An Overview of the Integration of Problem Based Learning into an existing Computer Science Programming Module

J. O'Kelly, A. Mooney, J. Ghent, P. Gaughran, S. Dunne, S. Bergin
Department of Computer Science, National University of Ireland, Maynooth, Co. Kildare, Ireland.

Abstract
In this paper we present an overview of the use of Problem Based Learning (PBL) in a first year Computer Science programming module. Traditionally, PBL was not employed in any of the programming modules within the Department of Computer Science and assessment and learning for this module was on an individual student basis. We outline the problems that we encountered with our previous approach for teaching this module and our rationale for enhancing our approach through PBL.

1. Background
Computer Science as a subject has proven problematic with first year students; in a recent study of Irish Undergraduate University courses, a 26.9% non-completion rate was documented for computer related studies [1]. Prior to this academic year, the first year programming module consisted of three one-hour lectures, complemented by a three-hour lab, per week. Tutorials with voluntary attendance were also available during the lab times, and periodic class and lab exams were given as course assessment in addition to an end of year exam. With regard to learning how to program, the students' foremost difficulty seems to have been transfer of knowledge from lectures to the laboratory sessions. In general, verbal feedback indicated that the syntax (or grammar) of programming did not seem to be the problem, but the areas of semantics and algorithms were; problem definition through to solution in a specified language (in this case, Java) was the key obstacle. Students could understand specific examples of abstract concepts detailed in lectures yet could not apply them to new yet similar lab assignments. As a result, frustration among the students was often high. In order to try to combat these problems, it was felt that a change to the structure of the course and the integration of PBL workshops would benefit the students and alleviate some of their problems. This paper is composed of two main sections; the first outlining how we integrated PBL into our existing introductory programming module and the second providing a critique of our approach and recommendations.

2. Overview of our PBL implementation
In this section we present the approach used for integrating PBL into our introductory programming module. We describe the set-up process, types of problems used, the learning outcomes, group issues and assessment.

2.1 Problem criteria and problem type
We based the design of our problems on a defined set of criteria to ensure our problems provided a constructive environment for learning. Although the lecturer initially designed the workshop problems, a weekly meeting was held to discuss the problems according to a set of criteria. This set consisted of seven initiatives: each workshop problem should be engaging, engender multiple viable hypotheses, allow enquiry, be based on current curriculum, represent real-world problems, sustain engagement and provide accessible resources for subsequent learning.

The problems subsequently created fall into three broad categories. Firstly, *extendible conceptual problems,* that is problems that ensure the students in the workshops focus on core concepts of computer programming in order to solve a problem. These problems involve no programming but

require the students to understand programming related concepts. These problems also allow for increased levels of difficulty to be added to the problem once a solution is found to ensure that the problem sustains the student's interest. *Non-extendible conceptual problems* help a student to understand programming-related concepts without performing any programming. This type of problem has just one solution and is not extendible. *Programming problems* are typical computer programming problems that the group try to solve collectively. This type of problem aids the weaker student as he/she gets to see how a stronger student solves a programming problem.

2.2 Transfer of knowledge to programming problems

Knowledge can be transferred directly or indirectly depending on the type of workshop problem presented to the students. The transfer of knowledge is directly related to the learning issues in a workshop problem as Fig. 1 illustrates. Learning issues are what we anticipate the student will learn during a workshop.

As well as the learning issues shown in Fig. 1, each type of problem teaches the student to work in a team to break a problem into solvable units (step-wise refinement). Learning issues such as concept development requires the student to apply basic programming concepts ideas, for example looping, recursion, conditional statements [2], to problems of incremental difficulty. Indirect transfer of knowledge involves the student applying a concept learned in the workshop to a computer programme while direct transfer of knowledge involves the generation of a computer programme in the workshop.

