CHAPTER 3
DESIGN OF RC STORY

3.1 INTRODUCTION

Requirement elicitation is the initial and important phase of requirement engineering process that plays a vital role in the success of software development process. Requirements elicitation phase allows to collect requirements of a system from different stakeholders. The outcomes of this phase are validated requirements, which are the main inputs for subsequent phases of requirement engineering such as, documentation, management, and project planning. Ambiguous requirements are one of the causes of software project failures. Practices of requirements elicitation such as, interviews, questionnaires and requirement artifacts (for example, user requirements document), play an important role during elicitation of unambiguous requirements and communication between the customer and developer [13, 77]. Requirements elicitation is equally important during maintenance of a software product as it is in new development. Requirements which arise for the maintenance are known as change requirements.

The change requirements for maintenance come in the form of problem reports. An unambiguous change requirement during maintenance reduces risks and contradictions among customers and maintainers. The elicitation of change requirements is an important part of software maintenance process but there does exist very few artifacts available for elicitation and documentation of change requirements. Change requirements in software maintenance are elicited and documented using existing artifacts such as, Software Problem Report (SPR) and Software Change Request (SCR).

Users of the existing system write the problems in the form of SPRs. Each SPR contains only one problem [13]. The SPR form
contains information such as, software name, version number, priority, problem description, operating environment, and the recommended solution. Criticality and urgency of the problem are also written by the person who is reporting the problem. A problem is considered to be critical if an important feature of the software is unavailable. If the person reporting the problem provide the solution date then it is treated as urgent otherwise the software review board decides the releasing date. All SPRs are submitted to the maintenance team for diagnosis. The maintenance team prepares SCRs if they decide that a software change is required to solve the problem. The SPRs and related SCRs are discussed in the meeting of software review board. The software review board takes the decision of selection or rejection of an SPR [14, 15, 16].

The SCR form contains information such as, sequential code and requirement details to be changed. It also contains request initiator detail, submission date, and system name with version number. The information about change request is written in the form of configuration details, i.e., software component and documentation component. The SCR form also contains reasons of initiating the change, type or priority of change and detailed functional and/or technical information about the change. The SCR forms are the key inputs for maintenance planning and estimation. The existing change requirement artifacts such as, SPR and SCR are unable to handle the problems of poor visibility and lack of communication between stakeholders [18, 30]. In SPR and SCR, there is very limited involvement of the customer, which is the very crucial part of XP approach. Therefore, the existing artifacts are not suitable for the extreme programming based maintenance process.

Agile methodologies have their own techniques and artifacts for requirements elicitation. User stories are generally used for the requirement elicitation and specification in XP. A user story is a feature that is desired by the customer in the software [74]. 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 [75]. User stories are written by the customer through index cards which can also be written and submitted using web applications. Developers may provide support to the customer during story writing. The index cards are pinned on a board by development team where they work.

These small stories work as a token for conversation between development team members as well as between customer and developer [76]. A good user story follows 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 the team, Testable means good acceptance criteria) [75]. An example of user story is shown in Figure 3.1 and Figure 3.2. There are two sides of user story i.e., front side and back side. The front side describes requirement description whereas; the back side states the acceptance criteria.

Front side of user story template has following sections; namely, title, description, priority, story points, and back side contain section for acceptance criteria. In title section, title of the user story is written in two or three words. It should begin by a verb phrase of present tense in active voice. Title is written in the middle of the top section. The description section contains one to two sentences about the requirements. It is the central part of the index card. Through the priority, customers provide importance of the stories. Generally 1, 2, 3 etc., priority scheme is used; where, number 1 denotes the most important stories. The story point of the story is filled by the developer and it denotes the development time in number of days. For example, story point 2 represents 2 working days. If developers feel that a particular story would consume more than two weeks then it is divided into multiple stories.
Title: Create User Account

<table>
<thead>
<tr>
<th>Acceptance Test: CUA1</th>
<th>Priority: 1</th>
<th>Story Points: 2</th>
</tr>
</thead>
</table>

As an Administrator,

I want to be able to create User Accounts

so that I can grant users access to the system.

