Impact of Outcome-Based Education on Software Engineering Teaching: a Case Study

Hong-Ning Dai, Wei Wei
Macau Univ. of Science & Technology
Macau SAR, China
hndai@ieee.org, wewei@must.edu.mo

Hao Wang
Norwegian Univ. of Sci. & Tech.
Norway
hawa@ntnu.no

Tak-Lam Wong
Douglas College
Canada
tlwong@ieee.org

***Abstract*— This paper investigates the impact of outcome-based education (OBE) on students' learning achievement from a software engineering (SE) program. It is not easy to transform an SE curriculum from traditional knowledge-based education (KBE) method to OBE method since it requires us to identify the outcomes clearly and map the outcomes with the expected capabilities of students. We first give a briefing on our SE program and outline the curriculum, then investigate the impact of OBE in two selected courses in SE program, with the completion of one course being the prequisite for admission into the other one. Experimental results show that OBE can greatly improve the learning effectiveness of students and teaching quality.**

Keywords—Outcome-based education; Software Engineering; Information Technology curriculum

I. INTRODUCTION

Outcome-based education (OBE) approaches have received extensive attention recently. An outcome is a clear learning result that learners must demonstrate at the end of the learning phase. In particular, learners should demonstrate what they can actually do with what they have learned and know. Compared with traditional education methods (such as knowledge-oriented education and skill-oriented education), OBE has the following benefits [1]: 1) *Clarity*. Teachers know what they will teach in courses and students have clear goals to be achieved after taking courses. 2) *Flexibility*. Teachers have freedom to choose any method to teach students. 3) *Involvement*. OBE emphases on student involvement. In this manner, students are expected to actively participate into the learning activities instead of passively memorizing knowledge like traditional education methods. The learning effectiveness can be greatly improved.

In our university, we have three information technology (IT) related majors: Computer Technology and Application (CTA), Electronic Information Technology (EIT) and Software Technology and Application (STA). The curricula of these three programs were revised in 2009 under the approval of Tertiary Education Services Office of Macau SAR. The previous teaching and learning methods used in our programs were largely designed in accordance with the principles in knowledge-oriented education (KOE), in which instructors delivered knowledge and skills to students. This method presumes that student learning outcomes can be guaranteed by a provision of high quality and large quantity of instruction However; we found that those teaching pedagogies associated with KOE are likely to result in learning at a superficial level, which means students can recite the concepts or the knowledge after taking the courses but fail to demonstrate problem-solving capability and initiatives to solve authentic problems. This becomes even worse in some upper-division courses, e.g., operating systems and network programming.

We are now reforming our education approach from traditional KOE to OBE. Compared with other disciplines, the OBE approach is relatively new to IT or computer science [2] [3] [9] [10]. Although it is shown in [4] [5] [6] that OBE can improve the perceived learning outcome perception of students and enhance the teaching quality, there are some limitations and drawbacks in OBE as shown in [7], which may potentially affect the learning quality [8]. Therefore, it is worth to investigate the impact of OBE on IT-related courses.

The research question is what the impact of OBE on IT education is. To be more specific, we are interested in find out, comparing with the KOE approach, how effective the OBE approach is in terms of improving student learning outcomes. Therefore, we investigate the impacts of OBE on IT education in this paper. In particular, we have initiated a real intervention of OBE in three IT-related majors for nearly three years. As indicated in a case study on courses of STA major, we demonstrate that this OBE method has greatly enhanced the learning effectiveness in our IT curricula and has improved the problem-solving capabilities of our students.

The remainder of the paper is organized as follows. Section II gives an overview of our OBE approach. Section III then presents learning activities in our curricular. Section IV gives the evaluation on learning performance. Finally, we conclude the paper in Section V.

II. OBE APPROACH

A. Overview of IT Curricula

There are three IT majors in our faculty: Computer Technology and Application (CTA), Electronic Information Technology (EIT) and Software Technology and Application (STA). When implementing OBE approach in all the three majors, we have carefully designed 10 curricular outcomes as shown in Table I. Most of them emphasize on the practical problem-solving capability of students. We have formally enforced this OBE approach in all three programs since January 2015.