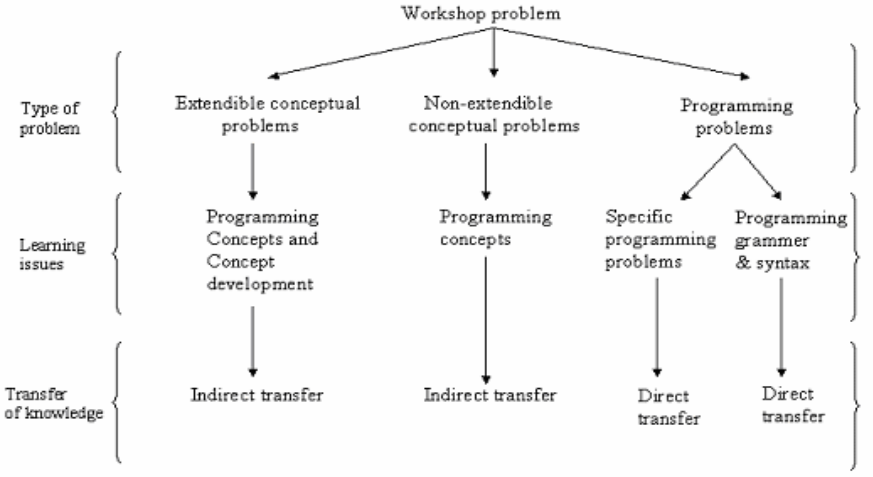

Fig 1: Workshop problems

Students taking the introductory programming module come from three distinct disciplines: Software Engineering, Arts and Science. Students were randomly placed into groups with the constraint of maintaining a gender balance. In this context we viewed a gender imbalance to occur when a group was entirely comprised of a single sex or the group contained one lone male or one lone female. Productivity is also affected by group size [3]. When deciding group size, our ideal was eight students. However due to practical restrictions, group sizes ranged from seven to ten in practice which worked to varying degrees of success. Workshop sessions were held on a weekly basis, each lasting ninety minutes. During these workshops, students were presented with an authentic problem to solve. Ground rules for behaviour and goal achievement were set by the group and reviewed regularly. In addition to functioning as a group member, internal group roles such as: chairperson, scribe, reader and archiver were assigned. These roles, detailed in Table 1, were valid only to the resolution of the

problem under investigation. Each group was entirely self contained and autonomous. Intervention by the facilitator took place only when problem-solving *process* broke down.

Roles	Responsibilities
Chairperson	Manage the group. Ensure all group members understand problem. Encourage participation from every group member. Resolve/ address issues which may arise from contravening the group rules.
Scribe	Write down all ideas proposed by the group on a whiteboard.
Reader	Read out the problem, and any supporting documentation to the group.
Archiver	Transcribe the notes from the whiteboard and distribute to all members of the group.

Table 2: The internal group roles and responsibilities employed in our PBL implementation

Props were used to great advantage by the majority of groups, particularly sensory and kinaesthetic learners. Each group was provided with an identical prop set. A sample of some items in a typical prop set is listed in Table 2. The content of the prop set was built up during the year as both lecturers and facilitators identified items, which were thought may assist groups in visualising the particular concepts being addressed.

Props	Evidence of Use
Abacus	Problems involving problem decomposition, pattern identification and looping structures.
Cards	Assisted the discussion of complex sorting and searching techniques
Toy vehicles	Used to identify salient characteristics and behaviour of objects
Alphabet scroll	Many groups used this prop in a data encoding/decoding assignment.
Pegs	Problems involving problem decomposition pattern identification, looping and recursion.

Table 3: A selection of items in a typical prop set, coupled with evidence of prop use.

Process evaluation played a crucial part in refining our PBL approach. Our observations of individual and group performance enabled us to revise the style of problems posed, the structure of the group itself and determine the effect of individual characteristics on group dynamics. Formatively the facilitators assessed students' performance on an individual basis, including individual participation, cognitive understanding and ability to back up statements and on a group basis, including intra-group participation and quality of the final solution.

