A Student Project in Software Evaluation

Michael F. Czajkowski, Cheryl V. Foster, Thomas T. Hewett, Joseph A. Casacio, William C. Regli, Heike A. Sperber

1Department of Mathematics and Computer Science
2Department of Psychology, Sociology, and Anthropology
3Department of Media Arts
Drexel University
3141 Chestnut Street
Philadelphia, PA 19104
+1 215-895-2461
(hewett@drexel.edu)

ABSTRACT
Properly educating computer scientists involves teaching effective means to properly engineer a system. An important part of such engineering work is ensuring that the computing system is both useful and usable. While many systems out there today are difficult to use, performing usability engineering on a system during its development has been shown to be an effective way to make a system more usable. The problem is fitting practical experience into the curriculum. This paper discusses a case example of how a team of undergraduate students learned to take a software system during its developing stages and perform effective usability engineering following the "thinking out loud" methodology.

1. INTRODUCTION
An undergraduate course in Human Computer Interaction taught by one of the authors has gone through several evolutionary changes in its 20-year history. At one time the course focused on readings and examinations. More recently the course involved students in the development of paper prototypes of personally useful applications [4]. Currently, the course focuses on designing and evaluating complex working environments for specialists (e.g., engineering designers, scientific problem solvers, etc.).

The final course project requires teams of students to conduct a "Thinking out loud" and Protocol Analysis study [2] to explore the needs of an expert problem solver. Typically student project teams are formed during the 6th week of a 10-week term. Whenever possible, the teams have varied disciplinary backgrounds (e.g. the team which did the project described here consisted of students from Computer Science, Psychology, Design Arts, and Information Systems). This paper describes the preparation of the students and a case example of a project done by one such team of undergraduates. The goal is to illustrate how much can be accomplished, the types of lessons that can be learned from such projects, and the benefits to the students.

The project consisted of a formative evaluation performed by a team of undergraduates on a software tool designed to support conceptual mechanical design, CUP [1]. CUP is a virtual environment that allows engineering designers to sketch out, in 3D, a conceptual layout for a new product model and to specify major structure-behavior-function relationships among assembly components.

2. STUDENT PROJECT
The project presented in this paper used formative evaluation procedures to gather feedback from target users. While software is under development formative evaluation is a useful tool for keeping the project on course toward successful completion of its goals. Also, redesign based upon feedback from a representative sample of the target users yields an artifact superior in quality and time/cost effectiveness [6].

2.1 Why an Empirical Evaluation Project?
An important characteristic of evaluation-guided development is its ability to support a human-centered focus for the software being developed. With hundreds of computer applications generated each year, it is important to ask how well these applications perform in the workplace for which they were designed. The creators of software systems are faced with increasingly complex issues to address, not the least of which involves consideration of which functions to provide for the user of the computing system. Early and repeated user feedback can save development time and costs by early identification of features not needed by users.

For example, in collaborative engineering systems, engineering designers are faced with an unprecedented number of new, complex tools—ranging from simulation and analysis systems to instant prototyping and fabrication services available over the Internet. It is never easy for software developers to combine all constraints effectively into an artifact that is both powerful and simple to use. One important part of software testing involves considering the types of experiences encountered and work-processes performed by the system's end users on a daily basis.
The purpose of evaluation-guided development is to ensure that a computing system can serve the target user in a manner that is efficient and fosters productivity. After the artifact is released, summative evaluation will give developers a good idea how their creation performs compared to alternatives.

2.2 Classroom Preparation of Students
To prepare students for making software design choices based upon empirical feedback from problem domain experts, a thinking out loud project is an integral part of their course on Human-Computer Interaction (HCI). At the beginning of the term the students are told they are going to be conducting a study of one or more real users engaged in one or more real design tasks. The term “design” is interpreted flexibly and some teams have used the project to understand the thought processes of such diverse individuals as a landscape architect or a recording studio engineer. This helps students to think about and identify functional requirements for an initial design. Other groups of students have chosen to use project and methodology to evaluate the usability of a piece of software currently under development.

Classroom preparation of the students for these projects consists of three distinct stages. First, the students are provided lecture material covering the importance of attending to the usefulness and usability of software. For example, during the class offering in which the student authors of this paper participated, lectures were supplemented and related to the arguments suggesting that the failure of economists to find economic benefit from the information technology in the workplace is related to badly designed software [7].

The second stage of preparation of students for their course project consists of a series of group discussions and activities focused on small group problem solving. At first these exercises consist of simple activities such as developing a group consensus about some aspect of the course material. In the class period following their group discussion a randomly selected subset of the discussion groups presents the results of their discussion to the class as a whole. Additional motivation for developing a shared understanding of the material is created by basing a quiz on the topic of the group discussion.

