Teaching Object-Oriented Systems Analysis to Non-IT Students: A Practical Experience

Nor Iadah Yusop
Faculty of Information Technology
Universiti Utara Malaysia
Sintok, Kedah, MALAYSIA
noriadah@uum.edu.my

Abstract

Teaching software development course is not easy even to people with computer science qualification, let alone to those without such background. However, the author personal experience has proven that this is not impossible. This paper aims at sharing the method adopted in teaching object-oriented systems analysis to postgraduate students without academic qualification or experience in information technology or computer science. Most of them are merely end-users to a number of ubiquitous applications software such as word processors. For the said course, three aspects were given special emphasis: concepts and theories familiarization, application of the theories, and familiarization of technical contents. In view of their background, with respect to the aspects addressed, the author would suggest that the method applied in the teaching of systems analysis is considerably successful for they manage to produce the deliverables required with acceptable quality.

1. Introduction

Currently, Faculty of Information Technology (FIT), Universiti Utara Malaysia (UUM) offers two bachelor programs: Bachelor of Information Technology with Honours and Bachelor of Multimedia with Honours, and we are in the process of offering the Bachelor of Computer Science with Honours. On top of these undergraduate programs, FIT also offers programs at postgraduate levels namely the Master of Science in Information Technology [MSc. (IT)], Master of Science in Information and Communication Technology [MSc. (ICT)], Master of Science in Intelligent System [MSc. (Int. Sys.)], and Master of Science in Technopreneurship. Except for MSc. (IT) which is also available by research, all other postgraduate programs are conducted by coursework. However, students are required to complete a final project in order to successfully graduate with the degree. All undergraduate and postgraduate programs are also offered to international students. The medium of instruction for all programs is English. Thus, English proficiency is mandatory.

Besides English proficiency and other basic requirements, candidates for all postgraduate programs other than the MSc. (ICT) are required to hold bachelor degree in information technology, computer science or equivalent areas. MSc. (ICT) program on the other hand, is opened only to candidates with bachelor degree from fields other than information technology or computer science. The candidates must possess at least 2.5 for their cumulative grade point average (CGPA) or must have at least three years of working experience in order to be eligible for the program. Candidates with degree in ICT discipline or related area are not allowed to enroll for MSc. (ICT) program [1]. Despite many programs offered by FIT, in this paper the author will only focus on MSc. (ICT).

Students enrolled for this program must successfully fulfill the minimum of 33 credit hours comprising of nine courses and one final project. Each course carries three credit hours and the project is of six credit hours. Core courses weight 18 credit hours including Research Methodology subject, and nine hours of electives. Other than the Research Methodology, the core courses these students must take include Principles and Techniques in Programming, Database Application Development, Computer Systems and Networks, Internet Technology, and Information Systems Development. The objects of the discussion in this paper are the Information Systems Development (ISD) course and the ISD students for semester ending April, 2006.

Brief background information about the faculty and its programs are described in this section. Section 2 focuses mainly on the ISD course's background as well as the corresponding students enrolling for the course. Succeeding section describes the approach adopted in

Proceedings of the Sixth International Conference on Quality Software (QSIC'06)
0-7695-2718-3/06 $20.00 © 2006 IEEE

teaching the course. While presenting the challenges faced in accomplishing the course objectives, Section 4 also shares some of the success stories. Concluding remarks and recommendations follows in Section 5.

2. Background

This section provides some insight about the students enrolled for the ISD course for semester ending April 2006, and also the ISD course itself.

The author had been very fortunate to have a small class comprising of fourteen students. Though small, their academic qualifications and experiences differ very much. Since the program is also offered to international students, it has attracted the interest of students from other countries too. Table 1 summarizes the class composition in terms of their qualifications and country of origin.

Table 1. Class composition

Country of Origin	Academic Qualifications
Malaysia	Education (3) Library Science (1) Financial Accounting (1) Business Administration (1)
Libya	Electrical Engineering (2) English Language (1) Business Administration (4)
Thailand	Business Administration (1)

More than half of the class is international students. All of them are without academic qualification in information technology or computer science. With regard to working experience, only four out of fourteen (28.6%) has at least or more than three years of working experience while others are fresh graduates. Being locals or foreigners, their knowledge about the concepts in information technology (IT) or computer science is considerably low although some of them have been using computers mainly for word processing. Thus the main challenge is to make them familiar with the IT concepts.

