Designing RC Story for Software Maintenance and Evolution

Jitender Choudhari
Medi-Caps Institute of Technology and Management, Indore, M.P., India
Email: jeet_159@yahoo.co.in

Dr. Ugrasen Suman
School of Computer Science and IT, Devi Ahilya University, Indore, M.P., India
Email: ugrasen123@yahoo.com

***Abstract*—Most of the software maintenance processes are mainly based on traditional software development process, which uses traditional Software Change Request (SCR) form as a requirement artifact. In this paper, we have designed a requirement artifact Request for Change (RC) story for software maintenance, which is based on extreme programming and proposed RC story format validating using case study of college portal. In order to find the impact of RC story on software maintenance, we have applied RC story on iterative maintenance life cycle using extreme programming. RC story in software maintenance process can help to resolve the problems of poor visibility of the project and lack of communication in maintenance process.**

***Index Terms*— Maintenance process, Maintainability, Reengineering, Software Engineering Process.**

I. INTRODUCTION

Extreme Programming (XP) is a software development methodology, which is intended to improve software quality and responsiveness to changing customer requirements. XP is one of the important implementation of agile philosophy. XP is a light-weight methodology for teams of approximately 10 people developing software in the face of vague or rapidly changing requirements [5, 7]. XP builds upon various existing and common sense practices and principles, but applies these to extreme levels. For example, code review, testing, designing, and refactoring are preformed continuously, rather than at dedicated phases of the software process only.

XP assumes that the development team makes use of modern development environments (Smalltalk, Java), and aims at taking maximal advantage of the resulting benefits. XP is performed in short iterations, which are grouped into larger releases. The planning process is depicted as a game in which business and development determine the scope of releases and iterations. The customer describes features via user stories, which are informal use cases that fit on an index card. The developers estimate each of the user stories. User stories are the starting point for the planning, design, implementation, and acceptance test activities conducted in XP. Software maintenance is a regular phase of XP but practitioners require a dedicated process model for legacy system reengineering and maintenance.

Software maintenance is the process of modifying a software product after delivery to correct faults or to implement new functional requirements thereby improving the performance, reliability, and adaptability for change request in the product for a modified environment [1]. Software maintenance is a complex and life long process and on the other hand extreme programming is an existing. The process of software development with its challenging practices. New requirement in software maintenance is accepted in the terms of SCR form [1, 10].

An iterative maintenance life cycle using extreme programming uses RC stories and existing software as input, performs various activities, and produces updated product [11]. The proposed model produces a modified product inheriting quality attributes such as improving maintainability of software, increasing productivity of maintenance team, reducing cost and effort of software maintenance. The existing requirement artifact such as user story is mostly designed for software development.

In this paper, we will discuss story writing in agile and requirement artifact of software maintenance in Section 2. The proposed RC story format for maintenance process is discussed in Section 3. It also describes the field specification of RC story. Validation of Proposed format through case study is discussed in Section 4. In Section 5, we state impact of RC story on various phases of iterative maintenance life cycle using extreme programming. The future work and concluding remarks are discussed in Section 6.

II. BACKGROUND

A. User Story Writing in Extreme Pogramming

User stories are actually narrative texts that describe an interaction of the user and the system, focusing on the value a user gains from the system [12]. A good user story uses the INVEST model (Independent means reduces dependencies easier to plan, Negotiable means details added via collaboration, Valuable means provides value to the customer, Estimable means it too big or too vague, Small means it can be done in less than a week by

doi:10.4304/jsw.7.5.1103-1108

the team, Testable means good acceptance criteria) [12]. A typical template of user story has 3 parts: title, description, and acceptance criteria. If the description becomes lengthy means more than will fit on an index card, customer should revisit the user story. It is likely it needs to be split into several stories. The purpose of a user story is to encourage collaboration. A user story is a have a future conversation; it is not meant to document every aspect of the work.

The customer specifies scenarios to test when a user story has been correctly implemented. A user story can have one or many acceptance tests, whatever it takes to ensure the functionality works. Acceptance tests are black box system tests. Each acceptance test represents some expected result from the system. Customers are responsible for verifying the correctness of the acceptance tests and reviewing test scores to decide which failed tests are of highest priority. Acceptance tests are also used as regression tests prior to a production release. A user story is not considered complete until it has passed its acceptance tests. Acceptance criteria enrich the understanding of the story.

