

Architects or builders; scaffolding or duck† tape?

Russel Beale
University of Birmingham/BCS
UK

ABSTRACT

In this paper I reflect on the role of HCI Education in University level courses. The theme of the conference is ‘Architecting the Future’ and I analyse what we mean by this in terms of HCI Education. As a community we seem to have moved from fundamental HCI issues through usability and user-centered design, and last year were all about design and creativity – have we moved one step beyond again and moved to architecting the future? And if so, is this appropriate for our students?

Four principles for HCI Education are presented that address the pressures that students and the curriculum is under. Further approaches to assisting HCI Education are given, in the light of case study experiences.

Author Keywords

HCI Education, reflections, design, context

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI).
J.4. Computer Applications: Social and Behavioral Sciences

INTRODUCTION

The theme for HCED2008 is that of ‘Architecting the future’. The conference website says that we

“would like to move a step forward and have an in-depth reflection on the required foundations of future HCI education. We wish to enrich and integrate our knowledge of the design processes that are used in the various design domains [...] by peeling away the domain specifics, identifying what is universal and what is different, and what common methods and tools can be identified. We would like to investigate how to better handle and integrate the border conditions impacting on our domain [...] and the influence of cross-cultural

issues. We would like to critically compare learning contexts. We thus aim to discuss ‘Architecting the Future’ of the HCI and design education.”

The Oxford American Dictionary defines an architect as “a person who designs buildings and in many cases supervises their construction; a person who is responsible for inventing or realizing a particular idea or project”

and if we use this concept to consider the website text, it seems to me that the concept of ‘architecting the future’ of HCI is working at a high level; it is drawing out generalities, focusing on universalities. In common with (good) architecture, it also examines the context of the situation, trying to develop appropriate concepts that fit with the surroundings and meet the differing demands and pressures imposed by the external environment. All these are laudable, noteworthy aims, but it is interesting to put them into context.

In 2007, HCIED looked at Creativity and Experience; in 2006, it looked at theory, design and innovation; in 2005 it looked at frameworks. This seems to show a trend from the practical (in 2005) through to increasingly abstract and esoteric concepts – through design to creativity and now architecture. I propose that this is due to two factors. The first is a practical one: a broadening of the audience for HCI Education, and hence a wider set of backgrounds requiring a broader encompassing theme. The second is semi-practical – there is a clear need for HCI to address more than just users sitting at computer screens and working with keyboards and mice at a windowed interface: HCI has to accommodate the variety and complexities of mobile systems ubiquitous computing, secure transactions, Web 2.0, social networking, and the myriad of new and upcoming technologies that offer different ways of interacting with digital systems and integrating them into our lives.

In principle, therefore, as educators we are embracing change, and adjusting our approaches to suit. But are we actually doing the correct thing for our students?

THE REALITIES OF HCI EDUCATION

From personal experience, the realities of HCI Education are somewhat different to the concepts described above. At

my University, I have been involved in teaching HCI for many years, and have fought long and hard for it to be an integral part of the curriculum. I am supported in any battles by the Director of Student Learning and Teaching, and the majority of my colleagues are in agreement that HCI is a critical part of undergraduate and Masters Education. However, the practicalities of teaching it have a major impact on the courses. The HCI courses have been revamped recently, and in 2005-6 the second year undergraduate HCI module was lost, with the only compulsory HCI that undergraduates get being about 1/3 of a 20 credit course: the problem is that the other 2/3 of that course are about the details of Java: students tend to focus on the programming components to the detriment of the concepts that I am trying to teach them in the other part. This loss was balanced by the introduction of a 3rd year course in HCI, optional, so that those interested could take it further. Designed to be taught to 20-30 students, at the last minute the school's teaching committee decided that Masters students didn't have enough options, and opened the course up to them as well. This had two impacts: the first was that the small group teaching and self-directed study had to be radically altered at the last minute to cope with a change in numbers from 20 to 90 students, and the second was that the course was now being taught to a mixed group, comprising those that had done some HCI before, and those that had done none. This ran again like this in 2006-7, before I reorganized things once again: this mixed course was not working. We altered the structure, but the only acceptable structure was to put on a specialist course for the Masters students, assuming no previous HCI. In addition, the HCI element for the second year was altered to accommodate the needs of final year projects, and so focuses more on interface design and coding of GUI elements. The more advanced HCI course has, for the time being, been lost.

