Lessons for (Pedagogic) Usability of eLearning Systems

The ActiveMath Group:
Erica Melis et al.
University of Saarland and DFKI
Saarbruecken, Germany
melis@dfki.de

Markus Weber
ergosign GmbH
Saarbruecken, Germany
weber@ergosign.de

Abstract: The usability of an eLearning system is a key feature for its success. This paper describes several aspects of the usability evaluation of the user-adaptive eLearning system ActiveMath to exemplify how usability engineering techniques can be applied to and adapted for an eLearning system and what the particularities of an eLearning system compared with a 'conventional' application are. Topics tackled in the usability evaluation include the presentation and manipulation of the user model, the implementation of support systems and the design of on-screen texts. The benefits of an iterative approach to system design and of a close collaboration of developers, usability engineers and interface designers who have their expertise in diverse fields are described.

Introduction

With the wide-spread usage of computers, software, and the Web in every corner of our lives more and more people use computational tools. Typically, the users are not specifically trained in computer science and they want to receive computational support for their tasks rather than to spend much time with and focus much attention on learning how to operate a computational tool. Hence, it is not enough to develop a *useful* program but necessary to make it *usable*, by using methodological approaches from human-computer-interaction (HCI).

However, there are many computational systems and web pages around who have a badly designed user interface. This may have various reasons:

- a usable GUI has never been requested from the developers or they do not understand what the underpinnings of a usable GUI are
- users readily adapt to user interfaces and therefore, the need for a good user interface is neglected by the developers
- developers do not collaborate with designers or usability labs and therefore, there is rarely an opportunity to improve the usability by an interdisciplinary effort.

This paper describes several aspects of the usability evaluation of the eLearning system ActiveMath to demonstrate how important an interdisciplinary collaboration is and to exemplify how general usability engineering techniques have been applied to and adapted for an eLearning system and what the particularities of an eLearning system compared with a 'conventional' application are.

Usability and eLearning

The main goal of usability engineering is to optimize the user experience with an interactive system. Obviously, 'optimal user experience' means different things for different systems, e.g. finding the desired product as fast as possible on an eCommerce site or managing incoming phone-calls in the best possible way with a call-center application. This implies that – even though there are heuristics that serve as guidelines in designing and evaluating an interactive system, such as consistency, error prevention and clear feedback (for examples cf. Nielsen 1994) – the work of the usability engineer always involves a creative part in finding ways to adapt existing knowledge on best

practices in system design to the system that is currently under evaluation. eLearning systems are no exception to this rule. In fact, they are probably amongst the more challenging objects of evaluation for usability engineers. This is due to the fact that the eLearning sector is very heterogeneous with respect to content delivered as well as technologies employed, which makes it particularly necessary to incorporate knowledge from educational psychology and tailor the process of usability engineering to the particular system evaluated.

Making an eLearning system usable basically involves two aspects: technical usability and pedagogical usability. Simply put, technical usability involves methods for ensuring a trouble-free interaction with the system, while pedagogical usability aims at supporting the learning process. Both aspects of usability are intertwined and tap the user's cognitive resources. The main goal should be minimizing the cognitive load resulting from interaction with the system in order to free more resources for the learning process itself. A prerequisite for doing so is the usability engineer's detailed knowledge about human learning in general and learning goals and processes in a content domain in particular.

Case Study: Usability Evaluation of ActiveMath

ActiveMath is a web-based intelligent learning environment (for mathematics) that has been developed at DFKI Saarbrücken and Universität des Saarlandes. (cf. Melis et al. 2001) It generates learning material user-adaptively, i.e., dependent on the learner's goals, learning scenarios and preferences (input via menus), mastery level (self-assessment input at first system entry and updated during the learning process), and generically integrates several back-engines useful for exploratory learning – among them the computer algebra systems Maple and MuPad that are also used to generate local feedback in exercises. A dictionary serves as a tool to on-click view the meaning/definition of a concept and its dependencies with other concepts, etc. ActiveMath dynamically generates learning suggestions. Other useful tools in ActiveMath are the dictionary, personal and open notes, an adaptive annotation of the table of contents, etc.

