Adaptive Reusability Risk Analysis Model (ARRA)

[1]G.Singaravel [2] Dr.V.Palanisamy [3] Dr.A.Krishnan
[1] Professor, [3] Dean, Department of CSE, K.S.R.College of Engineering, Tiruchengode,TamilNadu,India
[2] Principal ,Info Institution of Engineering, Coimbatore, TamilNadu, India

Abstract
Software Engineering is a quality software producing strategy. In which software development life cycle involves a sequence of different activities during the development process. Coding is the process of execution of the software with sample data to ensure that the software works correctly without any opposite motion of risk. Risk analysis is an uninfluenced process to find out any error in software development life cycle as a precaution for risk in software projects. Object Oriented Program is different from the former because of its special features such as Polymorphism, Inheritance, Data Abstraction, and Encapsulation. Besides, in Object oriented language it is easy to identify the relationship between the object, element, attributes etc., Reusability is a precedent to the transformation process of the objects from one project to another with similar properties. Risk analysis helps to avoid the adaptive reusability problem for transformation of coding. Its main feature is that it encourages the concept of reusability which paves the way for use of functions and packages. But certain risk might occur during coding phase of programming in this methodology. This research work has hence resulted in the design and development of an analysis model termed as ARRA to deal with this problem. This innovative model ARRA would be useful for reducing the coding risk for software development team which is particularly working under the C++, Java, Objective-C and Smalltalk platform environment.

Key words:
Software Engineering, Risk Analysis, Object Oriented programming, Set Theory.

1. Introduction

Risk is a major threat in any software development process. Risk analysis thus helps in mitigating the risk factors during the early stages of development[1],[7]..Reusable components present in the software have to be analysed for the risks and reuse[2]. Therefore, the Adaptive Reusability Risk Analysis (ARRA) models need has been established.

'ARRA' model is to analyze the risks encountered during the usage of reusable components in the system. Reusable components play a vital role in considerably reducing the development time of the system. This model helps to analyse risk involved in the system after the application of software reusable components. It ascertains the various risks occurred in the system on account of software reuse. It makes the system efficient by focusing on all the relevant risks which occur due to software reusable components and its impact.

2. Purpose of the Model

The main aim of this model is to search for similar systems, find out the necessary software reusable components which help to reduce the development time and all the possible risk factors which are encountered during the system development process. ARRA model detects reusable related risk and makes adaptability analysis in Figure 1.

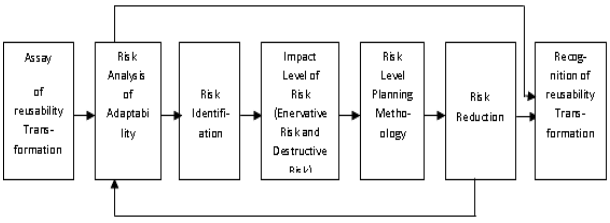

Fig.1. Adaptive Reusability Risk Analysis Model

2.1. Model Description

2.1.1 Assay of Reusability Transformation

Reusability Transformation is the process of copying and pasting a near similar code to the proposed system from an existing system which reduces the complexity in the code development. Reusability is the one which helps to identify the similar codes present in the project which perform similar functions when compared to the functionalities of the proposed system. Identifying the similar system which performs the same functions as the proposed system. It helps to build the new system with the same particulars. Reusability Transformation is the process of utilizing the software code from one system to another which performs the similar functionality and it can examine the codes in a particular system which performs the similar function as the target system is termed as one which resembles like another. Assay of reusability transformation is to ensure the reusability software code before transformation to other system.

Manuscript received February 5, 2010
Manuscript revised February 20, 2010

2.1.2 Adaptability Risk Analysis

After identifying the reusability code which are to be retrieved from the system, the various parameters such as function name, return type of the function and number of parameters which are to be used in the function are determined.

The various parameters which include the essential features such as return type of the function play a major role in performing the various operations. Also it includes the data type of the parameters, which are used in the function and the number of parameters in the function. Functions and parameters from the development software are to meet the requirements of the new software. This module checks the adaptability of the software code with the system by taking the various parameters into account. They make the system comfortable for using the software reuse code in future. The system comprises of various levels of matching based on the number of matching parameters obtained from the software code.

2.1.3 Risk Identification

Risk Identification is the process of finding out the various risks which occur due to the reusable software code which are identified and utilized in the system. The possible risks which occur due to the reusable software codes are analyzed and its impact of usage in the proposed system. This process makes our system to find out the risk and risk relevant measures which result in the improvement of the system.

2.1.4 Impact Level of Risk

After identifying the risk in the system and its has to analyze the impact level of risk in the system. The two type of risk involved in software development are **Enervative and Destructive risk.** This two type of Risk can be defined in terms of its impacts level on the software code before recognition of reusability.

