A CONCISE SURVEY OF RISK MANAGEMENT IN SOFTWARE REUSABLE COMPONENTS

M. Iyapparaj 1 Dr. S. Sureshkumar 2 V. B Santhoshraj 3

Senior Asst. Professor, Dept. of CSE, K.S.R College Of Engineering, Tiruchengode, Tamilnadu, India 1
Principal, Vivekanandha College Of Technology For Women, Tiruchengode, Tamilnadu, India 2
Final M.E-CSE, K.S.R College Of Engineering, Tiruchengode, Tamilnadu, India 3

ABSTRACT: Risk analysis is a process that involves mitigation of errors during various processes of software development. Early identification of risks should shorten the software development completion time. It will also provide efficient solutions. A complete risk analysis along with identification, monitoring and management plan is sure to ensure high quality software products. In this paper reviewed different perception of reusability and the techniques followed to mitigate the risk arose while reusing components.

Keywords: Software Engineering, Software Reusability, risk Management, Object oriented programming

I. INTRODUCTION

Software Engineering is a systematic approach to development, operation, maintenance and upgradation of software. In general, Software Engineering is a discipline which provides tools and techniques to develop quality software in an orderly fashion. It provides an engineering approach that is used to build software for end users using well defined, well managed, consistent and cost effective processes. The Software Development Life Cycle (SDLC) is the sequence of different activities that take place during the development. Any software development consists of five phases namely Requirement Analysis, Design, Implementation, Testing and Maintenance (Pressman 2000). The SDLC begins with the identification of the requirements of a software and ends with the formal justification for the development of software against that requirement. SDLC is the period of time that starts, when a software product is conceived and ends when the product becomes dormant.

II. NEED FOR RISK ANALYSIS IN SOFTWARE PROJECT

Occurrences of errors during the execution of a program need to be predicted before implementation. This is generally done through Risk analysis. For instance, classifying data types with variables would reduce the risk of errors. Literatures have pointed out the requirements of risk analysis for these types of problems. A risk is a potential problem which may or may not occur. Though software development is a very difficult process that determines risks and taking practical measures to avoid them, risk analysis becomes mandatory. In addition, to manage the risks, in order to reduce the complexity and the efforts taken for it, also becomes mandatory (Boehm 1991).

A.) Software Risks: The risk has two characteristics,

1) Uncertainty: The risk may or may not occur.
2) Loss: Unwanted consequences or losses will occur if risk is present.

To quantify the levels of uncertainty and the degree of loss associated with the risk, the type of risks should be categorized as follows: Risks which are external to program coding have been delimited to the study for this research. Four of these risks namely

1) ‘People Risk’ would lead to reduction of efficiency,
2) ‘Project Risk’ is sensitive to project cost
3) ‘Technical Risk’ is related to the time of implementation and
4) ‘Business Risks’ influence the software viability. Hence these issues are considered for the research study.
5)
III. CONCEPT OF REUSABILITY

A system is said to be reusable, if it has the capability of using its components in an orderly way. In short, a part or whole of one system can be incorporated to assemble a new system. As far as software is concerned, it makes the reusability simpler and competent to frame a new module which is free from bugs and risks (William Frakes and Carol Terry 1996). Table 1 depicts the merits and demerits involved in the concept of reusability.

<table>
<thead>
<tr>
<th>Merits</th>
<th>Demerits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increases adaptability</td>
<td>Lacks in tool support</td>
</tr>
<tr>
<td>Reduces process risk</td>
<td>Creating and maintaining a component library</td>
</tr>
<tr>
<td>Reduces time and cost</td>
<td>which creates overhead</td>
</tr>
<tr>
<td></td>
<td>Adaptability in reusability is cumbersome</td>
</tr>
</tbody>
</table>

Table 1. Reusability Merits & Demerits

IV. LITERATURE REVIEW

The technique of programming is widely brought up into two categories which are as follows:-

1) 1. Procedure oriented programming
2) 2. Object oriented programming

These two vital techniques ie., Procedure Oriented Program and Object Oriented Program are different from each other (Firesmith 1993 and Ryder et al 2005).

The former language which is procedure oriented usually follows algorithm-centric approach in which a program is driven by an algorithm that traces its path of execution from beginning to end. Data is an external entity the operation of which is defined upon by the algorithm. Fundamentally, these classes of programming languages are characterized by data that are considered and separated from the operations or program part. Algorithm which is the primary most stepping stone, which being the driver, with data being subsidiary to the algorithm, the procedure oriented program just breaks up a larger job into a number of subtasks and subprograms independently and conveniently as functions and subroutines. The functions and the subroutines which makes a complex job with best output, put together as a whole forms an efficient, easily debugging capability rich and a best programming part.

