Automated Tool to Assess Pair Programming Program Quality

Mazni Omar\textsuperscript{a}, Rohaida Romli\textsuperscript{b}, Azham Hussain\textsuperscript{c}

\textsuperscript{a,b,c}Graduate Department of Computer Science, College of Arts and Sciences, Universiti Utara Malaysia, 06010 UUM Sintok, Kedah, Malaysia

\textsuperscript{a}E-mail : mazni@uum.edu.my
\textsuperscript{b}E-mail aida@uum.edu.my
\textsuperscript{c}E-mail : azham.h@uum.edu.my

ABSTRACT

This paper aims to present an automated tool that has been developed to assess pair programming program quality. The tool known as Java Quality Measurement Tool or JaQMeT is used to assess specifically Java program quality. There are two program quality factors that can be assessed which are correctness and complexity. Pair programming program will be graded using JaQMeT. Then the results will be used to evaluate the effectiveness of pair programming. JaQMeT is only at its initial stage. It is an initial effort to facilitate the lecturers to reduce workload on grading programming assignment and specifically to assess pair programming program quality. Although JaQMeT has its several limitation but it is hoped that JaQMeT can be extended by using web-based technology and capable to check others program quality.

Keywords

Pair programming, program quality, automated tool

1.0 INTRODUCTION

Pair programming is one of the core practices in Extreme Programming. It is claimed that pair programming can promote team work and thus produce high quality software (McDowell, Werner, Bullock & Fernald, 2002; DeClue, 2003; William & Kessler, 2003; Hanks, McDowell, Draper & Knjajic, 2004; Xu & Rajlich, 2005).

In order to assess program quality, there is a need to develop a tool. The tool can assist the programmer specifically to automate the process of evaluating program quality. On the other hand, the tool can also reduce workload of teaching staff or lecturer in grading the programming assignment.

Pressman (2001) defined program quality as conformance to explicitly state functional and performance requirement, explicitly documented development standard, and implicit characteristic that are expected of all developed program. There are various quality factors in order to assess program quality such as correctness, complexity, maintainability, portability, and others. In this study two main quality factors will be used which are correctness and complexity. Correctness is a degree to which the program performs its required function (Pressman, 2001). Most correctness assessment is empirical and based on checking a program’s output from inputs. If the program meet its requirement, formally a program is correct. Correctness is a straightforward criterion to formulate the program quality. Meanwhile, software complexity metrics is developed to identify parts of program that are likely to be difficult to test, understand, or error-prone.

Most studies on pair programming investigate an impact of pair programming on program quality and then provide empirical evidence on the effectiveness of the practices. However the main aim of this study is to develop an automated tool to assess pair programming program quality. The tool will measure two main quality factors which are correctness and complexity of the program.
2.0 RELATED WORKS ON PAIR PROGRAMMING QUALITY TOOL

In this section, related works of the study which includes pair programming quality tool will be described. A summary of pair programming literature which focuses on program quality is given in Table 1: Summary of Literature. Most of the literatures were obtained from the ACM and IEEE databases.

<table>
<thead>
<tr>
<th>Author(s)</th>
<th>Quality Metrics</th>
<th>Method to measure program quality</th>
<th>Using Automated Tool?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Williams et al. (2000)</td>
<td>Functionality</td>
<td>Using automated testing tool that executed by impartial teaching assistant</td>
<td>Yes</td>
</tr>
<tr>
<td>Ciolkowski &amp; Schlemmer (2002)</td>
<td>Complexity</td>
<td>Using evaluation tool to compute worst-case complexity of system</td>
<td>Yes</td>
</tr>
<tr>
<td>McDowell, Werner, Bullock &amp; Fernland (2003)</td>
<td>Functionality, Readability</td>
<td>Using student’s score on graded programming assignment</td>
<td>No</td>
</tr>
<tr>
<td>DeClue (2003)</td>
<td>Functionality</td>
<td>Using student’s score on graded programming assignment and qualitative survey</td>
<td>No</td>
</tr>
<tr>
<td>Gehringer (2003)</td>
<td>Functionality</td>
<td>Using student’s score on graded programming assignment</td>
<td>No</td>
</tr>
<tr>
<td>McDowell, Hanks &amp; Werner (2003)</td>
<td>Functionality, Readability</td>
<td>Using student’s score on graded programming assignment and qualitative assessment which include functionality, style and holistic</td>
<td>No</td>
</tr>
<tr>
<td>Hanks et al. (2004)</td>
<td>Complexity</td>
<td>Using JavaNCSS to calculate source code metric of</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 1: Summary of Literature

There are various methods to measure program quality. Some studies that focus on functionality quality metrics have used student’s score on graded programming assignment to assess pair programming program quality (McDowell, et al., 2003; DeClue, 2003; Gehringer, 2003). In contrast, Williams et al. (2000) have used automated tool to assess functionality quality metric in their study.

