Int. J. Engng Ed. Vol. 20, No. 3, pp. 494–502, 2004
Printed in Great Britain.

0949-149X/91 $3.00+0.00

An Assessment Strategy to Determine Learning Outcomes in a Software Engineering Problem-based Learning Course*

GEORGE G. MITCHELL and JAMES DECLAN DELANEY
Department of Computer Science, National University of Ireland, Maynooth, Ireland.
E-mail: georgem@cs.may.ie

This paper briefly explores the role of experiential learning in software engineering education, focusing on Problem-based learning. An existing assessment strategy for grading individual students in a small group Problem-based learning setting is described. Although the student grades obtained may be a reflection of the course success, and standard questionnaires are also employed to monitor student feedback, the authors devised a method to determine how the students themselves perceive the success of the course in terms of their own learning outcomes. As well as complementing the existing assessment strategy, this would allow them to evaluate the possibility of integrating student self-assessment into the overall assessment strategy and would act as a valuable feedback mechanism in fine-tuning the course. The results indicate that students perceive a marked increase in their knowledge as defined by the course curriculum. In addition, there is a significant difference between how course facilitators grade the students and how the students rate their own knowledge. Interestingly, no obvious correlation was found between the academic results of the students at the end of the previous year and their subsequent results in the Problem-based learning course.

INTRODUCTION

WITHIN undergraduate software engineering education, learning-by-doing or experiential learning has always been important. Individual capstone projects during the final year are almost universally accepted, while group-based projects are becoming more common across all years of software engineering degree programs in many universities [1–6]. One important approach to providing a group-based project experience is Problem-based learning (PBL), a collaborative learning process where students work in groups to solve real software engineering problems in a simulated work environment. One of the key challenges associated with small group PBL is to award students grades that accurately reflect both their collaborative effort and their individual contribution to the team project. This grade, together with standard feedback questionnaires, may reflect the success of the course in meeting the course curriculum objectives and the authors believe that this can be complemented with an approach that integrates how students perceive their own learning outcomes.

In this paper the authors present the method they have devised to determine how students perceive their own learning outcomes following a small-group PBL software engineering course. They give sample data and provide an analysis of this data to illustrate the validity of their approach. Before this, we describe the approach adopted by the authors to grade individual students in a group Problem-based learning setting. To set the scene, the next section introduces experiential learning in software engineering education, focusing in particular on PBL and the documented evidence for the effectiveness of PBL in teaching software engineers.

EXPERIENTIAL LEARNING AND PROBLEM-BASED LEARNING IN SOFTWARE ENGINEERING

There is general agreement between industry, universities and accreditation bodies that learning-by-doing should be given more emphasis in software engineering education [4, 7–10]. Learning is the process whereby knowledge is created through the transformation of experience [11]. Approaches to providing this concrete experience differ, ranging from the traditional 'chalk-and-talk' paradigm [9], where a lecturer presents information to a group of students, to Problem-based learning, where students themselves seek out the information they require to solve a given problem. The following paragraphs summarise

* Accepted 18 November 2003.

some of these approaches in software engineering, concluding with a discussion of Problem-based learning.

- Studio approaches are based on the concept of 'reflective practice' introduced by Schön [12]. Kuhn *et al.* describe studio-based learning as a technique that simulates how architects evaluate and assess previous architectural solutions devised by both themselves and by others to design and construct new buildings [13]. When this approach is applied to software engineering, learners are given a large software engineering project to conduct. During subsequent tutorials they meet to analyse their approach to solving the problems and to analyse solutions others have adopted in the past. They learn through this continuous reflective process.
- Gonsalvez and Atchison have implemented a similar 'studio-based approach,' where 25% of the content of the information technology degree program is dedicated to practical work sessions during which students experience the development of software systems under industrial conditions [14]. Their approach differs from that of Kuhn *et al.* in that they employ formal teaching to teach new course material.
- The 'learner-centered approach' of Seffah and Grogono [15] focuses on the needs, skills and interests of software engineers. It is a distance learning approach for retraining unemployed engineers in software engineering, using e-mail, Internet, FTP and discussion groups. The course is based on a multiple project approach, with each project building on the previous one. In this way learners build upon their past experience, as well as sharing their experiences with other learners.
- Jeffery describes a 'project-oriented approach' to teaching software engineering [16]. This approach uses a chalk-and-talk paradigm with regular group discussions on a single assignment. The approach is based on the spiral model of software development and consists of five iterations of three weeks each.
- In the 'real-world object oriented design' approach of Ouyang and Wang, a single group software engineering project is executed in parallel with two separate courses in networking and software engineering [17]. The courses are delivered using the chalk-and-talk approach. In contrast, little or no guidance is given to the learners with the project, which involves the development of an object-oriented system under real-world conditions.