The main notion of the procedure oriented language is to make the process of debugging a program and to increase the number of reusable procedures in programs. The importance of program reuse is thus realized by software professionals. By adopting reusability techniques in programming, the cost of building software would be reduced considerably which have paved way for the emergence of Object Oriented Language. Khan Al-A’ali and Girgis (1995) have described the Object oriented programming language to be data or Object centric.

These programming languages which are described to be object oriented plays role in the promotion of objects which in turn helps in identification of the relationship between data and object that enhances creativity in the technique of reusability. When certain common data and objects are used in various different programs, it will greatly contribute in the reduction of time of development for the programmers and thereby increasing the efficiency of the program. Reusable components are chosen in such a way that they will be best suited for the activities in the implementation phase, to reduce risk. The implementation of code reusable components is mainly focused in this thesis.

Nunamaker et al (1989) put forth brief ideas of approach in order to understand the highly complex process of development of software and to improve the productivity of software including the using and designing of automated software development tools. The study of human factors in software development and application of software productivity measurement and evaluation technique are some other processes.

An environment to facilitate the reuse of software components is described. Such an environment supports the identification, creation, categorization, storage, selection, usage, and maintenance of reusable components. A meta system environment that allow users to define functionalities, structures, and constraints of various software components is discussed. Informations regarding all these components is being used by a knowledge-based system in order to support the selection, configuration, and distribution of the components that are reusable.
The rate of reuse (as shown by Nam Yong Lee and Litecky (1997)) significantly depends upon

1) reuse capability,
2) software development effort,
3) object-oriented design capability,
4) repository development effort,
5) Ada technology capability, and
6) domain capability.

The management of changes in reusable software was discussed and carried out by David Kane et al (1997) and his fellow members. Organization which undertake software reuse need to manage changes in component which grow due to expanded usage. Often, this is an expanded life cycle of software asset. The six patterns to support reusable software that emerged out of this discussion report are
1) develop a shared platform  
2) maintain the reuse platform identity  
3) integrate reuse and tie to the bottom line  
4) reuse code  
5) treat reusable components like product and  
6) merge clones.

Each of this pattern must be described with their problem statement and with their solution of reusable software. United State Navy Software Maintenance Organization management has created a process improvement activity for upgrading the overall quality of internal software maintenance. This was exclusively explained by Charette et al (1997). It provides a simple consensus method of risk assessment and mitigation.

The primary focus of this approach is on addressing the real risk rather than perceived ones. They state that there is a significant difference between maintenance oriented risk management and development oriented risk management. They have listed a suggestion of specific risk management information as input to specific risk management process.

V. CONCLUSION

In any software development set up, professionals differ from each other in their attitude, technical skill and ability. Talented software professionals form a great asset to an organization. They, therefore, reduce the levels of risks involved in successful product development.

The extensive literature reviewed in this paper, has indicated that there is no model available for adaptive reusability with reusable components in risk analysis for object oriented programs. An innovative model has been proposed in the name of ‘Proficient Risk Extenuation Model (PRE-M) for Adaptive Software Reusable Components’ and the design and developmental details along with its validity should be measured to focus on mitigating the risk occur in software component reusability.

REFERENCES

BIOGRAPHY

M. Iyapparaja is presently a Senior Assistant Professor, Department of CSE, K.S.R. College of Engineering, Tiruchengode, Namakkal, Tamilnadu, India. He received the B.E degree from Anna University, Chennai and M.E degree from Anna University of Technology, Coimbatore in 2006 and 2010 respectively. His research interests include Software Testing, Software Engineering and Agile Testing. He is a member of ISTE.

Dr. S. Sureshkumar is presently the Principal, Vivekanandha College of Technology for Women, Tiruchengode. He received the B.E degree from National Engineering College, M.S, degree in Software system from Birla Institute of Technology and Science, M.Tech., degree from Indian Institute of Technology, Kharagpur and Ph.D., degree from Anna University, Chennai in 1988, 1993, 2000 and 2009 respectively. He has published 30 numbers of papers in refereed international journals and conferences. His research interests include Image processing, parallel processing and Energy. He is a member of ISTE and IET.

V.B. SanthoshRaj is presently doing his Final M.E degree under the department of CSE, K.S.R. College of Engineering, Tiruchengode, Namakkal, Tamilnadu, India. He received the B.Sc degree in Computer Technology from Anna University, Chennai and MCA degree from Autonomous, Anna University of Technology, Coimbatore in 2009 and 2012 respectively. His project area of interests includes Software Testing and Software Engineering. He is the member of IEEE.