Determination of Risk During Requirement Engineering Process

1 Saima Amber, 2 Narmeen Shawoo, 3 Saira Begum
Jinnah University for Women
5-C Nazimabad Karachi

1 amberashfaq@yahoo.com, 2 nshawoo@gmail.com, 3 saira_begum@hotmail.com

ABSTRACT

Software developments is an investment activity that depends on how each phase in SDLC performs its job. Each phase has some unseen defects that should be acknowledged and removed earlier with the intention that other phases are saved from their awful consequences. These defects are caused by ignorance of risks which are related to each phase. These risks are handled by Risk management that is considered during design phase. In this phase counter measurements are complex. Research illustrate that the cost of errors fixing increases 50 times as to fixing errors in requirement analysis phase. This paper suggests a framework in which risk management is executed within Requirement Engineering (RE) process. Three models of risk management are considered, these models identify risky functional requirements. These models are compared on the basis of risk identification methodologies. A new model is derived which is based on UML oriented approach for modeling and reasoning about risk during the requirements analysis process.

Keywords: Requirement Engineering, Risks, UML

1. INTRODUCTION

Software requirements gathering, the initial step of software development cycle, needs a clear and unambiguous process for gathering requirements; however it is difficult to filter usable information from different sources. Software requirements are categories into many groups of business requirements, functional requirements, non functional requirements, performance requirements and security requirements.

Functional requirements represent how product works. Any confusion in functional requirements will have an effect on product’s functionality therefore inconsistency on functional requirements should be removed at initial stage. This paper performs risk management’s activities at requirement engineering level. The purpose of these activities is to find risk associated to functional requirements. In designing phase or later stages it is most probably to discover such type of errors which are related to starting stage [1]. Fixing these errors takes time thus cost of re-work becomes too high. This situation put a bad impact in front of customers. Therefore risk factor is calculated which is determined how much attention and time is given to implement this requirements. A survey is also performed about process of requirement engineering. This survey covers software houses of Karachi, Pakistan.

This survey helps to understand the process of requirements engineering with the activities of risk management.

2. PROBLEM ANALYSIS

Advance technologies and technical expertise enhance skills of developers and organization get success to achieve their goals. It is observed that in spite of advance technologies large numbers of complex IT projects have failed to deliver on-time and on-budget. Most organizations expect to cancel at least one software project per year at enormous cost. Approximately 66% of projects fail to meet business objective [1], [14].

These problems are faced by many organizations, unable to understand stakeholder’s requirements or may not define the scope of projects properly and might not consider organization’s objectives and benefits. These organizations have no such plans that may help them to get desired goals. These problems are created because

1) Requirement phase is not accomplished properly.
2) Efforts that are applied on coding phase are not estimated properly.
3) Testing efforts are not properly estimated.
4) Team’s skill is under/over estimated.
5) Due to customer modifications, it becomes difficult to deliver software project on time which results as extend in project delivery dates.

These reasons represent that there is a need to perform Risk management from the start of project. Risks are introduced and analyzed along the stakeholders’ goals and counter measures are introduced as part of the system’s requirements [4].

C. J. Davis, Fuller, Tremblay, & Berndt found accurately capturing system requirements is the major factor in the failure of 90% of large software projects,” echoing earlier work by Lindquist who concluded “poor requirements management can be attributed to 71 percent of software projects that fail; greater than bad technology, missed deadlines, and change management issues”. The cost
of this failure is enormous. Another study found that failed or abandoned systems cost $100 Billion in the USA[9]. According to the [10], The Standish Group’s CHAOS survey of over 350 organizations and 8000 projects produced the following results:

1. 16% delivered within budget and schedule
2. 31% cancelled before completion
3. 53% overrun in budget or schedule
   - 89% average budget overrun
   - 122% average schedule overrun
   - 61% average of originally specified content

3. PROBLEM RESOLUTION

The primary focus of this research is Software Requirements Specification (SRS) which is developed during the RE phase. At this point functional requirements are listed and find out those requirements which have high risk. Functional requirements are converted in UML scenario. A model is derived which calculate risk of these requirements. Calculated risk determines how much attention is required to fulfill this requirement.

Countermeasures are elaborated and then accommodated as a refinement of the design, when a limited number of changes are still possible and they may introduce the problem of revisiting the initial requirements [11].

4.1 Risk Management

Risk is defined as a combination of two factors: probability of malfunctioning (failure) and the consequence of malfunctioning (severity) [4].

\[ \text{Risk} = \text{Probability of undesired event} \times \text{Severity of undesired event} \]

Risk Management is defined as the activity that identifies a risk, assesses the risk and defines the policies or strategies to lessen the risk [12].