B. An Example of User Story

Problem Statement: The client has requested the ability to search for education providers by provider specialty within a teacher selection site.

Requirements Statement: The provider search screen shall provide the ability to search for providers by provider specialty.

The User story for above problem can be written in the following format.

Title: Search for providers by provider specialty.

Description: As a provider search user, I need the ability to search for providers by specialty so that I can more efficiently refer students to specialists.

Acceptance criteria: The provider search mechanism has the ability to enter a specialty. The specialty search will have a list of provider specialties from which to select. Searching via the provider specialty will return a list of matching specialists or a message indicating that there are no matches.

If there are more results than can fit on one page, the system will provide the capability to view the list in pages or sections.

C. SCR Form as a Requirement Artifact in Software Maintenance

Requirement in software maintenance is accepted in terms of Software Change Request (SCR) form which contains following information such as a sequential code, requirement details to be changed. It also contains request initiator person detail, submission date, system name with version number. The information about change request in the form of configuration details (software component/ documentation component).

The SCR form also contains reason of initiating the change and type/priority of change and detailed functional and/or technical information about the change.

III. RC Story for Maintenance Process

RC story is a requirement artifact in iterative maintenance life cycle using extreme programming. It provides customer collaboration and simplifies requirement engineering process of software maintenance. The RC stories should be written by the customers for software maintenance. The proposed RC story card is usually a 3×5 inches card have two sides. Front side of RC story contains change request information details and its description, which is shown in Figure 1. Back side of RC story card contains acceptance criteria for approving change request, which is shown in Figure 2. The size of RC story can be increased for the clear visibility in standup meeting. It can be broken down into small enough components that they may be delivered in single development iteration.

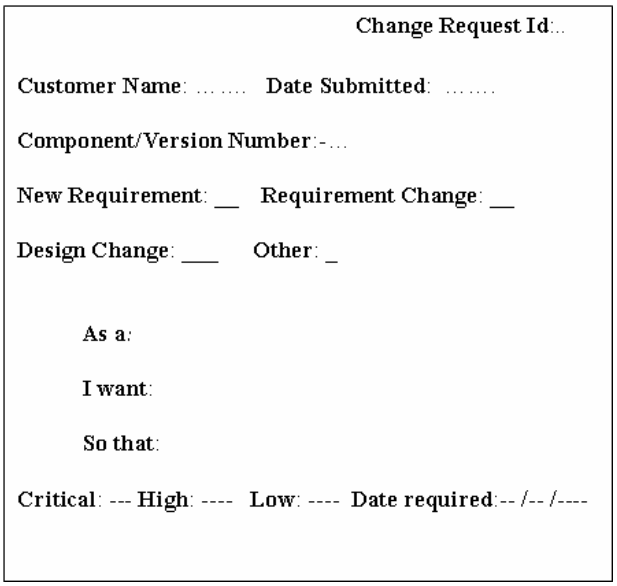

Figure 1. Front side of RC story.

A. Elements of RC Story

RC story card contains following fields for the customer to write a change request in proper manner that is understandable by a software developer and its team.

Change Request Id: A sequential number beginning with the organizational code which is allotted by customer.

Originator: Name of customer formulating the RC story.

Date Submitted: It contains date of RC story submission to Contractor.

System Name and Version Number: Name of system/Version number of software/documentation to be changed.

Change Type: Type of change being requested. Place a "X" in the appropriate area. Specify other.

New Requirement: Requirement was not identified in original specifications.

Requirement Change: Requirement needs to be altered.

Design Change: Original design needs to be changed.