Figure 3.1: Front-Side of User Story

Acceptance Test: CUA1

1. If I am an administrator, I can create user accounts by entering the name, email address, mobile number and account number of the user.

2. The system notifies me that it has sent an email to the new users containing an auto-generated password.

Figure 3.2: Back-Side of User Story
Acceptance test is an important section of a user story. If story having a single acceptance test then it is written on the back side of index card; otherwise, a unique identifier of the acceptance tests is written in the acceptance test section. Acceptance test is a test case written by the customer with the help of developers. Developers create automated acceptance tests to verify the user story. Customer can specify the user stories in details using the acceptance tests. User stories and their acceptance test case(s) are used to verify the functionality of the system.

Fields of user story are not sufficient from maintenance perspectives. Therefore, there is a need of dedicated change requirement artifact for the iterative maintenance life cycle using XP [54, 96]. We have designed, Request for Change (RC) story, a change requirement artifact based on uses story of XP and SPR form of the maintenance. The proposed RC story is written by the customer for bug reporting, enhancement request or for the inclusion of new requirements.

The proposed RC story is used as base during the maintenance process. The organization of the chapter is as follows. The proposed RC story format for maintenance process is discussed in Section 3.2. The proposed format is illustrated through a case study in Section 3.3. In Section 3.4, we state impact of RC story on various phases and activities of iterative maintenance life cycle using extreme programming. Finally, Section 3.5 describes the summary of presented research.

3.2 RC STORY FOR MAINTENANCE PROCESS

RC story is a requirement artifact in the iterative maintenance life cycle using XP. It provides customer collaboration and simplifies requirement engineering process of software maintenance. RC stories should be written by the customer. The proposed RC story card is
usually a 3×5 inch card, which has two sides, i.e., front side and back side. Front side of RC story contains change request description along with the other related details. The format of front side of RC story is shown in Figure 3.3. Back side of RC story card contains acceptance criteria for approving change request or the recommended solution, which is shown in Figure 3.4. As the daily stand-up meeting is performed on the story board in the XP based maintenance approach, therefore, size of the RC story card can be increased for the clear visibility. In general, RC stories are developed in iterations, where each iteration has two week duration. If maintainers find that the size of an RC story is more than two week then it can be divided into multiple small stories. Various elements and applications of proposed RC story are described in the following subsections.

3.2.1 Elements of RC Story

RC story card contains following fields for the customer to write a change request in a proper manner that is understandable by maintainers.

*Title:* Name of the RC story written in two or three words beginning with the verb phrase of present tense.

*Software Name:* Name of the software system to be changed.

*Version Number:* Version number of the software system to be changed.

*Operating Environment:* It shows the operating environment of the software system.

*Customer Name:* Name of the customer formulating the RC story.

*Priority:* Priority indicates the importance of RC story with respect to other stories. Generally, numbers 1, 2, 3 are used for priority, where
<table>
<thead>
<tr>
<th>Title:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Name:</td>
<td>Version Number:</td>
</tr>
<tr>
<td>Customer Name:</td>
<td>Priority:</td>
</tr>
</tbody>
</table>

As

I want

so that

**Figure 3.3: Front-Side of RC Story Template**

<table>
<thead>
<tr>
<th>Acceptance Test/ Recommended Solution</th>
</tr>
</thead>
</table>

**Figure 3.4: Back-Side of RC Story Template**
number 1 is the most important RC story as compared to story number 2, and so on.

**Story Points:** Story points denote the development time in number of days. This entry of RC story is filled by the maintainers.

**Change Description:** Detailed functional and/or technical information about the expected changes are stated in the following format:

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

**Acceptance Criteria or Recommended Solution:** 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 results from the system. Customers are responsible for verifying the correctness of the acceptance tests. Acceptance tests are also used as continuous integration tests prior to the production release. An RC story is considered to be complete if it has been passed its acceptance test criteria. In this section, customer can suggest a recommended solution for the problem described in change description section.