Most of the current risk assessment methodologies can only be used to estimate in the later stages of the software life cycle, typically from design models or code. As a result theses methodologies can identify risks but have limited capability in preventing these risks from occurring.

Software requirement’s risk addresses the possibility of suffering a loss of any functional or non-functional requirement of the software system. It is more feasible to make changes to the software system under development in the early stages of the software development cycle [2].

4.2 Risk Management during Requirements Engineering Process

The probability of occurrence of Low Level Risks is more than that of High level and Medium Level Risks. 38% of overall project Risks are Low Level Risks, 31% are Medium Level Risks and 24% are High Level Risks [Fig. 2].

In software engineering, risk is usually considered and analyzed during, or even after, the system design.
Extent are less in number even than Medium Level Risks. On the later phases the number of risks of high level extent decreases and the number of risks having low level risks increases. The major affecting risks lie in the earlier phases that can affect the cost and schedule of the project. It can be noted from the table and graph that the ratio of Medium Risks is high in the first phase and in the Testing phase and Degree of Low Level Risks are high in later phases. High Level Risks are found mostly in the early phases of software development [11].

Another reason for proper requirement engineering need is that the cost of error correction in the requirement phase is low than the cost of error correction in the later phases. If the errors are not corrected in the requirement engineering phase, the project can be cancelled due to time and budget overrun. Bob Glass’s in his book “Software runaways” says that improper requirement engineering management has often proved to be the biggest cause of the software projects disasters.

5. MODELS OF RISK ANALYSIS WITHIN RE CONTEXT

5.1 Goal- driven software development risk management modeling (GSRM)

The Goal- driven software development risk management modeling (GSRM) framework is established to assess, reason, control, and trace software development risk. This model consists of four layers of Goal layer, Risk obstacle layer, Assessment layer and Mitigation layer. The main focus is to integrate risk management activities within RE phase so that risks are identified and controlled from the early stage. As per in figure 2[1].

The reasons for considering the approach within RE are that poor requirements are one of the main causes of the project failure [13], cost relates to fix errors during the testing phase is fifty times more than the cost of fixing in RE phase [14].

5.2 Risk Tree Assessment

Risk tree possesses many events. The lowest level events are called primary events. In the middle, intermediate events exist and the highest level event is called the top event. All the events are connected in a tree by gates that show the relationship between successive levels of the tree. Risk tree depicts the logical interrelationships of the basic events that lead to the top events.

The risk tree is represented by three types of graphic symbols: logic gates, events and transfer triangles. The most common symbols and basic components used for risk tree construction and analysis are shown in Table 1.

<table>
<thead>
<tr>
<th>Graphic symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Δ</td>
<td>Transfer</td>
</tr>
<tr>
<td>□</td>
<td>AND gate</td>
</tr>
<tr>
<td>□</td>
<td>OR gate</td>
</tr>
<tr>
<td></td>
<td>Intermediate event</td>
</tr>
<tr>
<td></td>
<td>Elementary basic event</td>
</tr>
</tbody>
</table>

**Methodology**

The probabilistic assessment of risk tree consists of calculating the probability of a top event starting from the probabilities of the primary events. This can be done directly when the risk tree does not have any repeated events. When the risk tree possesses repeated events, for exact calculations, minimal sets of the risk tree should be passed.

The risk tree model can be converted into a mathematical model to compute the probabilities. The example of "AND" and "OR" operators are represented in Figure B2 and B3.
Case1: When input events are independent

Equation for AND gate

\[ P = \sum p_i \]

Equivalent probability equation for AND gate is also represented as

\[ P(S) = P(A) \cdot P(B) \ldots \ldots \ldots (1) \]

Equation for OR gate

\[ P = 1 - \sum (1 - p_i) \]

Equivalent probability equation for OR gate is also represented as

\[ P(S) = P(A) + P(B) - P(A) \cdot P(B) \]

\[ = 1 - (1 - P(A))(1 - P(B)) \ldots (2) \]

In above equations, \( n \) is the number of input events to the gates, \( p_i \) is the probability of failure of the input events and it is assumed that the input events are independent.

Case2: When input events are dependent

For and gates

\[ S = A \cap B \]

\[ P(S) = P(A) \cdot P(B|A) = P(B) \cdot P(A|B) \]

And for OR gate is:

\[ S = A \cup B \]

\[ P(S) = P(A) + P(B) - P(A \cap B) + P(A) + P(B) - P(A) \cdot P(B) \]

Some risk events are dependent on each other. If one of them does not occur, the other one does not occur either. These risks can be shown with AND gates [3]. Risk tree structures can be shown in figure 3 of [3].

C  Software Requirement Risk Assessment through UML

This model assess risk at requirement level using Unified Modeling Language specification of the software at early development stages.

