Comparing Hybrid Tool for Static and Dynamic Object-Oriented Metrics

Babur Hayat Malik¹, Javaria Khalid², Hafsa Arif³, Ayesha Sadiqa⁴, Amara Tanveer⁵, Asia mumtaz⁶, Zartashiya Afzal⁷, Samreen Azhar⁸, Muhammad Numan Ali⁹
Department of Computer Science and Information Technology
University of Lahore, Chenab Campus, Gujrat Pakistan

Abstract—Software metrics are created and used by the distinctive programming associations intended for assessing, guaranteeing program excellence, activity, and software recovery. Software metrics have turned into a basic part of programming growth and are utilized in each period of the product development life cycle. Software metrics essentially measure programming items like plan source code and help us in taking technical and administrative choices. The desire of this examination is to play out the relative investigation of static and dynamic metrics. In any case, software quality characteristics, for example, performance, execution time and dependability rely upon the dynamic exercises of the product artifact. Due to every one of these variables, we favor dynamic metrics instead of customary static metrics. With the assistance of customary static metrics, we are not capable to analyze different actualities of programming. There are various types of this OO static and dynamic equipments. In this paper we have played out a similar investigation of different OO static and dynamic metrics tools and find out the hybrid too is counted as best one extraction of both, static and dynamic characteristics from mobile Android applications. The source code and a Docker compartment is utilized by open source tool in only three phases pre-static, static and dynamic examination.

Keywords—Software metrics; static metrics; dynamic metrics; Object Oriented (OO)

I. INTRODUCTION

A software metric is fundamentally a software engineering track which relates to the various software developments and dimensions. One effective tool used for software product analysis is software metrics [1] [2] [3]. It plays a major role in the analysis and improvement of software quality along with measurement of software complexities [4]. An appropriate software model is required for the development of reliable software. ISO 9126 is one of the quality models that uses software metrics [5] [6]. Several tools are required for making of software quality models which intends to do metrics calculations. Though, these tools are also required to produce accurate data [7]. Software metrics are categorized into three parts: product metrics, process metrics, and project metrics, as shown in Fig. 1.

Results are specified by a standard unit known as "Metric". It is used for evaluation of software processes, products, and services. Different authors have proposed several object-oriented (OO) metrics which are quite famous in the present software development environment [9]. These are different from standard metrics as they use objects instead of algorithms as a key object [10]. Traditional metrics are not eligible in determining the quality as intricate projects are enforced through OOD design practices, so they are required [11]. Somerville [12] described metrics in two types known as static and dynamic. Static metrics analyze code before executing it whereas dynamic metrics analyze code during code execution. In this research, static metrics is more focused on the understanding of procedural and object-oriented programming languages [4]. In this paper comparison of Static and dynamic OO tools are proposed. They are more emphasized for finding object-oriented metric tools on the basis of several parameters.

This paper is written in several sections. Firstly, Section II describes the literature work of various Object-oriented Static and dynamic metrics tools. Then, in Sections III is discussed the differentiation between Static and Dynamic Metrics. Various types of object-oriented Static and Dynamic Metrics are presented in Section IV. In Section V, the comparative study of OO Metric Tools is performed. Lastly, Section VI, presents the conclusion of this article.

Fig. 1. Software Metrics [8]

II. LITERATURE REVIEW

Various OO metrics are developed until now which differ in their properties and features. The main purpose of this paper is to find out huge OO metric computational tools on the basis of their properties. Complex metrics to be resolved are still an issue whereas in traditional OO some metrics like CK and MOOD are considered quite helpful in the development of software [13].

Munson and Hall [14] identified the program complexity level along with three processes of functional, fractional, and operational complexity. Mayo et al. [15] discussed the quality attribute of the interface which calculates modules complexity and dynamic metrics when it's executed.
Honglei et al. [16] presented metrics definition, types, and history. Measurement of software complexity is one important factor and it's also related to software development price factor.

Hassoun et al. [17] proposed Dynamic Coupling Metric (DCM) for object level coupling that considers program execution as it is used to measure objects coupling during runtime. Though it also estimates the runtime complexity and system comparison at meta-level along with those systems which have no reflective features.

Singh and Singh [18] presented four class-level dynamic couplings for identifying object-oriented systems quality. They are more determined in finding key coupled classes consisting of most active classes during runtime. Gupta [19] presented three dynamic coupling metrics which consists of foremost relations between objects during runtime, i.e. aggregation, inheritance, etc.

Mayo et al. [20] defined both automated Interface and Dynamic Metrics. The first one is used for identifying modules complexity whereas dynamic metric calculates quality factor during execution. Hays in [21] identified OO systems testing and compared them with conventional programming language testing.

