Available online at www.elixirpublishers.com (Elixir International Journal)

Computer Science and Engineering

Elixir Comp. Sci. & Engg. 44 (2012) 7188-7191

ISSN: 2229-712X

Risk assessment visualization model for software development

Rasmita Dash and Rajashree Dash

Department of School of Computer Science and Engineering, Siksha O Anusandhan University, Bhubaneswar, Orissa, India.

ARTICLE INFO

Article history:

Received: 4 January 2012;
Received in revised form:
17 February 2012;
Accepted: 3 March 2012;

Keywords

Risk management,
Risk assessment,
Development Model,
Visualization tool.

ABSTRACT

Software risk management is the practice of assessing risks that affects the software development projects, process or products. Most software development confronts great risks and risks might occur in the whole development process. This paper explores the different risks involved in various phases of the software development process and defines mitigation steps after analyzing risks. The objective of this research is to construct a visualization tool for software risk assessment for all phases in software development process.

Introduction

The main challenge in dealing with risk management is to provide the user with meaningful visual tool [4]. Although, the current decision makings and risk assessment methodologies include visualizations, not all phases in software development are included [1].

As a result the risk assessment from one phase to another can be significantly complicated. A significant way to show the relationship of risk assessment from one to another is to integrate all phases in software development. By doing this, software developers can discover the risks earlier and help them to determine the execution of their risk plan.

With the knowledge of risk management, software developers will be ready to define a standard process and methods when developing software.

In this paper, we will discuss on the traditional software development process model - "Waterfall Model". The waterfall model has a clear objective where each process takes the input from the previous step.

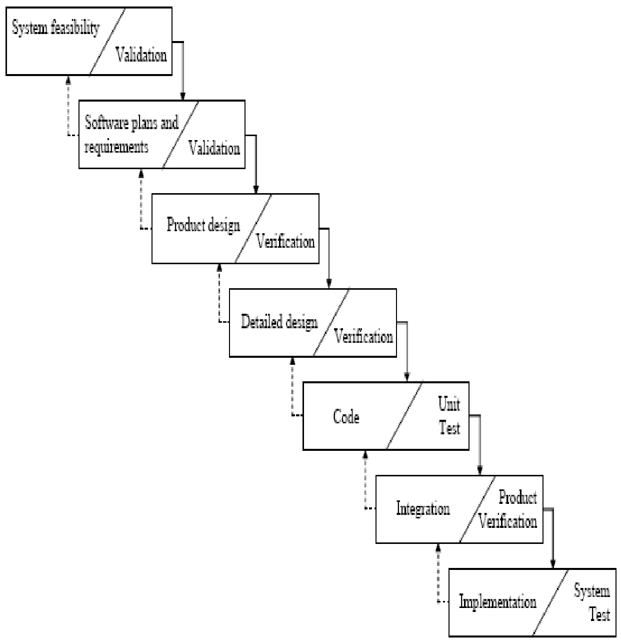
We have considered few risks that can be arise in different phases of software development. And we have discussed their effects on the project parameters. Finally we have taken a visualization tool that is based on this traditional software development process model.

Methodology

In order to produce good software, it is important to take risk into account. The basic problem is that in waterfall model the project risk remains higher throughout the software process development.

As stated by [4], others risk analysis techniques and tools such as decision trees, causal analysis, gap analysis, Pareto analysis, sensitivity analysis are very well defined but used only limited visualization tools.

Ease of use is required to construct visualization tools which represent the interconnected of stages in software development projects.



Firstly "Waterfall" model was selected for the analyses of risks in the development phases. Here risks involving in the software development phases were identified. The identified risks are analyzed for their impact on cost, schedule, repute of the organization, some other phases and some other factors. Further the probability of occurrence of these risks were also identified and analyzed in Table: 1.

With the visualization standards which consists of all phases in this traditional development approach, this visualization tool can be a blueprint which applicable to commonly problems in software development projects. A visualization tool should be thought of as a user control task and not as report component [8]. In actual software development, a project seldom reaches a situation where the developers cannot control the software development, or need considerable help from a senior manager in order to get the

Tele:
E-mail addresses: rasmita02@yahoo.co.in

project back on track. The more complex a system is, the more the we must be able to predict early on, in the development process, those components of the software system that are likely to have a high fault rate [2]. As been mentioned earlier, there is no specific tool that can manage risk for the entire development. The risk assessment are usually done in particular phase depend on the purpose and the type of software produced

Proposed Tool

The objective of this research is to construct a visualization tool for risk assessment for all phases in software development. Risk Assessment Visualization Model (RAVM) is divided into 5 modules that cover the basic phases according to Waterfall Model. There are planning, requirement analysis, design, coding & implementation and integration modules. The modules of RAVM are shown in Figure 3. Next, we present the proposed risk assessment methods used in RAVM modules

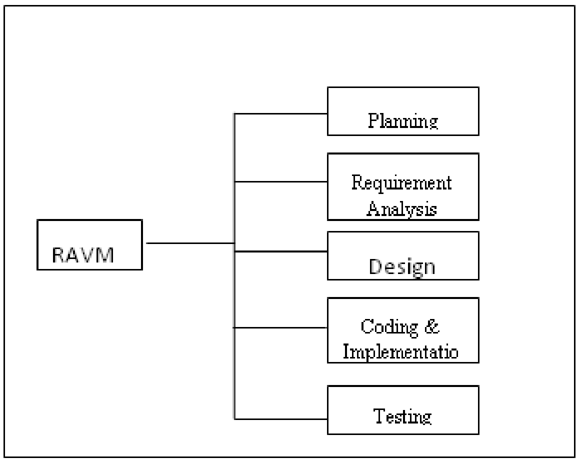


Fig.2. Risk Assessment Visualisation Model

To begin with, software was developed based on the project development description. As the first modules in RAVM, the initial phase of the development is planning. This phase is planned at general level and the most detailed level will be covered in next phase.

