Challenging the Norm in the Teaching of Practical Computer Science

David Lillis, Martina Naughton, Alex Cronin,
Joe Carthy, M-Tahar Kechadi and Rem W. Collier

School of Computer Science and Informatics
University College Dublin

{david.lillis, martina.naughton, alex.cronin, joe.carthy, tahar.kechadi, rem.collier}@ucd.ie

Abstract

The teaching of practical sessions in Computer Science frequently tends to follow a standard pattern: large numbers of students work in isolation on a particular assignment, enlisting help from whichever demonstrator is available at the relevant time. This model has a number of inherent difficulties. In some cases, each demonstrator may not have the same approach to solving the problem at hand, which can lead to confusion amongst students. Also, it is frequently the case that demonstrators find it difficult to identify those students in most need of additional assistance as the numbers involved are prohibitively large.

This paper describes the restructuring of a first year undergraduate computer science module in UCD. An Active Learning Laboratory was built, mimicking that of the University of Minnesota to allow learning to take place with a group focus: students provide support to others, share their work with their class and actively work to problem-solve both independently and with benefit to their peers. This facilitated the subdividing of classes into groups, to which a specific demonstrator was assigned, so as to bridge the gap between students and educators by helping to build stronger relationships between them. Encouraging students to work as groups aids interaction between them, and also strengthens the learning for students that aid classmates with the material.

The practical aspect of this year's COMP 10050 module involved the use of a 3D, interactive, animation, programming environment for building virtual worlds called Alice (developed in Carnegie Mellon University) that the students used to create their assignments and projects. In addition to the restructuring of the practical sessions, the course content was also altered so as to place the work in which the students engage in a better context. This includes engagement with the strong research community in the school, as well as industry professionals, so as to see interesting and practical applications of relevant technologies.

1. Introduction

Motivated by the desire to increase Computer Science retention rates amongst undergraduate students, UCD School of Computer Science and Informatics decided to encompass a number of well-document strategies in the "Software Engineering Project I" (COMP 10050) module. The objective of these strategies was to address such issues as increasing student motivation (by emphasising the "fun" side of computer science), improving their awareness of the applicability of computer science skills and diversifying the learning environment by encouraging students to learn from their peers by utilising active learning principles.

This paper describes the restructuring of the first-year undergraduate computer science module COMP 10050 for the 2008/09 academic year. This restructuring consisted of three principal changes: the introduction of the Alice programming environment, the use of Research Visits and Seminars and the building of an Active Learning Laboratory. These are discussed in detail in the following sections. The effectiveness of these changes was evaluated using a survey that students were invited to complete [1].

2. Alice Programming Environment

Alice is an innovative 3D programming environment, developed at Carnegie Mellon University (CMU), that makes it easy to create an animation for telling a story, playing an interactive game, or a video to share on the web [2]. Alice is a teaching tool designed as a revolutionary approach to teaching and learning introductory programming concepts.

This software was selected as the foundation for COMP 10050, as it has been shown that the programming visualisation environment offered through Alice is highly motivating to college students, can improve their programming skills, and can aid overall retention within Computer Science [3]. It uses 3D graphics and a drag-and-drop interface to facilitate a more engaging and less frustrating first programming experience. Moreover, working with an easy-to-use 3D graphics environment is attractive and highly motivating to today's generation of media-conscious students. Another reason it was chosen is because the 3D modelled classes and instantiated objects in Alice provide a very concrete notion of the concept of an object and support an "object-first" approach [4, 5]. Finally, the drag-and-drop editor prevents students from making syntax errors that are prevalent for beginners.