The frequent tribulations such as, poor visibility of the project and lack of communication between maintenance team members can be resolved using proposed RC story format. The proposed RC story format also increases customer collaborations. The traditional models of software maintenance use the SPR form for requirements elicitation and the selected SPRs are documented in SCRs. RC story is written by customer, whereas SCR is filled by any developer side agent. RC story contains approval criteria in the form of acceptance test, whereas SCR does not have any criteria to test the change requirements. RC story
3.2.2 Applications of RC Story

RC story is an effective tool for identifying and documenting change requirements in maintenance. It provides significant business value in terms of planning, allowing the client to effectively collect numerous requirements into a concise way 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 the estimation of maintenance projects as well as to track the velocity of projects. Where, velocity indicates the number of story points developed in a single iteration. The proposed artifact 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.

3.3 CASE STUDY

RC story is valuable artifact in software maintenance. It provides customer collaboration and simplifies requirement engineering process of software maintenance. A university provides online academic facility through a web portal for its affiliated colleges. 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 the teacher has to submit data one by one into specific format containing text boxes at the time of marks submission on web portal instead of submission single MS-Excel file at a time which contain records of all students. 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 web portal. It is observed that the process is very time consuming, especially when we have huge records. Also, it is very difficult to
verify the entered data, if the 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 proposed changes in the above data entry process using RC stories for software maintenance. The RC stories will be helpful to acquire requests for change. The requirement statement and RC story are 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 fields, which are displayed on existing screen to save data. This requirement change process reduces time effort of data entry as well as reduces mistakes.

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

**Title:** Upload marks from MS-Excel format.

**Software Name:** university portal

**Version Number:** v1

**Operating Environment:** Vista

**Customer Name:** W. William

**Priority:** 2

**Story Points:** 4

**As a:** User of university web 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/ Recommended Solution:** The marks submission procedure allows to accept a document file in MS-Excel format through browsing and uploading process. The
existing data entry form accepts an MS-Excel file and displays all records one by one in existing text boxes and a save button, which accepts one record and displays next record after submission.

The above requirement statement can be represented in the form of RC story template as shown in Figure 3.5 and Figure 3.6. 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 university has reduced their time of marks entry. Also, it is observed that mistakes observed during marks entry have been reduced. RC story is the best artifact to articulate requests and also, it includes communication between end users and software developers.

3.4 IMPACT OF RC STORY IN XP BASED MAINTENANCE

The iterative maintenance life cycle using extreme programming is a RC story driven approach. The RC stories defined for a system are the basis for the entire maintenance process, as shown in Figure 3.7. The iterative maintenance life cycle using extreme programming consists of seven phases; namely, identification and categorization, planning, analysis, design revision, change implementation, acceptance testing, and release. It uses RC stories and old software as input and performs all the phases and produces a modified product inheriting quality attributes, such as improved maintainability, increased productivity of maintenance team, reduced cost and effort of software maintenance. In this process model, the specifications for changes or enhancements in the existing functionality or reported bugs are considered in the form of RC stories. The impact of RC stories on each phase and activities of iterative maintenance life cycle using extreme programming is discussed in the subsequent paragraphs.

The first phase of XP practices based maintenance model is the identification and categorization of change requests. It deals with
As a: User of university portal

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

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

---

**Acceptance Test/ Recommended Solution**

The marks submission procedure has the facility to accept a document file in Excel format and enter data in the text box of web portal and submit them online. The existing data entry form may be enhanced with browsing tool, which accepts an Excel file and displays all records one by one in existing text boxes and a save button, which accepts one record and displays next record after submission.
seeking change requests in the form of RC stories as an input and produces validated and categorized RC stories as output. The RC story has two parts: a description of the change requirements and user acceptance criteria. The requirements description of an RC story consists of a problem statement and its priority. This requirements description section of an RC story supports to system analysts or service engineers for identification and categorization of RC stories. As the RC stories are written by end users of the existing system, the on-site customer can easily understand and support to system analysts or service engineers for the examination of an RC story for reasonableness. RC stories are categorized in one of the maintenance categories, i.e., corrective, perfective, or adaptive maintenance. The RC story is the appropriate artifact for applying the planning game practice of XP in identification and categorization phase of the maintenance process.