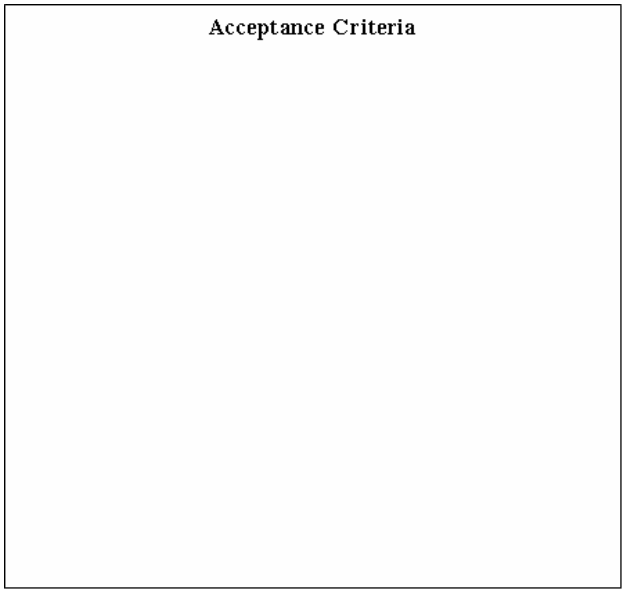

Figure 2. Back side of RC story.

Other: Indicates other than above change types. Specify in the change description area of RC story.

Change Description: Detailed functional and/or technical information about the change in following format:

As a [user role] I want [change to be made in existing software] so that [user can achieve goal by fulfilling its requirement by change].

Acceptance Criteria: The customer specifies scenarios to test when a RC story has been correctly implemented. A RC story can have one or many acceptance tests, whatever it takes to ensure the functionality works. Acceptance tests are black box system tests. Each acceptance test represents some expected result from the system. Customers are responsible for verifying the correctness of the acceptance tests and reviewing test scores to decide which failed tests are of highest priority. Acceptance tests are also used as regression tests prior to a production release. A RC story is considered to be complete if it has been passed its acceptance test criteria.

Priority: Ranking to identify action or response to an RC story. Place a "X" in the appropriate area.

Critical: A change in operational characteristics that, if not accomplished without delay, will impact system operability.

High: A change that, if not accomplished promptly (e.g., prior to the next production cycle), will impact system effectiveness.

Low: A change that can be planned, scheduled, and prioritized.

Date Required: The date the change is needed.

The frequent tribulations such as poor visibility of the project, lack of communication in maintenance process can be resolved using proposed RC story format and increases customer collaboration. SCR form used now a day is an existing way of requirement change gathering during software maintenance process. RC story is written by customer as SCR is filled by any developer side agent. RC story contains approval criteria in form of acceptance test, whereas SCR do not have any criteria to test for requirement change. RC story resembles XP user story thereby daily standup meeting is possible.

B. Applications of proposed RC Story

RC story is an effective tool for identifying and documenting future maintenance requirements. It provides significant business value in terms of planning, allowing the client to effectively collect a numerous of requirements into a concise plan that could easily be communicated to a vast and diverse audience of project stakeholders in terms of Daily Stand-up meetings. It is used in estimation of maintenance project as well as to track the velocity of project. It is also used to create work breakdown structure (WBS) in maintenance. By using RC stories, detailed discussions on what would be delivered and when could be facilitated by simply laying the cards out on the table and organizing them by release.

IV. CASE STUDY

RC story is valuable artifact in software maintenance procedure. It provides customer collaboration and simplifies requirement engineering process of software maintenance. There is a graduation college in MP India, TCPS, which provides online academic facility through a portal for its stakeholders. The teachers of college can submit marks of students using online data entry form. Generally, the teachers prepare MS-Excel sheet to maintain student evaluation record. It is observed that at the time of marks submission on college portal, the teacher has to submit data one by one into specific format containing text boxes. The same process of marks submission is repeated for all the subjects in all courses. This requirement was provided to developers at the time of initial version of the portal. But it is observed that the process is very time consuming and erroneous. There is no other way to enter data if we are having huge records. Also, it is very difficult to verify the entered data, if the teacher of technical staff has made mistakes during data entry. Therefore, there is a need to change in the existing system that could reduce time as well as prevent mistakes during data entry. Thus, we have tried to propose changes in the above process using RC stories for software maintenance. The RC stores will be helpful to acquire requests for change. The requirement statement and RC story is discussed as follows:

Requirements Statement:

The huge data of students' marks can be accepted in MS-Excel file format, which contains column entries. After uploading MS-Excel file, the process receives records one by one through text boxes, which are displayed in existing screen to save data. This requirement change process reduces time effort of data entry as well as reduces mistakes.

The User story for above problem can be written in the following format:

Title: Upload marks from MS-Excel format.