Mohsin, Shaikh, and Zeeshan Kaleem [22] presented the idea of code comprehension with a combination of Software metrics and techniques called Program Slicing. It is basically coded automation analysis for coupling, cohesion, and complexity.

Debbarma, Mrinal Kanti et al. [23] described the comparison of static and dynamic metrics and analyzed them in terms of regression testing that helps in effort and time estimation used during testing.

III. TYPES OF METRICS

In various real-time applications, there is a small number of the most eminent metrics that are analyzed. There are different categories of metrics that are presented below:

A. Traditional Metrics

In an object-oriented system, traditional metrics are commonly applied to the methods that include the class operation. "A method is a component of an object that operates on data in response to a message and is defined as part of the declaration of a class". Methods reveal how a problem is fragmented into different sections. Two traditional metrics are Cyclomatic complexity and size (line counts) [24].

B. Object-Oriented Metrics

Object-oriented software metrics emphasis on measurements that are functional to the conceptions of classes, coupling, and inheritance. Encapsulation metrics are applied for classes, not for modules. Information Hiding is measured & enhanced due to Inheritance complexity is additional, the level of abstraction can be measured by Object Abstraction metrics. These are as follows:

- Metrics correlated with Class
- Metrics associated with Methods
- Metrics Encapsulation
- Measurement of Cyclomatic complication
- Metrics used for Inheritance [25, 26].

1) Static metrics: This Metric is the outcome of non-executable code. Static metrics describe system features from design through maintenance. Earliest Metric used for Static is [27] (LOC/KLOC) examine the throughput of a software package. In earlier 1990, McCabe was the most powerful metric for examining the intricacy of cyclomatic [28] complexity. Complexity is evaluated from the graphical representation and various mathematical equalities. In 1976 McCabe [29] demarcated the cyclomatic complexity metric. It measures the total numbers of independent routes over a software component.

2) Dynamic metrics: These are resultant of source code investigation. When code is running it evaluates what is really happening. Dynamic metrics comprise complication events and processes beneficial in consistency demonstrating at the same time [30]. When software is executing its values are reliant on the involvement or experimental information. From coding to maintenance system aspects are classified by dynamic metrics [8]. The comparison of static and dynamic metrics with its merits, demerits are shown in Tables I and II.

<table>
<thead>
<tr>
<th>TABLE I. STATIC VS. DYNAMIC METRICS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Static Metrics</strong></td>
</tr>
<tr>
<td>1. Its nature is always static.</td>
</tr>
<tr>
<td>2. It is simpler and easier to collect.</td>
</tr>
<tr>
<td>3. OO software attributes are difficult to examine.</td>
</tr>
<tr>
<td>4. It takes less time as compared to dynamic analysis of software.</td>
</tr>
<tr>
<td>5. It is available at the early stages of the software development life cycle.</td>
</tr>
<tr>
<td>6. For software quality prediction its results are less accurate.</td>
</tr>
<tr>
<td>7. More Tools are effortlessly available to accomplish this examination.</td>
</tr>
<tr>
<td>8. Its implementation is done on the code.</td>
</tr>
<tr>
<td>9. It deals with structural aspects of the system.</td>
</tr>
<tr>
<td>10. It identifies vulnerabilities in a runtime environment.</td>
</tr>
<tr>
<td>Serial No.</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>
IV. CURRENT ISSUES AND CHALLENGES

- After negotiations upon dynamic metrics, it has definitely perceived that currently not any metrics available for testability at execution time of the software systems.
- Its benefit includes accuracy and precision; however, they are more difficult in evaluation to static ones. Therefore, a good hybrid approach is required.
- For the analysis of different software aspects pseudo dynamic metrics is another auspicious research prospect readily accessible to researchers.
- It can be certainly observed from the survey of many research studies conducted by different authors that dynamic metrics are examined and tested using a project that is not bulky [31].

V. RESULTS

We have to concern together static along with Dynamic Metrics to realize the deviation. After comparing both of these metrics we concluded that dynamic metrics analysis gives result at execution time of programs whereas static analysis at rest of the SDLC process. So, for dynamic analysis data is collected with the help of tool based on either Java or C++ based application, then apply a statistical tool to measure the quality of the product. Dynamic analysis can give a better result than static analysis.

AndroPyTool executes different tools in order to extract wide-ranging features from an input set of Android samples. All these features and the evidence that they symbolized are organized in three dissimilar classes (pre-static, static and dynamic), both the features and how they are extracted.

