A Computer Science HCI Course

Beryl Plimmer

Department of Computer Science
University of Auckland
Private Bag 92019
Auckland, New Zealand
Tel: *+64 93737599*
Email: *beryl@cs.auckland.ac.nz*

Can a computer science student learn to be a designer and a psychologist as well as a computer scientist? Unlikely, but they can learn to appreciate what other disciplines offer HCI. The need for computer science students to understand the big picture, that HCI is multidisciplinary in nature, has been recognised for many years. Yet successfully integrating HCI into a computer science degree is still difficult. Our thesis is that an appropriately structured course can take advantage of the existing knowledge of students for each to learn more than they otherwise would, and experience the contributions other disciplines make to HCI. This paper presents the theoretical background for this thesis and our experiences with delivering the course in a New Zealand University. In this course, by carefully defining the project requirements, the students experienced designing and prototyping a program where psychology, design and computer science contributed to the software creation process.

Keywords: HCI Education, Computer Science Education, Student Group Projects

1 Introduction

Human Computer Interaction (HCI) skills are increasingly important for computing professionals. There is no doubt that an ever-increasing percentage of program code is devoted to providing a simple and effective user interface. At the same time many of the algorithmic techniques yesteryears' students needed to master are provided by the operating system or programming language.

Computer scientists are not expected to be HCI specialists [Douglas et al., 2002]. However they do need to have a good understanding of HCI principles and appreciate the roles that others, with different expertise, can play in the design process. According to Lethbridge [2000] HCI/user interface design skills are ranked second (to negotiation) in the list of skill gaps for software professionals. This can be attributed to the rapid change in the nature of interfaces and the cross disciplinary nature of HCI. The challenge for computer science educators and students is to first accept the

importance of HCI and then to learn to work with people with different backgrounds to design and deploy better computer interfaces [Douglas et al., 2002].

There are suggestions that HCI should be integrated across the curriculum in Software Engineering degrees [Phillips & Kemp, 1996], and that a more cross disciplinary approach is necessary [Milewski, 2004]. Other institutions have developed degrees or diplomas specifically in HCI. However, New Zealand is a small country; there is not sufficient demand to justify an HCI qualification.

This course is situated in a computer science department that has diverse interests. Yet, most of the students find employment as commercial programmers where HCI knowledge is useful. Our approach is pragmatic; by carefully structuring the course it can include the basic theories behind HCI and, via a project, give students the experience of designing and building software where HCI fundamentals are at the heart of the process.

The structure of the remainder of this paper is as follows: the next section provides a background from both HCI and educational psychology. HCI, as a discipline, is discussed alongside its penetration into the New Zealand software development industry. This section also provides a brief précis of relevant educational psychology and a description of the student demographics. These ideas are then pulled together to suggest a teaching and learning strategy. Section three explains the plan, linking the elements of the learning experience to educational and HCI theories. Section four describes the implementation giving examples of the student work. The evaluation and discussion in section five includes survey results and comments from the students and faculty.

2 Background

Planning a HCI course requires consideration of the desired learning outcomes and the underlying educational psychology. This section looks briefly at HCI and the real-world application of HCI as a basis for desirable learning outcomes. Educational psychology and the attributes of typical computer science major students provide a background for the pedagogy. These disparate ideas are then brought together to suggest a teaching and learning approach for this computer science, HCI course.

2.1 HCI as a discipline

HCI is truly a multidisciplinary field drawing on many different strands of research. The three main contributing areas are design, psychology and computer science. From design HCI draws the creative inspiration and knowledge of what is pleasing to humans. Psychology provides knowledge of human physical and mental capabilities. Lastly computer science provides the technology to build the interactive environment.

Human computer interaction is a very recent area of research. Early computers were operated by specialists who were trained in the specific requirements of the system. Now computers are everyday tools, with interfaces integrated into everyday objects. Advances in hardware and artificial intelligence are continually pushing the boundaries of what is possible.