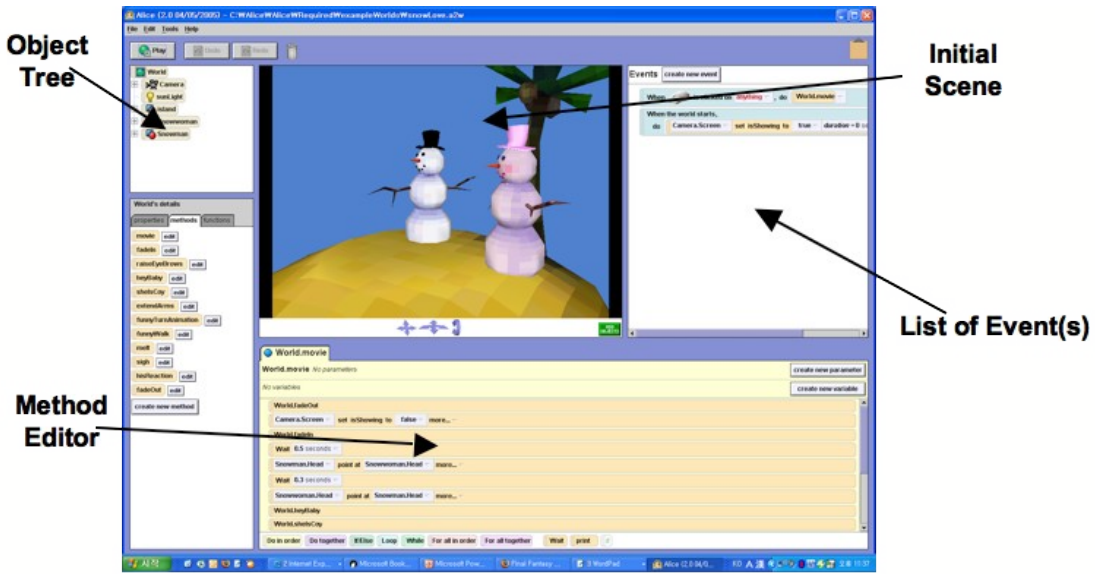

Figure 1: The Alice Programming Environment

A screen-capture of the Alice Programming environment is shown in Figure 1. The interface displays an object tree (upper left) of the objects in the current world, the initial scene (upper centre), a list of events in this world (upper right), and a method editor (lower right). The overlapping window tabs in the lower left allow for querying of properties, dragging instructions into the code editor, and the use of sound. The methods editor allows users to edit both new and existing methods that are already associated with in-built objects. All program statements written within each method are displayed using pseudo-English instead of traditional Java syntax. This allows novice programmers to develop fun animations without prior computing experience. Overall, Alice's simple, clear design, as well as the speed with which one can make changes and just 'experiment' is what makes Alice unique and valuable in an academic setting.

One of the objectives that the students must fulfil in order to complete COMP 10050 is that they must design and develop a moderately complex piece of software. To do this, the students used Alice. For these assignments, the students worked in pairs, as discussed in Section 4. They created animations that used the in-built objects and methods in Alice, but also created their own methods in order to complete the assignments. These assignments also involved the use of arrays, for loops, conditional statements, thus reinforcing basic programming concepts that the students were learning in another module named Introduction to Programming (COMP 10020) (which was being taught in parallel to COMP 10050).

The nature of Alice is such that a wide variety of interesting topics can be used for students' assignments. These include:

1. **Gaming/Simulation:** any interactive game or simulation of their choice, e.g. Tetris

2. **Interactive-based Learning:** any interactive animation that explains and describes some concept targeted at primary school students.
3. **Sport:** an animation re-creating a famous sporting event.
4. **Entertainment:** re-enact a scene of their choice from a movie or music video.
5. **Virtual Tours:** create a virtual tour of a building or tourist attraction.

At the end of the semester, a small competition was held, where students were given the opportunity to submit their worlds for presentation on the last day of term. Prizes were awarded for the best animations. Overall, the quality of the submissions was excellent, to the extent that seven top prizes were awarded.