Example: Program for Enervative and Destructive risk

```
#include<iostream.h>
#include<conio.h>
void main()
{
int a,b,c;
clrscr();
cout<<"Enter two numbers";
cin>>a>>b;
for (int i=1;i<a;i++)
{
for (int j=1;j<b;j++)
{
cout<<i<<"*"<<j<<"="<<i*j<<"\n";
}
}
getch();
}
```

Enervative Risk is the risk which has its impact on the system which leads to the reduction of efficiency of the entire system. If an error occurs in the system, it reflects by reducing the performance of the system. It executes in a normal fashion such that it won't affect the normal execution. Rather it reduces the efficiency up to which the system produces the outcome.

In above example program, the execution of the loop depends on the value given for the integer variables in the program which determines the execution time of the program. This will leads to the reduction in the efficiency of the system and achieving the expected output is also delayed. Hence it is termed as Enervative risk.

Destructive Risk is the risk which destructs the entire system by generating the unexpected outcome from the proposed system. These are the errors which make the system inefficient by producing the wrong outcome which entirely leads to the failure of the system.

In the above example program, the loop variable is declared as char which produces wrong output to the system. This leads to the destruction of the system and hence the impact will continue throughout the system due to its effect in the value of the variable by producing the wrong output. Hence it is called as destructive risk.

Enervative and destructive risk gives a detailed idea for the impact of the risk on the system and utilizing the software code effectively to reduce these risks and increases the efficiency of the system.

2.1.5 Risk Level Planning Methodology

According to the risk level, and it can plan the risk reduction factor which helps to reduce the risk level can be performed by focusing on the main effectively risk which leads to many consecutive sub risk factors.

2.1.6 Risk Reduction

Risk can be reduced by reusing the code in the proposed similar system. So, that it reduces the development time and the risk factors. These reusable components can reduce further risk occurrence in the future system implementation.

2.1.7 Recognition of Reusability Transformation

After identifying the Reusability software code components, it will be tested for utilizing the reusability in the future project. This helps to reduce the development time and also supports the reusability concept of Object Oriented programming Language. If it is not recognized to reuse the software code, again it takes iteration process until the adaptive of reusability.

3. Characteristics of the Model

ARRA model focuses mainly on the reusable components which are identified from the group of similar system which performs the similar functionality. The Characteristics of the Model are

- The ARRA model offers the object oriented system which helps to reuse the various components of the system.
- It supports the similar system functionalities and makes the system efficient and transferable.
- It focuses on the functions and packages which are reusable and transferable
- It manages procedure oriented and structure oriented systems by converting them into an object oriented paradigm which produces an efficient system which satisfies all user requirements.
- It ensures the efficient functioning of the system after the successful transformation of the reusable components.
- It finds out all possible risk factors in the reusable components.
- It is an iterative process, which reduces the risk until the zero effective level is reached.

4. Tested Model Strategy

The following testing strategies are covered by the ARRA model.

(i) Smoke Testing
(ii) Comparison testing
(iii) System testing
(iv) Regression testing
(v) Acceptance testing

4.1. ARRA Model Comparison with Different Model

***** The table 1., listed the comparison of ARRA model with its working principle and its risk coverage of different model[3] and it clearly understood that there is no model available for Adaptive Reusability Risk Analysis (ARRA) model.

Table 1: Comparison of ARRA model

Model Name	***Year***	***Working principle of model***	***Risk Coverage***
G risk – model	2006	Continuous improvement of software processes and products.	Software factories
ARAMS (adaptable risk assessment modeling system)	2007	Computer based modeling and database – driven analysis system developed, provides the common framework for linking disparate model and database.	Measuring data, human and ecological.
ARRA (Adaptive Reusability Risk Analysis) Model	* Proposed model	Adaptive checking of Reusable components in Object Oriented Programs.	Risk cover in function in C++ and package in JAVA for reusable code transformation.

4.2. Object Oriented Programming Concept in ARRA Model

Object oriented languages provide a standard class library that can be extended by user, thus saving a lot of coding and debugging effort[8].Object Oriented Programming construction of reusable components easily modify and extend implementation of components[5]. Code reuse is possible both in conventional language as well as in Object oriented language and greatly enhances the possibility of reuse.

Table 2: Parameter performance of reusability component

S.No	*Parameter of Performance Efficiency (PPE)*	*Reusable component(Functions) in C++*	*Reusable component(Packages) in C++*
1.	CPU Utilization	More	More
2.	Memory Utilization	Moderate	Moderate
3.	Response Time	High	High
4.	Staff experience effort	Less	Moderate
5.	Rework Effort	Less	Less

Add.cpp and Mypack. Java are the two programs which are used as an example to analyse the various Parameter performance for reducing the unnecessary efficiency by using of reusable components function in C++ and package in JAVA are listed in table 2.