2.2 The real world

We are preparing students to enter the New Zealand workforce as computer science graduates. The reality in New Zealand, as it is in many other places [Greenburg, 1996], is that HCI and interaction design are not widely recognised as an important part of software development. There are about 10 people practicing as fulltime HCI/usability professionals in the entire country [Mankelow, 2004]. New Zealand industry is characterised by many small organisations (< 5 staff) and very few organisations with more than 50 IT development staff [Ministry of Economic Development, 2004]. Greenburg pointed out in his 1996 article [Greenburg, 1996] that in many workplaces there would be little or no knowledge of HCI and usability. This is likely to be the position many of our graduates find themselves in.

In addition best practice for designing and testing interfaces involves ongoing contact with real users. User-centred design assumes that users take part in the early discovery phases of design and usability testing. The reality of software development is that real users are often not available [Greenburg, 1996]. A result of this is that a common complaint about current interfaces is that the computer scientist has designed it for himself [Turban, 2003]. It is often suggested that the computer interaction experience is absolutely fine if you are a 20 to 30 year old male, but for the rest of the population computers are incomprehensible.

Yet, the demands of real world systems make it increasingly important for interfaces to be well designed and well engineered. Computers are experiencing an ongoing change of audience [Turban, 2003] and computer interfaces are frequently ubiquitous and imbedded. In most instances we must assume that the user will have no training, and no access to a manual or on-line help. Therefore the interaction must be intuitive for the user.

One of the goals for this course is to prepare the students for the local workplace. A place where knowledge of HCI and usability are often minimal, where there may be no access to real users, but the software must be such that it is usable by the target audience without instruction.

2.3 Education

Our approach in the design of this course is not unique. We have drawn on well defined educational psychology principles. First, the ideas espoused by Vygostsky [1978] that people learn more, and perhaps more effectively, from their peers than they do from the teacher. Second, constructivism and the learning cycle, that people learn most by learning about the theory, doing and then reflecting on what they have done [Kolb, 1984]. This is similar to activity theory that Nardi [1996] advocates for use in HCI. Last, group projects provide a wide range of benefits for students [Koppelman et al., 2000], however group work also presents some challenges for teachers, particularly in how to fairly assess the individual.

2.4 Course

This project is a major part of a third year computer science ‘Introduction to HCI’ course. The course is run over fourteen weeks, twelve teaching weeks with a two week break in the middle. There are three one hour lectures a week and each student attends one two hour laboratory a week. The majority of the 100 students in the course are CS majors, in a traditional three year under-graduate Bachelor of Science degree. The

requirement for the major is eight CS papers at least four of which must be at third year. This is the only undergraduate CS course the department offers that has HCI or interface design as a part of its learning outcomes. However, we are situated in a large university (30,000 students) and the BSc structure is such that students can, and do, select courses from a wide range of disciplines including fine arts, psychology and media studies. More than half the students are studying towards either a 'double major' or 'con-joint' degree. A double major is a degree with majors in two science disciplines such as computer science and mathematics, physics or psychology. A con-joint is two degrees; BSc/BCom is a popular choice for CS students. Many of the students are in their final semester of study.

The demographics for the most resent offering of the course were: gender, 70% male, 30% female; ethnicity, 40% Chinese, 26% Pakeha (New Zealanders of European descent), 10% other Asian, 11% other, 8% Indian, 5% Maori; age, median 23, 4% of the students over 30. There are two main groups of Asian students (Chinese, other Asian and Indian), international students and new immigrants. The official languages in New Zealand are English and Maori; all instruction in the department is in English and foreign students must meet English proficiency levels. The mix of cultures in the class adds a enjoyable diversity of prior experience including different social norms and school systems.

2.5 Teaching approaches

Bringing together the diverse roots of HCI, the local industry and our student population we have indeed a very large landscape in which to situate the course! One HCI course can but introduce the principles and make the students aware of the wider issues in HCI. The lectures deliver the basic theoretical principles and the laboratories and project enrich the course with practise. Based on Vygostsky's [1978] principles we hypothesise that the students can learn more from each other about design, a subject that is difficult to teach in a computer science course, and psychology than we have time to 'teach'.