Third stage preparation for the course project consists of classroom-based exercises focused on building up the skills needed to conduct either a requirements gathering or formative evaluation study. Once students are in their project groups the group discussions and in-class exercises are more tightly focused upon practicing skills needed to conduct their course project. For example, one classroom discussion consists of reverse engineering the design of a hypothetical being based upon information given about its use of a set of tools. For example, in one exercise students are asked to assume they are extra-terrestrial alien archaeologists visiting an Earth that is barren of life. Their goal is to infer characteristics of users from a standard PC. They are, however, required to develop a “picture” which includes only the necessary characteristics, and as many as possible of the sufficient characteristics, of human PC users. This discussion is followed by group presentations of the results. Presentations are conducted in the format of a mini-conference and students from other groups are allowed to ask for justifications for conclusions drawn by the group currently making its presentation. The discussion and presentations help to illustrate three major points. One point is that different design teams will often produce different answers to the same questions. A second point is that the typical set of design-requirements under-specifies the problem solution so that there is actually a range of possible solutions that satisfy those design requirements. Finally, the exercise provides ample illustration of how easy it is to be trapped by the assumption that one is similar to the user and thereby drawing erroneous conclusions that are not tested against the data.

In another classroom exercise the students are required to actually conduct a short thinking out loud usability study involving a paper simulation of a badly designed menu-based word processor. In this simulation at least one student each plays the role of an observer, a user, the CPU, which provides instructions to the user, and the Co-Processor, which handles the screen displays. The instructions and screen displays are already prepared and written out on a sheet of paper. (Only the CPU sees the written instructions.) In addition to the practice with the evaluation methodology, this exercise demonstrates for the students the importance of prior preparation for and pilot testing of their study procedures. Finally, the project teams are provided with at least 2-3 hour class periods during which they are expected to meet for discussion and planning of their final course project and for planning the preparation of their project report.

3. AN EXAMPLE STUDENT PROJECT
There are several techniques that can be used to gather information about how engineers design mechanical and/or electronic artifacts (e.g., questionnaires, observational studies, etc.). The method used in this evaluation study involves thinking out loud and protocol analysis. An extensive review of work on this procedure [2] has shown that analysis of protocols, and of retrospective verbalizations by problem solvers, can be a useful source of information in understanding the expert problem solving process. When applied to gathering feedback needed to evaluate the usability of software this technique is equally useful [5].

The use of thinking out loud combined with retrospective procedures (i.e., post-study discussion of particular points) was employed in this study because it allows tracking of the thought processes and intentions of the users while they are engaged in a given task. Asked to speak their minds while completing the task, the participants were able to provide the experimenters with useful information as to how sample users go about their work processes. Post-study discussion with the users immediately following their problem solving activities allowed for further clarification of difficulties that the users encountered during their use of the software.

3.1 What is Being Evaluated?
This portion of the paper describes one cycle in the evaluation-guided development of a 3-D modeling tool for conceptual understanding and prototyping in engineering design. CUP enables a user to specify a spatial layout of components and subassemblies in a mechatronic artifact (an object designed with both electronic and mechanical components and/or subassemblies). CUP provides the means for a user to specify structural, behavioral, and functional (S-B-F) information about the various components and subassemblies.

3.2 Evaluation Criteria
Part of an evaluation of CUP should be rooted in knowledge of the engineering design process and how engineers actually work
in practice. Consequently, the students doing this project developed evaluation criteria based on the task-related needs of CUP users. The main focus was, "Does CUP's interface really enable designers to create conceptual designs of artifacts?"

3.3 Participants
Participants in the study were undergraduate volunteers recruited based on educational background and experience. Although CUP is eventually expected to provide support for working professionals, at this stage of testing it was desirable to find and eliminate any basic usability faults with the system before examining the full range of functionalities using the more expensive time of experienced engineering designers. It was also desirable to have a range of test participants to allow for comparisons between relatively experienced and relatively inexperienced student users of CAD systems. The seven participants used in the study were all Drexel University students with varying educational backgrounds with a focus on majors in engineering. All participants had at least a novice proficiency in computing. Only one participant had prior experience with CUP.

3.4 Task Selection
Members of the evaluation team experimented with CUP themselves to develop a set of design tasks that would be approachable by both engineers and non-engineers alike. At this stage of evaluation guided development the goal was to evaluate the intuitiveness and functionality of the CUP interface. For the users in this study, the simple design of a number 2 lead pencil was selected because it required the use of most of CUP's main features. It was expected that each participant would create 4 separate objects to represent the graphite, wood, eraser, and metal ring of the pencil and then link the objects together with some description of the relationship between the linked objects.

3.5 Study Procedures
First, each participant was given a brief overview of the project and the application. Participants were then given a brief, open-ended preliminary questionnaire to assess their general level of computer expertise and background. The investigator then read the instructions for thinking out loud to the participant, followed by 2 or 3 practice thinking out loud problems adapted from [2]. Each participant was then presented an example of a template object already designed with CUP. Next the participant was allotted 3 minutes to explore and get an understanding for the software before being given the task of creating a conceptual design of a pencil.