In addition, students were evaluated on their fulfilment of duties in a specified role. Furthermore, peer assessment took place both within the group and between groups. The former involved the evaluation of group performance and role performance. This enabled groups to bond better, while the latter provided opportunity for more subjective evaluation of actual work submitted. Competitive inter-group assessment was also applied during some workshops and also at a national level. Students were not summatively assessed for their performance in a workshop.

3. Critique and Recommendations

We present a critique and recommendations on three core areas: PBL process, group issues and training.

3.1 PBL Process

In this section we consider four aspects of the PBL process: (1) the use of props, (2) the problems, (3) the review process and (4) the assessments. (1) The use of props was important in our PBL implementation. However, the introduction of a new prop at a workshop was too evident to the group that it could be used in solving the problem and therefore influenced the problem-solving process. (2) As the students' ability in computer programming improved the problems became more programming related. We feel that this approach worked well and resulted in an increase in student confidence levels in their lab work. (3) The weekly review meetings often required the facilitators and lecturer to spend

considerable time refining the problem to reach agreement. We found these meetings to be beneficial to the PBL process. (4) The students enjoyed the formative workshop assessments and felt that it gave them closure on the problems and allowed them to see how other groups were performing.

3.2 Group Issues

Initially, the students were divided into groups of 7 to 10. Although, some groups of this size worked well, the facilitators observed that sometimes up to three people were uninvolved in the group process. Furthermore, it was observed that this was often shy people who needed to be encouraged before they would become involved. Prior to the commencement of Semester 2 the groups were reorganised and a maximum of 6 students were assigned to each group. This reduced number in each group encouraged the shy students to become involved. In addition, we found that the dynamics within a group was critical to how successful the group performed and that a balance of personalities helps to achieve harmony in a group. We did not observe any major gender issues and a survey indicated that over 95% of students did not feel that gender was an issue within their group [4].

3.3 Training

At the end of the first semester the facilitators and lecturers participated in a second training session on PBL. This session was valuable as it gave the facilitators an opportunity to reflect on the PBL implementation and to identify bad-habits and inconsistencies among the facilitation style. Due to time constraints we did not hold any induction for students participating in PBL. This resulted in the students not having any understanding of PBL prior to the first workshop, which we felt caused initial confusion. In future we intend to hold induction sessions prior to commencing PBL workshops. Finally, we observed that PBL has resulted in a change in learning for our students. Typically, students learn on an individual basis but the introduction of PBL has meant that they had to learn to work in a group, to compromise, to listen and to realise that there is more than one way to solve a problem.

3.4 Recommendations

Although, we promote the use of props, we recommend that all props are determined before a course commences and are available to the students at every session. We intend using more formative assessment next year and recommend its use to other institutions. We believe that gender was not an issue due to how we had organised the groups and would suggest other institutions follow a similar allocation approach. We would strongly recommend a mid-term review of the PBL approach as it refreshes the process in peoples minds and also ensures all involved are employing the same approach. We believe that student induction would be advantageous for the success of this approach and would recommend that other institutions provide induction to students undertaking such a PBL course.

4. Conclusion

In this paper we presented the integration of PBL into an existing introductory programming module. We outlined the set-up of PBL workshops, the types of problems used, group structures, learning outcomes, assessment procedures and process evaluation. Initial results from the introduction of PBL are positive and we intend to use a similar approach in the next academic year.

References

[1] Morgan, M., Flanagan, R., Kellaghan T., **A Study of Non-Completion in Undergraduate University Courses** (2001)

[2] Savitch, W., **JAVA An Introduction to Computer Science & Programming**, Prentice Hall

[3] Ellis, S., Dick P., **Introduction to Organizational Behaviour**, McGraw Hill (2000), 78

[4] O'Kelly J et al. **Initial findings on the impact of an alternative approach to Problem Based Learning in Computer Science.** Pleasure By Learning, (PBL 2004), Cancun, Mexico