The common characteristics among these approaches are learner focus, reflective practice and real-world simulation. These are also key aspects of Problem-based learning (PBL), a generic teaching technique that originated in McMaster Medical School in Canada during the late 1960s [18]. Prior to McMaster, Michigan State University had proposed a technique called 'focal problems' based on similar ideas, but which never became popular [18]. PBL reverses the traditional approach to learning and has been classified as a category of experiential learning [19]. It presents the student with a real-world problem and the student must explore the solution domain in a self-learning capacity. Thus, the students learn the course material through guided practical application of their knowledge to a suitable task. Its introduction into educational courses is related to a constructivist theory of learning [18, 20], where a learner's current knowledge is challenged and, through interaction with other people and with the broader environment, the learner constructs new knowledge. The lecturer's role is to scaffold the students' learning process so that the lecturer becomes a facilitator. Small group PBL is becoming an increasingly popular approach to teaching software engineering skills and this is discussed in detail by Ellis *et al.* [21] and Woods [22].

However, within software engineering and computer science, mixed views regarding the effectiveness of PBL have been documented. McCracken & Waters describe the application of PBL to a software development course [23]. They identified a number of shortcomings in the PBL methodology and performed an ethnographic study to determine the students' viewpoint. This study was conducted by observing and by audio recording of the meetings held between students. Among their conclusions they advise that learners need to be given skills for team building, that learners need assistance in identifying their individual learning goals and that they need to be taught to distinguish product from process. Fekete and Greening describe the application of PBL to teaching first-year computer science subjects and consider such an approach so successful that they have adopted it as the basis for facilitating first-year learning of computer science [24, 25]. Cavedon and Harland explored a PBL approach to teaching artificial intelligence and state that although necessary, the appropriate form self-assessment should take/needs more work [26]. Studies of the outcomes of PBL in other domains have also been documented [27–30].

We note in passing that any evaluation of PBL in a real-world situation may be subject to the Hawthorne Effect [31]. This means that the positive attitude adopted by students and academic staff to the PBL approach may, in part, be a direct result of the psychological effect of participating in a new teaching paradigm or being subjected to experimental evaluation.

Thus far we have explored teaching approaches that encourages experiential learning in software engineering education. In the following sections a new course assessment strategy for small-group PBL is presented and analysed within the context of a software engineering degree program. As a first step, we explain the assessment strategy that has been adopted to evaluate student performance on the course and then proceed to examine the success of the course in terms of the learning outcomes as perceived by the students themselves.

ASSESSING STUDENTS IN A PBL GROUP SETTING

One of the key difficulties in implementing a PBL course is to formally assess and grade the learners [20, 32]. Assessment provides continuous feedback to the learners regarding their progress and to the facilitators regarding the learning outcomes in relation to the pre-established learning goals. This allows the course to evolve and change and gives the facilitators the chance to redirect learner efforts. The assessment issue within PBL is discussed in detail by Savin-Baden [33] and she notes the difficulties associated with the facilitator acting as an assessor, in particular in relation to its effect on the power dynamics within a PBL group.

The experience of the authors in setting up and facilitating a small group PBL course as part of a third year computer science and software engineering degree program [34, 35], has led them to identify four core issues that need to be addressed when assessing students in a group project setting. These are:

1. Assessment of the group performance.
2. Assessment of the individual contribution to the group.
3. Assessment of the project deliverables.
4. Assessment of the course success.

In the present study, assessment of the group, the individual and the project deliverables in the small-group PBL project course, was centred on three skills groups, which were identified from the course objectives. Following identification of the skills groups, various assessment techniques were chosen and employed to assess each of them. The skills groups and associated assessment techniques are summarised in Table 1.

The assessment techniques outlined above focused on the group, the individual and the project deliverables and enabled the facilitators to grade the learners based on a weighted sum of a group and an individual mark. In previous years, the individual student grades obtained using this method were combined with standard end-of-course feedback questionnaires to determine the course success. However, in the course being discussed in this paper the authors were also interested in gauging the success of the course from the learners' point of view, in particular to determine whether they perceived that the learning outcomes were being achieved. This would allow them to evaluate the possibility of integrating student self-assessment into the overall assessment strategy and would act as a valuable feedback mechanism in fine-tuning the course and choosing suitable problems in the future as well as complementing the existing assessment strategy. They therefore devised a technique to measure how individual students perceive the development of their own learning over the duration of the course. The next section describes this technique.

AN EXPERIMENT WITH A NEW ASSESSMENT STRATEGY

Assessment objectives

In PBL it is hoped that the problem drives the learning process [21]. However, research demonstrates that in general assessment tends to drive the students' learning [36]. Savin-Baden [33] highlights the fact that the assessment methodology must fit the type of learning promoted by PBL and is an important part of the learning process itself. As a result, the assessment methodology employed needs to be linked to the teaching methodology adopted and it should be planned at the same time as the course itself is being planned [37]. In any assessment procedure it is important to identify the key assessment objectives [38, 39]. In the small-group software engineering PBL course under discussion, the assessment objectives focus on implementation skills, teamwork skills and problem-solving skills as determined by the course objectives and summarised in Table 1.