TABLE I. PROGRAM LEVEL OUTCOMES

No.	Outcome description
(a)	Apply fundamental knowledge of mathematics, algorithmic principles, computer theory, and principles of computing systems in the modeling and design of computer-based systems that demonstrate an understanding of tradeoffs involved in design choices.
(b)	Analyze a problem, specify the requirements appropriate to its computing solution, design, implement, and evaluate a computer-based system, process, component, or program that satisfies the requirements.
(c)	Apply design and development principles in the construction of software systems of varying complexity.
(d)	Use current skills, techniques, and tools necessary for computing practice.
(e)	Function effectively as a member of a team to accomplish a common goal.
(f)	Understand professional, ethical, legal, social, and security issues and responsibilities.
(g)	Analyze the local and global impact of computing on individuals, organizations, and society.
(h)	Write effectively.
(i)	Give effective oral presentations.
(j)	Recognize the need for, and an ability to engage in, continuing professional development.

B. *Case Study on STA Major*

Our study in this paper is mainly focused on STA. The overriding teaching objective of STA major is to train students to become software engineers, make them familiar with the development of enterprises scale software and organizations. In other words, it targets on the development of students' practical problem-solving capability in software development. Students are required to complete at least 160 units of courses, including major courses (106 units), general education courses (36 units), and final year project (18 units). Take Fig. 1 as an example, where the knowledge of data structures is the necessity to take a course on operating systems.

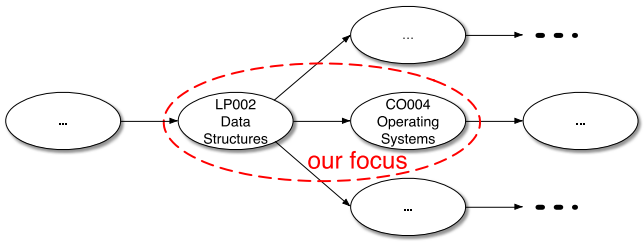

Fig. 1. Prerequisite relationships in STA program, where only part of the whole relationship diagram is shown.

In this paper, we attempt to investigate the impact of OBE in STA major. In particular, we consider two typical courses in STA major: Data Structures with course code LP002 and Operating Systems with course code CO004, where a completion of LP002 is the prerequisite for taking course CO004. We then test the effectiveness of OBE by evaluations of two groups of students (experimental group and control group). Details on OBE evaluations will be shown in Section IV.

III. ASSESSMENT METHODS AND LEARNING ACTIVITIES

We have carefully designed courses LP002 and CO004 to fulfill the program level outcomes as listed in Table I. In particular, we offer students with lectures, tutorials, exercises and programming assignments so that they can learn from practical examples. In order to assess the performance of students, we also design assessment approaches. Note that the assessment approaches should be mapped to our program learning outcomes.

A. *Assessment Methods*

We first give the mappings from assessment methods to program learning outcomes of course LP002 (Data structures) in Table II, where a tick indicates a mapping from the assessment method to the learning outcomes (otherwise it is left with a blank).

TABLE II. MAPPINGS FROM ASSESSMENT METHODS TO PROGRAM LEVEL OUTCOMES OF COURSE LP002

Assessment method	%	Assessment related to intended learning outcomes									
	Weight	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)
Participation	5					✓	✓			✓	✓
Programming assignments	15	✓	✓	✓	✓	✓			✓	✓	
Exercises	15	✓	✓	✓	✓			✓	✓		
Midterm exam	15	✓	✓	✓	✓		✓	✓	✓		✓
Final exam	50	✓	✓	✓	✓		✓	✓	✓		✓
Total	100%										

Similarly, we then give the mappings from assessment methods to program learning outcomes of course CO004 (Operating Systems) in Table III, where a tick indicates a mapping from the assessment method to the learning outcomes (otherwise it is left with a blank).

TABLE III. MAPPINGS FROM ASSESSMENT METHODS TO PROGRAM LEVEL OUTCOMES OF COURSE CO004

Assessment method	%	Assessment related to intended learning outcomes									
	Weight	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)
Participation	10					✓	✓			✓	✓
Labs	20	✓	✓	✓	✓				✓		
Projects	40	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Exams	30	✓	✓	✓	✓		✓	✓	✓		✓
Total	100%										

Comparing Table II with Table III, we can find that LP002 and CO004 share common outcomes. In order to evaluate the impacts of OBE, we investigate how the outcomes of the prerequisite course LP002 affect course CO004. To quantify the effects of OBE, we choose outcomes (a), (b), (c) and (d) to make the evaluations.

B. Learning Activities

LP002 is a second-year course, which introduces the fundamental algorithms and data structures in computer science. In particular, this course will concentrate on sorting, searching and graph algorithms as well as the fundamental data structures such as linked lists, queues, stacks, trees and graphs. Students are expected to be able to analyze the learned algorithms, evaluate the performance of algorithms, design and implement the learned algorithms.