Description:
As a: User of college portal

I want: The ability to submit data in MS-Excel file format.

So that: User can submit huge data properly and accurately in a limited time.

Acceptance Criteria: The marks submission procedure has the facility to accept a document file in MS-Excel format through browsing and uploading process. The existing data entry form may enhance with browsing tool, which accepts an MS-Excel file and display all records one by one in existing text boxes and a save button, which accept one record and display next record after submission.

The above requirement statement can be represented in the form of RC story template as shown in figure 3 and Figure 4. RC story was specified to software maintenance team by end users. The next version of portal is developed with above RC story. It is observed that the college has reduced their time 85% of marks entry. Also, it is observed that around 95% mistakes during marks entry have been reduced.

Change Request Id: *s4*

Customer Name: *W. William* **Date Submitted**: *10/01/2009*

Component/Version Number:-*v1*

New Requirement: __ **Requirement Change**: _X_

Design Change: ___ **Other**: _

As a: *User of college Portal*

I want: *The ability to submit data in MS-Excel file format*

So that: *User can submit huge data in properly and accurately in limited time*

Critical: - *X* - **High**: ---- **Low**: ---- **Date required**: *12/2/2009*

Figure 3. Front side of college portal RC story.

RC story is the best artifact to articulate requests and also it include communication between end user and software developers.

Acceptance Criteria

The marks submission procedure has the facility to accept a document file in Excel format and put data in text box of portal and submit them online. The existing data entry form may enhance with browsing tool which accepts an excel file and display all records one by one in existing text boxes and a save button which accept one record and display next record after submission.

Figure 4. Back side of college portal RC story.

V. Impact of RC Story on Iterative Maintenance Life Cycle using eXtreme Programming

The software industry has many approaches for software maintenance based on traditional software development process. We have proposed an iterative maintenance life cycle using extreme programming, which is shown in Figure 5[11]. It consists of seven phases, namely; analysis, planning, change design, change implementation, regression/system testing, acceptance testing, and delivery. The proposed approach uses RC stories and old software as input and performs all the phases in the proposed iterative maintenance model. Finally, it produces a modified product inheriting quality attributes such as improving maintainability of software, increasing productivity of maintenance team, reducing cost and effort of software maintenance.

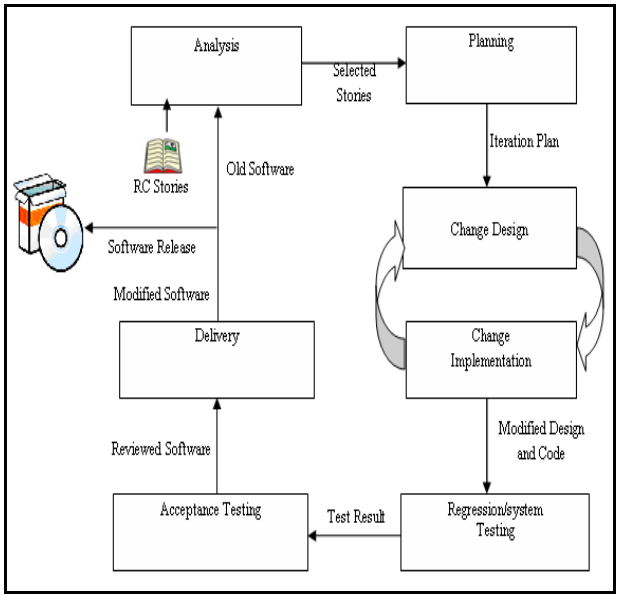

Figure 5. Iterative maintenance life cycle using eXtreme programming

The user of the existing system may request for some additional functionality or may report bugs in it. The specification for change or enhancement in the existing functionalities or bug reporting will be in the form of RC story. The impact of RC story on each phase of proposed life cycle is described in the following subsections.