Evaluation of Alice

Question		1 - Strongly disagree	2 - Disagree	3 - Neither agree or disagree	4 - Agree	5 - Strongly agree
1	Alice is stable and almost never crashes when I'm using it.	28%	42%	10%	14%	6%
2	I like Alice	36%	20%	18%	20%	6%
3	I would like if I could see and edit the Java code behind my animations in Alice.	4%	15%	19%	33%	29%
4	Alice is intuitive and easy to use.	16%	30%	12%	38%	4%
5	Alice helped me to understand basic programming concepts (such as For Loops, If Statements, Arrays etc.)	19%	22%	14%	38%	7%

Table 1: Survey results relating to Alice

The results of the survey that relate to Alice are presented in Table 1. With regard to whether the student enjoyed using Alice overall, it is worth comparing the responses to Questions 1 and 2. This result suggests that those who did not like Alice overall, are those students whose installations of Alice crashed quite frequently. During the course of the module, students with large animations had trouble saving their work due to memory limitations within Alice. Many students also found that Alice's performance decreased significantly as the size of the animation increased. Therefore this partly explains why a significant proportion of the students did not enjoy using Alice. However, despite this drawback, it is also reasonable to conclude that students who had little or no trouble with their Alice installations are those who did liked and enjoyed using Alice overall.

One of the design features of Alice is that it conceals the animation's underlying Java code from the user. This is an advantage for users who have no prior Java experience. However, the COMP 10050 students had completed an introductory Java module (COMP 10010) in the previous semester, and were also attending a "following-on" module to COMP 10010 in parallel with COMP 10050. Therefore, they had some programming experience, and as such are not completely novice programmers. Question 3 tries to assess whether or not the student would prefer to view the underlying Java code that sits behind their animations. The responses show that over half of the students (62%) agree that they would prefer if they could view and edit the Java code directly. One reason that may help to explain this, is that as an animation becomes very complex, the drag and drop interface is almost too limiting, and as such makes completion of complex for/while loops difficult to complete. This result suggest that for complex animations, an option to permit users to view and edit the underlying Java code directly might be a useful add-on in later versions of Alice.

With regard to Alice's usability and ability to help students to understand basic programming concepts, it is noteworthy that there is a high correlation between the responses to Questions 4 and 5 in Table 1. This suggests that those who found Alice easy to use (approx. 42%), found it beneficial when attempting to understand basic programming concepts such as objects, methods, control structures (such as if statements), repetition (i.e. the use of for/while loops), arrays/lists and the use parameterisation. However, a significant proportion of the students (approx. 46%) did not find Alice easy to use, and as a result did not feel that it help them to understand programming basics. One possible explanation for such results refers back to a point made earlier regarding Alice's simplistic drag-and-drop user interface, and its inability to facilitate the creation of complex functionality easily.

3. Research Group Visits and Seminar Series

One observation that is frequently made regarding the teaching of computer science is that students often find it difficult to understand the relevance of the material they are being taught in a practical sense. This is particular true in the early stages. As with other disciplines, at this stage students are furnished with the basic foundation skills required for the remainder of their degree course and indeed for their further careers. Unlike some other disciplines, however, the teaching of computer science in third level generally is initially aimed at novices in the field. The principal reason for this is that Irish second level education frequently contains no computer-related content. Even where this is present, computing education in secondary schools very rarely ventures beyond the teaching of word processing and other office-related skills, which are far removed from the science of computing. As a result of this, it is unfair on students to assume any computer science knowledge

prior to the commencement of the course.

Because a third-level computer science education must begin at the most basic level, there is a relatively long journey for students before they see the fruits of their work in terms of being capable of building interesting applications and systems. Additionally, the complexity of many of those applications and systems that they may be using in their everyday lives is far, far higher than anything students would be required to create during their degree programme, not least during their first year.

Research Group Visits

As an effort to bridge this comprehension gap, as part of the restructuring of the COMP 10050, it was decided to introduce the students to a variety of research groups working within the School of Computer Science and Informatics. The School has a very active research community, including over 100 postgraduate research students, along with research staff. School researchers publish hundreds of articles each year at international conferences, journals and books. As a result, there is immediate access to a large number of academics who build systems and applications of various types, in addition to being at the cutting edge of developing new technologies.

In order to gain a greater understanding of some of the practical applications of the skills being learned in their first year, students were brought to attend meetings of a number of the research groups in the school. Having the actual developers of these systems present facilitated the students' understanding, as they were given the opportunity to question them directly about interesting angles in their research, as well as being able to clarify points of confusion.

