Incorporating Human-Computer Interaction into the Undergraduate Computer Science Curriculum

Sarah Douglas, University of Oregon,
<douglas@cs.uoregon.edu>
Laura Leventhal, Bowling Green State University, <leventha@cs.bgsu.edu>
Marilyn Tremaine, New Jersey Institute of Technology, <tremaine@acm.org>
Craig E. Wills, Worcester Polytechnic Institute, <cew@cs.wpi.edu>
Bill Manaris, College of Charleston, <manaris@cs.cofc.edu> (Moderator)

Introduction

This special session presents issues, approaches and experiences related to incorporating Human-Computer Interaction (HCI) into the undergraduate CS curriculum. Since the publication of the ACM SIGCHI Curricula for Human-Computer Interaction in 1992 [1], CS educators have tried various implementations of these guidelines. These implementations have been mainly offered as elective courses or modules within other courses, partly because the CS mainstream has been slow to recognize the importance of user interface design and HCI issues in software development.

Today at least 50% of the code written for software applications is specific to the user interface. Average end-user expertise is continually dropping due to the proliferation of inexpensive computers available to the general public. Accordingly, user interface effectiveness has become increasingly important in software development, as the user interface has the power to “make or break” a software product. Through its emphasis on user-centered design, task analysis, and usability evaluation among other topics, an HCI course addresses more than 1/3 of the guidelines and imperatives of the ACM Code of Ethics [2]. This includes issues related to professional review, evaluations of computer systems, social responsibility and quality of life, and non-discrimination and dignity of end-users.

CS educators have begun realizing the importance of incorporating HCI into the educational experience of students. Approximately 40% of the CAC-accredited degree programs include an HCI course at the upper level; however, this course is offered mostly as an elective (37% of degree programs) as opposed to a required course (3% of degree programs) [3]. Also, HCI is included as a core knowledge area in Computing Curricula 2001 [4, 5, 6]; unfortunately, only six hours have been devoted to it, as opposed to, say, 36 hours for architecture.

Sarah Douglas

The ACM IEEE Curriculum 2001 will have a major impact on the content of international undergraduate computer science programs for the next ten years. As such, it represents a significant opportunity to revise, revitalize and modernize curriculum in the exploding field of computer science. Does the proposed curriculum adequately represent new and important areas such as HCI? Has the perception of HCI as ‘not really part of computer science’ changed? What should be the role of HCI in a computer science curriculum?

To answer these questions I will draw on two experiences. The first is my participation in the creation of Curriculum 2001 as the Chair of the Human-Computer Interaction Knowledge Focus Group. Our HCI Focus Group is composed of seven internationally well-known HCI educators. As requested by the Curriculum 2001 guidelines we developed a proposed HCI curriculum that has now been integrated into the overall proposal called ‘Steelman’. The second relevant experience is my twenty years as a computer science professor pioneering the development and teaching of HCI courses at the University of Oregon.

To jump to the punch line, although the Curriculum 2001 has recognized a need for HCI, it’s been a struggle. My participation has been an exercise in politics that has ultimately yielded a more positive and significant role for HCI in the undergraduate computer science curriculum. However, in my opinion, HCI content is still not adequately represented. I don’t believe this is for lack of trying. I believe it is because there are some basic assumptions about the content of computer science held by many academics that prevent the full inclusion of HCI, and a tendency of the HCI research community to isolate itself from mainstream computer science thus failing to relate HCI content to standard topics in computer science.

Laura Leventhal

Recent drafts of the Computing Curricula 2001 have suggested that undergraduate Computer Science Curricula must include units in software engineering (SE) and HCI. While the first inclination might be to offer separate SE and HCI courses or to incorporate minimal HCI concepts in a traditional SE course, we suggest an alternate approach.
At Bowling Green State University, our HCI course is a required course for undergraduate Computer Science majors. The students in the course are typically second semester sophomores or juniors. The heart of the course is a presentation of the usability engineering lifecycle. As we teach this material, we also introduce students to concepts of the software engineering lifecycle. We cover “traditional” HCI concepts such as interaction styles and assessment, but in the context of a lifecycle model. Our goal is to familiarize the students with specific HCI concepts as well as the larger software and usability engineering concepts at the same time. Our students complete a group, term project in which they develop a prototype user interface. In the course of the project, the students complete a number of phased activities, including task analysis, design and assessment. Many of our students go on to take our full course in software engineering as a senior-level elective, following the completion of our HCI course.

Marilyn Tremaine

Human-Computer Interaction has traditionally been included in an undergraduate Computer Science Curriculum as an elective in the student’s third or fourth year. The course is either (a) a survey course talking about various elements of interface design, e.g., menus and input devices or applications that have a large user component, e.g., software visualization programs, prototyping tools and groupware; or (b) it is primarily a Computer Science course that has students building cool interfaces, especially multimedia interfaces. These courses do not provide needed HCI training for the computer science student.

There are three problems with the above approach. First, it does not reach all students. User Interface Design is so pervasive today that all students need to be exposed. Second, it teaches students that HCI is a special sub-discipline of Computer Science not an integral part of the field and third, the courses that are taught do not prepare students for the problems they will encounter in their career.

The approach I recommend is to teach HCI in the same basic first and second year courses that data structures, algorithms and graph theory are taught in. For example, as students learn recursive functions and are given the problem to develop a Towers of Hanoi program, they can build two screen representations of the problem, one using more external states than the other. Testing the programs on friends will demonstrate the importance of providing external memory for problem solving tasks. Computer scientists are not expected to be user interface specialists, but they should be able to work with these people. As such, the course in HCI should cover how HCI is incorporated in the software development lifecycle and have the students build a user application using the HCI techniques they learn in the course.

Craig E. Wills

The WPI Computer Science department has offered a junior-level HCI course since the 1980s. Not only do we have this course, but it is a core course that all Computer Science majors are encouraged to take (we do not have course requirements). Actually, most students do take this course. As an instructor, I find it can be a frustrating course for students because there are few “right” answers, but instead many guidelines. Students also are confronted for the first time with the realization that an intuitive interface of their design is not so obvious to their peers. I think this frustration is healthy. I think peer reviewed projects are extremely valuable for both the reviewer and reviewee.

A recent IEEE Computer article ranks knowledge areas in terms of their importance to software professionals [7]. HCI/user interfaces are ranked number two (behind negotiation) as the topic with the greatest knowledge gap – where importance most exceeds current knowledge.

Overall the interactions of HCI with other computer science topics, particularly systems and the Web is a topic worth exploring. I think portions of HCI could be introduced in other courses such as Software Engineering, Networks, Multimedia and Webware. Student projects are also a natural place for students to use and evaluate user interfaces.

We have explored this approach through a NSF-supported grant to set up a lab for HCI, Networks and Webware. I believe Webware sits at the convergence of Networks and HCI topics and that many natural interactions between topics can be investigated via the Web.

References