We employ the following learning activities: 1) lectures, in which instructors introduce the topics and students are encouraged to ask questions or involved with class discussions; 2) tutorials, in which instructors or teaching assistants offer the Q/A discussions on the assignments or programming projects; 3) programming labs, in which students can complete the programming exercises or projects under the guidance of instructors or tutors.

CO004 is a third-year course, which aims to introduce the basic design principles and implementation techniques of Operating Systems to students by offering several small projects. Following the guidance and instructions, students should be able to implement the fundamental parts of a UNIX (LINUX) system such as a process scheduler and a file system. Compared with LP002, CO004 more concentrates on the practical aspect while we have the similar learning activities such as lectures, tutorials and labs.

IV. EVALUATION OF OBE METHOD

In order to test the effectiveness of OBE method in STA major, we select courses LP002 and CO004. Besides, we also select two groups of students:

1) Control group: The 2013 intake students who took course LP002 in September 2014 and course CO004 in January 2016;

2) Experimental group: The 2014 intake students who took course LP002 in September 2015 and course CO004 in January 2017.

Note that the OBE method has been formally enforced in our faculty in January 2015. Thus, the students in control group were taught in traditional method (i.e., KBE) when they took course LP002 in September 2014. They then were taught in OBE method in CO004 when they took course CO004 in January 2016. Between the two cohorts of students, they share similar demographic information (such as gender ratio and family backgrounds). They obtained the admission to study in the program with the similar academic performance in the university entrance examination, and have experienced the same group of lecturers in year one. Differently, the students in experimental group were taught in OBE method in both LP002 and CO004. We choose the same size for control group and experimental group, i.e., the number of students is 40.

A. Comparison on learning performance with/without OBE

We first investigate the effectiveness of OBE method in control group and experimental group in course LP002. In particular, we make a comparison on the learning performance of control group (without OBE) with that of experimental group (with OBE). Note that we choose the same teaching materials and design the same learning activities (lectures, tutorials and labs) for both the aforementioned groups.

In particular, we conduct a statistical analysis on the final scores of both control group and experiment group. Table IV shows the results.

TABLE IV. FINAL SCORE STATISTICS OVER TWO GROUPS OF STUDENTS

	Control Group (2013 intake)		**Experimental Group (2014 intake)**	
	No.	**Percentage**	**No.**	**Percentage**
<60	22	55.00%	13	32.50%
60-70	3	7.50%	4	10.00%
70-80	8	20.00%	8	20.00%
80-90	4	10.00%	6	15.00%
>90	3	7.50%	9	22.50%
Total	40	100.00%	40	100.00%

As shown in Table IV, we can see that there are more students in experimental group who can achieve high score (i.e., above 90) than those in control group; moreover, fewer students in experimental group fall into the low score range (i.e., below 70). In summary, the number of students scoring higher grades increases and the number of students scoring lower grades decreases after introducing OBE method. This result indicates that OBE method can significantly improve the effectiveness of learning performance.

B. Impact of OBE on follow-up courses

We then investigate the impact of OBE on other follow-up courses. In particular, we taught the 2013 intake students in LP002 by traditional method (KBE) and taught them in CO004 by OBE method. Our assumption is that the students who have been taught in OBE approach in previous semester can demonstrate a higher level of learning outcomes than those who have not in subsequent semester. In other words, we ask if the students in experimental group achieve higher score in CO004 course than students in control groups with special reference to their performances in relation to learning outcomes a, b, c, and d (see Table 1 for reference).

We conduct a comparison study on the two groups students. The control group consists of students who first took LP002 in traditional method and took CO004 in OBE method. The experimental group consists of students who took both LP002 and CO004 in OBE method. We employ the same learning activities for the control group and the experimental group. The activities include: 1) seven labs, 2) two projects, 3) two computer-based exams. We mainly evaluate the performance of students on the outcomes (a), (b), (c) and (d) of CO004. Please refer to Table I for the detailed descriptions of outcomes (a), (b), (c) and (d). Among them, outcome (a) is relatively difficult to be achieved since it requires the strong problem-solving capability of students.

Table V shows the mapping from learning activities to outcomes (a tick means a mapping and a blank means no mapping). It is worth mentioning that exams 1 and 2 emphaze on different outcomes. In particular, exam 1 mainly tests the capability of using Linux/Unix commands or other tools while exam 2 tests the comprehensive capabilities of programming, problem-solving and analyzing.