The success of the course in meeting the initial

Table 1. Summary of course objectives, skills groups, and assessment techniques

Course Objective	Skills Group	Assessment Technique	Mark Awarded
Implement a software design specification and to produce software documentation based on best practice	Implementation skills	Final product and relevant documentation (summative assessment by facilitators)	30% (group)
Operate in a team environment; to contribute to the team; to organise the team and assign roles and responsibilities and to integrate any industrial experience with their theoretical knowledge	Teamwork and leadership skills	Group presentation and interview (summative assessment by the course facilitators and a third party)	40% (group)
Think through a problem, analyse a situation, deal with pressure and communicate with clients	Analytical thinking, problem solving and inter-personal skills	Individual reflective journal, individual interview and peer/self assessment feedback forms (formative assessment)	30% (individual)

objectives is normally assumed to be reflected in the final mark awarded by the facilitators to each student, a higher mark meaning greater success. However, in keeping with the PBL paradigm, facilitators and learners are co-evaluators. This co-evaluation is usually achieved by a process of peer and self-assessment of students, together with anonymous feedback forms [22, 40]. With a view to incorporating a learner self-assessment mark into the final grade of future courses, the authors set out to determine how students evaluated themselves in terms of perceived learning outcomes. Two core objectives of learner self-assessment were identified as part of the small-group PBL software engineering course. These were:

1. To assess the *knowledge gained* as perceived by the learners.
2. To provide feedback from the students to the facilitators on material which should be covered in tutorials and integrated into the assigned problem in future years.

Knowledge gained refers to the software engineering knowledge the learners acquired during the small group PBL project course. This knowledge was partitioned into three broad knowledge areas as follows:

- knowledge carried forward;
- knowledge acquired;
- knowledge consolidated.

Three key software engineering subject areas were assigned to each of these knowledge areas. Table 2 summarises both the knowledge and subject areas.

The authors aim to develop an appropriate assessment methodology to determine the perceived student learning outcomes in each of the knowledge and subject areas and this is described in the following subsection.

Assessment of perceived learning outcomes

To assess perceived learning outcomes, a measure of the students' perceived knowledge at the beginning of the course must be compared with their perceived knowledge at the end of the course. Such a measurement fits well with PBL, which emphasises the importance of reflection and self-assessment.

An initial benchmark for comparison with the final perceived learning outcomes was obtained by choosing a suitable assessment methodology after examining a number of assessment techniques used in education such as sentence completion, short-answer questions, questionnaires with a Likert scale and multiple-choice questionnaires (MCQ) [19, 41].

To facilitate a quantitative analysis of the gathered data, a customised self-assessment questionnaire with a Likert scale was used. All of the questions followed a similar format. Learners were asked to evaluate their knowledge in a number of topics by giving each a score.

The self-assessment strategy aimed to assess the three broad knowledge areas described earlier. Each knowledge area consisted of three subject areas, which in turn consisted of a number of relevant topics. In all, there were 64 topics in nine subject areas. The topics were selected based on the course objectives and the curriculum content and learners were required to give themselves a score reflecting their knowledge in each topic. Some of the topics were chosen to assess the validity of learner responses. For example, in 'knowledge acquired' topics relating to CASE tools, UML and UML Modelling Tools were interrelated and a consistent rating would be expected. Other topics were purposely included, although the students would have had no reason to study those topics as part of the degree curriculum. In such cases the ratings would be expected to be very low. Examples of some topics that learners had to evaluate within three subject areas are given in Table 3. An important aspect of the forms was that the Likert-style scores were extended to range between 0 and 1000 instead of the standard 0 to 5. Zero indicated very poor or strongly disagree while 1000 indicated excellent or strongly agree. A scale from 0 to 1000 was chosen to avoid neutral answers, which may result from using a narrower scale [42]. However, there is the risk that the same score has a different value for different learners. This risk is minimised by using a difference value rather than an absolute value to measure the learning outcome.

