ISSUES WITH OBJECT ORIENTED PROGRAMMING ASSESSMENT IN MALAYSIAN UNIVERSITIES

Norazlina Khamis, Sufian Idris
Faculty of Computer Science & Information Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia
azlina@um.edu.my

Faculty of Information Science & Technology, National University of Malaysia, 43600 Bangi, Selangor, Malaysia
sufian@ftsm.ukm.my

ABSTRACT

Object oriented (OO) programming is one of the most popular programming paradigms nowadays. It is not surprising that most of Computer Sciences degree programs offer OO courses as part of their curriculum. One of the biggest challenges for educators who teach programming courses is deciding how to assess each student’s programming skills. How do we assess whether the students have acquired the right OO skills after they have completed their OO courses? None of the current assessment techniques would be able to provide this answer. It appears reasonable that a better understanding of how to assess OO skills is needed by developing a new approach of assessment. It is even important in the context of Malaysia where there is currently a growing concern over the level of competency of Malaysian IT graduates in programming. This paper will discuss the issues and problems with current OO programming assessment methods used in Malaysian universities. In this research, surveys were conducted in several universities in Malaysia to investigate how educators assess OO programming skills amongst undergraduate students. Issues and problems related to current assessment will also be identified. This is followed by the description of a proposed research framework to develop the assessment approach as mentioned.

INTRODUCTION

The most influential programming paradigm today is object oriented programming (OOP), and it is widely used in education and industry. Backlund (Backlund, 1997), in his research wrote that OOP has been acknowledged for its naturalness, understandability, extensibility, reusability and ease of maintenance. In the literature, OOP has been suggested as a solution to many problems. Robins and Rountree (Robins and Rountree, 2003) in their research stated that the demands and interest in programming have grown increasingly and this lead to the popularity of the programming courses being offered. Recognizing the importance of equipping students with object oriented knowledge and skills, it is not surprising that most Computer Science degree programs offer object oriented-related courses.

OOP has always been considered as an advanced course and it was introduced late in the curriculum. Normally procedural programming is taught in introductory programming courses. This is gradually changing. Currently, more universities have started to introduce OOP as a first year of programming course. However, teaching OOP remains difficult. Learning new knowledge is hard and in general, people are reluctant to learn new things. Educators found that when students learn to program in object oriented style, they find the task difficult as they are so used with programming in the procedural style. According to Kolling (Kolling, 1999), it takes an average programmer 6-8 months to switch her/his mindset from procedural to object oriented. It is often we can find that students claim their source code were using object oriented style, but if we examine carefully, their source code is actually in procedural style.

Teaching and learning programming has become more advanced everyday. These advances require educators to understand issues on how programming is learned, what the expected cognitive outcomes for learning to program are, what are the levels of programming skills that have been achieved by the students and how to assess these skills. It has been found that
current research on the above mentioned issues only address on the first two issues (McCracken et. al, 2001), (Lister et. al, (2004), (White and Sivitanides, 2005).

In this report we present the results of an investigation carried out to identify issues and problems with current OOP assessment methods amongst Malaysian universities. This will be followed by a description of a proposal for developing a framework for assessing OOP skills.

OOP ASSESSMENT
According to McCracken et. a.l (McCracken et. al, 2001), programming is one of the skills that every computer science student is expected to master. Numerous studies have been conducted to identify the issues with regard to learning and teaching how to program, and these studies have helped computer science educators to improve the teaching of programming courses (Bennedsen and Carpersen, 2004), (Wei et. al, 2005), (Lahtinen and Ahoniemi, 2004). There are also a number of research on predicting performance and success of students in programming (McCracken et. al, 2001), (McKeithen et. al, 1981), (Graham and Verbyla, 2003).

Most of the OOP courses in Malaysian universities usually take one semester or 14 weeks. How do we assess whether the students have acquired the right object oriented skills after they have completed the courses? What are the OOP skills that are being assessed? To investigate these issues further, we conducted a survey amongst lectures in a few universities in Malaysia. We hope to identify in the survey the OOP assessment approaches that are being used by these universities.

Method
We distributed a questionnaire amongst OOP educators in a few universities in Malaysia. Among the universities that participated were; University of Malaya (UM), National University of Malaysia (UKM), International Islamic University (IIU), University of Science Malaysia (USM) and University of Technology Malaysia (UiTM). The main purpose of this survey is to investigate the current approaches used in these universities when conducting OOP assessment. The participant if this survey comprised of educators that are involved either in teaching or doing research on OOP. We identified ten (educators) who agree to participate in this survey. From this survey, we found that most of the universities used a combination of approaches when conducting their OOP assessment. The approaches that have been identified are as follows: written examination, practical examination, programming assignment and viva session. Figure 1 shows a summary of the approaches of assessment for OOP courses. A detailed discussion on the results of this survey can be found in (Norazlina and Sufian, 2007).

