Method for Risk Assessment at Software Design Level by Applying UML

Mohd Shamsh Alam¹, Mohd Arif², Mohd Sadim³

Abstract—By doing risk assessments, different kinds of risks, involved in the entire software development life cycle are recognized and measured. It improves the software quality and minimizes the cost and resources if measured in the early stages of SDLC. A method for the risk analysis at design level by using UML is considered here to calculate the risk factor we used state chart and sequence diagram.

Index Terms—Risk Analysis, State Diagram, Sequence Diagram, UML.

I. INTRODUCTION

Risk assessment is somehow related with software testing. This is a method of validating and verifying a software product that weather meets the user’s requirements or not. Testing and risk analysis basically starts right from the requirement gathering stage of a software project and is mainly done at the time during the development process. Traditionally testing is done after the requirement is freeze and coding process has been completed. If any error found because of fault requirement analysis then change in design and coding is required. Therefore testing after coding is more expensive. This can be minimized if risk assessment and testing is done at early stage. The risk management defined by Barry Boem [01] has two stages, risk assessment and risk control. Identification of risk involved, analysis and prioritization that comes under risk assessment. Higher risk should be tested as early as possible and frequently after categorization on the basis of risk attributes. Since risks adversely impinge on the project outputs.

Risk can be categorized into many like project, technical, and business risk [02], in addition to this, it may also be like performance risk, budget risk, and schedule risk [03]. So we can say that recognition of all the risk is tough task and assessment at early stage by means of some method is good in terms of process, business and finance. Here we are trying to analyze the risks involved in software project at design level using UML behavioral diagrams of a project. Since only UML diagrams are available at the initial stage of a software project.

II. BASIC CONCEPTS OF SOFTWARE ARCHITECTURE AND UML DIAGRAMS

Software architecture is understood as the complete outline of a system. It describes the in and out of a system by components and connectors for its design and evolution [IEEE 1471]. A component may be class, an object, functions or collection of related functions. A connector describes the relationship between components.

UML has many types of diagrams which is divided into two categories, Structural and Behavioral diagram. Class, component, composite, structure, deployment, object, package and profile diagram comes under structural diagram since these describes the structure and used in documenting as model. Behavioral diagram describes the functionality of the systems. The UML diagrams that we used in this approach are as

- Use case Diagram
- State Chart Diagram
- Sequence Diagram
- Interaction diagram

CCFG (Concurrent Control Flow Graph) is used when the simultaneous control flows in a program has to represent [04] while for sequential control flow CFA [05] methods are used. The CCFG is required to explain the control flow of a sequence diagram. There is some sequence diagram which calls other sequence diagrams.

III. REVIEW OF RELATED WORK

Software quality can be measured by maintainability, reusability, understandability of a design [06]. Yacoubet al. [07] introduces a metrics to quantify the design worth at early development stage. Almendors-Jimenez and Iribarne [08] also proposed a method for use case by means of sequence diagram. Their objective was to provide semantics of use case relationships and described how to map each relationship. We followed their approach. Garousi et al. [04] also proposed a method for the analysis of control flow of UML diagrams.
Usually at coding stage the control flow analysis is done but this approach can also be used at the early stage of SDLC. As per the experts knowledge different risk assessment methods has been given at requirement phase [09][10]. It does not take into account any structural and behavioral dependencies between interacting objects of the system. Cortesselaet al.[11] also given a risk assessment method using UML diagrams. It estimates the performance failure probability which helps to resolve risky scenarios along with uncertain software components.

IV. ANALYSIS OF RISK AT DESIGN LEVEL

Information Flow Complexity: It is used to measure the program complexity and defined as,

\[ IF = \text{length} \times (\text{fan_in} \times \text{fan_out})^2 \]

Where length is the number of lines of code, fan_in describes the number of components called this component and the parameters passed to it where as fan_out describes the number of components called by this component and the parameters passed from it to others. This IF can also be useful at design level where length is the cyclomatic complexity.

Cyclomatic Complexity: Introduced by McCabe in the year 1976, that measures the program complexity by measuring the independent paths in a program. CC is measured from control flow graph by using the following formula,

\[ CC = \text{Number of edges (e)} - \text{Number of nodes (n)} + 2. \]

V. CASE STUDY

Library management system is taken as case study in which four components user, librarian, member record (MR), books are considered. User component can issue, renew, return, and reserved book. Librarian manages user, book and member record. Member record keeps information about members and their request. Six connectors have been used for the communication between components. The use case diagram is shown in fig.1.
Second International Conference on Advanced Developments in Engineering and Technology (ICADET-15), INDIA.

In the above diagram each usecase shows some high-level functionality and one or more scenarios. Each scenario is realized by using sequence diagram. For each scenario we calculate the risk factors of components and connectors. The risk factors are calculated as a product of the dynamic complexity/coupling and the severity value. Then domain expert assign this value to the component/connector. Then CCFG is constructed for each scenario as per the sequence diagram to estimate the risk factor.

For each component \( i \) involved in scenario \( s \) the risk factor \( rsf_i \) is calculated as,

\[
rsf_i^s = DCm_i^s \times svrt_i^s
\]

Where \( DCm_i^s \) \((0 \leq DCm_i^s < 1)\) is the normalized dynamic complexity and \( svrt_i^s \) \((0 \leq svrt_i^s < 1)\) is the severity level of component \( i \) in that particular scenario \( s \).

For each connector between two components \( i \) and \( j \), scenario \( x \), the risk \( rsf \) is calculated as,

\[
rsf_{ij}^x = DCn_{ij}^x \times svrt_{ij}^x
\]

Where \( DCn_{ij}^x \) is normalized dynamic coupling and \( svrt_{ij}^x \) is the severity level of connectors between two components \( i \) and \( j \).

Then cyclomatic complexity of a component \( i \) in scenario \( s \) is calculated as,

\[
CC_i^s = q_i^s - p_i^s + 2
\]

And when \( fan_{in} \) and \( fan_{out} \) is known then the dynamic complexity of component \( i \) in scenario \( s \) is defined as,

\[
dcm_i^s = CC_i^s \times (fan_{in}^s \times fan_{out}^s)^2
\]

Now for a component \( i \) in the scenario \( x \), the normalized dynamic complexity \( DCm_i^x \) is estimated as,

\[
DCm_i^x = dcm_i^x / \sum_{k \in S_x} dcm_i^k
\]

Now for calculating the scenario level risk factor using CCFG. The risk factor \( M_i \) of each scenario

\[
M_i = \sum_{i=1}^{n} R_i = \sum_{i=1}^{n} (R1 + R2 + R3 + ...)
\]

VI. CONCLUSION

In this a process of early risk assessment using UML diagram like use case and sequence diagram is given. Initially risk factor of components and connectors is calculated using UML diagram then CCFG is used for scenario level risk estimation.

REFERENCE


Mohd Shamsh Alam, M Tech Scholar, Department of Computer Science & Engineering, Al Falah School of Engineering & Technology, Dhauj, Faridabad, Haryana, India, +91 99999 52558, (e-mail: mshamsalam@gmail.com)

Mohd Arif, M Tech Scholar, Department of Computer Science & Engineering, Al Falah School of Engineering & Technology, Dhauj, Faridabad, Haryana, India, +91 9910834004, (e-mail: mohd.arif.jmi@gmail.com).

Mohd Sadim, Assistant Professor, Department of Computer Science & Engineering, Al Falah School of Engineering & Technology, Dhauj, Faridabad, Haryana, India, +91 9891650087, (e-mail: sadim4@gmail.com).