Methodology

This methodology has two components.

1. UML specification, which help to find complexity of scenario based specification.
2. Functional Failure mode, which help to find severity of failure scenario mode.

The risk factor of a scenario in a failure mode is obtained by combing the complexity of failure mode in that scenario and the severity of the failure.

UML use case model: It describes system’s functional requirements in terms of use cases. Use cases are realized in scenario. Each scenario is modeled by sequence diagram, which shows the interaction of various components in that scenario.

- The probability of malfunctioning of system is proportional to its complexity.
- The complexity is calculated at scenario level, by using UML sequence diagram.
- Function Failure Analysis (FFA)

Cyclomatic complexity is define as

\[ CC = E - N + 2 \]

where

- \( CC \) = Cyclomatic complexity
- \( E \) = the number of edges of the graph
- \( N \) = the number of nodes of the graph

D  An Extension in GSRM model (Derived Model)

This model has a layer based approach and consists of four layers. This model is derived from two models, Goal-driven software development risk management modeling (GSRM) framework and Software Requirement Risk Assessment through UML.

This model is applied on Software requirement specification. Function requirements are dig out from SRS
and risk factors are found out from those requirements which have high risks. After that I apply risk assessment technique to calculate risk. Calculated risks determine how much attention is required to implement this requirement. This model has four parts.

1. Scenario based Functional requirements
2. Identification of Risky requirement
3. Risk Assessment
4. Mitigation

Part 1: Scenario based Functional requirements

In this part, functional requirements are transformed into UML use case scenario. A number of SRS contains functional requirements in form of use case scenario. A functional requirement can consist of multiple case scenarios.

Example:

Use Case 1: Login

Primary actor: User

Pre-condition: internet connection available

Main scenario:

1. Start the application. User prompted for login and password.
2. User gives the login and password.
3. System does the authentication.
4. Main scenario is displayed.

Alternative scenario:

5. Authorization fails.
6. Prompt the user that he typed the wrong password.

Part 2: Identification of risky requirements

This part identifies the potential risk factors as a problem from the early development components that negatively influence the goal. Same problem upset more than one goal such as misinformation, human errors and requirement errors. Generally, these risk identification is done through checklist, questionnaires and brainstorming session with the stake holders. Here we follow a set of questionnaires based on the early development components as well as brainstorming session to identify these risks. The identified risks are analysis further through the assessment part.

Part 3: Risk Assessment

This part consists of three components.

- McCabe’s cyclomatic complexity
- UML’s Sequence diagram
- Functional failure Mode

a. McCabe’s cyclomatic complexity

Cyclomatic complexity is computed using the control flow graph of the program (we convert use case scenario into control flow graph). The nodes of the graph correspond to indivisible groups of commands of a program, and a directed edge connects two nodes if the second command might be executed immediately after the first command. Cyclomatic complexity may also be applied to individual functions, modules, methods or classes within a program. Mathematically, the cyclomatic complexity of a structured program is defined with reference to a directed graph containing the basic blocks of the program, with an edge between two basic nodes if control may pass from the first to the second (the control flow graph of the program). The complexity is then defined as

\[ CC = E - N + 2 \]

Where

- \( CC \) = cyclomatic complexity
- \( E \) = the number of edges of the graph

Table 2: Cyclometic complexities of control structure

<table>
<thead>
<tr>
<th>Control Structure</th>
<th>Cyclometric.C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>1</td>
</tr>
<tr>
<td>If than else</td>
<td>2</td>
</tr>
<tr>
<td>While</td>
<td>2</td>
</tr>
</tbody>
</table>

Role of model: This model mapped all high risk functional requirements into control flow graph so that cyclometric complexity will find.

b. UML’s Sequence diagrams

UML’s Sequence diagrams are used to represent or model the flow of messages, events and actions between
the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behavior which can be difficult to extract from static diagrams or specifications. Actors, Objects, Units and Separators are the Components of Sequence Diagram.

Role of model: In this thesis, high risk functional requirement is mapped into use case and then this use case convert into UML sequence diagram so that number of message at a failure mode is/are obtained.

c. Functional failure Mode

A failure modes and effects analysis (FMEA), is a procedure for analysis of potential failure modes within a system for classification by the severity and likelihood of the failures. A successful FMEA activity helps a team to identify potential failure modes based on past experience with similar products or processes, enabling the team to design those failures out of the system with the minimum of effort and resource expenditure, thereby reducing development time and costs. Failure modes are any errors or defects in a process, design, or item, especially those that affect the customer, and can be potential or actual. Effects analysis refers to studying the consequences of those failures.

Severity is the consequences of a failure mode. Severity considers the worst potential consequence of a failure. Failure mode is categorized into one of the four classes, based on its severity.