5. Properties of Functions with Relation to Set Theory

This property deals with the elements which are common in nature with the proposed system and the old system. If it so, then it can make use of that function in the proposed system. It assures that when the elements of the system are similar and it can utilize the function for another system. Consider any two projects P1 and P2. In the project P1, have to check the similar elements which can be reusable in the project P2 through the reflective, transitive and symmetric properties as given below:

1. Project P1 consists of swap functions which are reused in the project P2 with some modifications like variable name and function name. If it is reused without any modification from P1 to P2 and that process is termed as reflexive which is the replica of swap function. i.e.., $A \subseteq A$ (Reflexive).
2. Project P1 contains the swap function which is reused by the Project P2 and the same function can be used in some other project say P3 which in turn gives that the function from Project P1 can be directly used by the Project P3. i.e., $(A \subset B) \wedge (B \subset C) \Rightarrow (A \subset C)$ (Transitive).
3. Consider a system which consists of add (a, b) function that accepts the input parameter from the user which can be symmetric if and only if it gives the same output for the two exchanged inputs a and b. i.e.., $(a, b) \in R \Rightarrow (b, a) \in R$ for all a, b in set A. (Symmetric). Thus the inclusion property is satisfied.

Reflexive, Transitive and Symmetric properties are satisfied through the Project 1 and Project 2[9].

6. Results and Discussion

The Table3: Shows the properties that ensure before and after the transformation of reusable components. Some properties can be used in functions of C++ and some other properties can be used in packages of JAVA. By using this property of ARRA Model, reduces the risk factor of the software coding. This reduces the overall risk factors of the project by using reusability component for cost reduction, time series reduction, finding the enervative and destructive risk in software coding development. This model also has an iteration process until the recognition of transformation in reusability component. This model ensures the effective functioning of the system under various factors.

Table3: Properties for Ensuring the Reusability Component.

Before transformation of reusability	*After transformation of reusability (checklist after adaptability of reusable components)*
• Usability and plan for reuse • Understandability and clarity for transformation • Interoperability and inheritability for testing of transformation. • Portability and flexibility for if any modification or components reused before transformation • Validity checking for reusability • modularity	• Coupling and Cohesion • Integra ability • Extensibility • Usability and clarity • Correctness • Understandability • Reliability • Portability • Flexibility • Testability • Efficiency • Validity • Functionality • Maintainability

References

[1] Boehm B. W. (1991), 'Software Risk Management: Principles and Practices', IEEE Software, vol.8, no 1, pp. 32-41.

[2] Fairley R (1994), 'Risk management for Software Projects' IEEE Software, vol. 11, No.3, pp.57-67.

[3] Foo S. W. and Muruganantham A (2000), 'Software Risk Assessment Model'. Proc. of the 2000 IEEE International Conference on Management of Innovation and Technology, 2, 536-544.

[4] Han Van Loon (2007), 'A Management Methodology to Reduce Risk and Improve Quality', IT Professional, pp. 30-35.

[5] Ivar Jacobson (1996), 'Object-Oriented software Engineering, A Use Case Driven Approach', Revised printing, Addison-Wesley.

[6] Jingwen Cheng (1994), 'A reusability-based software development Environment', International Conference on System Sciences, Vol. 19, No. 2, pp. 57-62.

[7] Pressman R.S. (2000), 'Software Engineering: A Practitioner's Approach', Fifth Edition, Mc Graw-Hill International Edition.

[8] Ryder B.G., Mary Lou Soffa and Margaret Burnett (2005), 'The impact of software engineering research on modern programming languages', ACM Transactions on Software Engineering and Methodology, Vol. 14, No. 4, pp. 432- 477.

[9] Tremblay J.P and Manohar R (1989), ' Discrete Mathematical Structures with application to Computer Science', McGraw Hill.

G.Singaravel received his B.E in Electrical and Electronics Engineering from kongu Engineering college, Perundurai, Bharathair University and Master Degree in Computer science and Engineering from Madurai Kamaraj University, Madurai. He is doing her Ph.D in Software Engineering at Anna University, Chennai. He is at present working as an Professor in the Department of Computer Science and Engineering, K.S.R. College ofEngineering, Tiruchengode, Namakkal District, Tamilnadu. He field of interest is Software Engineering ,Computer Architecture .He is a member of MISTE and MCSI.

Dr.V.Palanisamy received his B.E degree in Electronics and Communication Engineering from PSG College of Technology, Coimbatore and Master Degree in Communication systems from University of Madras. He also received his Ph.D in Antennas Theory from Indian Institute of Technology, Karagpur. Since 1974 he has been working in various capacities in the Department of Technical Education in Tamilnadu. He is at present working as a Principal of Info Institute of Engineering, Coimbatore. His field of interest is Electronics, Antennas, Image Processing, Communication Systems and VLSI Technologies.

Dr.A.Krishnan received his PhD Degree in Electrical Engineering from IIT, Kanpur. He is a IEEE senior member. He is now working as a Dean at K.S.R. College of Engineering, Tiruchengode and guide at Periyar University, Salem and Anna University, Chennai. His research interest includes Control System, Digital Filter, Power Electronics, Digital Signal Processing, and Artificial Intelligent Techniques. He is a visiting professor in ISTE chapter and at foreign universities. He has been Published more than 250 technical papers at various National and International Conferences and Journals.