IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 8, July 2015.
www.ijiset.com
ISSN 2348 – 7968

Software Reliability Prediction and Improvement Strategies for Web Applications

Tejinder Kaur, Kailash Bahl
Computer Science and Engineering Department
Patiala Institute of Engineering & Technology for Women

ABSTRACT

Software reliability is an important aspect of software quality. Software reliability is defined as “the probability for failure-free operation of a program for a specified time under a specified set of operating conditions”. is one of the key attributes when discussing software quality. Software reliability for web applications is measure via different metrics and testing techniques. In this paper, we characterized the metric, techniques, errors for web applications. This paper also examine’s the potential for reliability improvement. Based on the characteristics of web applications and the overall web environment, we classify web errors and to check reliability using different metrics parameters.

Keywords: Reliability, web applications, reliability metrics.

1. INTRODUCTION

Software reliability plays very important role in the field of computer science. Software reliability is the probability that software will work properly in a specified environment and for a given amount of time. The IEEE defines reliability as “The ability of a system or component to perform its required functions under stated conditions for a specified period of time.”[3] The area of software reliability covers methods, models and metrics of how to estimate and predict software reliability. This includes models for both the operational profile, to capture the intended usage of the software, and models for the operational failure behavior. The latter type of models is then also used to predict the future behavior in terms of failures. Before going deeper into the area of software reliability, it is necessary to define a set of terms. Already in the definition, the word *failure* occurs, which has to be defined and in particular differentiated from *error* and *fault.*

Software reliability is defined as a probabilistic function and comes with the notion of time, it is noticeable that software reliability is different from traditional hardware reliability and it is not a direct function of time. Hardware reliability with the heavy usage and as the time goes electronic and mechanical parts may become “old” add wear-out but software will not rust or wear-out during its life cycle. Software will not change over time unless intentionally changed or upgraded. Software reliability is based on the concept of failures.[3]

1.1 Aim of the Research

Reliability is usually defined in terms of a statistical measure for the operation of a software system without a failure occurring. Software reliability is a measure for the probability of a software failure occurring. Reliability is the probability that a system, or a system component, will deliver its intended functionality and quality for a specified period of "time", and under specified conditions, given that the system was functioning properly at the start of this "time" period [4].

2. MOTIVATION

Looking specifically at the aims of our research above, we begin to formulate some of the most obvious questions surrounding these issues. For instance, it will be interesting to explore the kinds of pressure the manager faces in trying to bring the agile principles to life. With agile processes promoting the concept of 'self-directed teams' [1], Is there room for the conventional project manager or does this role also need to evolve to suit the principles of the new paradigm? How are the agile manager roles different from the traditional manager roles?

How do organizations adapt to a radically new framework such as agile? Does it merely take up gradation of technical skills or a complete change in outlook and the way the organization works?

Finally, as outsourcing becomes a common practice, we are faced with other challenging questions. How difficult is agile project management for outsourced or off-shored projects? Does communication become a prime concern or does management suffer at the hands of trying to synchronize distributed teams spanning different continents and time zones? Is agility thrust upon the teams that the projects are outsourced to by their parent

ISSN 2348 – 7968

companies or do they freely choose to follow agile processes?

These are the issues we hope to investigate in our case studies

3. BACKGROUND

The research focuses on agile methods like eXtreme Programming (XP) and Scrum and use grounded theory as a qualitative research method. Before we describe the details of the proposed research, we discuss these concepts in more detail.

3.1 Agile Methods

Agile methodologies follow iterative and incremental style of development that dynamically adjusts to changing requirements and enables better risk management. The four basic principles of agile as defined by the Agile manifesto [2] are:

Individuals and interactions over process and tools,

- working software over comprehensive documentation,
- Customer collaboration over contract negotiation, responding to change over following a plan.