The types of research being carried out by these groups varied widely. So as to give a flavour of the type of work the students saw, here are a few of the research topics being addressed by the groups in question:

- **Search Engine Technology:** as used by companies such as Yahoo! and Google.
- **Recommender Systems:** used by companies like Amazon that build profiles of users' preferences to recommend products based on the likes of other users with similar tastes.
- **Robotics:** programming software to control robots in a real-world environment.

It can be seen from these examples that much of the work the students saw during their research visits is very closely related to technologies that they use every day.

Survey Results on Research Visits

The results of the survey that relate to the Research Visits are presented in Table 2.

Question		1 - Strongly disagree	2 - Disagree	or disagree3 - Neither agree	4 - Agree	5 - Strongly agree
6	I found my visits interesting	10%	14%	13%	48%	14%
7	I understood the material that was presented.	10%	13%	16%	51%	10%
8	I have an increased interest in the research investigated by the group that I visited.	14%	28%	23%	32%	3%
9	My research visit has increased my interest in computer science overall.	12%	13%	30%	41%	4%
10	I think that it is beneficial for undergraduate students to see active research carried out in the school.	1%	1%	16%	58%	23%

Table 2: Survey results relating to Research Visits

With regard to the level of interest and understanding exhibited by the students, it is noteworthy that there is a high correlation between the answers to Questions 6 and 7. This result suggests that although some material may have been aimed at too high a level for the students' level of expertise, it is reasonable to conclude that those students that understood the material tended to find the experience an interesting one.

Questions 8 and 9 relate to the impact on the level of interest that the students have in the work of the research group they visited and in computer science in general, respectively. Although a large proportion of students did not find an increased interest in the specific subject area of the visit, a larger proportion encountered an increased interest in computer science overall. One possible explanation for such results is that the students were not given the opportunity to choose which research group they were interested in visiting. It is possible that permitting more student input into the allocation of students to research groups will improve such figures in the future.

Finally, the results for Question 10 show a very positive result. Only 2% of students disagreed that research visits are beneficial to undergraduate students. This strongly suggests that these research visits are a worthwhile component of the module and argues for their continuation, bearing in mind the conclusions drawn from the earlier questions. That is, presenters should be more mindful of the knowledge levels of the students being presented to, and students' preferences should be taken into account when allocating students to research groups.

Seminar Series

In addition to being exposed to the research being carried out within the school, industry speakers were also invited to give seminars, with the aim of aiding them to understand the software development process in an industrial setting. In all, three seminars were given by representatives from Microsoft, Google and UCD's Human Computer Interaction (HCI) research group.

Survey Results on Seminar Series

The survey results relating to the Seminar Series are presented in Table 3 below.

Question		disagree1 – Strongly	2 – Disagree	or disagree3 – Neither agree	4 – Agree	agree5 – Strongly
11	I found the seminars interesting	6%	13%	19%	57%	6%
12	The seminars helped me to gain a better understanding of how software is developed in industry.	6%	19%	22%	45%	9%

Table 3: Survey results relating to Seminar Series

Questions 11 and 12 of the survey evaluate the students' opinion of the Seminar Series. The majority of students agreed that the seminars were interesting and that they resulted in greater understanding of the software development process in industry. However, the sizable minority that did not gain benefit from the seminars must be addressed also. Unfortunately, the survey omitted to query the students' understanding of the material being presented, which may have been a factor, as it appears to have been for the Research Visits. Also, the focus and scope of the seminars was quite broad, with some preferring to focus on the practical aspects of software development and others being of a more theoretical nature. Perhaps more consistency in the material being presented would benefit students in forming comparisons between the companies represented. In any case, it can be concluded that although most students gained benefit from the series, further engagement with the remaining students is required to discover reasons why their experience was not entirely positive.

4. Active Learning Laboratory

Based on the success of a pilot project run by the Office of Classroom Management in the University Of Minnesota (UM) the UCD School of Computer Science and Informatics (UCDCSI) decided to implement an Active Learning Lab (ALL).