Thus we have gone from a situation in which we had more HCI courses, at different levels, including an advanced one, to a situation in which the only compulsory HCI is actually interface design, and where the HCI course for the Masters students has to start from scratch. We have therefore lost both space in the timetable, and have to start with a lower background level of knowledge of HCI issues, than ever before. Whilst this is a specific situation, discussions with some colleagues at other institutions suggest that similar pressures exist there[2].

The reason for this is not hard to find. UK Universities are competing harder and harder for decreasing numbers of students, and even those not prepared to compromise on the quality of their applicants are finding that they need to teach more basic computing and programming than before, and this puts pressure on the timetable higher up. Broadly speaking, we have many students with less exposure to computing, sometimes less committed to it as an intellectual pursuit, sometimes less than fully engaged with the educational process, and as educators we have to work

harder to cover the basics. This means that many courses have to assume less initially, and go slower. Given this, it is hard to see how we can embrace the worthy concepts of a much broader HCI – this requires that we do more, at a higher level, and yet the pressures are such that we have to actually do less, at a lower level!

Addressing this is a job for the entire staff, but one that requires significant curriculum change, and the practicalities of academic life (the recent focus on the research assessment exercise, the continuing pressure to publish and gain grant funding, the relative lack of recognition for teaching-led initiatives in personal promotion prospects), and the sheer scale, and risk, of a full curriculum revamp means that this is an unlikely exercise. Many computer science curricula are somewhat overburdened with computer science theory: there rages a debate about the appropriate levels at which to teach. In a few places, the fundamentals are the theoretical principles, with the students left to learn languages by themselves, on the whole – in many more places, there is significant effort expended in teaching a programming language. In our institution, the debate has moved away from teaching low level things such as assembly language, though we still have a strong theoretical basis to our teaching of Java, our language of choice – but a course on Web 2.0 has met with some negative criticism because it is seen as too lightweight and not theoretical enough. In addition, we suffer from a similar problem to many other UK Universities- people expert in new technologies, their principles and uses, are not attracted to work in academia, and hence the material, if taught, is not done as well as some of the other subjects, in which the domain experts are best employed in computer science departments.

SCAFFOLDING, OR DUCK TAPE[†]?

Our vision as HCI Educators may be to create architects of the future: people who conceptualize new software applications new forms of interaction, user support and communication, and who develop these concepts into innovative, usable, useful designs. But is the reality more that we have to deal with people who just want to get something built, who have an appreciation of the concepts but a practical need to be efficient? Or worse, people with less knowledge, a focus on just getting the job done to a satisfactory level, and moving on to the next thing? Are we actually dealing with architects, or with builders? Are we dealing with people who want to get it all right, do it safely and correctly, and will take the time, or are we dealing with those who want to cobble together something that works,

[†] there is some debate as to whether the term is “duck” tape or “duct” tape – originally designed for the US military in World War 2, the waterproof tape started life as “duck” tape, but when appropriated for households after the war, it was primarily used for joining air-conditioning duct work – hence “duct” tape. Both are therefore correct.

something that will do, that may not last, but seems okay for a while?

As the forums may say, YMMV. Your Mileage May Vary. The reality of HCI education will depend on which institution you come from, what the focus of the teaching is, what the quality of your student intake is, and how flexible and creative your approach to curriculum development is.

The biggest question of all is: how do we deal with this?