Figure 3.7: RC Story Driven Iterative Maintenance Life Cycle using XP
Planning phase consists of two planning activities, i.e., release planning and iteration planning. Release and iteration planning sub-phases use RC stories along with the priority and the type of maintenance fields. Developers along with customers make decisions about the grouping of RC stories into patch release iterations. Generally, RC stories, which belong to the same category of maintenance, are developed in a single release. If the maintenance category of an RC story is *urgent repair* then it is implemented irrespective of analysis and design. As the corrective maintenance can improve the design of existing legacy code, initial iterations of maintenance process are assigned to corrective maintenance. At this stage of the maintenance life cycle, the planning poker technique is used for the estimation and RC stories are the key inputs for the estimation. In the planning poker technique, the individual RC stories are presented for the estimation. It involves experts from all disciplines of software development. The agreement between developer and customer is signed based on included stories and agreed upon release date. During iteration planning, RC stories are divided into tasks, which are assigned to the programmers.

Feasibility and detailed analysis of RC stories are performed in the analysis phase. This phase considers the impact of RC stories on rest of the modules with consideration to their security and safety implications. The analysis phase concludes with estimation of cost, size and duration for RC stories. Feasibility analysis starts with the identification of the impact of modifications. The purpose of this activity is to evaluate the scope and impact of modifications and then to decide whether or not to proceed for implementation of an RC story. A valid RC story and its relevant artifacts serve as input to this activity. The impact of modifications is analyzed roughly based on existing documents, specifications and source code with the purpose of identifying affected products, artifacts and modules that need to be modified. The impact of modification on the business process with short and long-term costs is also evaluated.
Another activity performed during feasibility analysis is the analysis of alternative solutions. It identifies and assesses the different alternatives for a given RC story. The RC story with its relevant artifacts becomes an input for this activity. During assessment, prototyping techniques can be useful in evaluating alternatives. The best alternative is selected amongst the possible solutions, which is implemented at a later stage. The on-site customer can supply support for identification and selection among alternative solutions. The identification of safety and security implications is the last activity of the feasibility analysis phase. The purpose of this activity is to identify all other aspects that can compromise success with regard to safety and security. Input for this activity is an RC story with detailed information about the system. Current safety and security issues along with the data security policy of the organizations are also used as inputs for this activity. Firstly, identification of vulnerable areas of the system is performed and their successful identification is followed by analyzing the RC story. The detailed specification of safety and security implications is performed in the later stages of the process model. Output of this activity is a list of the aspects that should be considered in detail during the later stages, i.e., detailed analysis, design, and testing.

On a positive feasibility report, a thorough analysis is required for design, which is performed during detailed analysis. The first activity of the detailed analysis phase defines firm requirements for the modification. It establishes and maintains agreement of what the system should perform after implementation of an RC story. It determines the scope of an RC story that provides a basis for the design. The RC story, analysis report and original requirements specification document are inputs for this activity. In case of corrective maintenance, the problem can be reproduced to understand the requirement. Use case diagrams can be used to represent the user requirements. This activity results into the analysis model, which represents the system or its affected units.
Another activity during detailed analysis is to identify the modification and analyze their impact. This activity assesses the impact of modifications in the system, which provides a basis for the design. Inputs for this activity are RC story, analysis model, and relevant artifacts. The impact of modification on the other components of the system is identified in this activity. This activity produces improved analysis model that provides a general idea about the intended functionality, the elements that need to be modified, and the impact of changes. Effort estimation is the last activity of the detailed analysis sub-phase. The SMEEM model, which is discussed in subsequent chapter, can be applied to calculate the volume of maintenance using RC stories. As the effort estimation of a maintenance project is highly influenced by the existing design documents and source codes, SMEEM incorporates value adjustment factors such as, documentation quality, structuredness etc., with different intensity levels to adjust the story points of RC stories.