ISD aims at providing students with necessary theories and practice of systems development. Students are required to carry out a software development project in groups. However, the project mainly focused on the software analysis and design phase using object-oriented technique. At the end of the semester, they have to submit a full report of their projects. They are also exposed to Rational Rose as tool to assist them in their analysis and design. Upon completion of the course, the students are expected to possess certain levels of understanding on methods and techniques of systems analysis and design, and to models the requirements using computer-aided software engineering tool.

However, this paper focuses only on the teaching methods applied to accomplish the earlier part of the syllabus that is the systems analysis including the modeling. The later part, i.e. the design part, applies similar teaching method.

3. Teaching methods

As mentioned in the earlier section, the students have considerably low knowledge even about the basic concepts of IT. Making matters worse, pre-requisite of IT-related subject for the said course in the current course structure is nil. However, some of the students do take IT-related subjects such as Principles and Techniques in Programming, Internet Technology or other core or electives courses concurrently. Apart from providing them the knowledge about the what's and how's of systems development, the main challenge is to familiarize them with concepts related to information systems such as system, software or information system to name a few. In addition, familiarization with the basic concepts and theories of software and software development, as well as the technical contents of the subject matter, and the application of the theories to the software development itself, need to be given. The rest of this section describes the teaching methods adopted to achieve these. These methods were applied throughout the semester.

3.1. Concepts and theories familiarization

At the beginning, getting grasp of the concepts is important because they are being used over and over throughout the duration of the course. During the first two weeks, using lectures, the relevant concepts are made known to them. Questions were asked to ensure that they get good grab of the introduced concepts. Sadly to say, many of them were grappling in familiarizing themselves with the concepts. However, repetition is impossible for there is a list of topics scheduled for the whole semester to be covered. Nonetheless, during the subsequent lectures on theories of software development, the definitions of the concepts are reiterated when necessary. Meanwhile, questions were asked to make sure that they are with the rest of the class. As time goes by, the author found that their familiarity increases for there is no more puzzled faces when the concepts were mentioned in the class.

Once the concepts and theories of software development are familiarized, they are ready for the

application of it. To begin with, the students have to have an information systems project to work on. Specifically, they are required to work on software development project. Following subsection describes the processes and methods the author took in facilitating the students in their groups' projects.

3.2. Application of the theories

Software development theories provided to the students were put into application in information systems development projects. The nature of the projects varies, but they started from scratch.The project was carried out in groups of three to four students. They were instructed to form their own group. To ensure their interest on the project, each group was asked to agree upon an information system they "wish to have" and produced a description of the system. The reason for the "wish to have" feature will be described later in this section. In total, four projects were proposed.

Having had the "wish to have" systems, the groups are required to present their descriptions. The author, playing a role of a facilitator, evaluated each description to see its viability. Among the characteristics looked into is the size of the system which is measured based on the number of expected main functionalities that the system may have. The floor put on was four and ceiling was six. The author considers more than six as reasonably complex for novices. This is important to ascertain that the complexity of the system they want to have is medium. Too simple would not make the learning "fun" and they may not learn as much. However, too complex would make learning difficult as this course is not easy in the first place.

In practice, usually users do not develop their own information systems. They commonly outsource it to other parties. As mentioned in many software engineering books [2, 3], the developers ought to develop software that meets the user (client) requirements. Therefore, to make the students feel that they were working on real projects, they have to play the roles of developers and clients. For the purpose of the projects, each group played both roles, i.e. there are moments when they will play the roles of clients, as well as developers. However, in this course, the students are not required to do the implementation part (programming). Instead, their tasks end at the design stage. In this case, they mainly played the role of the analysts.

The mechanism used in switching the roles is by assigning the development of the "wish to have" system to other group, i.e. developer group. The owner of the "wish to have" system will become the client to the developer group and they are assumed to know what they want the system to provide. This is very important because the developer groups will be consulting them for "approval" on the continuation of the project to the next stage. However, the facilitator was there to assist them in making such decisions. Hence, in this case, each group would not develop its information system itself; instead others would do it for them. Figure 1 depicts the groups' "wish to have projects" and the projects that they were responsible for developing. Mutual exchanged were not planned. Once the roles and the projects' assignments were settled, the analysis phase begins.