TABLE V. MAPPING FROM LEARNING ACTIVITIES TO OUTCOMES IN CO004

Activities	Outcome (a)	Outcome (b)	Outcome (c)	Outcome (d)
Labs				✓
Projects	✓	✓	✓	✓
Exam 1			✓	✓
Exam 2	✓	✓	✓	✓

We next give the statistical analytical results on the outcomes (a), (b), (c) and (d) of CO004 in control group and experimental group. Note that each percentage score is calculated by averaging the obtained points over the total points in each outcome over every student; this procedure is involved with the calculation of points in different learning activities (according to the mappings in Table V). Table VI lists the comparison of outcomes in CO004 in different groups of students.

TABLE VI. COMPARISON OF OUTCOMES IN CO004 IN TWO GROUPS

	Control Group (2013 intake)	Experimental Group (2014 intake)
Outcome (a)	50.9%	74.1%
Outcome (b)	67.8%	69.3%
Outcome (c)	61.2%	75.2%
Outcome (d)	73.8%	91.3%

We can see from Table VI that the students from the experimental group can achieve higher outcome scores than those from the control group. This result implies that there are impacts on OBE method on the follow-up courses. Recall that the students from the experimental group in CO004 were taught by OBE method while those from the control group in CO004 were also taught by OBE method. However, unlike the students from the experimental group, the students from the control group were taught in non-OBE method when they took LP002, which is a prerequisite for taking course CO004. As shown in our aforementioned results in LP002 (refer to Section IV-A), the students from the control group performed worse in LP002 than those from the experimental group. As a result, the capabilities (i.e., the outcomes) that they obtained from LP002 were poorer than those from the experimental group; it consequently affects their performance in the follow-up course CO004. Take Table VI as an example again. The students from the control group achieved outcome (a) in the proportion of 50.9%, much lower than those from the experimental group (i.e., 74.1%). This result implies that OBE method can improve the learning performance of students.

V. CONCLUSION

In this paper, we investigate the impact of outcome-based education (OBE) method on information technology (IT) education. We have implemented OBE method in three IT related majors for more than two years in our faculty. In this work, we select two representative courses in Software Technology and Application program in our faculty; one course is the prerequisite of the other one. We give the course descriptions and analyze the mappings from learning activities to learning outcomes. Experimental results demonstrate that OBE method can greatly improve the learning effectiveness of students, especially in problem-solving capability in software design and implementation. In the future, we will investigate the impact of OBE on other courses and the whole curriculum.

REFERENCES

[1] Bouslama, Faouzi, Azzedine Lansari, Akram Al-Rawi, and Abdullah Abonamah. "A novel outcome-based educational model and its effect on student learning, curriculum development, and assessment." *Journal of Information Technology Education: Research* 2, no. 1 (2003): 203-214.

[2] Wong, Gary KW and H. Y. Cheung. "Outcome-based teaching and learning in computer science education at sub-degree level." *International Journal of Information and Education Technology* 1, no. 1 (2011): 40.

[3] Bansal, Srividya, Ajay Bansal, and Odesma Dalrymple. "Outcome-based Education Model for Computer Science Education." *Journal of Engineering Education Transformations* 28, no. 2 & 3 (2015): 113-121.

[4] Rigby, Steve, and Melissa Dark. "Using outcomes-based assessment data to improve assessment and instruction: A case study." *ACM SIGITE Newsletter* 3, no. 1 (2006): 10-15.

[5] Au, Oliver, and Reggie Kwan. "Experience on outcome-based teaching and learning." *Hybrid Learning and Education* (2009): 133-139.

[6] Pang, Mary, To Ming Ho, and Ryan Man. "Learning Approaches and Outcome-Based Teaching and Learning: A Case Study in Hong Kong, China." *Journal of Teaching in International Business* 20, no. 2 (2009): 106-122.

[7] Brady, Laurie. "Outcome-based education: a critique." *The Curriculum Journal* 7, no. 1 (1996): 5-16.

[8] Havnes, Anton, and Tine Sophie Prøitz. "Why use learning outcomes in higher education? Exploring the grounds for academic resistance and reclaiming the value of unexpected learning." *Educational Assessment, Evaluation and Accountability* 28, no. 3 (2016): 205-223.

[9] Wang, Haodong. "Enhancing Java programming teaching effectiveness for u-shaped class students." *Journal of Computing Sciences in College* 31, no. 1 (2015): 117-125.

[10] Cooper, Stephen, Lillian Cassel, Barbara Moskal, and Steve Cunningham. *Outcomes-based computer science education*. Vol. 37, no. 1. ACM, 2005.