The analysis phase includes activities to determine whether to accept or reject the request of RC stories submitted by user. Analysis is conducted at two levels, i.e. feasibility analysis and detailed analysis. Feasibility analysis identifies alternative solutions and assesses their impacts and costs. Whereas detailed analysis defines requirements for the modification and devises a test strategy. This phase contains key process of technical evaluation for RC story, which includes identification of software that affected (e.g., operating system software, application software) and all dependent or affected subordinate interfacing applications that may be affected (Modules/ Screens/ tables/ files/ document) by the change [10, 11]. The phase also involves impact analysis, an activity by which the software engineer assesses the level, time estimation and identifies of the lifecycle stage(s) affected by the change [10, 11]. The output of analysis phase is in form of feasibility report for RC stories, detailed analysis report, and change impact analysis report.

Planning phase mainly focus on development of strategies such as release plan, iteration plan for completion of RC stories and plan for developers meetings to improve visibility and communication between team members. The list of RC stories is finalized by the technical team as well as the product team during release plan. Typically, technical team will reveal the estimate from impact analysis and the product manager or customer will evaluate the same.

Iterative development of a system implements the requirement for change into small sets of iterations. Typically, iteration is a small release with a set of RC stories or functionalities including planning at the beginning of iteration to generate iteration plan for the execution of release plan. The developer has to complete acceptance and unit testing for each RC story completed during the iteration phase. Each of the stories that completes the maintenance phase i.e. output of the testing phase, will be confirmed and approved with the customer intervention. Failed story stories acceptance test will be analyzed in iteration plan meeting and new stories will be created to complete the same. Finally, system release plan and release date will be finalized.

The maintenance projects are observed to be complex; therefore the clarity of vision to the team members is a prerequisite for maintenance. The clarity of vision focuses on improving the visibility, better co-ordination, and better communications between team members and thereby a quality product development can be an immediate qualitative result. Stand up meeting on RC story board is a solution for the clarity of the vision, co-ordination and communication. Stand-up meeting enables the communication within the team members, i.e. developer, managers and amongst the different group working on a same project. A briefing regarding the tasks for the day, technical issues and other information is shared among the team members. The issues are discussed among team members to find a solution. Each team in the group takes fifteen to thirty minutes to review their progress. Each person takes a couple of minutes to discuss what they did in the previous day and a little about what they will do. Discussion is held and queries are resolved for further processing of the tasks by the team members.

The design change for a RC story in the system under maintenance is performed in change design phase. Output of design phase is updated design baseline. This entails using all current system and project documentation, source code and databases of existing software and the output of the analysis and planning phase. Activities include the identification of affected software modules, the modification of software module documentation, the creation of test cases for the new design, and the identification of regression tests. The architecture of maintenance system may not support contemplated change, as the concepts relevant to the change are delocalized in the code of the application domain. One approach to implement the change is by restructuring the software first and to localize the concept in one location, and then to change it. In this behavior preserving transformation process, the behavior of program is preserved whereas there is a change in the architecture. The change can be performed into two steps making the changes easier to be implemented. Firstly, to transform the architecture so that the change will be localized and then to make the change in it.

Implementation of RC stories is performed in change implementation phase. It takes the design document and source code as input and produces updated software as outcomes. Change implementation includes activities of coding and unit testing, integration of the modified code, and integration and regression testing. Change implementation is accomplished through extreme programming practices that include pair programming, coding standards, test driven development and customer collaboration. Pair programming for change implementation in extreme programming is critical to improving collaboration and greatly facilitates mentoring and improvements in engineering practice. In pair programming, both developers discuss design issues, try to find a common understanding, write test codes and produces code for given RC story. The developer holding the keyboard is entering code and another developer conducting an immediate code review, thinking about the overall design, additional test cases, and potential simplifications. The observations from research work done on individual and pair programming teams state that the work is more effective and there exist a much higher level of communication between pair rather then working independently on individual issues [7, 8]. The overall productivity also becomes high. There were on average approximately 67% more issues fixed than previous periods [4, 5, 9]. Coding standard or common standard practice used in implementation improve software quality

as well as keep the code consistency to have better maintainability and clarity.

The change implementation approach appreciates test driven development providing courage to change the system therefore, it is an important part of software maintenance process. Test driven development promotes constant improvement of test coverage quality. We also use code reviews to back up the lack of great test coverage. Customer collaboration in change implementation is a customer service team or the customer setting the priorities and generates acceptance tests for each RC story. This extreme programming practice elicits proper software maintenance needs.