Table 3: Classess of severity

<table>
<thead>
<tr>
<th>Classes</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minor</td>
<td>0----0.25</td>
</tr>
<tr>
<td>marginal</td>
<td>&lt;0.5</td>
</tr>
<tr>
<td>Critical</td>
<td>&lt;0.75</td>
</tr>
<tr>
<td>Catastrophic</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Role of model: A failure mode is based on how the scenario fails during execution and how severe it is. This is performed by domain expert. Different forms are available for analysis of FFM. A scenario could fail at several places but we consider the software failure not hardware failure.

Working Formula of Derive model

| Risk factor is calculated as the combine factor of severity and complexity. |
|-----------------------------|------------------------|
| $R_f = \text{complexity} \times \text{severity}$ |

| Complexity of the scenario = Cyclomatic complexity * Number of message |
|-----------------------------|------------------------|
| Severity (S) =0 < S <1 |
| Values of Risk factor (Rf) = 0 < S <1 |

Part 4: Risk Mitigation

This is the last part, where the possible control actions are identified and selects the most suitable ones to mitigate the risk and thereby to attain the goal. Once the goals, risk factors and risk events are identified and analyzed by goal, risk -obstacle and assessment layer, then this model focuses to implement the suitable cost effective counter measure as early as possible.

6. CONCLUSIONS

Development of Software suffers from cost overrun, schedule delay and undesired user needs. Project’s cost is determined in terms of time, money and efforts. Change in any factor will dominate the other one. It must exceed budget and efforts that could be used in another project. According to the researches, improper process of the requirement engineering is the immense source of this problem. This thesis helps to find risky functional requirements and calculate their risks. UML performs a vital role to find risk through the multiplication of probability of failure scenario and its severity. In this thesis, a new model is derived which provide an easy and proper way to assess risk factor. McCabe’s cyclomatic complexity process is used to find complexity of use case scenario. This process mapped all high risk functional requirements into control flow graph. UML sequence diagrams are used to model the flow of messages, events and actions between the objects or components of functional requirements.

A project’s vision is the ultimate condition that defines working state of project and scope define the boundary of project. 82.35% of the software houses prepare project’s vision as a part of their requirements document they gather information from the users of the system.

Concluded facts show that 52.94% organization convert complex functional requirements in use cases scenario.58.82% organizations use different analysis models for requirements analysis that gives an accurate set of requirements. 47.1% Software organizations apply risk
management in requirement engineering phase therefore risk factor is assessed of risky requirements. 88.23% of them don’t use any risk analysis techniques such as Tree assessment structure for risk calculation and Goal driven Software Risk Management.

47.2% of the software development industry is embracing this modeling language for requirement analysis and the subsequent phases of software development lifecycle. 53% of organizations find out risk factor that determines how much attention and time is applied to implement this requirement.

52.059% of the organizations answer that they track a proper process of requirements engineering but 43.235% organizations don’t pursue any process and remaining of them don’t response over it.

REFERENCES


[5] Requirements Engineering: A Roadmap by Bashar Nuseibeh Department of Computing Department of Computer Science, Imperial College London SW7 2BZ and Steve Easterbrook, University of Toronto, 6 King’s College Road, U.K. Toronto, Ontario M5S 3H5, Canada Email: ban@doc.ic.ac.uk

[6] Research direction in RE by Betty H.C. Cheng Michigan State University 3115 Engineering Building East Lansing, Michigan 48824 USA chengb@cse.msu.edu, Joanne M. Atlee University of Waterloo 200 University Ave. West Waterloo, Ontario N2L 3G1 CANADA jmatlee@uwaterloo.ca

[7] Risk Analysis as part of the Requirements Engineering Process by Yudistira Asnar (yudis.asnar@dit.unitn.it), Paolo Giorgini (paolo.giorgini@dit.unitn.it)

[8] Software Economics: A Roadmap by Berry Boehm

[9] Requirements Elicitation – What’s Missing? by Bill Davey RMIT University, Melbourne, Australia, Bill.Davey@rmit.edu.au and Chris Cope Latrobe University, Bendigo Australia.

[10] Achievements and Challenges in Software Resource Estimation Barry W. Boehm, University of Southern California 941 W. 37th Place Los Angeles, CA 90089 +1 (213) 740-5703 boehm@usc.edu Ricardo Valerdi Massachusetts Institute of Technology 77 Vassar Street Cambridge, MA 02139 +1 (617) 253-8583.


[15] Software Requirement by Karl E. Wiegers


[18] Measuring the Complexity of a UML Component Specification by Sajjad Mahmood, Department of Computer Science, La Trobe University Melbourne, Australia and Richard Lai Department of Computer Science, La Trobe University, Melbourne, Australia.