The chosen assessment method forced the learners to give numeric responses and this facilitated the quantitative analysis of the data. The perceived learning outcome for each subject area was calculated by subtracting the initial score from the final score for each topic, and then averaging the scores

Table 2. Knowledge areas and subject areas

Knowledge area	Subject areas	Comment
Knowledge carried forward	• Network protocols • Operating systems • Programming skills	These are knowledge areas that the students should have acquired prior to the group project course.
Knowledge acquired	• Software engineering • Databases • Web authoring	This is the primary pedagogical objective of the PBL course.
Knowledge consolidated	• Software use • Resource use • Communication skills	Knowledge that transfers from being theoretical to being practical

Table 3. Learners gave a score to each topic between 0 and 1000 to reflect their knowledge of that topic

Knowledge area	Subject area	Topic
Knowledge acquired	Software engineering	CMM
		VDM
		Project Management
		Object Oriented Design
	Databases	SQL
		database type 1
		database type 2
	Web Authoring	HTML
		CSS
		ASP
		PERL

of all topics across a single subject area for each learner. An overall perceived learning outcome figure was obtained by averaging across all subject areas. These are represented mathematically in Equations (1) and (2).

$$P_s^k = \frac{1}{n}\sum_{j=1}^{n}(T_f^j - T_i^j) \quad (1)$$

$$P_o = \frac{1}{m}\sum_{k=1}^{m} P_s^k \quad (2)$$

where,

P_s^k = perceived learned outcome in subject area k;
n = number of topics in subject area k;
T_f^j = the final score for topic j;
T_i^j = the initial score for topic j;
P_o = overall perceived learning outcome;
m = number of subject areas.

The results are analysed and discussed in the following section.

ASSESSMENT RESULTS AND DISCUSSION

The results of the initial and final questionnaires were tabulated and analysed. The perceived learning outcome for each subject area was calculated as described previously. In all, 44 learners participated in the self-evaluation process. An initial analysis was made by comparing the individual mark awarded by the facilitator to the individual result obtained by that learner across all curriculum subjects at the end of the previous year based mainly on a traditional teaching paradigm. It might be expected that academically strong students would also perform well under a PBL paradigm. A second plot was made comparing the overall self-perceived learning outcome for each learner with the facilitator's evaluation of the same student during the PBL course. These two plots are shown in Fig. 1 and Fig. 2 respectively.

From Fig. 1 we have identified three distinct groups (A, B and C) and two individual students (D and E) for analysis purposes. Most of the students belong to groups A and B. Group A are those students who performed satisfactorily under traditional teaching methods, but who have performed significantly better when PBL was employed. These students expressed satisfaction with the team-based environment and benefited from collaboration with stronger students on a continuous basis during the entire project. Group B are those students who performed well under

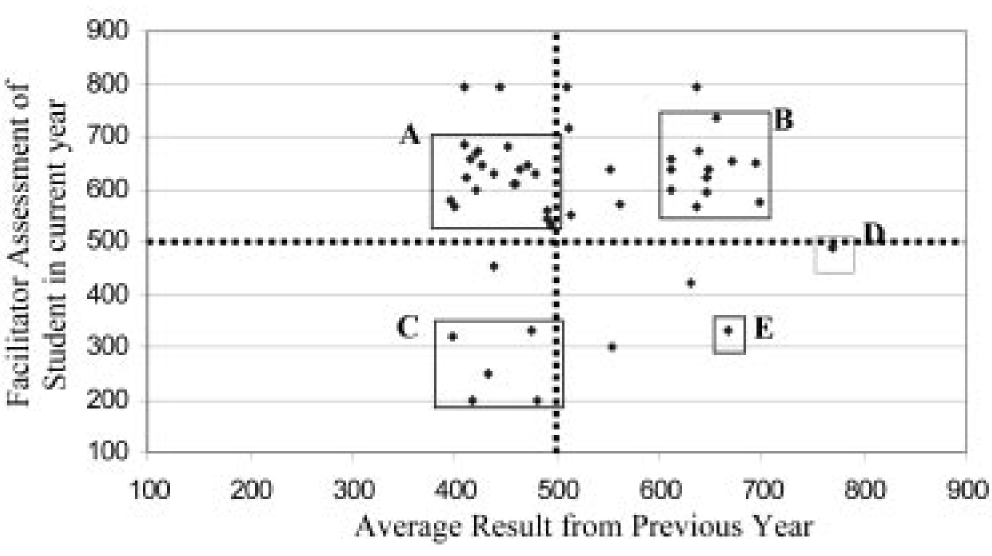

Fig. 1. Plot of average result across all subjects from previous year against final facilitator mark in PBL-based course for each learner.