The user interface has the following states and components that had to be user-tested:

- login page
- choice menus
- self-assessment
- (adaptive) content presentation
- table of contents
- dictionary
- student model
- interactive exercises with back-engines
- presentation of example solutions

The developers of the ActiveMath Group and the cognitive psychologists and interface-design specialists from ergosign GmbH had the opportunity to collaborate on both, the functional and pedagogic usability of the eLearning system ActiveMath. This effort included design sessions and empirical usability tests with people using ActiveMath in seminars and people who belong to the group of prospective users of ActiveMath (students of mathematics and students enrolled for teacher training). We report about some general and crucial results in order to help other developers to avoid wrong development paths and pitfalls in their user interface design.

Selected Findings from Case Study

User Model and Knowledge Visualization

ActiveMath makes use of user modeling to adapt content presentation to the individual needs of the learner. Personalization has a high potential for efficiently supporting the learning process, but ongoing changes in the interface (e.g. when links are added or removed) can confuse the user. So, it is important to provide the user with a framework that supports her in developing a coherent mental model of the system and its adaptivity.

Regarding the manipulation of the student model by the user, the general usability principle of *consistency* has been applied to this functionality by suggesting to refrain from using two different kinds of presentation of the user's knowledge level (colors in one case and numerical values in the other) and to present the mastery level with three colors throughout the application. The use of the colors is still differentiated enough to convey the mastery level. Moreover, colors do not overburden the user like the use of numerical values can. This is an example for the mediating function the usability engineer has between the developers and users of a system: the developers have good reasons for representing the knowledge level of the user as finely grained as possible (as the foundation of the user model), while some 'translation' into an understandable code for the learner has to be made, who is interested only in whether her mastery of a certain topic is good, average or poor and who usually will not differentiate more, when she is asked to evaluate her own competency.

Additional thoughts had to be invested for finding suitable forms of visualizations for the content structure relations. This can only be done after gaining insight into the structure of the content domain (mathematics) and by collaborating with the content experts in close feedback circles. The collaboration led to the suggestion and evaluation of expandable/collapsible structures and knowledge map visualizations for the student model that go beyond structures found in conventional applications without a pedagogical background. This shows that the general usability principles have to be combined with results from learning and pedagogical psychology as well as insights into the content domain.

Support Systems

Another area where general guidelines from usability engineering have been merged with didactical knowledge is the help system of ActiveMath. Basically, different kinds of support should be available for different user needs such as guided tours to quickly get accustomed with the program and detailed help for reference purposes. Help should be easily accessible and provide specific support for the student's problem solving. In the case of ActiveMath, this was not sufficient. First of all, technical help on handling the program and didactical support concerning the content had to be distinguished. A second distinction separates novice and advanced learners.

For technical help, the basic guidelines for online-help design apply. Pedagogical support is different, however. Firstly, while it is appropriate to immediately provide the full solution in technical help concerning a problem with handling the system, this may not be the best option in tutorial help. For instance, when the support system will only provide a hint on the correct solution of a mathematical problem instead of providing the solution itself, this can motivate a learner to work on the problem herself in an active way instead of just receiving the correct answer.

Secondly – quite different from conventional help systems – ActiveMath distinguishes active and passive help on the basis of findings from learning and problem solving research. Active support is directly communicated to the user, whereas passive support has to be explicitly requested.

Such a distinction makes sense when adapting to the student's level of expertise and self-guidance. Novice users tend to use an eLearning system in a passive way and therefore need guidance and explicit support by the system even if they do not request it. Advanced learners tend to better self-regulate their learning and can actively request help from the system while active help may be regarded as obtrusive in the extreme case. This notion was also validated during user interviews. Therefore, it was suggested to give advanced learners the option of toggling active support and/or only provide active support only, when the user is clearly wrong. This example shows how expertise in the design of online help system can be combined with a user analysis and knowledge from cognitive psychology to design the help system for an eLearning system.