SUGGESTIONS FOR ADDRESSING THE PROBLEM

Given this broadening of scope of HCI coupled with increased curriculum pressures and a mean reduction in the quality and engagement of the students, we need to take measures to address this.

There is not time to teach HCI principles, theories, and practical skills, interface design, relevant coding approaches, theories, and so on for even conventional systems, let alone new styles of computing in the current curriculum.

I make the following four suggestions for addressing this:

1. Make students aware that users are not like them

Most students initially assume that users are the same as they are: the usual arguments for HCI courses and user-centered design apply here. By focusing on this, by demonstrating through practical experiments, interviews, user involvement, and case studies, that users are very different to systems designers, the students take away a deep message that (hopefully) impacts on everything that they subsequently do. By emphasizing this conceptual shift, from concentrating on the programming, or the practical nature of the task, to asking what the user wants and expects, the seeds are sown for a questioning of processes and approaches that do not take account of this.

In my classes, I have achieved this at different stages using some visual illusions, getting novices to play with a new interface, discussing case studies with the class, and, most recently, by getting the students to design a software system to support other students in their first week at University, and getting them to interview and design a system based on those needs. The times when realization dawns, when a student sees that what they thought was important/obvious turns out not to be so, is very pleasing, and happens at different stages with different students.

2. Provide them with a design process they can use

I teach a user-centered design process, and hang a number of tools and techniques of each of the different stages. This does two things: it focuses their attention on design as a process, including the need to involve the user where possible, and it gives them an extensible framework onto which new techniques can be added as they develop through their University and professional career.

In my classes, I get students to undertake a design project – this is true for both the semester-long Masters course, or the 4 weeks of HCI shoehorned into the Java programming course. They do some user needs analysis, they prototype, evaluate, revise and refine, and (sometimes) build something. The practical experience of the process demonstrates its benefits, even over 4 weeks.

3. Make students aware of what they do not know

If we accept that we cannot teach everything, a major goal has to be made clear to the students that what they learn is only a small part of potentially useful knowledge.

A design project is a good tool for this – it allows you to push their assumptions, discuss tangentially-related topics, and make them question things in new ways. Even without time to explore these areas, by spotlighting disparate areas in the space of knowledge, research, skills and practice that form HCI, their education is augmented by an awareness that they are not yet expert in this domain.

4. Pick an appropriate strategy for exploring the educational space

"I" approaches are effective: "T" ones are more so if you have time; "Π" ones are ideal if you have even more time or the students more experience.

These categorizations refer to how you explore the space of HCI work. The HCI domain is envisaged as a rectangle, with topics along the horizontal axis, and depth on the vertical axis – broad and shallow approaches span the top of the space in a horizontal band, whilst narrow and deep approaches focus on specifics in depth.

It has been my recent experience that doing more in depth work on a specific subset of topics is more beneficial (and addresses points 1-3 above more effectively) than trying to cover everything very briefly. This is the "I" approach. Students need specifics – without concrete examples, the concepts seem unrelated to everyday work, and only through detailed analysis can they begin to appreciate the difficulties and subtleties inherent in our field.

If you have more time, then a broad scan of many areas, followed by an in-depth study of one, is even better – it contextualizes the work, addresses point 3 very effectively, and yet gives specific skills and awareness that the shallow approaches find hard to achieve.

The "Π" approach does 2 areas in depth, linking them with the broad overview. By covering 2 things in depth, the space is more fully mapped out for the students, and they can compare and contrast the in-depth areas.

ADDITIONAL STRATEGIES FOR EFFECTIVE EDUCATION

Design the course around the student cohort

One approach I have previously used to great effect (though not for the revamped courses as yet) is spending the first

lecture finding out what the students think HCI is all about, and then ascertaining what their interests are: it may be mobile systems, or the internet, or social networking, or gaming. Working with them, we then devise an HCI syllabus for the next 10 weeks that uses this domain as the exemplar for all the HCI work that we subsequently undertake.