The project runs for the first eight weeks of the course (six teaching and two break). The overall goal for the project to encourage the students to think deeply about the interface and interaction design, and also to appreciate the roles of design and psychology in HCI. It draws on the individual strengths of the group members in design and psychology to add to the learning experience.

3 Plan

In order to achieve the goals a great deal of thought was put into the project scope and requirements. We considered which parts of the project would be fully defined and in which parts the students would have latitude to explore. The learning plan includes timeline, scenario, information sources, group formation and assessment plan.

3.1 Scenario

The scenario needs to be quite specific to allow the students to focus and limit the size of the project. The problem must specify a user or users quite different from the

'average' class member so that the students can be in no doubt that they are not the primary audience for the software. Also, if the interaction is restricted to a non-standard subset of a normal PC the students are obliged to think of different ways standard functionality can be achieved. Finally the topic needed to be such that it provides design opportunities, some challenges for the *geeks* in the class and be achievable in the eight week timeframe.

3.2 Resources

We wanted the students to research the requirements from secondary sources as this exposes them to the wide variety of information that is available. This reflects the reality that many software development companies do not involve users in the development process. Another practical reason for not using real people, other than class members, is that using people imposes a quite onerous ethical approval process on us. We, certainly, did not want to use class members, as they can not represent a naive user.

A choice left to the students is the implementation environment. The prototype could be built using any programming language available on the university network. The network computers are standard PCs running Microsoft Windows XP and languages available include Java, C++, Visual Studio .Net C## and VB.

3.3 Assessment

There are two assessment deadlines, one three weeks into the project and another at the end of the project. At the first date the background research, user requirements and a non-functional prototype must be presented and handed in. This is timed so that the students complete this phase before doing the programming. The presentation encourages them to formalise their ideas and lets each group see what the other groups are planning. They may review their designs after the presentation. The second hand-in and presentation is of the prototype software and also a short report on the experience. The background research, user requirements and non-functional prototype will be checked for completeness and a reasonable design. The prototype will be reviewed against Nielsen's [1994] usability heuristics.

Assessment is often problematic with group projects; it is difficult to judge each individual's effort verses the team effort. However, the computer science department has a policy of minimising the course work's contribution to the final grade and retesting course work in controlled assessments (test and exam). Accordingly the entire project contributed 8% to each student's total grade. Clearly this does not represent the effort required for the projects. Test and exam questions directly related to the project contributed further 20%. This assessment approach means that the project can be considered more a formative than summative: neither the students nor teachers need be overly concerned about how much each individual receives for the project as opposed to the group score.

3.4 Integration with theory

A number of elements of the project were directly linked to the theory portion of the course. The use of personas and interface storyboards and wizard-of-oz techniques are explained in the theory part of the course and expected to be used in the project. The theory also considers human physical capabilities such as vision, and cognitive capabilities such as problem solving strategies. There was a clear expectation that the user will be able to use the software without help or instruction, therefore the students must design within expected abilities. The theory also discusses where HCI fits into the software development life-cycle and which techniques are suitable at which stages.

4 Implementation

In this section we describe in detail the implementation of this project into the introduction to HCI course offered in the second semester 2004 (July – November). The assignment specification was given to the class in the first lecture.

4.1 Scenario

The problem was to design and build a drawing package for a specific six year-old child. According to the scenario the child, Lindsay is disabled and can only interact with the computer using an eye-gaze device (the students emulated this with mouse clicks). Lindsay wants to be able to draw pictures like his/her classmates but drawing with normal drawing software is too slow. Lindsay's teacher saw an article on some software that let the user put together a picture from pre-existing bits [Ruder-Finn, 2003] but feels the software is inappropriate for a six year old. The project is to design and prototype a drawing program specifically for Lindsay that he/she can operate alone.