There are differences between the traditional ways of software development and the agile style of working. Some of the prominent ones are highlighted in table 1

Table 1: traditional Vs Agile Project Management

Categories	Traditional	Agile
Development Model	Traditional	Iterative
Focus	Process	People
Management	Controlling	Facilitating
Customer Involvement	Requirements gathering and delivery phases	On-site and constantly involved
Developers	Work individually within teams	Collaborative or in pairs
Technology	Any	Mostly Object Oriented
Product Features	All included	Most important first
Testing	End of development cycle	Iterative and/or Drives code
Documentation	Thorough	Only when needed

3.1.1 eXtreme Programming (XP)

XP was created by Kent Beck [3], who compiled a collection of good practices and took them to the extreme. Programming. Its mostly targeted at small to medium sized projects and has gained rapid acceptance and practice over the world.

The five XP values are communication, simplicity, feedback, courage, and respect. Its hallmark principles are planning game, small releases, metaphors, simple design, refactoring, pair programming, testing, collective ownership, 40-hour work week, on-site customer, coding standards, and continuous integration.

Customers provide the specification of required functionalities in the form of user stories [4]. They are written concisely in non-technical formats and focus on the needs of the user avoiding any design details. They help the developers to estimate the implementation time and go into the release planning.

Each short iteration achieves a handful of tasks and its recommended that a steady project velocity be maintained. Developers work in pairs and perform unit tests and integrate code often. The customer is ideally available on-site and is closely involved in the development through rapid feedback. Refactoring the code to renew obsolete designs and remove redundancy allows for a higher quality product to be produced. Testing is an important part and is undertaken frequently in form of unit tests and acceptance test. Finally lengthy documentation is avoided and optimization is left till last.

3.1.2 Scrum

Scrum is another agile development methodology developed by Jeff Sutherland and formalized by Ken Schwaber.

The roles involved in this process are Product Owner, Scrum Master, and the team. The Product Owner is responsible for maintaining the correct business perspective. The Scrum Master works with the Product Owner and facilitates the team. The team should contains seven (plus/minus two) members.

Activities include sprint planning, sprint review, and scrum meeting. A sprint is usually 2 to 4 weeks of development time where a set of selected issues are worked on. The sprint review reviews the previous sprint in terms of tasks achieved and the next sprint details are

ISSN 2348 – 7968

defined. The Scrum Master leads a daily 15 minutes meeting where each member briefly describes their tasks and concerns.

The artifacts produced are named Product Backlog, Sprint Backlog, and Burndown Chart. The product backlog is a list of product features prioritized by value delivered to the customer [5] and is maintained by the Product Owner. The sprint backlog refers to the development tasks that are needed in order to implement a feature and is a subset of the product backlog. The burndown chart shows the total work remaining in a sprint.

4. AGILE PROJECT MANAGEMENT

Project management is an integral and indispensable part of any software development process. Managing the teams, customer relationships, cost reduction, risk management, maintaining project time line and budget constitute the crux of project management. Although these basic tasks remain the same, their execution details are slightly different for each agile framework. What is drastically different- is the way of thinking.

The role of the project manager has undergone considerable changes with the evolution of agile methodologies. While Scrum has introduced the Product Owner and Scrum Master, XP has invented the roles tracker or coach.

Sanjiv Augustine and Susan Woodcock explore the role of the project manager and propose the concept of visionary leader as opposed to an uninspired taskmaster.[1] While traditional management was viewed as governing and commanding, experienced agile project managers are preaching of 'self-directed teams' with 'light touch' leadership[1]. Similar sentiments are resonated by Mary Poppendieck in a panel discussion titled Agile Management – An Oxymoron? [8] where Poppendieck notes " *I distinguish management tasks – getting the maximum value from the dollar – from leadership tasks – helping people to excel. Leaders are required. Managers are optional.*" It will be interesting to observe the new face of project managers in an agile setting through the course of this research. We plan to interview managers about their perceived roles and how different do they find it from the traditional manager roles.

Our research also aims to focus on the process and problems of transitioning into an agile framework. Experienced practitioners suggest having a checklist to assess the company's need and readiness for embracing the agile wave. In a paper by Nerur et al. from the University of Arlington, Texas [9] various issues related to transitioning into an agile environment are mentioned. They are broadly divided into people-related, process-related, and technological issues.