An informative overview of this environment as implemented by UM can be found on their website [7]. It contains detailed descriptions, classroom diagrams, video responses and a recommendations report following a survey of the students, staff and other stakeholders [8]. The report took place in the autumn of 2007 and based on its findings a decision was undertaken by UCDCSI to implement this concept.

As illustrated in Figures 2.1, 2.2 and 3 great efforts were made to mimic UM in providing a comfortable learning environment with modern well maintained equipment. UCDCSI's ALL became available in late February 2009. Since its opening it has played host to Computer Science undergraduate practical session involving both the use of laptops and pen & paper, masters modules, human resource seminars and professional modules covering software engineering and cybercrime. In each case the group orientated nature of the lab proved very beneficial.

Figure 2.1: ALL with 4 student tables, wall monitors and the instructor panel

Figure 2.2: Student table, whiteboard and class projector screen (wall mounted monitor seen belongs to different student table)

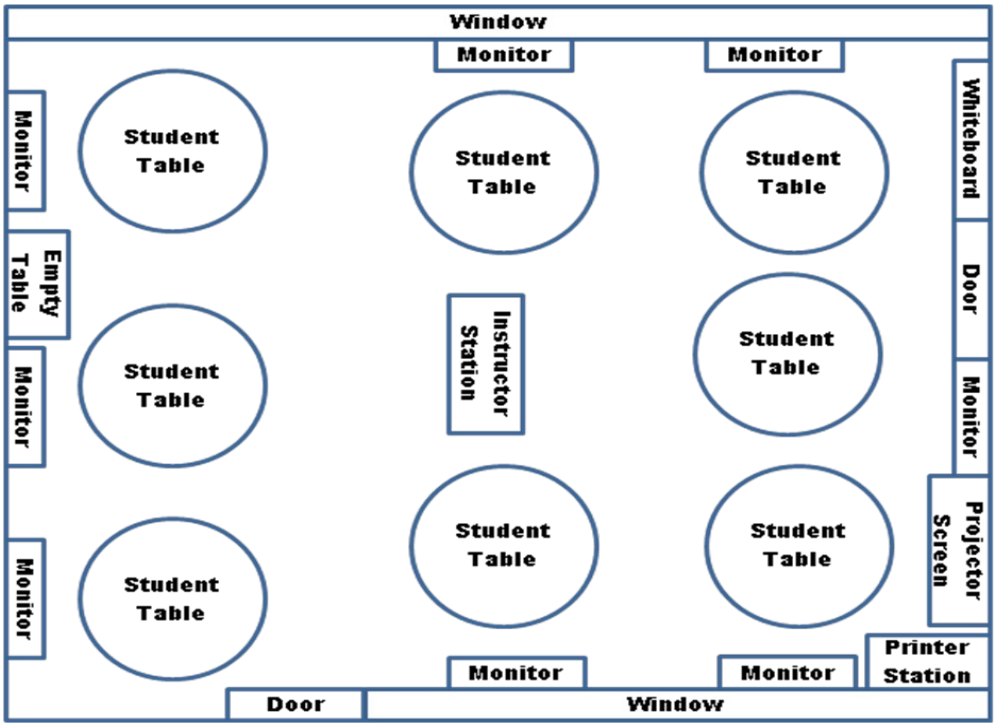

Figure 3: Diagram of UCDCSI ALL

Survey Results on Active Learning Lab

The lab environment provided by the ALL is considerably different from that of a standard lab in terms of the physical space, the demonstration support and the access to new technologies. The following tables show the survey responses relevant to the ALL. We also highlight the main differences between the ALL demonstration model adopted by COMP 10050 to complement the ALL environment and the demonstration model adopted by stand labs.

