A Module Component Analysis Approach for Testing Aspect Oriented Programs

Annu
Student, M.Tech
Deptt. Of Computer Sc., Shri Baba Mastnath Engineering College, Rohtak, Haryana
cse.annu@gmail.com

Sunita
Asstt. Professor,
Deptt. Of Computer Sc. & App., Shri Baba Mastnath Engineering College, Rohtak, Haryana
Kashisuni5@gmail.com

ABSTRACT: Aspect Oriented programming is one of widely used programming paradigm used in most of business applications. The Object Orientation enables to write a code in expressive and reusable form. Such code only represented in the modular form but also uses some libraries and external codes. Number of modules and external component access in a code affects the quality of a program code. The presented work is about the software quality analysis in terms of Aspect Oriented features. We have collected some software metrics to take the relative decision. In this work the concern is given to 4 main metrics called coupling, cohesion, module interference Metric. By assigning the weightage to these all metrics an overall system metrics is defined. The work is presented in the form of a software tool that will perform the analysis on java program but the presented metrics are implementable for all Aspect Oriented languages.

Keywords: Aspect Oriented, Module Interference, cohesion, Coupling, Software Quality

1. INTRODUCTION

Software engineering deals with the development of software but as it includes some of the external codes as well components the direction of viewing the software as the individual object is changed. Now it is the collection of different software modules or the components that are integrated in such a way a new software is developed. With the inclusion of the software components the complete life cycle of the software is changed. Now it needs to test and estimate each software components individually. And if these components are already in running mode in some other application we need to just perform the interfacing of the current application with these software components. Now there is need to test and estimate the interfacing between code and the software components. To understand the whole concept the definition regarding the software components and the software reusability must be clear.
1.1 Software Components

Some basic properties of the software components are:

(i) A software component can be a code block, module, function, class, control or the project or software itself.

(ii) The software component can be language dependent or language independent.

(iii) A software component can be end product or it can be extendable.

(iv) A software component is the unit of interfacing that conceptually specify its internal and the external interfacing with main application.

(v) A software component can also be a deliverable software object.

(vi) A software component can be online or the offline product or code.

As we can see a software component is not an individual term it is the basic concept that gives the software reusability in some way. Any kind of internal or interfacing in a software in the form of individual component is represented in the form of software components. Each of the software language defines most of software components in different way.

1.2 Software Metrics

As the number of components available on the market increases, it is becoming more important to devise software metrics to quantify the various characteristics of components and their usage. Software metrics are intended to measure the software quality and performance characteristics quantitatively, encountered during the planning and execution of software development. These can serve as measures of software products for the purpose of comparison, cost estimation, fault prediction and forecasting. Metrics can also be used in guiding decisions throughout the life cycle, determining whether software quality improvement initiatives are financially worthwhile (Sedigh et al., 2001). A lot of research has been conducted on software metrics and their applications. Most of the metrics proposed in literature are based on the source code of the application. However, these metrics cannot be applied on components and component-based systems as the source code of the components is not available to application developers. Therefore, a different set of metrics is required to measure various aspects for component-based systems and their quality issues.

1.3 Software Reusability

Software reuse is the process of implementing or updating software systems using existing software components. A good software reuse process facilitates the increase of productivity, quality, and reliability, and the decrease of costs and implementation time. An initial investment is required to start a software reuse process, but that investment pays for itself in a few reuses. In short, the development of a reuse process and repository produces a base of knowledge that improves in quality after every reuse, minimizing the amount of development work required for future projects, and ultimately reducing the risk of new projects that are based on repository knowledge.

2. Related Work

Lalji Prasad, Aditi Nagar[1] has defined a work on the analysis of software system for different structural and object oriented metrics. This metric based component estimation along with relationship analysis is performed under the procedure based analysis. Author discussed the metrics such as LOC, cyclomatic complexity, cohesion and coupling metrics. Author defined the analysis under the class level analysis so that the software product error detection and correction can be performed over the system. The estimation of software system under the operational measures is also performed to analyze the software quality. The operational features of the project and product is also discussed along with coupling analysis so that the estimation of software related metrics will be obtained. The paper has also discussed the software coupling measurement under the structural analysis. Zeeshan Ali Rana[2] has defined an estimation on software products to analyze the software system under defect analysis for object oriented software system. Author defined the work to perform the defect prediction in the software system and analyze the software system.
effectiveness under module level defect analysis. The defects are analyzed under different vectors such as number of defects. Author defined the coupling based analysis, association analysis, dependency analysis and the interface analysis. The class and inheritance analysis is performed to analyze the software product effectively. The UML diagrams are constructed to establish the relationship between classes. These kind of software system also able to present the static view of system so that the decision regarding the software quality can be done. Software system analysis is performed under the complexity metrics based analysis. Barry W. Boehm[3] has presented resource based software estimation scheme for software quality analysis. Author defined budget analysis approach to improve the software product analysis. Author performed analysis under different testing aspects.

E Da-wei,Xiamen[4] has defined an improved metrics based complexity model for object oriented programming. Author defined the complexity analysis under multiple aspects so that effective software development under method analysis will be performed. Author defined the analysis under the software complexity and method analysis so that the software design model will be improved. Author defined the development process with the specification of cost model under the size and volume analysis so that the prediction to the software system will be done effectively. Author defined the structural complexity model under integral factor so that the development effort will be reduced. Author defined a size and complexity based model for development of software system under cost estimation. Author defined an object oriented program so that effective software development will be done. Author defined a predictive software analysis under quality modeling so that the software system analysis will be done. Author defined an effective software evaluation and measurement for software system under traditional metrics analysis. Author defined a metrics suit with coupling, cohesion so that the software cost estimation will be done effectively[5][6].