In the design revision phase, changes are performed on the existing design artifacts without affecting the integrity of the existing system. The documentation of software modules is modified initially during the design revision phase. Inputs to this activity are an RC story and relevant affected modules. The purpose of this activity is to convert RC stories into specification documents represented in the form of modules. The specification module is represented using CRC cards. This design activity uses refactoring and metaphor practices to keep the system design simple and extensible. Outputs of this activity are design elements such as, class diagram and subsystem design, which represent different views of the system.

Another activity during the design revision phase is to create test cases for the new design including safety and security issues. The purpose of testing at this stage is to develop a test model that can be used during verification and validation of an RC story. Input for this activity consists of existing testing artifacts and revised design
artifacts from the first activity. In this process, an RC story and design models are converted into acceptance and system test cases, respectively. Outputs of this activity are test plan, test cases, test procedures, and test scripts. Planning for integration testing is the last activity of the design revision phase. At this stage, it is easier to identify affected areas that can help in the planning of integration testing activity. Defects, which are discovered earlier, existing test cases, and design information from the design model are the inputs for this activity. The test plan is established according to the modified system and then test cases are designed and implemented accordingly. Test plan, test cases and test script are the outputs of this activity.

During change implementation phase, coding, unit and integration testing are performed on the basis of different tasks of a single RC story. This phase takes updated design documents and existing source codes including production and test codes as inputs and produces updated software as an output. The pair programming, TDD, refactoring, and continuous integration practices of XP can be used during this phase. During this phase, an RC story may be divided into multiple tasks. Initially, a structure for the change implementation is prepared that represents the structure of the code change along with the consistency. The detailed design documents are inputs to this activity. It starts from the analysis phase, where modified components are already identified. Here, an implementation strategy is also prepared that describes a sequence of modifications. Planning the integration of modified module is also performed at this stage. Implementation models and integration plans are outcomes of this activity.

Performing changes in the code is the core activity of change implementation phase, which aims to develop components using design objects. The design model, implementation model, existing source code, and existing documentation are inputs for this activity. This activity can be performed by the code change approach, which employs the TDD, pair programming and refactoring practices of XP. Firstly,
unit test classes are written or existing test classes are modified according to the requested changes. In case of test class unavailability, unit test classes are written according to the structure of actual class. The program comprehension can be applied using pair programming while writing unit tests. In the initial iteration, emphasis should be given to writing unit test classes to obtain more and more coverage for existing code that helps the design of actual classes. Test and actual class creation or modification is performed using the pair programming practice. The refactoring practice can also be applied in small steps to incorporate code changes. Outputs of this activity are unit test classes along with actual classes. During this activity, a sustainable pace is maintained with the help of the 40-hour week practice of XP. The collective code ownership practice can also be followed here so that every programmer can be aware of changes in different modules of the system. The daily stand-up meeting at the RC story board can also be followed for briefings regarding the tasks for the day, technical issues and for information sharing among the team members.

The RC story driven acceptance testing is performed to ensure that the components and the system as a whole provide expected results. The acceptance test scripting activity starts from the analysis phase for all those RC stories that are planned for the implementation. Initially, test design is performed for an RC story. The RC story as an input to this activity contains acceptance criteria for change requirement. A user interface designer creates a prototype by means of an RC story’s acceptance criteria to design the tests. A test designer designs the test script on the basis of customer feedback. User approved test cases for acceptance testing is the outcome of this activity.

3.5 SUMMARY
Change requirement elicitation is the initial and important phase of software maintenance process. The existing artifacts of change requirements elicitation and documentation such as, SPR and SCR are
unable to handle problems of poor visibility and lack of communication between stakeholders. These artifacts are not suitable for extreme programming based maintenance process. The user story of XP as a change requirement artifact is not sufficient for maintenance perspective. Therefore, there is a need of dedicated change requirement elicitation artifact for the iterative maintenance life cycle using extreme programming. We have designed RC story format, a change requirement artifact, based on user story of extreme programming and SPR form of maintenance.

The RC story improves customer collaboration and simplifies requirement engineering process of software maintenance. The frequent tribulations such as, poor visibility of the project, and 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. It is illustrated with the help of a case study of university web portal.