Question		disagree1 - Strongly	2 - Disagree	or disagree3 - Neither agree	4 - Agree	agree5 - Strongly
13	I find the new lab comfortable and feel that it provides ample space to carry out my practicals.	0%	3%	6%	38%	54%
14	The new layout makes it easier for me to communicate with members of my lab group and my demonstrator.	0%	0%	6%	46%	48%
15	I find that sitting at a table with the same group of students each week is beneficial.	1%	6%	20%	42%	30%
16	I find that having the same demonstrator each week is beneficial.	3%	4%	9%	43%	41%
17	I worked closely with my partner on assignments that required us to work in pairs.	28%	16%	16%	28%	13%
18	Working with a partner is better than working on my own.	19%	17%	23%	25%	16%
19	My lab group used the wall monitors in the new lab.	12%	17%	14%	45%	12%

20	I find that seeing the work of other students on the wall monitors is beneficial.	3%	1%	26%	52%	17%
21	I prefer the layout of the new lab better to the layout used in other laboratories within the School.	1%	6%	13%	36%	43%

Table 4: Survey results relating to the ALL

The physical space

As can be seen from responses to Questions 13, 14, 21 above students liked the room, the group table of 10 and preferred the ALL layout to the standard lab layout of rows of desk all facing one direction. The same room previous accommodated 64 students and now accommodates 80. Even with the increased numbers the lab seems more airy than before due to the round tables.

Demonstration Support

Question 16 and 17 responses highlight the perceived advantages of a student sitting with the same group of student each week and being supported by the same demonstrator. Below in Table 5 the key differences of the ALL and standard lab environments are highlighted followed by a more in depth discussion

	ALL	Standard Lab
Demonstrator Student Ratio	1:10	1:8
Demonstrator assists same students each session	Yes	No
Records and is aware of student attendance	Yes	No
Demonstrator is aware of current student marks	Yes	No
Demonstrator is aware of student strengths weaknesses	Yes	Somewhat
Demonstrator is aware of amount and nature of support experienced during current lab session	Yes	No
Demonstrator is responsible for and proactive in assisting students	Yes	Somewhat
Demonstrator is on first name terms with students	Yes	some
Students belong to a well defined support hierarchy (pairs, table)	Yes	Somewhat

Table 5: Demonstrator role comparison table: ALL vs. Standard Lab.

The following information is represented clearly in Table 5. A reduced demonstrator ratio can be accommodated in the ALL as the demonstrators have more clearly defined roles resulting in a more efficient use of their time. As the demonstrator is responsible for recording student attendance, marking student assignments and monitoring their progress they become acutely aware of the abilities of the students under their direction. As each demonstrator is responsible for all students at his table he must ensure to divide his time appropriately. Students who are not performing their own work between demonstrator support periods are quickly identified, those with common issues can be brought together by the demonstrator more easily for a mini tutorial. The pair environment means that if assistance has been provided to a pair they may then reinforce one another's understanding through discussion.

In a standard lab environment the sharing of solutions is often discouraged in an attempt to avoid

students simply copying another's work. The student pair must discuss their work as they are producing a joint project. As the level of the table can vary greatly it is often the case that stronger students identify solutions to aspects of the assignments and provide support to weaker students thus reducing the work load on demonstrators. The potential pitfall of one student in the pair doing the majority of the work is addressed by conducting an oral test with each student.

It is worth noting that a table and pair driven learning environment can be more demanding from the perspective of the teaching assistant:

- Initial allocation of student pairs
- Extended student absences or students switching sessions required pair reallocation.
- Fair assessment requires pair and individual assessment criteria
- With improved channels of communication more student queries are received.

The improved student satisfaction and the knowledge that students are bonding more closely with class makes make the additional workload worthwhile.

It would certainly be beneficial to having an orientation session for the demonstrators to assist them in managing the group dynamic to promote pair and table interaction, as some time may be required for them to become comfortable with the new teaching environment. For example: If a demonstrator sees himself as the sole source of knowledge for his group of 10 students he may feel under pressure to have all the answers, he should be made aware of support available from the lecturer, teaching assistant, other demonstrators and be prepared to ask pairs of students at his table to attempt to research solutions themselves and report back to the table.

