to solve the limitations of planning game as discussed in section 4.

4. ENHANCING XP

Most of the developers find it hard to secure a design in the head without some means of writing various abstractions: requirements, relationships, and collaboration [20]. That is why models help in communication and discussion of abstractions. Trying to preserve the nature principle of XP, the model used should be simple with modest details.

UML provide better medium for this purpose rather than programming code. Wills [7] found that it is impractical to use program code and informal stories as the only means of communicating the requirements and the architecture of the system especially when it involves a large team with varied experiences. Watts Humphrey[21] enumerates that lack of design documentations in XP practice, limits XP to only small programs and makes it difficult to take advantage of reuse opportunity. The orthodox XP is code-centered rather than design-centered which might not be serious for small programs but it can be disastrous when programs are larger than a few thousand lines of codes or when the work involves more than a few people.

XP practices can cause major architectural design problems when it focuses on doing only what is immediately necessary [21]. Design that only based on user stories are not sufficient enough when it is impossible to capture long-term goals early on in the beginning of a system’s lifecycle, thus leads to project failure. A model that is simple and easy to be understood by both the developers and customer helps for better communication of system development.

Due to the limitations of XP, we proposed to incorporate UX Model in XP methodology as illustrated in the following Figure 2.

It concentrates on screen description, storyboard scenarios and navigational paths of the system. The notation used should be simple for better understanding for both customers and developers. Since it simplifies the communication of requirement process, it will preserve the principles of XP. In addition, it provides a clearer view point of the system, thus improve the communication of user requirements in web application development between the users and developers.

![Figure 2: The proposed model](image-url)

The proposed model was put to test with a series of workshops where each workshop was a simulation of the enhanced planning game that integrate UX Model. The usability of the method is defined in the ability to produce a satisfactory development output from the decisions made at the planning game meeting. The workshops were carried out with a group of students, who are in the process of developing a system for conference...
management. The procedures of the workshop is illustrated in Figure 3. The system development was a part of deliverables where the students are restricted by time scope of the course.

However, ToxBoe[22] stated in his paper that one of the biggest problems Häitinen faced when trying to test a combination of XP and GUIDe that the test subjects did not have any prior experience with XP. Acknowledging the suggestion, this project only focused on the planning aspect of XP, without anything to do with the actual development of the system including testing of the programming which will also be postponed for future workshops with experienced programmers. The system development are carried out by the student without researchers intervention, nevertheless the development must still follow the XP principles.

4.1 Procedure of the workshops

Each workshops carried out with the students were slightly different from each other in its procedure of how each part is carried out.

![Figure 3: Procedure of the workshop](image)

**4.1.1 Creation of User Stories**

In this step, a collection of user stories were created in each of the students group. Role playing approach were used where a student from each group played the customer role. The qualifications and knowledge of the system needed were provided to the ‘customer’ before the workshops.

**4.1.2 Release Planning**

Release planning is the step where user stories were estimated and grouped into iterations. The estimation of how much the time used to implement each user stories were skipped since the participants had no exposure and experience in doing time estimation. Instead, the judgements of the amount of possible user stories to implement in the next few iterations were made.

**4.1.3 UX Model Design**

This is the stage where the integration is made. Based on the prioritized user stories, developer models the UX Model by hand (which we believe it is much better to have modelling tools like Rational Rose). The UX Model must be a simple design focusing only on customer viewpoint where the system should become easier to understand for better communication. UX Model in this proposed model only involve the screens content and descriptions, the storyboard scenario and navigational map of the web application system.

**4.1.4 Creation of Acceptance Test**

One or more acceptance tests are created for each user stories that was chosen for the next iteration. The acceptance test is a basic step-by-step scenario that should be able to run when the user stories is implemented. After the acceptance tests are created, the students will start doing the system development to develop the prototype. The acceptance test are statements of stating the user requirements based on the user stories.

In the next planning game, the acceptance tests were verified in term of whether the test could successfully run or not in order to test the method. In other word, whether the deliverables functioning as mapped to the acceptance tests created.

5. FINDINGS

The method was tested specifically on the effectiveness of the introduced method using a well
tested questionnaire on effectiveness adopted from Empirical Software Engineering Journal, a work of Damian et. al [23]. It has shown positive results toward the effectiveness of this integration model.

The findings reveal a significant association between the level of involvement in planning game sessions and perceived effectiveness of the proposed model. The adopted questionnaire was also intended to address issues such as participation and satisfaction of the process, immediate and long-term perceived benefits of the proposed model [23].

By integrating UX Model in Planning Game, there are chances of interfering with the basic principles of the original XP. The principle that might be in jeopardy is the working software over comprehensive documentation. As long as the additional documentation facilitates more than it harms, this principle should not be in danger. Documentation in XP is not prohibited, just less preferred in comparison with working software. However, as long as the additional documentation is kept at the minimalist and simple, the principle is still safe and secured.

REFERENCE