A computational system for the software system under the software development rules was defined by the author under cost, timeline and quality analysis. Author defined the software development under software modeling and analysis. Author has defined a professional practice analysis and model for development process so that effective development model will be presented. A computational work to perform the software development and software product analysis under cost and time analysis was done by the researcher. Author defined a parametric analysis on software system under software quality and software history analysis. Author defined the software process and software design mechanism to analyze the software system under software evolution and software management analysis. Author defined the specific development model so that the development process will be improved. Author defined a set of management approaches to analyze the software system under software quality analysis. Author defined the management analysis program under the specification of research analysis and development by validating the software management under software improvement. Author defined the software measurement analysis under the defined framework so that the software measurement validation is performed. Author defined the structural model for software development so that the attribute relation analysis will be performed. Author defined the entity analysis so that the software development and validation will be performed. Author defined the software measurement based program to analyze the software system under rule specification and to analyze the program under different metrics. Author defined the effective path generation so that effective software measurement and testing will be applied over it[7][8][9].

3. PROPOSED WORK

Software quality analysis is one of the key concerns for all software stakeholders under different aspects of software estimation. One of such aspect is the software reliability. Software testing is one of the effective approaches to estimate the software reliability or the software quality. The presented work is the estimation of aspect oriented software system using the structural testing approach. Aspect Oriented programming generates the software modules under new perspective called aspects. Aspect basically defines the function or the event of the software system that itself collect number of attributes and functions incorporated to the individual entity. The presented structural analysis mechanism will be performed to analyze individual component of the software system. Once the individual module analysis will be performed, at second stage, the module interaction analysis will be performed. This analysis will include the interaction of a module with other as well as the interfacing with the external environment will be analyzed. In the final stage, the collaborative analysis of all these modules will be performed to perform the system integration analysis. In this section, complete software quality will be analyzed. The analysis will be done on real time software application that will include the statistical analysis as well as metrics oriented analysis. The presented work will be implemented on real time aspect project.

In this work, an effective metric based analysis approach is defined to perform the complexity analysis for software system. The analysis is here defined on software or module level component. The components include the attribute based analysis, function based analysis and nested components based analysis.

Finally, the sum of complexities of these methods will give the overall Reusability interface complexity of the target component. There exist no precise criteria to measure the various reusability respective to different quality characteristics, like complexity, maintainability, interfacing etc. The software components are the main objects that perform the code reusability. Today most of the business applications and others are component-based systems.

The proposed study aims to estimate degree of reusability on the basis of black-box components and component -based systems.
The work proposes and validates metrics for Interface complexity, of the system. The study uses methods and the properties of the component with its interfacing. It will estimate the software quality in terms of software component reusability.

The proposed work is about estimating the software reusability in a software program. Software components are one of the major factors that provide the software reusability. The software component can be a control, a function, a module etc. In java the software component can be described as a Bean component or the class and in vb or .net it can be defined as an active Control etc.

Now to estimate the software reusability in terms of software components we need to estimate some regarding factors such as

1. We have to find out different kind of software components or the codes that are providing the software reusability. We are representing all kind of code and the components as a single unit called components.
2. We have to find the kind of reusability. It means in what way the component become the part of main application. e.g. It is being used as a parent class or it is used as the composition etc.
3. Now we have to find the interface Metrics of the component with the main program.
4. We have to estimate the relationship between different components. It is possible that different components may have some bonding in themselves.

The flow of presented work is shown in figure 2.

Figure 2: Flow of Work
3. RESULTS

The presented work is implemented in matlab environment. MATLAB supports the entire data analysis process, from acquiring data from external devices and databases, through pre processing, visualization, and numerical analysis, to produce presentation-quality output. The work is tested on dummy module based project representation. The results obtained from the work is given here under.

![Attribute Based Complexity Analysis](image1.jpg)

**Figure 3: Attribute Based Complexity Analysis**

Here figure 3 is showing the attribute based complexity analysis of an aspect oriented program under different module respective to components. The complexity is here analyzed in terms of attribute based analysis. The figures shows the component 1 is more complex in terms of attribute level complexity.

![Method Based Complexity Analysis](image2.jpg)

**Figure 4: Method Based Complexity Analysis**

Here figure 4.3 is showing the method based complexity analysis of an aspect oriented program under different module respective to components. The complexity is here analyzed in terms of operation based analysis. The figures shows the component 6 is more complex in terms of method level complexity.

4. CONCLUSION

The proposed work is about to estimate the software quality for aspect oriented program. The work is defined to analyze the software system in terms of integrated component analysis. The complexity analysis shows the effective analysis of software system.
REFERENCES

[1] Lalji Prasad, and Aditi Nagar,” Experimental Analysis of Different Metrics (Object-Oriented and Structural) Of Software”, 2009 First International Conference on Computational Intelligence, Communication Systems and Networks 978-0-7695-3743-6/09 © 2009 IEEE.


[9] E Da-wei and Xiamen,” The Software Complexity Model And Metrics For Object-Oriented”, 1-4244-1035-5/07@2007 IEEE.


© 2014, IJCSMC All Rights Reserved 476