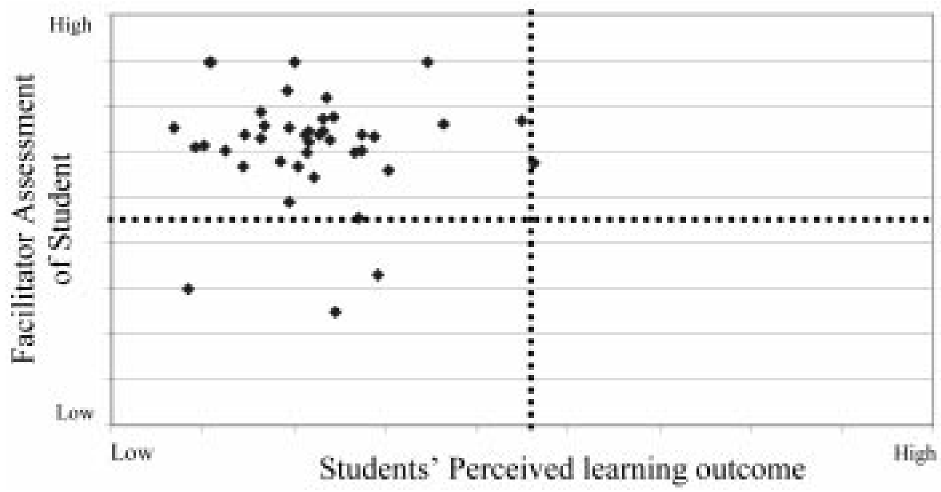

Fig. 2. A plot of perceived learning outcome (difference between start and end of course) against final facilitator grade for each student.

both the PBL and the traditional teaching approaches. We consider them to be solid workers and are high achievers irrespective of the teaching methodology or assessment technique employed. In contrast group C indicates those students who performed poorly in both methods. D and E indicate two students who were academically strong under traditional teaching methods during previous years but who obtained a low grade in the PBL course. On speaking to these students we found that they were unhappy with any form of assessment other than end of semester written examination, that they lacked enthusiasm and motivation because they had lost interest in their degree program.

Figure 2 illustrates that there is no apparent correlation between students' perception of their own learning outcomes and the facilitators' perception of the student learning outcomes. Almost without exception the students' perceived learning outcomes is lower than that perceived by the facilitators. This means that either the students underestimate their own learning outcomes, or the facilitators overestimate the students' performance. It should be noted that the facilitators' grades are based on an absolute mark given at the end of the course, whereas the students' perceived outcome is based on a difference between their evaluation at the start and end of the course. For this reason, the low evaluation by the learners may be explained by students overestimating their initial knowledge in a particular topic or underestimating their final knowledge in the same topic. If this were true, it would suggest that during the course of the project students realised that their knowledge was less than they had perceived it to be. Such a realisation would be important, as people cannot learn what they perceive they already know.

Figure 3 depicts the improvement that the students perceived in their own knowledge in each of the nine subject areas and in the three knowledge areas. The improvement in the knowledge acquired area exceeded the improvement in

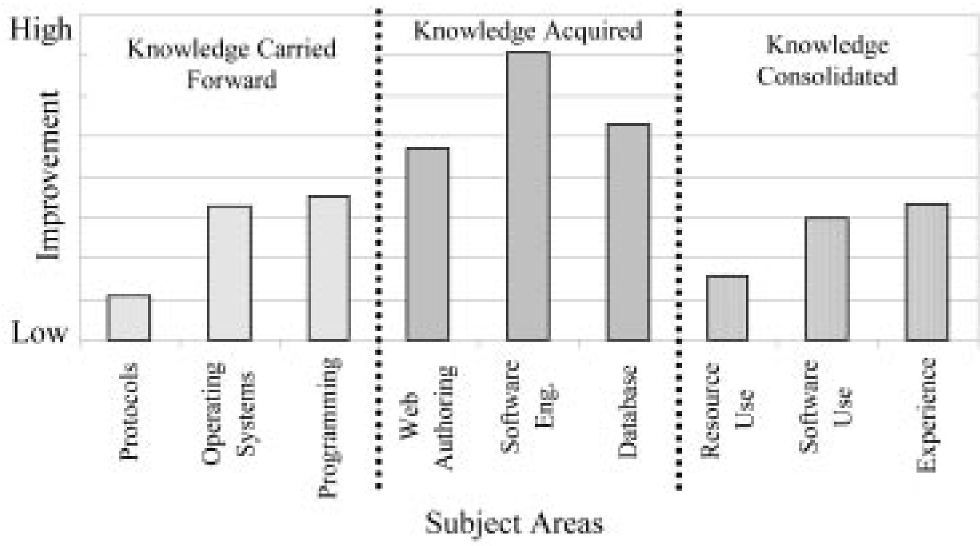

Fig. 3. Student-perceived improvement for the three knowledge areas and for the subject categories within those knowledge areas.