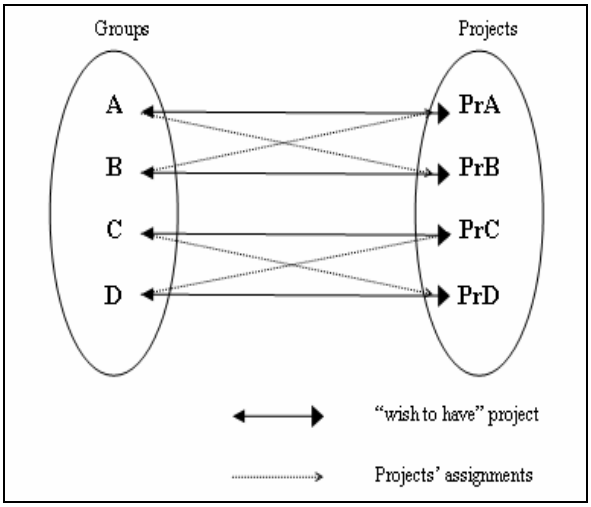

Figure 1. Projects' Assignments

During the analysis phase, most frequently the students worked in their groups. The groups have to be closely supervised and monitored to keep them on track. As facilitator, the author requested feedbacks from each group at the end of the meeting sessions. These feedbacks can either be in written, or oral (presentation), or both. Through presentations, the students' progress and achievements can quickly be ascertained. Meanwhile, the "clients" could also confirm or "approve" whether their requirements were correctly understood.

Pertaining to the project itself, the steps adopted during the analysis phase adhere to the descriptions provided in many software engineering [2, 3], and systems analysis and design books [4, 5]. To begin with, the developer groups gather the requirements of their clients. The main technique used in requirement gathering was interview. The first interview was made during class where instructions were given and monitoring was done. It was between the clients and the developer groups. Basically what was done is that

the developer asked, and the clients answered accordingly. Since there was not much time allocated to the first session, the facilitator had to ensure that they were really working on fact findings about the clients' "wish to have" system. The interview was carried out to get the basic requirements of the system. Further clarification of unclear or incomplete requirements shall be referred to the clients in other meetings and probably using other means. Other meeting are left to the developer teams and their clients to decide, and usually as the need arise.

After the requirements were gathered, the developer groups must really understand the requirements stated by their clients. To accomplish this, activity diagram was used to graphically represent the gathered requirements. This was the starting point for them in using symbols to represent the requirements. For this purpose, they were taught how to draw the diagram and immediately applied to their project. It was not easy and in fact the greatest challenge was to have them to represent the requirements in graphical form. However, learning by doing has proven to be effective in this case though the students did not get the diagram right for the first time. In subsequent sessions they managed to show some improvement on the diagram. After they had understood the requirements, they were asked to classify the requirements into functional and non-functional requirements [2, 4].

Activity diagram was not the only diagram or model that they have to produce. With regards to systems analysis, a bunch of other models need to be developed. Prior to that, deep understanding on what's and how's of modeling, i.e. the familiarization of the technical contents of software development, is highly desirable. This was another big challenge the facilitator ought to face. Next sub-section addresses this issue.

3.3. Familiarization of technical contents

To start working with object-oriented systems analysis (OOSA), it is desirable that the students have at least some knowledge on object-oriented programming language (OOPL). The reason is merely that similar concepts are applied in OOSA. Unfortunately, the students did not posses as such. The hardest time is to make them understand the OOPL concepts such as class, inheritance, and messages to name a few, let alone the concepts to software development itself such as the use case diagram, class diagram, or state chart diagram. However, perhaps because of the learning by doing method adopted in teaching and learning, they manage to get grasp with some of the concepts; at least the concepts necessary to get their analysis part done.

Subsequently, apart from having them to classify the functional and non-functional requirements, what need to be done next is to have them to model the functional requirements using the use case diagram. The use case diagram and other diagrams produced are then modeled using the Unified Modeling Language (UML) of Rational Rose. At this point, they ought to learn about how to create the necessary UML models [6]. Besides, they also need to know how to document or draw the models using the tool. Again, learning by doing takes the lead.

Apart from the above aspects, others are fairly comparable with other non-technical subjects. Following section quotes some of the success stories and highlights major challenges faced in delivering the course contents successfully.

4. Success stories and challenges

Apart from the complexity of ISD, another big challenge the author faced was the students' proficiency of English is considerably low, spoken and written even with English proficiency requirement is stated in the admission requirement. Though not everybody, the majority shows so. This increases the difficulty levels in making the students get what they need to get. However, the methods adopted shows that it could be done. The students in fact find the class interesting. This is shown in Table 2 on report by the University Teaching and Learning Centre (UTLC) of the university about course assessment.