Pair Work

Question 18 yielded a split decision as to whether or not working on an assignment in pairs was beneficial. In each of the first 4 lab sessions it was a requirement that the students work in pairs operating one laptop to complete the assignment. Attendance rates of stage one students are something of continuing concern for universities and in pair work poor attendance hits hard. When a partner is absent a student is encouraged to work with another pair and although in certain cases this is beneficial the dynamic of the pair relationship breaks down resulting is less support both during and after the practical sessions.

Question 17 responses indicate a split decision as to whether or not students worked in pairs. Although we do not have an established statistical correlation to say "if student and partner attended

regularly then the pair benefited" informal feedback from both regularly attending and poorly attending students support this hypothesis.

It would be our considered recommendation that group sizes be increased from 2 to 3 and that measures be taken to improve attendance rates.

Wall Monitors

Questions 19 responses indicated that some students used the wall monitors to share their work with others and some didn't. Public presentation of work is something that the majority of students would not have experienced at university before, and are understandably apprehensive about engaging. The teaching staff encouraged students to participate on a voluntary basis and we pleased to see that those who did appeared to gain confidence from the experience.

Question 20 responses show that even though there were mixed levels of enthusiasm with regard to sharing their work (Question 19), students were happy to learn from the work of others, and in the majority of cases responded with positive feedback and encouragement to those who had displayed.

5. Conclusions

We successfully re-developed the COMP10050 Software Engineering Project 1 in three ways.

Teaching programming skills through the colourful, drag and drop, user friendly "object-first", 3D programming environment Alice, allowed students to see all the aspects of program creation from a functional and object driven approach as opposed to getting bogged down in syntax. Although Alice was well received by many of the students, our evaluation also showed that, Alice is best suited for the implementation of low complexity animations by users who have no prior programming experience. Since COMP 10050 students had prior programming knowledge, higher complexity animation assignments were assigned to maintain student interest. As a result, many students encountered problems with Alice, largely caused by it memory limitations and stability issues.

Exposing students to motivational and informative seminars and postgraduate research group visits allowed students to see the rewards of pursuing our undergraduate computer science program. Allowing students to choose the research visits they attend, more closely prescribing the complexity level, and having a more consistent format to the sessions will increase the already strongly positive student responses to bring them in line with the benefits students feel can be derived from such an exercise.

Through our new Active Learning Laboratory we have provided a group work environment where

module content can be more easily discussed and, where solutions emerge from student driven interaction. It also improves the cohesion and communication quality between the student, the demonstrator and the lecturer. Students felt positive in relation to the physical space, the group work and the support, but there were mixed responses to the pairs work due to absenteeism.

All our conclusions are supported by our survey results and informal feedback from the module's students, demonstrators and lecturer. Based on our findings, the authors recommend that rolling out such a module, with adjustments (discussed throughout the paper) within the school on a permanent basis would be beneficial to the students overall.

Bibliography

[1] COMP 10050 Introduction to Software Engineering 1 module information http://www.csi.ucd.ie/csimodules/comp10050-software-engineering-project-1

[2] ALICE Programming Environment Homepage: http://www.alice.org

[3] Moskal, B., Lurie, D., and Cooper, S. (2004). *Evaluating the effectiveness of a new instructional approach.* In Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education. D. Joyce and D. Knox (eds.). ACM Press, New York, 75-79.

[4] Cooper, S., Dann, W., and Pausch, R. (2003) *Teaching objects-first in introductory computer science.* Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education, Reno, NV, February, 191-195.

[5] Dann, W., Cooper, S., Dietzler, K, Dragon, T., Ryan, K., and Pausch, R. (2003) *Objects: Visualization of behavior and state*. In Proceedings of the 8th Annual Conference on Innovation and Technology in Computer Science Education, Thessaloniki, Greece.

[6] COMP 10050 Software Engineering Project 1 student evaluation form: http://www.csi.ucd.ie/content/software-engineering-project-1-comp10050-evaluation-form

[7] Office of Classroom Management University Of Minnesota Active Learning Classrooms information. http://www.classroom.umn.edu/active-learn-room.asp

[8] Active Learning Classrooms Pilot Evaluation: Fall 2007 Findings and Recommendations, University of Minnesota http://www.classroom.umn.edu/ALC_Report_Final.pdf