The gender of Lindsay was determined by the makeup of the group (see below). Lindsay was the opposite gender to the majority of the group. Given the demographics of the class most groups were developing for a girl. If the group was evenly balanced, male and female, they could choose the gender of 'their Lindsay'. In this case the gender had to be specified with the group registration.

4.2 Resources

We provided a number of resources on children's drawing developmental stages from the library and internet [for example: Druin et al., 1997; Golomb, 2004]. We did not make any provision, or actively encourage the students to talk with children. As described above it is quite common to rely on secondary data for user needs and we wanted to make the students aware of how much research there is readily available. And, a practical consideration, the university has very strict ethical processes to work with children. Of course many of the students have young family members that they could relate this problem to or know primary school teachers with whom they could discuss the domain.

Students were required to find out about eye-gaze products from the internet and selected a particular product that their Lindsay hypothetically used. While they could use any programming language available on the university network some felt a bit daunted by the prototyping task. We provided a two hour tutorial on the Microsoft

Visual Studio .Net Tablet SDK ink classes that offer a set of easy-to-use ink capture and manipulation methods. This allayed their fears about implementation difficulties.

4.3 Group formation

The students formed their own groups with the following restrictions and recommendations. Group size was set at four; however groups of three or five were acceptable when necessary to balance numbers. The class split into approximately thirds for the laboratory sessions, all the members of a group had to be in the same laboratory session. We recommended that each group had as least: one person with art/design skills (any highschool course counted), one person with some knowledge of psychology or educational psychology (any undergraduate course), and one 'A' programmer.

In the tutorials there was time for group formation. To get to know each other and each others' skills each member of the class wrote him/herself a name label and added to it a red dot if they knew about design, a blue dot if they knew about psychology or educational psychology and a grade in the range A – C of their programming ability.

We were a little nervous as to whether enough students would have skills in psychology and design. About 30 students had some knowledge of psychology or educational psychology and about 35 students had design skills. The students then had time to chat and form provisional groups. It was left to them to ensure they had an appropriate mix of skills. The provisional groups work together on a small lab exercise (on Fitts law) this gave them an opportunity to get to know each other better before the groups were finalised. A couple of groups formed around existing friendships, rather than skills. If they didn't have the skills within the group their work did suffer. We did not interfere as we felt that this was a part of the learning process.

By the following week everyone had to notify the course administrator of their group. We gave some suggestions on the ways that they divide the work such as research into requirements, interface design, functional requirements and programming. They were not required to account for the division of work but were reminded that it would all be in the examination. Students could appeal to us if there were group problems. While we are certain that some people did more work than others, no one came to us and complained about their group.

4.4 Research and non-functional prototypes

Most of the groups did a very thorough job of researching into children's drawing and eye-gaze interaction. In week three the groups presented their requirements and non-functional prototypes. We were delighted with what they created.

From their research the groups discovered that six year-old children are generally in the schematic drawing stage. At this stage children are drawing pictures to convey a story. They have fixed presentations of objects but this may be varied to emphasise an import feature, for example they may have a standard representation of the family dog, but if the picture is about the dog eating the family dinner the dog's mouth may be much larger than normal. Spatial awareness is developing so objects are set in relative adjacency, but all sit on the same base-line, a 'horizon' that typically runs across the picture. Size is relative to importance rather than a representation of reality.

They also were required to find out about eye-gaze interaction. There are some clear indications as to what is possible with this type of interaction that they were expected to identify. However, for the next stage of the project, the development of the prototype they emulated eye-gaze with mouse-only interaction.

They were encouraged to develop a low-fidelity prototype of the interface first, but could present either a low or high-fidelity design. Figures 1 & 2 show two of the sketches that one group produced. The first sketch gives the general layout of the form in drawing mode and briefly describes the function of each of the buttons in a list down the right-hand side. They produced similar sketches for: save, make stamp, make colour and picture gallery. Figure 2 shows a colour mock-up of the interface where flaps are folded over the different areas so that the prototype can be used as a storyboard. This group used the mock-up as a prop for a use-case based presentation where they described the process of Lindsay making and saving a picture.