On-Screen Texts

Another area in which general usability guidelines have to be adapted for eLearnings system is the creation of on-screen texts. When reading texts on screen, the reading behavior of a user differs from the book-reading behavior. This leads to general usability guidelines for the design of on-screen texts. One of the guidelines is to keep on-screen texts short. Obviously, the modifications that can be made to text in learning material are constrained by the requirement that it has to serve the learning and understanding of the user. The constraints for texts in the domain of

mathematics are especially strict as they are inherently structured and any transformation (especially shortening them) may compromise their validity. In addition, the intended user of ActiveMath uses the system in order to gain deeper knowledge on university mathematics rather than just browsing. Therefore, this user has a higher tolerance for text-length than the average user of an eCommerce website. So, the recommendations on modifications of text focused on making sure that the texts contributed by different authors were similar in style and that external sources should be clearly indicated so that no confusion results on the learner's side. Other recommendations dealt with the visual design of text in order to make it as pleasant to read as possible through font type and size, (sparse) use of graphical elements to structure content, etc. These recommendations do not imply that we advocate simply feeding whole textbooks into eLearning systems. As a designer one should be aware of the fact that users tend to print out long texts to read them on paper and acknowledge this behavior. eLearning users have a higher tolerance concerning text length, if there is some motivation to learn from screen such as the adaptivity in the case of ActiveMath. So, the reading experience should be pleasant while it should still be possible to easily produce paper versions of the content as with a dedicated feature in ActiveMath.

A superordinate demand for designers of eLearning systems like ActiveMath consists in finding ways to optimally integrate the usage of the system with the user's learning context such as classroom. For the usability engineer this means extending the scope of his examinations beyond the system itself and take into consideration the context in which it is used (an elementary requirement in usability engineering).

In the case study this meant reflecting the scope and limits of the current ActiveMath. In interviews, students stated that when they were using the system for learning, they often did this in combination with pencil and paper (for quick calculations). One approach for adapting the system to such a habit was to provide a means for 'scribbling' online. A step in this direction was made by giving the user access to a note pad. However, this tool could not handle mathematical symbols and, thus, was of limited use only. Instead, it was suggested not to spend lots of resources on the online replication of scribbling because it should not be the goal of ActiveMath to replace all other tools the learner is using. Instead, the system should focus on providing interactive tools that provide the user with added value such as step-by-step calculations of example tasks with corresponding feedback.

Many eLearning systems seem to fail because they aim at replacing proven methods of learning instead of improving the learning experience, where there is room for improvement.

Iterative approach

The iterative approach is widespread in the area of software engineering. The ideal procedure involves several iterations of prototyping / implementation and testing. In the case of an eLearning application, the focus on technical usability or pedagogical usability, respectively, shifts during the iterations. While the first testings serve to eliminate technical hurdles in interaction design, didactical design becomes more and more important during the following iterations. This makes sense because in order to examine the effectiveness of didactical measures, it has to be ensured that their effects can unfold and are not diluted by problems the user has with basic interaction design. Of course, problems with interaction design can still occur in later phases, but our experience shows that those usually require corrections more limited in scope than those issues discovered in earlier phases. Collaboration of the usability engineer with the content experts also intensifies as the testing and redesign proceeds. This procedure helps to tailor the approach of user-centered design to the specific requirements of the system.

References

Melis, E., Buedenbender, J., Andres, E., Frischauf, A., Goguadse, G., Libbrecht, P., Pollet, M. & Ullrich, C. (2001). ActiveMath: A Generic and Adaptive Web-Based Learning Environment, *Artificial Intelligence and Education,* 12 (4), 385-407.

Nielsen, J. (1994). *Usability Engineering*. San Francisco: Morgan Kaufmann.