Acceptance test is created based on user RC stories to ensure that a component or a system is providing expected result. Usually, customer provides a scenario i.e. use case along with the expected output of the system. The developer creates test scripts to capture the actual result from the system and confirms with expected static data given by customer.

VI. Conclusion and Future Work

Most of the requirement artifacts in software maintenance process are mainly based on traditional software development process, such as SCR form. The RC story is designed to assist the project manager of software maintenance to gather requirements. In our proposed process model the RC stories should be written by the end users for software maintenance. It provides end user collaboration and simplifies requirement engineering process of software maintenance. The frequent tribulations such as poor visibility of the project, lack of communication in maintenance process can be resolved using proposed RC story format. RC story may be used as a useful guide for the requirement gathering artifact of a maintenance project in any organization. The proposed RC story format has been used in various projects. We have shown here a college portal case study.

The Future work will focus on developing metrics and measures of software maintenance project for effort estimation based on story points. It can also incorporate value adjustment factors of software maintenance. The RC story can be tested for large projects to observe more practical results. The proposed RC story for software maintenance can further be explored for requirement engineering in maintenance of agile based product.

.

References

[1] B. P. Lientz and B. E. Swanson, “Software maintenance management”, Addison- Wesley publishing company, 1980.

[2] K. Beck, Extreme Programming Explained – Embrace Change, Pearson Education Low price Edition Asia, 2006.

[3] M. Fowler, Refactoring Improving the Design of Existing Code, Addison Wesley, 2002.

[4] A. van Deursen, “Program Comprehension Risks and Opportunities in Extreme Programming”, Proceedings of 8th Working Conference on Reverse Engineering, IEEE Computer Society, pp. 176-185, 2001.

[5] A. V. Deursen, Kuipers and Leon Moonen, “Legacy to the Extreme, pp.267-275”, 2002.

[6] H. Svensson and M. Host, “Introducing an agile process in a software maintenance and evolution organization”, Proceedings of the Ninth European Conference on Software Maintenance and Reengineering, Manchester, pp. 256-264, 2005.

[7] C. Poole and J. W. Huisman, "Using Extreme Programming in a Maintenance Environment," IEEE Software, vol. 18, pp. 42-50, 2001.

[8] C. Poole, T. Murphy, and J. W. Huisman, “Extreme maintenance”, Proceedings of the 17th IEEE International Conference on Software Maintenance, Florence, Italy, pp. 301-309, 2001.

[9] S. Shaw, “Using Agile Practices in a Maintenance Environment”, Intelliware Development Inc. 2007.

[10] K. H. Bennett and V. T. Rajlich, “Software maintenance and evolution: a roadmap”, Proceedings of the Conference on The Future of Software Engineering, ACM Press, Limerick, Ireland, pp. 73–87, 2000.

[11] J. Choudhari and U. Suman, “Iterative Maintenance Life Cycle Using eXtreme Programming”, Proceedings of the International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom), pp. 401 – 403, 2010.

[12] W. Nazzaro and C. Suscheck, “New to User Stories? “, available at http://www.scrumalliance.org/articles/169-new-to-user-stories, Apr. 2010.

Jitender Choudhari received his B.Sc. degree in Computer Science from Holkar Science College, Indore in 2003, M.Sc. degree in Computer Science in 2005 and M.Tech. degree in Computer Science (with Distinction) in 2010 from Devi Ahilya University Indore, India. He has Five years of teaching experience and pursuing his research in extreme programming area under the guidance of Dr. Ugrasen Suman Associate Professor, Devi Ahilya University. He is presently Assistant Professor at Medi-Caps Institute of Technology and Management, Indore, MP, India. He has published Three papers in national and international conferences and also guided more than 10 PG projects.

Dr. Ugrasen Suman has received his Master degree from Jabalpur University and PhD degree in Computer Science from Devi Ahilya University Indore, India. He is presently an Associate Professor at Devi Ahilya University. He is having around 10 years teaching and research experience. His areas of research are Software Engineering, Knowledge Management & Web Mining, Information System Design, Web Services, and Cloud Computing. He is currently guiding Six PhD scholars and Eight PG research scholars. He has published more than 30 research papers and also guided more than 35 PG projects. He was being a member of ACM-SIGSE. He is also working on a UGC-SAP research project of Data Mining and Software Engineering.