Finally, the last item on our research agenda is exploring the management of outsourced or off-shored agile projects. We hope to contact and collaborate with companies in India that deal with off-shored agile projects. We find that previous research on outsourcing in an agile environment suggests that there is a co-relation worth exploring. In a paper titled When XP Met Outsourcing [10], the researchers note that they " ***saw a strong awareness of the interactions between outsourcing arrangements and the XP process***". Sutherland et al.[11] note in their recent paper that in order to ensure success "***outsourced teams must be highly skilled Agile teams and project implementation must enforce geographic transparency with cross-functional teams at remote sites fully integrated with cross-functional teams at the primary site.***" We will aim explore these interesting angles further in our case studies. Having done some initial reading and exploring the literature available on agile project management, we now chart out our research road map.

5. Finding Agile Practitioners

The next important and challenging part of the research process is finding interested parties for interviews and observations. We searched for agile companies, groups, and organizations on the Internet and contacted them with details of our research. We also signed up on agile mailing lists and joined user groups of agile enthusiasts.

Searching through different avenues, we are planning to build a set of practitioners and companies that can be representative of the larger agile community. *We* must however acknowledge that our representative set of practitioners and companies is confined to those that exhibit their interest in participation.

5.1 Interviews

We plan to follow projects available online and hold interviews and observations at important milestones of the projects or at regular intervals mutually agreed upon by

ISSN 2348 – 7968

the interviewees. This is how we can get a complete picture of agile project management through the entire life cycle of different projects.

5.2 Data Collection

The data collected will be analyzed as per the principles of grounded theory. All materials collected will be stored in a secure and confidential way and will be destroyed at the completion of the research.

5.3 Building the Theory

As we gather more data from our case studies, we'll need to undertake in-depth analysis of all the information and follow the systematic coding procedures to arrive at theories. We will also need to revisit the data time and again in order to validate the emerging theories against the raw information and carefully avoid any biases or misinterpretations.

6 Conclusion

There is no lack of interest in the research community and the software industry for agile processes, as is evident from the research being conducted on agile practices and from the growing awareness of agile methodologies in the industry. This gives us the motivation and encouragement to explore another important issue in the agile sphere – Agile Project Management, one that demands greater understanding. Our research hopes to explore three specific issues, namely process and problems of transitioning into an agile framework, role of the agile project manager, and management of outsourced agile projects and derive theory to comment on the successful practices in all of these areas.

The agile community in India seems to be vast and very proactive. It was also encouraging to see their interest and willingness to participate in our research. Societies such as Agile Software Community of India or ASCI [12] are working to support and propagate agile practices in the India software industry and academia.

7. REFERENCES

[1] Augustine, S., Payne, B., Sencindiver, F., and Woodcock, S. 2005. Agile project management: steering from the edges. Commun. ACM 48, 12 (Dec. 2005), 85-89.

[2] http://agilemanifesto.org/ (Dec. 2007)

[3] Kent Beck, Extreme programming explained: embrace change, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1999

[4] http://extremeprogramming.org/ (Dec. 2007)

[5] http://www.scrumalliance.org/view/scrum_framework

[6] Strauss, Anselm and Glaser, Barney (1967). The Discovery of Grounded Theory: Strategies for Qualitative Research. Chicago: Adline.

[7] Strauss, Anselm and Corbin, Juliet (1990). Basics of Qualitative Research Grounded Theory Procedures and Techniques. Sage Publications.

[8] Anderson, L., Alleman, G. B., Beck, K., Blotner, J., Cunningham, W., Poppendieck, M., and Wirfs-Brock, R. 2003. Agile management - an oxymoron?: who needs managers anyway? OOPSLA '03. ACM, New York, NY, 275-277.

[9] Nerur, S., Mahapatra, R., and Mangalaraj, G. 2005. Challenges of migrating to agile methodologies. Commun. ACM 48, 5 (May. 2005), 72-78.

[10] Martin, A., Biddle, R., and Noble, J. (2004), When XP Met Outsourcing, Jutta Eckstein & Hubert Baumeister (Ed.)

[11] Sutherland, J, Viktorov, A., Blount, J, &Puntikov, N