Design and Analysis of Dynamic OO Metric Tool for Java Project

Sherry Chalotra
Assistant Professor, Pyramid College of Business & Technology, Phagwara, Punjab

Abstract: Dynamic OO Metric Tool is used for measuring different software metrics and characteristics of java programs by accessing class file or byte code and gives in-depth detail of the project in form of its number of packages, classes, their methods, constructors and fields. The metrics used in Dynamic Metric Tool are used to predict various characteristics at the earlier stages of software lifecycle by just compiling the completed modules and getting its detail by using the proposed tool in mathematical form. The evaluated metrics results are presented in a graphical user interface that gives exhaustive detail in a very clear and simple way. It establishes quality benchmarks to identify potential design problems at early stages of software lifecycle rather than putting it later ends. The overall aim of Dynamic OO Metric Tool is to calculate various software metrics and design attributes for java projects so as to reduce the complexity of maintenance and moving it towards design and coding phase.

Keywords: Object Oriented, Object Oriented Programming, Software Maintenance, Java, Design attribute

1. Introduction

Object Oriented Programming gets operational after its deliverance in order to adapt the rapid changes in IT industry. It’s the most complex process that consumes 67% of overall cost of software development. Thus software engineering methods or tools are required[11]. Software metrics play a crucial role in software re-engineering process that gives numerical measurement of particular aspects of target software and helps in identification of complex parts of the software that need restructuring [11]. In this paper software metrics are evaluated for Java software. A set of selected metrics are used to evaluate design attributes like, size, encapsulation, response from class, its communication and other class attributes and their relation to evaluate inheritance [1], [2]. Object Oriented Programming is typically based on objects, which are used to access the class, which accommodate data, field, attributes and methods in it. The most striking feature of OOP is the security that is being provided by encapsulation, information hiding, inheritance etc. by which methods can access the data of class to whom it is associated and the entire communication is only through objects by passing message to one another [4]. Encapsulation merges abstract data types with structured programming and divide entire system into modular objects which are responsible for their own behavior as each object has its state and behavior [4, 12]. Due to this object oriented programming gave such a platform to programmers where data is not directly accessible by the rest of the system and gave a secure way of programming as the data is accessed by calling specially written functions, called methods, which are bundled with the data [4]. The provision of re-usability of existing components and extending components as needed by defining new subclasses with specialized behaviors makes a software modular which is known as inheritance or open-closed principle of OOP [9]. The module is said to be open if it supports extension while a module is said to be closed if it has a well-defined stable state and further extension or interaction may lead to introduce errors in other modules.

Dynamic OO Metric Tool is designed for java programs for measuring various object oriented metrics like, methods, classes, fields, encapsulation and inheritance. As it is designed for java programs only, the input given to this tool is .java files [10]. Java programs are first compiled so that the source files (.java file) get converted into the byte code (.class files). These object files (.class files) are feed as an input to the tool and the use of the .class file over .java file is to provide an efficient result as .java files are just text files that can easily be modified. Also .java files are not readable by Java Virtual Machine (JVM) so it becomes important to convert it in to its byte code. Dynamic OO Metric Tool works in three different phases by creating classes dynamically at run time only. The structure is shown in figure 1.

2. Methodology

2.1 Design of Dynamic OO Metric Tool

Dynamic OO Metric Tool is designed for java programs for measuring various object oriented metrics like, methods, classes, fields, encapsulation and inheritance. As it is designed for java programs only, the input given to this tool is .java files [10]. Java programs are first compiled so that the source files (.java file) get converted into the byte code (.class files). These object files (.class files) are feed as an input to the tool and the use of the .class file over .java file is to provide an efficient result as .java files are just text files that can easily be modified. Also .java files are not readable by Java Virtual Machine (JVM) so it becomes important to convert it in to its byte code. Dynamic OO Metric Tool works in three different phases by creating classes dynamically at run time only. The structure is shown in figure 1.
2.2 Description of quality attributes

a) **Design Size:** A measure of the number of classes used in a design.

b) **Encapsulation:** Defined as enclosing of data and behavior within a single construct. In Object Oriented design, the property specifically refers to designing classes that prevent access to attribute declarations by defining them to be private, thus protecting the internal representation of the objects.

c) **Response for a Class (RFC):** It is a class level design metric. Response for a class is a set of methods that can be executed in response to a message received by an object of that class [1, 12]. Response for a class is defined by the number of methods available in the class. The number of available methods in the class is the sum of the number of local methods in the class and the number of methods called by the local methods.

2.3 Metrics for Class

Class metrics concern with various characteristics of a class such as attribute, relationship and object instantiation.

3. Results & Discussion

The main goal of OO Metric Tool is to compute various metrics for class for the java projects that is created dynamically. This tool provides an automated way for measuring the metrics from .class files of the java projects and gives in-depth detail of the project in form of its number of packages, classes, their methods, constructors and fields. Along with that, it measures other object oriented metrics and provides the result mathematically. It also computes design quality attributes. The representation of project’s attributes and mathematical count of metrics are represented in a graphical user interface. The GUI provides the better interaction between the user and the software. The results so generated are also very easy to understand as it is displayed in a friendly graphical user interface that provides options for the detailed information of the classes. This helps the team leaders and developers to have detailed view of all the attributes related java projects.

4. Conclusion

It is clear that to measure the quality of software at design level as well as code level, different tools are available. To measure different quality attributes each tool includes different set of metrics. The Dynamic OO Metric Tool allows the user to interact with the software and to measure project’s attributes and object oriented metrics mathematically by considering the object files of java programs. It also provides the facility of measuring the
design attributes like inheritance, encapsulation, size, response etc. that helps the team leaders and developers to have detailed view of all the attributes related java projects.

5. Future Scope

This research concentrates on calculating various metrics of class for the java projects only that are dynamically created. The future scope of this work is as follows:
1) It can be designed for software running on different platforms like C++, PHP etc.
2) It can be designed for software running mobile application platforms also like Android.
3) It can be designed for measuring the metrics of default packages.

References