In Pre Static it comprises extracting information without inspection of code and permits to categorize and to track the sample. It also includes the package name and the main activity name, which are found with Andro-guard. In Static analysis, it contains those features that are regained by analyzing the application at the code level. In this category, features such as API calls, activities, opcodes or permissions can be originated. In Dynamic Analysis, it includes Droid Box tool for this purpose, which allows to dynamically find dissimilar information in real time. The information gathered by the Droid Box tool includes: the use of cryptographic functions, loaded DEX classes in run time and the kind of operation, network connections, SMS, phone calls, started services, enforced permissions and information leaks detected. The detail diagram of AndroPyTool is shown in Fig. 2 [38].

VI. DISCUSSION

Various OO Metrics tools their description, merits and demerits are studied in this research paper. These tools are tabulated under various attributes that would be of interest to developers and researchers using the tools as elaborated in Table III. Our study has further pointed out the work and research findings that has been done till now to use of hybrid approach of static as well as dynamic metrics, although they have tremendous scope. Based on the analysis of existing dynamic metrics, we have tried to reveal potential research challenges and opportunities existing in the field of dynamic metrics. Best methodology that is suitable for pre-static, static and dynamic metrics is hybrid approach and its tool that is AndroPyTool.
TABLE III. COMPARISON OF STATIC VS. DYNAMIC METRICS TOOLS

<table>
<thead>
<tr>
<th>Tool Name</th>
<th>Description</th>
<th>Language</th>
<th>Availability</th>
<th>Authors</th>
<th>Tool Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckStyle [35]</td>
<td>Java Checkstyle is an improvement tool to enable designers to compose Java code that clings to a coding standard. Presently, Checkstyle gives checks that discover classplan issues, copy code, or bug designs like twofold checked to bolt.</td>
<td>Java</td>
<td>Open source</td>
<td>Oliver Burn</td>
<td>Static</td>
</tr>
<tr>
<td>FindBugs [35]</td>
<td>This is the Static Analysis tool and is open source that checks and study class files or JAR libraries for probable problems adjacent to a list of bug patterns by matching the byte code [5].</td>
<td>Java</td>
<td>Open source</td>
<td>David Hovemeyer and William Pugh</td>
<td>Static</td>
</tr>
<tr>
<td>StyleCop [35]</td>
<td>For plugins and customs rules, StyleCop provides an extensible framework to write down custom rules which match up to our requirements.</td>
<td>C#</td>
<td>Free</td>
<td>Andy Reeves, Chris Dahlberg</td>
<td>Static</td>
</tr>
<tr>
<td>JMT [36]</td>
<td>It only associates the Metrics with Java language.</td>
<td>Java</td>
<td>Free</td>
<td>Politecnico di Milano and Imperial College London</td>
<td>Dynamic</td>
</tr>
<tr>
<td>QMOOD++ [37]</td>
<td>QMOOD++ is easy and free of cost accessible in the runnable application and source code form. It handles the 30+ Metrics. QMOOD++ is an inclusive, multi-handler, multiprocessing, incorporated software tool.</td>
<td>C++</td>
<td>Free</td>
<td>Bansiya, Jagdish, and Carl Davi,</td>
<td>Dynamic</td>
</tr>
<tr>
<td>JMetric [11]</td>
<td>JMetric only works with Java. Its information is presented through tables and charts.</td>
<td>Java</td>
<td>Free</td>
<td>Commercial Tool</td>
<td>Dynamic</td>
</tr>
<tr>
<td>AndroPyTool [38][39]</td>
<td>AndroPyTool incorporate different analysis tools and Android applications Processing tools, in order to convey fine-grained reports drawing their individual performance and features.</td>
<td>Python</td>
<td>----</td>
<td>Alejandro Mart, Raul Lara-Cabrera, David Camacho</td>
<td>Hybrid</td>
</tr>
</tbody>
</table>

VII. CONCLUSION

A correlation of diverse software metrics and its major tools are presented in this comparative study. On the base of their major types like static and dynamic metrics, these are differentiated. At early stages of software development life cycle (SDLC), Static metrics are reachable easily. These metrics manage the overall structural qualities of the product framework and very simple to assemble. The unpredictability of static metrics has calculated the measure of exertion expected to create and keep up the code. In the latter stage of the software development life cycle, dynamic metrics are easily reachable.

These metrics confine the dynamic conduct of the framework and difficult to acquire and got from hints of code. After a virtual study of various static and dynamic tools are performed and broke down that hybrid tool is best in the greater part of the android applications. AndroPyTool, the primary objective is to furnish scientists and malware examiners with an incredible and coordinated device for extracting multi-source highlights from Android applications. In future work, more tools and features can be add on into AndroPyTool tool for better analysis and to improve the data analysis stages, in order to give more functionalities to the users.

REFERENCES