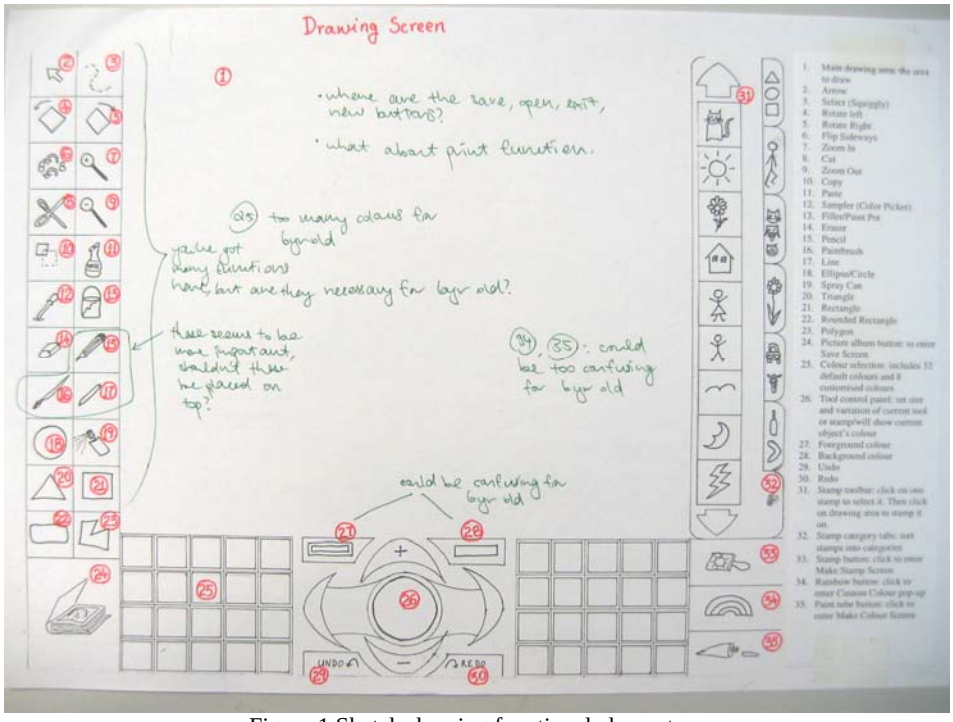

Figure 1 Sketch showing functional elements.

Figure 2 Interactive low-fidelity prototype with flaps that show changes in interface depending on tool or state.

Figures 3 & 4 show a different group's high-fidelity design. This group were very creative with their design. The rocket shown on the left of figure 3 and in figure 4 'lauched' as the program started-up. Once in position the door would open, the astronaut come out and introduce himself (their Lindsay was a boy) to Lindsay as the help agent (using text-to-speech) who could be called on at anytime by clicking the icons on the rocket door.

Another unique feature of this group's design were the drawing tools and colour icons shown at the bottom of figure 3. Their plan was to change the colour icons and colours for each different type of tool. Their argument was that different tools had different colour ranges and different drawing effects. For example the crayons are bright colours that laid down thick solid ink while the felts are lighter colours that produced slightly transparent ink. This group implemented both of these features in their functional prototype.

Figure 3 High-fidelity interface design.

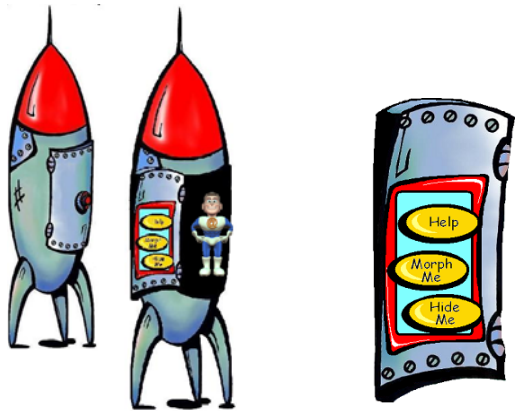

Figure 4 Detailed images of rocket astronaut and icons for design in Figure 3.