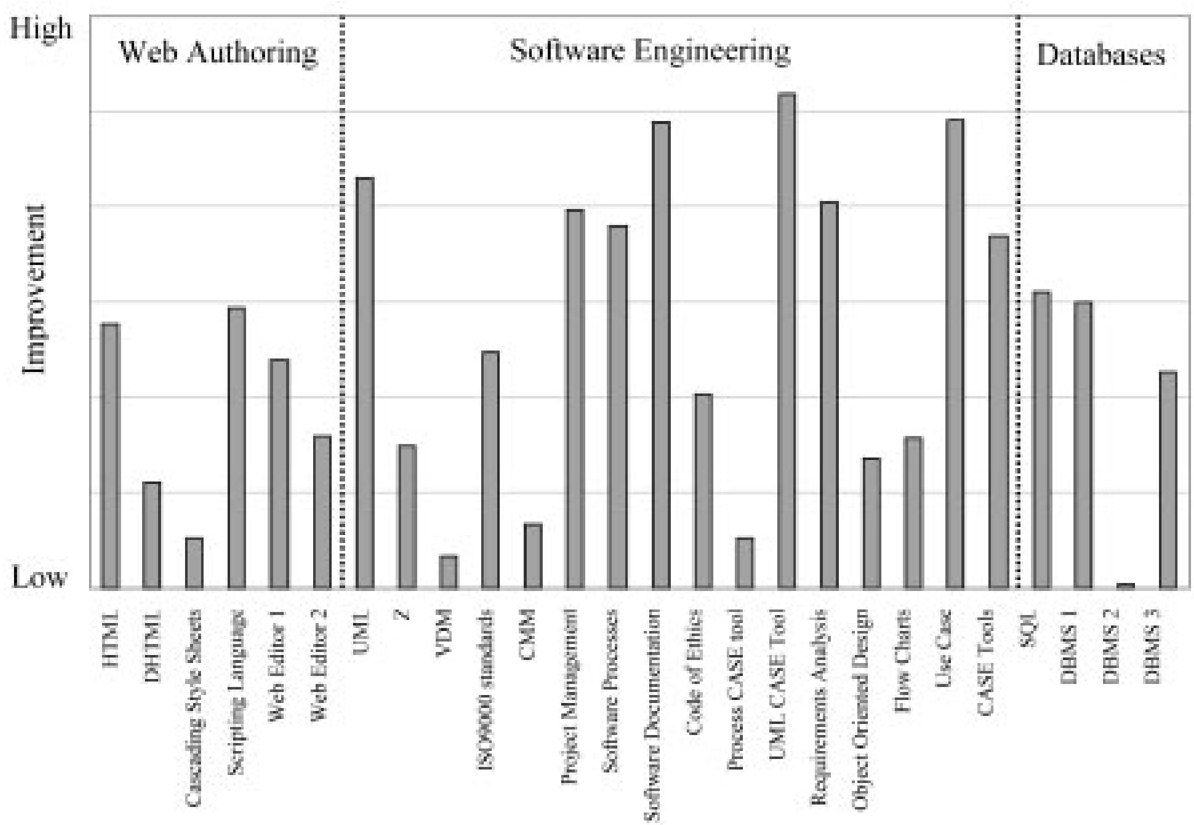

Fig. 4. Learner perceived knowledge increase for all questions in the knowledge acquired area that comprises the web authoring, software engineering and database subject areas.

the other two knowledge areas. Within the knowledge acquired area the students believed that they improved the most in the software engineering subject area. This improvement shows that the learners themselves were motivated to learn software engineering by encountering the concrete difficulties of developing a software system in a team setting. The role of the facilitator was to survey the knowledge areas being explored and guide learners in the appropriate direction. The improvement in the other two knowledge areas was more modest and naturally reflects the fact that the students focused on the knowledge needed to solve the problem at hand.

A more detailed breakdown of the subject areas can be made by examining the average score for each question. To illustrate this we analysed the knowledge acquired area and the three subject areas that comprise it. The perceived knowledge acquired in each topic is shown in Fig. 4. From this figure we note that in some topics the improvement is minimal. These correspond to areas that the learners did not have to study to complete the project. We also note that learners naturally focus on and improve in topics that helped them to successfully complete the project such as UML, documentation and requirement analysis. This reflects the importance that should be attached to selecting a PBL project. The choice should reflect the objectives of the course curriculum. The key areas of software engineering process show the greatest increase, indicating that learners are concentrating on the process rather than on the product to complete the project.

CONCLUDING REMARKS

This paper examined experiential learning in the context of software engineering education and, in particular, focused on a category of experiential learning called Problem-based learning. Assessment is a key issue in PBL and four core issues that need to be addressed when assessing students in a group project setting were identified. These were assessment of group performance, individual contribution, project deliverables and the course success. The importance of integrating student self-assessment into the overall grade, and the importance of obtaining feedback regarding the learning outcomes of the course, motivated the development of a technique to measure student-perceived learning outcomes. This technique was described in detail and test results were presented and analysed.