Ciolkowski & Schlemmer (2002) and Hanks et al. (2004) have performed pair programming experiments by using an automated tool to compute complexity of the student’s pair programming program. In addition, Arisholm et al. (2007) have assessed correctness of quality program using correctness analysis tool, automated test scripts and automated web-based grading tool. Besides, another method to measure pair programming quality in terms of readability and lines of codes is by using qualitative assessment (McDowell, Hanks & Werner, 2003; Xu & Rajlich, 2005).

3.0 DESIGN OF JaQMeT

An automated tool known as Java Quality Measurement Tool (JaQMeT) has been developed to assist teaching staff or lecturer to assess Java program quality. JaQMeT is able to assess two program quality factors which are correctness and complexity. It is designed based on the integration of two studies proposed by Rohaida, Fazilah & Mzini (2004) and Mawarny & Rohaida (2005). Proper enhancements and modifications had been made to fit with current requirements.

Two program quality factors were chosen to assess the quality of student’s program. These two types of program quality were adequate to assess program quality that applied basic Java programming concepts. Apart from that JaQMeT is used as quality measurement tool to assess program quality produced by student that applied pair programming practices. Figure 1 depicts a process of quality assessment in JaQMeT.
As shown in Figure 1, the process of measuring program complexity is done by implementing static analysis to the student’s program and program schema. Then, both programs are analyzed by comparing their complexity values to see their similarity results. Finally, the weight value and score are assigned to each selected metric and calculation of complexity mark is executed to obtain the mark given to the student’s program. There are two software metrics adopted (Abounader & Lamb, 1997; Xenos, Starrinoudis, Zikouli & Christtrodoulakis, 2000) for complexity checking in JaQMeT:

• Cyclomatic Complexity metric - measures the amount of decision logic in a single software module. Cyclomatic complexity is defined to be \( e - n + 2 \), where \( e \) and \( n \) are the number of edges and nodes in control flow graph, respectively. This cyclomatic complexity is measured for each method in class.

• Operation Complexity of a class metric - defined \( \sum O(i) \), where \( O(i) \) is operation \( i \)'s complex value. Summing up the \( O(i) \) in for each operation \( i \) in the class gives their metric value.

Meanwhile, the correctness of student’s program is assessed by using the equivalence partitioning technique. Equivalence partitioning technique is one of the black-box testing techniques and adopted to design the test cases used to assess the correctness of the program. Black-box testing examines the functional operation of the system. This means that the program is executed with given test data, and the output of student’s program will be compared with the program schema to determine whether the output is correctly produced (Chu et al., 1997). Specific score and weight were assigned to each test case as a measurement of the program correctness. The assessment result contains marks of student’s program.

There are eight main functions consist in JaQMeT:
- Assess Program Correctness
- Upload File Schema
- Set Weight Value
- Assess Program Complexity
- Set Test Data
- Upload Question
- View Assessment Result
- Upload Student Program

Figure 2 illustrates a use case diagram that maps respectively to all functions with the actors of the system. Each use case shows how an actor interacts with system and what the system does. The use case has a set of sequence actions and performs observable result to a particular actor, who interacts with the system.

As shown in Figure 2, there are two actors involved in managing the functionalities of the JaQMeT, namely lecturer and Java compiler. Lecturer is a person who evaluates student’s program and plays an important role in preparing and managing source needed in processing Java program assessment, managing program schema, and managing weight value for measuring correctness and complexity of program. Java Compiler is an external entity or independent software used to compile and interpret a Java program to be assessed.

All interfaces of JaQMeT system were developed based on number of use cases that have been defined. Figure 3 to 8 depict sample interfaces that map the predefined use cases.
The following briefly explained Figure 3 to 8:

- Upload Question interface is used to upload questions to be solved by students.
- Upload File Schema interface is used to generate schema output for a given question.
- Upload Student Program interface is used to upload a student’s program submitted to be assessed.
- Assess Program Correctness interface is used to assess the correctness of student’s program.
- Assess Program Complexity interface is used to assess complexity of student’s program.
- View Assessment Result interface is used to view student’s correctness and complexity quality results

A functional testing was conducted based on the strategy of black-box technique to ensure that JaQMeT has met expected requirements. Each predefined use case had been thoroughly checked its functionality. Use cases “Assess Program Correctness” and “Assess Program complexity” were tested by running it with predefined samples of program and samples of students’ program that were collected from the pair programming experiment conducted in this college.

4.0 RESULTS AND CONCLUSION

JaQMeT can help the lecturers specifically to reduce workload on grading programming assignment. Therefore it helps to reduce time to assess pair programming program quality. Student’s pair programming program quality which focuses on correctness and complexity can be graded by using JaQMeT. This result can be used to evaluate the effectiveness of pair programming practices. However JaQMeT also constrained by a few limitations such as it is a stand alone system and only can assess two quality factors; correctness and complexity for Java programming program. Therefore it is hoped that JaQMeT can be extended by using web-based technology as a web-based system might allow multiple accesses to the system. In addition a new tool that capable to check others program quality for other programming languages can be developed and hence integrate it with the tool developed in this study.

REFERENCES

McDowell, C., Hanks, B. & Werner, L. (2003). Experimenting with pair programming in the