![Figure 1: Assessment Approaches in Malaysia's Universities](image)

Issues
Several issues of this study raised points that must be addressed in future. Deciding how to evaluate each student’s programming skill is one of the biggest challenges for educators who teach programming courses (Chamillard and Joiner, 2001). From the survey, we found that there are number of approaches to evaluate student programming skills and normally,
respondents conducts assessment for students programming skill using these three popular approaches: written examination, practical examination and programming assignments. Students need to answer a set of questions or problems and they must apply the OO concepts that they have learned. They will be given a grade based on their answers. Most of the respondents highlight the problem with this approach is that grades do not reflect the actual programming skills that the students have acquired. Different educators have different expectation when marking the student’s work. So it is quite difficult to justify if student’s who get good grades in OOP at one university have the same OO skills with another students who also get a good grades in other university. Thus most of the respondents agree if there’s a guideline/framework which specifically list out OO skills that student’s should acquired when they learn OOP. The framework will serve as guidelines when designing questions for OOP.

Many authors have reported that Computer Science undergraduate students in their second year do not have the ability to write programs even though they obtained good grades in OOP courses in their first year (McCracken et. al, 2001), (Daly and Horgan, 2001), (Califf and Goodwin, 2002). This also happened in Malaysian universities. Through informal interviews with respondents, they found quite often students get good grades but still have problems with actual programming. Why grades does not appropriate for OOP assessment? One reason why grades are not suitable for OOP assessment is that the marks given by the educators may vary among educators who teaching the same course. Grades only represent the extent to which students have successfully met the university requirements and it is impossible to make inferences about what they know by looking at their grades.

Another issue that arise from this survey is does the assessment is design to assess student’s OO skill? What are the guidelines when educators design the questions for OOP assessment? Based from the survey, we found that only three out of five universities apply Bloom’s taxonomy concept when designing the questions for their OOP assessment. Bloom’s taxonomy is a hierarchical classification of cognitive skills and capabilities. The taxonomy is popular and useful as a tool to facilitate appropriate questioning (Edward, 1981).

One of the drawbacks of using this model is that it is only measures the cognitive skills and capabilities. What about OO skills? Can we identify OO skills based on this model? When these questions were highlighted to the educators, most of them who involved in this survey agree that there is a need to have a framework that specifically serves as a benchmark to conduct OOP assessment. From the framework, educators can easily identify either their student’s have acquired the OO skills. Thus, based on the results from the survey we would like to propose the development of a framework for assessing OO skills.

CURRENT WORK
Most of the assessment in OOP focuses on assessing the quality of the program produced by the students and not the skills acquired by the students (Patenaude et. al, 1999). It appears reasonable that a better understanding of how to assess these skills is needed by identifying a new approach/framework for assessing them. It will not only help Computer Science educators to evaluate each student’s skills in OOP program, but will also lead to better understanding of the pedagogy of Computer Science programming courses. We would like to investigate more by identifying what can be classified as OO skills? How do we define OO skills and how to measure these skills? To obtain the information, we will conduct another survey and interviews with OO educators that we have identified from our previous survey to participate in phase two (2) of the project (refer figure 2).

![Figure 2: Research Methodology](image-url)
Currently we are in the process of identifying OO skills by doing document analysis and interviews with OO experts. OO experts comprised of educators/professors from Computer Science faculties. The members of this group were chosen because they have high experience professionally or through their research in OO. Some members of this group also taught programming. The expected outcome from this phase is that we will have a set of OO skills students should acquire when they learned OO. We are unable to come out with a proposed set of OO skills at this time due to the timeframe associated with the submission of this paper. The proposed research approach for this project can be found in figure 3.

**Figure 3: Research Framework**

**CONCLUSION**

The main objective of this project is to develop a framework for assessing OO skills as well as an accompanying tool to assist educators in assessing object-oriented programming skills of Computer Science undergraduates. Learning to program is difficult and the capabilities for each of the student is also different. Thus it is important to conduct a proper assessment which can reflect the actual OO programming skills that the students have acquired after completing their OOP courses. It is hoped that this project will be able to provide a model on how to assess OOP skills. It will not only help Computer Science educators to evaluate each student's skills in OO programming, but will also lead to better understanding of the pedagogy of Computer Science programming courses.

**REFERENCES**