The authors believe that the assessment of perceived student learning outcomes is an important aspect in ascertaining the overall success of the course and the results they presented indicate that students are aware of increasing their body of knowledge following the PBL group project. The results also show that the learning outcomes as perceived by the students and the grades awarded by the facilitators are uncorrelated, but that the perception students have regarding their learning outcome is lower than that of the facilitator. This suggests that the students' evaluation would have to be weighted if it is to be used as part of a grading scheme. In addition there is no correlation between how students perform across subjects at the end of the previous academic year and their final grade in the PBL course. In general, students performed better in the PBL course. The students perceived an increase in knowledge in the key subject areas and topics that were part of the course objectives, showing that they are aware of acquiring and increasing their knowledge, and that they study topics that help them solve the problem. This underlines the importance of the choice of problem in focusing the self-learning process. It also emphasises the importance of the facilitator in guiding the learners, so that they avoid pitfalls that absorb time with very little return.

This paper describes a technique for measuring how students rate their own perceived learning outcomes. A future study might investigate if this form of self-assessment would work if the students knew it was part of a graded assessment strategy. We have also shown that students perceive an increase in their software engineering knowledge following a small-group Problem-based learning project. However, we are concerned about students' initial low perception of their own knowledge. What are the reasons for students having a low perception of their own knowledge? It may be that facilitators need to give a higher level of continuous feedback to students or it may be because students are not fully aware of their learning objectives. In the meantime we have shown that students perceive an increase in their software engineering knowledge following a small-group Problem-based learning project.

Acknowledgements—Thank you to Sean Delaney, BEd, MEd, Coláiste Mhuire Marino, Ireland, for his assistance, guidance and recommendations. Thank you to the reviewers of the paper who provided insight, direction and constructive advice.

REFERENCES

1. Boyer Commission on Educating Undergraduates in the Research University, *Reinventing Undergraduate Education: A Blueprint for America's Research Universities*, Carnegie Foundation for the Advancement of Teaching, January 1998.
2. S. L. Duggins and B. B. Thomas, An historical investigation of graduate software engineering curriculum, *Proc. 15th Conf. Software Engineering Education and Training (CSEET)*, Kentucky, USA, February 25th–27th 2002.
3. S. L. Collofello and M. Hart, Monitoring team progress in a software engineering project class, *29th ASEE/IEEE Frontiers in Education Conf.*, San Juan, Puerto Rico, November 10th–13th, 1999.
4. M. J. Oudshoorn and K. J. Maciunas, Experience with a project-based approach to teaching software engineering, *Proc. Southeast Asian Regional Computer Confederation 5th Conf. Software Engineering Education*, Dunedin, New Zealand, November 1994, pp. 220–225.
5. S. Henry, A project-oriented course on software engineering, *ACM SIGCSE Bulletin*, **15**(1), Feb 1983, pp. 57–61.
6. L. M. Northrop, Success with the project-intensive model for an undergraduate software engineering course, *Proc. 20th SIGCSE Symp. Computer Science Education*, 1989, pp. 151–155.
7. M. Sebern and M. Lutz, Developing undergraduate software engineering programs, *Thirteenth Conf. Software Engineering Education and Training*, 6–8 March, 2000, Austin, Texas, USA, pp. 305–306.
8. E. Wang, R. Wirtz and M. Greiner, Simulating corporate project engineering for freshmen, *Proc. Frontiers in Education Conf.*, November 1998, pp. 1313–1318.
9. Web Site of the IEEE Computer Society Educational Activities board: http://www.computer.org/education/
10. A. J. Kornecki, I. Hirmanpour, M. Towhidnajad, R. Boyd, T. Ghiorzi and L. Margolis, Strengthening software engineering education through academic industry collaboration, *10th Conf. Software Engineering Education and Training (CSEET '97)*, Virginia Beach, VA April 13th–16th, 1997.
11. D. A. Kolb, *Experiential Learning: Experience as the Source of Learning and Development*, Prentice-Hall, 1984, ISBN: 0132952610.
12. Schön, *Educating the Reflective Practitioner: Towards a New Design for Teaching and Learning the Profession*, San Francisco, Jossey-Bass, 1987.
13. S. Kuhn, J. E. Tomayko, O. Hazzan and B. Corson, The software studio in software engineering education, *Proc. 15th Conf. Software Engineering Education and Training (CSEET'02)*, 2002.
14. Gonsalvez C. and Atchison M., Implementing studios for experiential learning, *Proc. Australasian Computing Education Conf.*, December 2000, Melbourne, Australia, pp. 116–123.
15. A. Seffah and P. Grogono, Learner-centered software engineering education: from resources to skills and pedagogical patterns, *Proc. 15th Conf. Software Engineering and Training (CSEET'02)*, Covington, Northern Kentucky, Feb 25–27, 2002.
16. C. Jeffery, Tight spiral projects for communicating software engineering concepts, *Proc. 3rd Australasian Conf. Computer Science Education*, July 8–10, 1998, The University of Queensland, Australia, pp. 136–144
17. Y. Ouyang and S. S. Wang, Real-world object-oriented design experience for computer science students, *29th ASEE/IEEE Frontiers in Education Conference*, San Juan, Puerto Rico, November 10–13, 1999.