This approach has a number of advantages. It is pedagogically sound, since it both empowers the students in helping build a course tailored to their needs and interests, and also engages them much more effectively by utilizing their interests to motivate them to work. It also offers variety for me, the lecturer, in that I get to explore and teach different domains. It is not as much as it initially seems to be, if the principles 1-4 are followed, since many of the concepts are common across domains – I teach them about users, how they think, reason, learn and understand; they learn about design processes and the relevant components. It is just the focus and emphasis that changes, and the case studies used.

The disadvantage is that it is more work for me, in that I have to structure the material appropriately, find relevant case studies and research work as required, though the most awkward part is that this work has to be done after this initial lecture, during term-time, and so cannot be fitted around other demands on my time. This pressure, during an already busy part of the year, makes it harder to manage than would otherwise be the case. The other disadvantage is if there is no consensus amongst the students: some can feel that they have missed out if their topic is not focused on.

Use peer analysis and comment

In the design exercises my students undertake, they document them in a blog. This provides an easily accessible record of their work, which is accessible to their peers as well. The students are encouraged to monitor and read the other students work, and this provides a greater sense of community, some form of peer pressure to continue with the work throughout the semester (it is a continually developing design project), and allows them to comment on the work of others and to receive the benefit of peer perspectives[1, 4]. This helps them appreciate that others do not share the same principles and perspectives as they do (Point 1 above), and also reduces the need for me to provide feedback to them.

Blogging, being a Web2.0 and social computing approach, has also shown some success in engaging the students – they appreciate using a new(ish) technology, and through usage learn to understand its benefits and disadvantages[3].

Put HCI everywhere else

One way to address the pressures on the HCI parts of the curriculum is to integrate it into the other parts: instead of colleagues teaching just the waterfall model of software development, get them to discuss user-centered design

approaches, or agile programming. Instead of assessing programming just on algorithms used and object-oriented design, get it assessed on usability and interface design as well. Ensure HCI practitioners are invited as part of the industrial lecturers, if you have such a lecture series – or put one on if you don't. Dedicate part of the programming course to GUI programming, and teach aesthetic design/design patterns/usability/Apple | Microsoft | other guidelines/Neilsen principles in that part as well.

This may require you to participate in assessing other courses, or at least to collaborate with colleagues to integrate the parts you consider to be important in their modules – but it can lead to better educational outcomes.

Utilise the web

There is a wealth of information regarding HCI topics on the web – many of us make our material freely available online, and do not mind it being re-used elsewhere. There is little point in re-writing material if it has already been crafted, and it is not uncommon to find links to other web pages throughout the material I teach – if others have done a great job in explaining a topic, then it is better for me to use that material and talk around it than it is for me just to duplicate it myself. This approach is essential if you have designed the course around the current cohort and it takes you into unfamiliar areas, but useful however you are teaching.

CONCLUSIONS

We may ideally want to create architects of the future, and, in some cases, we have the students, time and opportunity to do just that. But many of us have to work at a lower, more basic, prosaic level, with builders – people who want to get a job done, in a manner which is sufficient to get them by – in a system that pressures us to do more with less. I have presented four principles by which we can hope to produce people able to be good at HCI despite these pressures, and some additional approaches from my personal experience that may assist others in achieving similar goals.

REFERENCES

- [1] Beale R., Blogs, reflective practice and student-centered learning. in *HCI 2007*, (Lancaster, UK, 2007), BCS Publishing.
- [2] Beale R.. University HCI—Squeezed Into Where? *ACM <interactions>* (September-October).(2005), 15-16.
- [3] Oravec, J.A. Blending by Blogging: weblogs in blended learning initiatives. *Journal of Educational Media*, 28 (2-3).(2003), 225-233.
- [4] Williams, J.B. and Jacobs, J. Exploring the use of blogs as learning spaces in the higher education sector. *Australasian Journal of Educational Technology*, 20 (2).(2004), 232-247.