With the gender mix of the class, most of the target applications were for a female Lindsay. Two of the all male groups made rather telling, and amusing, comments about their design for Lindsay. One said that their window surround was ‘yucky pink’ and they had lots of ‘pretty’ colours in the palette because girls like colours like this! Another group said that they had real colours like blue and red and then a few girlie colours like pink.

The examples shown are typical of most of the project designs. However, there were three groups who had simply not got into the problem/design space. Having this early presentation meant it was very clear to these groups that they were off track. One of these groups completely overhauled their design and created a good prototype. Almost

all the groups decided to change their specification or interface in some way as a result of their peers' questions or seeing other groups' presentations. We encouraged them to do whatever revisions they deemed useful and they could submit a revised design with their prototype.

4.5 Prototypes

The most frequent question we had to answer in the first couple of weeks was 'what language do we *have* to use for the prototype?' Many of them were quite concerned by the answer which was 'anything that is available on the university network'. We see this as an indication of the computer science centricity of many of the students.

The main teaching language in the department is Java. Most of students had also completed courses that use C++, Visual Studio .Net C# or Visual Studio .Net VB. The ease of implementing in Visual Studio .Net with the ink SDK meant that most groups used this for their prototype. There were three projects written in Java and one in C++.

We stressed that they were creating a prototype. Given the three/four week timeframe we were not looking for a complete or robust piece of software, rather a prototype that could be used to demonstrate the design and interaction principles. They were, however, expected to write clear, readable code.

All of the prototypes had basic functionality to facilitate drawing, erasing, new picture, print, save, load etc. There was a wide range of other features that groups provided derived from their research into children's drawing. Screen shots of four groups interfaces are shown in figures 5 to 8. A number of groups provided 'stamps'. Figure 5 shows a duck stamp while figure 6 shows a little dog; the dog (and all this group's stamps) is an animated GIF: the dog hops up and down. The group that created the prototype shown in figure 7 allowed Lindsay to save portions of his/her own pictures as stamps, the most recent of which are shown on a clipboard on the right of the screen. The final prototype shown here (figure 8) allowed the user to construct a creature by selecting a body, head, eyes etc, this screen shot shows the 'mouths' that could be added to the creature on the picture.

The prototype shown in figure 5, and a number of other groups, added a vocabulary list and an on-screen keyboard. Some went to considerable trouble to discover a suitable list of words and considered different keyboard arrangements. A number of groups provided a set of backgrounds, figure 8 shows a typical background.

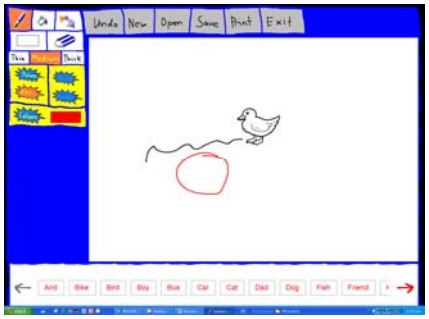

Figure 5 Duck stamp, freehand ink and word vocabulary.

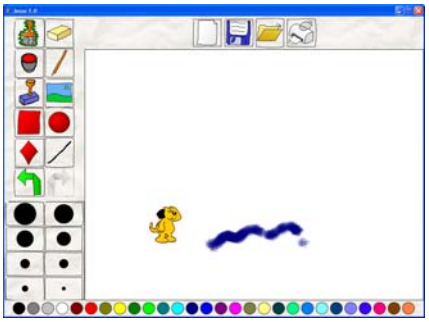
Figure 6 Animated stamp, varied width pen and colour palette.