18. G. Camp, Problem-based learning: a paradigm shift or a passing fad? *Medical Education Online Electronic Journal*, **1**(2), 1996.
19. S. Warner-Weil and I. McGill, (eds) *Making Sense of Experiential Learning: Diversity in Theory and Practice*, Milton Keynes: Open University Press (1989).
20. Tony Greening, Scaffolding for success in PBL, *Medical Education Online Electronic Journal*, **3**(4) 1998, Available from URL http://www.Med-Ed-Online.org.
21. Ellis A.(chair), Working Group on Problem Based Learning, *Resources, Tools, and Techniques for Problem Based Learning in Computing, ITiCSE 1998*, Dublin, Ireland, pp. 41–56.
22. D. R. Woods, *Problem-based Learning: Helping your Students gain the Most from PBL*, Waterdown, Canada (1995) ISBN 0-9698725-1-8.
23. M. McCracken and R. Waters, Why? When an otherwise successful plan fails, *ITiCSE 1999*, Cracow, Poland, pp. 9–12.
24. A. Fekete and T. Greening, Conveying technical content in a curriculum using Problem-based learning, *ACSE 1998*, Brisbane, Australia, pp. 198–202.
25. T. Greening, J. Kay and J. H. Kingston, Results of PBL trial in first-year computer science, *ACSE 1997*, Melbourne, Australia, pp. 201–206.
26. Cavedon L. and Harland J., Problem-based learning with technological support in an AI subject: description and evaluation, *ACSE 1997*, Melbourne Australia, pp. 191–200.
27. L. Berkson, Problem-based learning: have the expectations been met? *Academic Medicine*, **68**, 1993, pp. S79–S88.
28. D. T. Vernon and R. L. Blake, Does Problem-based learning work? A meta-analysis of evaluative research, *Academic Medicine*, **68**, 1993, pp. 550–563.
29. M. A. Albanese and S. Mitchell, Problem-based learning: a review of literature on its outcomes and implementation issues, *Academic Medicine*, **68**, 1993, pp. 52–81.
30. A. C. Nieuwenhuijzen Kruseman, L. F. J. Kolle and A. J. J. Scherpbier, Problem-based learning at Maastricht, an assessment of cost and outcome, *J. Education for Health*, **10**(2), 1997, pp. 179–187.
31. R. H. Franke and J. D. Kaul, The Hawthorne experiments: first statistical interpretation, *American Sociological Review*, **43**, 1978, pp. 623–643.
32. P. Spasov, Look, no timetable, *Proc. 30th ASEE/IEEE Frontiers in Education Conf.*, October 2000, Kansas MO, pp. S1B12–S1B17
33. M. Savin-Baden, *Facilitating Problem-based Learning Illuminating Prospectives*, SRHE and Open University Press, Milton Keynes, UK (2003) ISBN 0335210546.
34. J. D. Delaney, G. G. Mitchell and S. Delaney, Software engineering meets Problem-based learning, *The Engineers Journal*, **57**(6), July/August 2003, pp. 57–59.
35. J. D. Delaney and G. G. Mitchell, PBL applied to software engineering group projects, *Int. Conf. Information and Communication in Education (Proc. ICTE2002)* (Eds: Méndez-Vilas and J. A. M. González) book series: *Information Society*, Government of Extremadura, 13–16 November 2002, Badajoz, Spain, pp. 1093–1098, ISBN 84-95251-70-0.
36. P. Ramsden, *Learning to teach in Higher Education*, London: Routledge (1992).
37. N. E. Gronlund, *Assessment of Student Achievement*, 7th ed., Allyn and Bacon (2003) ISBN 0205366104.
38. S. Rogers and S. Graham, *Succeeding with Performance Tasks, Projects, and Assessments: The High Performance Toolbox*, 3rd Ed., Colorado, Peak Learning Systems (2003) ISBN:1889852074.
39. J. C. Grundy, A comparative analysis of design principles for project-based IT courses, *ACSE 1997*, Melbourne Australia.
40. S. K. Lechner, Evaluation of teaching and learning strategies, *Medical Education Online Electronic Journal*, **6**, 2001, URL http://www.Med-Ed-Online.org
41. D. E. Wilkins and P. B. Lawhead, Evaluating individuals in team projects, *SIGCSE 2000*, March 2000, Austin Texas, pp. 172–175.
42. J. S. Roberts, J. E. Laughlin and D. H. Wedell, Validity issues in the Likert and Thurstone approaches to attitude measurement, *Educational and Psychological Measurement*, **59**(2), April 1999, pp. 211–233.