RAVM in Planning Phase

In RAVM, a risk baseline is determined in planning modules so that risk assessed is related to work and not people. RAVM will use a method of formal risk assessment where users need to answer questionnaires based on a risk taxonomy checklist. It is most useful if the project managers conduct the risk assessment at the early phase of the projects. The inputs from questionnaires will be reserved in a risk repository for future reference. The risk repository contains list of all possible risk in different levels of software development. Once completed the interview-based risk assessment, RAVM will evaluate identified risks and categorize the risks. The evaluation will rate at scale

RAVM in Requirement Analysis Phase

In requirement module of RAVM, the goal for this phase is to understand the requirements document and stabilize the requirements as soon as possible. The primary risks are that wrong software will be developed if the requirements are not testable and fixed. As a result, the software projects will not complete as schedule. These are measured in terms of following attributes

- Understandability
- Ambiguity
- Completeness
- Consistency
- Volatility

For requirement risks, RAVM will indicate the problems based on the requirement metrics and attributes used. For example the higher the ambiguity terms, the higher the project is at risk. Then it determines how these risks affect the project parameters based on their sensitivity.

As for requirement changes, RAVT will still capable to assess the risk. The problem is the later the changes, the more resources need to fix them.

RAVM in Design Phase

As for design, it can be depicted by showing the data flow, control flow and the functional or non-functional structures. In RAVM, we proposed to use the following matrices that decides the sensitivity of the risks

- Understandabilitty
- ambiguity
- Completeness
- Cohesion and Coupling
- Fan- in/ Fan-out

By using this attributes, ambiguity can be measured by the amount of the detail, completeness is by number of modules. Fan-in is the count of calls to a given module and fan-out is the count of calls from a given module.

RAVM in Coding Phase

To evaluate risk in this phase a combination of static and dynamic model can be developed based on the source code. There are many approaches for static testing like review, walkthrough, inspection where as actually executing code with a given test case is called as dynamic testing.

RAVM in Integration Phase

As testing is the last part of the project, it's always under pressure and time constraint. To save time and money it is required to prioritize the testing work. How will prioritize testing work? For this more important and less important testing work needs to be identified. How will you decide which work is more or less important? Here comes need of risk-based testing. So RAVM will gather information from risk repository and evaluate based on the weight-age determined by users

Conclusion

The purpose of this research is to identify the different kinds of risk that can be raised in different phases of software development and to construct a visualization tool for risk assessment in software development life cycle. But not every risk factor is fully controllable and several risk exceed the authority of software manager. Nonetheless, risk analysis and assessment with a visualization tool is quite effective in the identification of significant problems. So it can provide enormous advantages to an organization by cutting down on costs and ensuring proper delivery as per schedule.

References:

1.Abdullah Tahir, Mateen Ahmed, Sattar. Ahsan Raza, Mustafa Tasleem,"Risk Analysis of Various Phases of Software Development", *European Journal of Scientific Research*, vol.40,no.3,pp.369-376,2010.

2. Adens Gillia,"The Role of Risk in a Modern Software Development Process",*TASSC Technical paper*, 2004.

3. H Ammar, T Nikzadeh, and J. B. Dugan,"A Methodology for Risk Assessment of Functional Specifications using Colored Petri nets", *Software Metrics*

Symposium, Proceeding, Fourth International, pp.108 – 117, 1997.
4. Hall, M. Elaine,” Managing Risk – Methods for Software System Development”, *Addition Wesley,1998.*
5. Hua He, Zhi-Yong Zhou, “Risk Management System of Construction Enterprise”, *Enterprise economy*, vol.281no.1pp.,68-69,2004
6. “Risk Management for Software: Learning to Contain, Mitigate and Manage the Uncertainties of Software Development “, *Tim Lister*.
7. The Treasury Board of Canada Secretarial .Integrated risk management framework[R], *Ottawa:Treasury* Board of *Canada*,1-17, 2002,
8. Lev Virine, Lisa Rapley,“Visualization of Probabilistics usines ModelsIn Proceedings”, *Winter Simulation Conference*, 2003.
9 W.Eric Wong, Yu Qi,Kendra Cooper,(2005), “Source Code Based Software Risk Assessing”, *ACM Symposium on Applied Computing*, pp.1485 – 1490,2005.

Attribute	Value	Description
Probability	Very likely	High chances of risk, Threat>70%
	Probable	Risk like this may turn into a problem once in a while 30%<threat>70%
	Improbable	Less chances of risk Threat<30%
Impact	Catastrophic	Unrecoverable failure of system information Cost turnover > 50%
	Critical	Minor system damage with recoverable operational capacity Cost turnover exceeding 10% and less than 50%
	Marginal	Minor systemdamage to project with recoverable loss of operational capacity Cost turnover < 10%

Table1.Sample Attribute Value

Software Development Phases	Risk
Planning	Financial justification Client knowledge Complexity Availability
Requirement Analysis	Functional requirements Non-functional requirements No resource planning
Design	Functional complexity Algorithmic complexity Module dependency Use of appropriate data structure
Coding & Implementation	Use of reusable components Expertise of reusable components No team coordination Resource insufficient
Testing	Difficult project module integration Project is complex to implement Running out of fund

Table.2 Risk in SDLC