During work on the design task, each participant was recorded and observed by one or more investigators who were out of the participant's line of sight. There was no social interaction between the investigators and the participants at this time except for needed cues such as "please keep talking". During the design task investigators noted any problems that the participant seemed to have (e.g., spending several minutes trying to label an object). When the user encountered a difficulty and asked something such as "What should I do here?" the response was to reply with a question such as "What do you think you might do here?" Coaching by an investigator was only allowed after it was clear that the user had encountered a problem that would prevent further progress. The nature of any difficulty that required coaching was noted and then the participant was shown how to solve the difficulty.

Immediately upon completion of the design task, the investigators asked participants to explain the source of their difficulties (regardless of whether they had been coached or not). During the thinking out loud portion of the study a few participants had to be routinely reminded to keep talking, but others freely shared their thoughts. At the conclusion of the post design interview, each participant was asked to complete a second brief questionnaire. This questionnaire asked participants to give any additional feedback about the software (problems, suggestions, etc.) that might not have been discussed during the interview.

3.6 Evaluation Results
Most of the usability problems identified in these tests fell into two main categories, those associated with object manipulation and those associated with tool bar and menu functionality. Although a full discussion of the usability problems with CUP was provided in the original project report, the focus of this paper is not to detail the results but rather to underscore the importance of the evaluation as part of the teaching model. Therefore, we provide here only an overview sample of some of each type of problem encountered by the users studied.

One major problem most participants encountered was with manipulating objects they had created. Translation of an object required use of the right most of 3 mouse buttons. Users wanted to use the left-most button. Rotation of an object with the mouse was felt to be inaccurate and uncontrollable. Re-sizing of an object was difficult for some participants because there was no visual reference as to the scale of the world visible on the screen. Use of the tool bar was generally preferred over use of the menu bar. The majority of the icons (e.g. the primitive, delete, and manipulation buttons) were easily understood. However, the group and ungroup icons were hard to distinguish from each other and difficult to identify as representing group and ungroup. Menus were often employed only after a user felt that he/she exhausted all possibilities of the mouse and tool bar. Most participants did not find the Help Menu. All participants were successful in creation of design primitives, but the design window that appeared initially was generally found to be confusing.

3.7 Discussion
This section of the paper has provided an overview of a case study application of evaluation guided software development methodologies to software systems for engineering design. While the results of the evaluation will not surprise those who are experienced with usability testing methods, the students learned some useful lessons for further evaluation guided development of CUP. We anticipate that future human-centered evaluation studies will continue to provide useful feedback for the ongoing evaluation guided development of CUP.

4. DISCUSSION OF STUDENT LEARNING
There are three particularly important pedagogical benefits that have followed from the class participation and student project described above. First, the students involved have had the opportunity to engage in a "hands-on" course project, which made a contribution to an ongoing software development project. They have learned experientially how important it is to involve formative usability engineering in the creation of such a system. Second, the students have been able to consolidate their classroom based learning into a body of practical knowledge that will enable them to utilize similar evaluation procedures in future projects.
Third, the students have been deeply involved in the process of preparing this paper for publication and conference presentation. Teaching the importance of HCI in the classroom is a topic that certainly deserves much more attention than it has been getting [3]. The students involved in this project learned a considerable amount of the importance of HCI as part of the software development cycle. The students were able to extract how users approach a problem of using an unfamiliar graphical application. Furthermore, the students benefited in understanding that effective usability engineering can be a tool to of attack the productivity paradox [7]. At the onset of the course, these students had no background in how to implement the thinking out loud protocol [2]. In performing the study described here they were able to gain knowledge of how to incorporate such testing and results into an ongoing system design process.

5. CONCLUSION
By having students learn how to perform formative evaluation of systems, we believe that they are more likely to design usable and useful systems in the future. More importantly, one goal the instructor set for these students has been accomplished; they successfully learned to conduct a usability study in which they systematically contributed to the on going development of an software tool through usability engineering. Effectively, by the end of the class they had developed a significant body of experience with an evaluation methodology that has become a de facto "industry standard" in recent years. Even if they never actually conduct a future requirements gathering or usability tests, they fully understand how to do so and how to communicate with and use the results provided by those who will do such work.

6. ACKNOWLEDGMENTS
This work was supported in part by National Science Foundation (NSF), Knowledge and Distributed Intelligence in the Information Age (KDI) Initiative Grant CISE/IIS-9873005 and Grants ENG/DMI-9713718 and CISE/CDA-9729827. Additional support was also provided by The National Institute of Standards and Technology (NIST) Grant 60NANB7D0092 and in part by the United Kingdom’s Engineering and Physical Sciences Research Council Contract GR/COGMS while the third author was a Visiting Fellow at the Creativity and Cognition Research Studios, Loughborough University, UK.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the other supporting government and corporate organizations.

7. REFERENCES