Figure 7 Collection of Lindsay's previous objects shown on left. These can be 'stamped' onto a new picture.

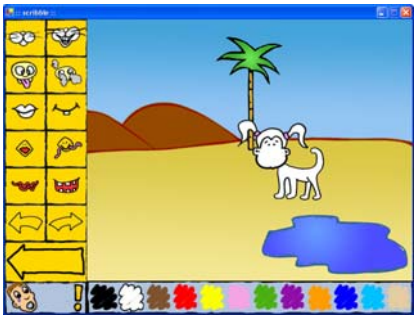

Figure 8 Background and creature created from body parts – nose/mouth options shown at left.

4.6 Presentations

After the mid-semester break (two weeks) a show-and-tell session was held in one of the labs. Each group was required to have their prototype running on a computer. We invited a few local industry people to the session too. Each group had one person looking after their demonstration and answering questions. Everybody else was encouraged to circulate and try out the prototypes.

Twenty-six prototypes and 100+ people meant that this session took on a festive air. The students and industry people really enjoyed looking at all the different solutions to the same problem.

4.7 Marking

Marking was undertaken for both parts of the project after the final hand-in. The first part of the project, delivered in week 3 was worth 50% of the marks. This 50% was split between user needs ¼, interface specification ½, and presentation ¼. The user needs were evaluated against what we would have expected them to find out about six year olds' drawing skills and eye-gaze interaction, and the reasonableness of the persona and design.

For the second part of the project, ¾ of the marks were for the prototype and ¼ for the review. The prototype was evaluated against Nielsen's usability heuristics [Nielsen, 1994] with an additional item for the project's special case, which was Lindsay's special needs. A small portion of the marks were allocated for professional looking code and innovation. In the project review the group was expected to reflect on the project and the process, describing what they did well, what they could have done better and features they were particularly proud of. In this section we looked for a thoughtful approach.

5 Evaluation and Discussion

Our goal with this project was to give students the opportunity to work on a software development project where HCI was the focus. To this end we deliberately set a

scenario that constrained the interaction and the target user could not be construed as being themselves. This made them research into the likely needs of the user and develop a persona. The timetable for deliverables was such that they needed to complete the non-functional design before they started to program.

The group work gave the students that had skills in psychology and/or design the opportunity to draw on those skills. It transpired that several of the students in the class were doing a double major degree in Computer Science and Psychology. One particular student in this category said in week three of the course 'now I know where my two majors fit together'. She has gone on to get a work with one of the only HCI consultancy firms in New Zealand.

We were a pleasantly surprised at the number of students who had excellent skills in graphic design. Several of these students made comments that they had chosen computer science as a career over design because there were more jobs in computer science. One student demonstrated a real flare for designing icons. She, through one of the other class members, has a contract to design icons for a local company that develops applications for mobile phones. We hypothesise that she can do this so well because she has a deep understanding of the function a particular button accesses and she is an excellent designer.

There were, of course, students in the class who did not have skills in either psychology or design. However they have had the opportunity to work in a team where the contributions of these other disciplines were vital to the project's success. Some commented that this experience has given them a better concept of the resources that are freely available about different types of users and that there are people other than information technology specialists who can make a significant contribution to a software project.

Of the 26 projects all but two demonstrated that the students had well understood the brief. The software was clearly targeted at six year-old drawing skills and they had thought carefully about the interaction so that a keyboard was not required. For example, most groups simplified file persistence so that the user would not have to understand or interact with the operating system's file systems. The remaining two groups presented rather poor imitations of a standard paint program. We hope that by seeing what other groups produced they may realise the possibilities.

Many commented that the presentations (in weeks 3 and 8) were inspiring. This was the first time that they had seen so many diverse approaches to the same problem. They remarked that this reinforced the theory of producing multiple ideas.

We conducted standard student surveys during the class. Overall the class rated well above average. Many students commented that this was the first group project they had done in computer science and that they enjoyed the group work. Others remarked that they were in their final semester of study and that the experience had made them rethink their approach to programming.