Table 2. Assessment of teaching method

Assessed Items	Value
Content is understandable	3.00
Interest in subject	3.15
Effective teaching aids	3.23
Opportunity to interact in class	3.23
Objectives were well explained	3.15
Encourage students to give opinions in class	3.31
Encourage students to think in class	3.23
Monitoring of students' understanding	3.38

Questionnaire about the course were given to all students at the end of the semester. It was anonymous. Among the items relevant to teaching methods adopted asked were shown in the Table 2. Analysis was done by the UTLC, and the report is given to individual instructor as guide to improve his teaching method.

The column 'value' in Table 2 represents the students' responses on the items asked. 1 indicates

strongly disagree, 2 disagree, 3 agree and 4 strongly agree. Results have shown that in general the students agree that the teaching methods used in delivering the course content are helpful. Apart from creating their interest and understanding in the subject matters, the methods also encourage them to interact with fellow friends during presentations and questions and answers sessions. The author also found that their communication skill do improve very much. Thinking is also encouraged when they were asked to express their ideas or to defend their decisions. Meanwhile they also agreed that their understandings were always monitored. Besides, they did show their understanding of the subject matters when they were able to produce the required documentation at an acceptable quality.

5. Conclusion and recommendations

In general, the main teaching methods adopted were lectures and "learning by doing". The author would consider the "learning by doing" involved in this case was not exactly an adapted version of Bucks Institute Educations' (BIE) Project-Based Learning (PBL) [7]. Though the aim is alike, that is to let the students discover what they need to know themselves. The reason is that it does not follow exactly the PBL framework suggested by BIE. In fact, nothing about it matches the BIE's PBL. Contrasting to PBL which has to be planned carefully, the method adopted was accidentally implemented. although learning by doing has been the major method adopted, lectures are still important in which major concepts and theories were taught and stressed. In the exercise, lectures and learning by doing are complementing each other. The major weakness in the method adopted however, was the inexistence of proper evaluation or assessment methods or rubrics. The actual students' performance on the global skills such as communication was not adequately assessed. However, their performance on the subject matter, i.e. mastery of the required skills and theories, are sufficiently measured through quizzes, examinations and written report.

Regarding the aspects emphasis as mentioned in previous section, reiterations of concepts and theories do help in increasing the students' understandings about the concepts. Immediate practice of theories taught about the how to's in systems analysis process during the lecture sessions do help them in mastering the skills they need to acquire. Interactivity between the instructor and the students is highly recommended and necessary. In addition, the ratio of computer to students ought to be one-to-one because each of them was taught to use the tool on personal basis. Once everybody knows how to use the tool, then only they were asked to work in their group.

The method adopted, though boosting up the students' interests and mastering of the subject matter, it requires very high commitment of all parties involved namely the instructor, students, and administrators. The commitment of the instructor and students is obvious. The administrator needs to give full support in terms of scheduling and allocating the time and place as well as ensuring the required facilities being provided for the success of the method or even the PBL implementations. Class duration need to be considerably sufficient to ensure the flow of instruction through practice is not interrupted and the implementation is very time consuming.

As mentioned in previous sections, the students do not have the basic in IT. To lessen the workload of the instructor in making the students know and understand the concepts in IT, and to increase the OOSA learning curve, the students ought to be given some pre-requisite courses. Though this will require restructuring of the program, the author personally feels that this would help the students as well as the instructors. Teaching such students requires a lot of time and effort of the instructors.

Anyone interested in implementing the method or the PBL need to plan its implementation very carefully, not only on the learning part but also the performance measurements. For PBL, BIE does have a framework of how to implement it. Perhaps, the learning by doing or active learning could better if PBL framework is suited accordingly. However, active learning alone might not be sufficient, it should be complemented by lectures or other conventional or modern teaching methods.

6. References

[1] *Postgraduate Academic Handbook 2005/2006 Session*: Universiti Utara Malaysia.

[2] I. Sommerville, *Software Engineering*, 7th ed: Pearson Education limited, 2004.

[3] T. C. Lethbridge and R. Laganiere, *Object-Oriented Software Engineering: Practical Software Development using UML and Java*. England: McGraw-Hill Education, 2001.

[4] A. Dennis, B. H. Wixom, and D. Tergarden, *Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach*, 2nd ed: John Wiley & Sons, Inc., 2005.

[5] S. Bennet, S. McRobb, and R. Farmer, *Object-Oriented Systems Analysis and Design using UML*, 2nd ed. United Kingdom: McGraw Hill International Editions, 2002.

[6] T. Quatrani, *Visual Modeling with Rational Rose 2000 and UML*. New Jersey: Addison-Wesley, 2000.

[7] "Buck Institute for Education."http://www.bie.org/. (Retrieved: 21 June 2006).