Human-Computer Interaction in Software Engineering Courses

DISCUSSION SUMMARY

Chris Phillips and Elizabeth Kemp

Introduction

A working group was convened to consider human-computer interaction (HCI) issues in relation to software engineering (SE) courses. Two broad questions were posed to the group:

1. Is this aspect (HCI) of software development adequately covered in SE courses?

2. Do contemporary graphical user interface (GUI) development tools over-emphasise the interface at the expense of functionality?

This report has been written to reflect the discussions by the participants and to set out specific recommendations concerning the issues raised. It does not necessarily represent the views of any individual participants, nor the official views of their academic institutions.

Is this aspect of software development adequately covered in SE courses?

There was a consensus that the short answer to this question was no! HCI tends to be regarded as an advanced level topic, taught at the third or fourth years of current undergraduate computing curricula, and is often viewed as an ‘add-on’. This is evidenced by the lack of emphasis given to HCI in many current SE textbooks, despite the fact that interface design involves techniques which span the lifecycle, and which are a part of software system design, e.g. task analysis. HCI is not well integrated with SE lifecycle models, and is often perceived as only a small part of overall system design despite the fact that “from the user’s point of view, for a modern application with a GUI, the interface is the system” [1].

Discussion was focused on developing a set of recommendations to address these perceived shortcomings. This included consideration of how SE and HCI issues might be brought together, of suitable approaches to teaching interactive software systems development which would help achieve this integration, and of methods for evaluating system designs produced by students. The following recommendations emerged:

1. Early exposure should be provided in SE courses to HCI issues and possibilities, including a wide range of interaction styles. The HCI and SE strands should be teased out and then integrated. The link between problem solving and interaction issues should be established. User-centred and use-centred approaches could be considered.

2. The notion of designing systems from the ‘outside in’ rather than from the ‘inside out’ should be introduced (the ‘black box’ perspective), partly through exposure to visual design tools. The distinction between interaction objects and software objects should be made clear.

3. Interactive software systems should be considered from the viewpoints of functionality, usability, and visual attractiveness. Examples should be included to illustrate these views, and to point up the tensions which can exist between them during system development, e.g. the order in which they should be considered. It has been stated [2] that a useful system should be both functional and usable. Principles which can be applied to the design of interactive software in order to promote its usability have been grouped under the broad headings of learnability, flexibility and robustness [3]. The latter includes task conformance - the degree to which the system supports the user tasks in the way that the user understands them. Usability thus goes beyond functionality.

4. Exposure should be provided to highly transparent, low impact, inclusive system development techniques, e.g. paper mock-ups, sticky notes. These have the advantage of removing barriers and placing users on equal terms in the system development team, and of providing a more comfortable and accessible style of development. They require ‘non-ennobling’ technology supporting the development of cheap, highly modifiable throw away prototypes.

5. The value of usability inspections at each stage of system development should be emphasised [4]. Nielsen and Molich’s [5] heuristic evaluation technique can easily be used by students to evaluate bibliographic systems, for example.

6. Interactive designs produced by students should be subjected to competitive evaluation and usability testing by both the teacher and the peer group.

7. The importance of the user’s mental model of a system should be emphasised. Different users may
construct rather different mental models of the same system. The ease with which a new user can construct a mental model is an important measure of the usability of a system. The notion of metaphor, as a means of assisting the user in reasoning about what is happening within the system, is significant here.

Do contemporary GUI development tools over-emphasise the interface at the expense of functionality?

It was felt that the latter part of this question might be usefully reworded as “...at the expense of the interaction”. Current GUI development tools were seen as easy to use, even by novices, but rather limited in scope. In particular:

1. They do not provide abstract models of the interaction.
2. No link is generally provided with interface design metrics.
3. They concentrate on the interface at the expense of the interaction, i.e. they are poor at representing dialogue, and are focussed on the development of screens whilst largely ignoring navigation issues.
4. They are generally limited to a specific widget set, or some close derivative, i.e. they do not support creative or innovative design.

There is a need for tools to be integrated within particular methodologies.

Conclusion

The above recommendations are not seen as exhaustive and further work is needed in order to refine some of the ideas. They also need to be considered in the light of recent international curriculum proposals in HCI (e.g. [6]). A dilemma to be resolved in this connection is the need on the one hand to define an HCI curriculum, and the need on the other hand to integrate this with SE education.

There was a consensus among the participants that HCI issues should become SE issues. The prominence of HCI at the conference, as evidenced by the keynote address, suggests that a beginning has been made.

References


Workshop Participants:
Tim Comber Southern Cross Univ, Aus
Larry Constantine Univ of Tech, Sydney, Aus
Damian Conway Monash Univ, Aus
Bill Davey RMII, Melbourne, Aus
Bob Godfrey Univ of Tasmania, Aus
Dan Hawthorn UNITEC, Auckland, NZ
Leo Homes CIT, Upper Hutt, NZ
Elizabeth Kemp Massey University, NZ
Xiaosong Li Auckland University, NZ
John Paynter Auckland University, NZ
Chris Phillips Massey University, NZ
Arthur Tatnall Victoria Univ of Tech, Aus
Ewan Tempero Victoria University, NZ
William Wong University of Otago, NZ