This was the first offering of an HCI course at this university. Some of the hard-core computer science students were surprised that the course de-emphasised programming. A couple commented that they did not think that it was computer science at all. We are prepared to live with that view and suspect that they may give a different answer after they have been working for a couple of years.

6 Conclusions and Future Work

This paper describes why and how we designed a project into a computer science course that drew on the students' existing knowledge of design and psychology. The project that we prescribed required them to learn about user requirements and constraints from the psychology literature and a novel interaction device. They were then required to design a creative interface for a user that we deliberately made very different from them. The targeted user together with the limited interaction required them to carefully reconsider standard interaction such as file save/load.

The prototypes that the groups created were inventive and demonstrated attention to HCI principles. Their comments showed that they had gained as much as we had hope from the experience. They enjoyed working together and having the opportunity to be creative.

We considered using the same project as a case-study for usability testing in the second-half of the course. Because this was the first offering of the course we decided to separate the two parts and use a different project for usability testing. However given the success of this project, next time we are planning to set a project with similar constraints and extend its use into the usability testing part of the course.

7 References

Douglas, S., Tremaine, M., Leventhal, L., & Wills, C., Incorporating Human-Computer Interaction into the Undergraduate Computer Science Curriculum, *Proceedings of SIGCSE '02*, Covington, Kentucky, pp. 211-212, 2002

Druin, A., Stewart, J., Proft, D., Bederson, B., & Hollan, J., Kidpad: A Design Collaboration between Children, Technologists, and Educators, *Proceedings of CHI 97*, Atlanta, pp. 463-470, 1997

Golomb, C. *The Child's Creation of a Pictorial World, 2nd Edition*: Mahwah, N.J. : L. Erlbaum Associates., 2004

Greenburg, S., Teaching Human Computer Interaction to Programmers. *Interactions, 3*(4), pp. 62-76, 1996

Kolb, D. A., *Experiential Learning: Experience as the Source of Learning and Development*. New Jersey: Prentice-Hall Inc., 1984

Koppelman, H., van Dijk, E. M. A. G., van der Mast, C. P. A. G., & van der Veer, G. C., Team Projects in Distance Education: A Case in Hci Design, *Proceedings of ItiCSE 2000*, Heisinki, pp. 97-100, 2000

Lethbridge, T. C., What Knowledge Is Important to the Software Professional? *IEEE Computer, 33*(5), pp. 44-50, 2000

Mankelow, T., *Usability in New Zealand*. Retrieved February, 2005, from http://www.optimalusability.com/downloads/presentations/UPA-07Sep2004.pdf, 2004

Milewski, A. E., Software Engineers and Hci Practioners Learning to Work Together: A Preliminary Look at Expectations, *Proceedings of CSEET'04*, 2004

Ministry of Economic Development., *Smes in New Zealand: Structure and Dynamics - 2004*. Retrieved February, 2005, from http://www.med.govt.nz/irdev/ind_dev/smes/2004/index.html, 2004

Nardi, B., *Context and Consciousness: Activity Theory and Human-Computer Interaction*, Cambridge, MA and London: MIT Press

Nielsen, J., Enhancing the Explanatory Power of Usability Heuristics, Proceedings of ACM CHI'94, Boston, pp. 152-158, 1994

Phillips, C., & Kemp, E., Towards the Integration of Software Engineering and Hci Education: A Cross-Disciplinary Approach, *Proceedings of OZCHI*, Hamilton, pp. 145-150, 1996

Ruder-Finn, I., *Mr Picasso Head*. Retrieved 1 Feb 2005, 2005, from http://www.mrpicassohead.com/, 2003

Turban, R., Approaches to Implementing and Teaching Human Computer Interaction, *Proceedings of ITCC'03*, pp. 81-85, 2003

Vygostsky, L. S., *Mind in Society: The Development of Higher Psychological Processes*. Cambridge MA: Harvard University Press, 1978