ADAPTIVE NAVIGATION SUPPORT IN A WEB-BASED SOFTWARE ENGINEERING COURSE

Maria Virvou, Victoria Tsiriga, Maria Moundridou
Department of Informatics, University of Piraeus,
80 Karaoli & Dimitriou St.
Piraeus 18534, Greece
{mvirvou, vtsir, mariam}@unipi.gr
Phones: +3014142269, +3014142131, +3014142133

Abstract
In this paper we introduce a Web-based Adaptive Educational System for the domain of Software Engineering in higher education. The system uses a combination of stereotypes and the overlay technique to construct and update the model of each individual student. A domain model representing knowledge about the domain of Software Engineering is also part of the system. Based on the information held in domain and student models, the system is able to adapt instruction to the knowledge level of each student. The system uses the adaptive hypermedia technique of link annotation to support the student while s/he studies theoretical issues or solves exercises.

INTRODUCTION

In recent years, most universities and an increasing number of big companies all over the world feel the necessity to offer virtual Web-based courses to educate their students or employees (e.g. Ebner et al., 1999; De Bra, 1996). A common complaint, among many others related to implementing tutorial software in educational settings, has been the incapability of educational systems to adapt to the particular learning needs of the student (Eklund & Brusilovsky, 1998).

Adaptivity is an essential matter especially in Web-based educational software, since distance learning systems aim at reaching a much more heterogeneous group of learners in settings where no teacher is available to help users during the learning process. This challenging goal forced in the recent years a number of research groups to engage in research on adaptive Web-based educational systems. As a result, quite a lot of Web-based adaptive educational systems exist, making use of various methods and techniques. Brusilovsky (1996) provides a detailed review of adaptive hypermedia methods, techniques and systems and distinguishes between two main adaptive hypermedia technologies: (a) *adaptive presentation*, the case where adaptation is performed at content level and (b) *adaptive navigation support*, which is performed at link level. Both these technologies have been evaluated and the results offer strong evidence that their use in an educational adaptive hypermedia system can have a positive effect on students' learning and comprehension of the domain (e.g. Boyle & Encarnacion, 1994; Murray et al., 2000).

In this paper we introduce a Web-based Adaptive Educational System for the domain of Software Engineering. The courseware is aimed to be used by undergraduate students in a computer science discipline. The system's main characteristic is that it offers navigation support to students, adapted to their individual needs and knowledge. To achieve this, the system is based on two models:

the domain model (representing knowledge about the Software Engineering domain) and the student model (representing knowledge about the individual student). In the subsequent sections of this paper we will describe the domain and student models and present how adaptive navigation support is performed.

REPRESANTATION OF THE DOMAIN

The key to adaptivity of the Software Engineering courseware is the system's knowledge of the structure of the domain being taught. The domain knowledge is represented in a form of concept network, depicting the relations between the concepts of the domain. Each node of this network represents a topic, which may consist of one or more topics or concepts. There are two types of links between nodes: part-of and prerequisite. A part-of relation points from a more general topic to a more specific topic or concept, which is one of its parts. For example, there is a part-of relation between the topic discussing the "Model of Life-cycle" and the topic explaining the "Waterfall Model". A prerequisite relation points from a concept to another, which is its prerequisite. For example, the concept of "statechart diagrams" in an object oriented methodology, points to the concept of "class diagrams" which is its prerequisite..

In addition, each topic has an associated difficulty level. There are four difficulty levels, namely "easy", "average", "difficult" and "very difficult". Finally, tests examining the knowledge that must be acquired by studying a topic are associated with a particular topic; these associations are also part of the domain knowledge.

MODELLING THE KNOWLEDGE LEVEL OF THE STUDENT

The Software Engineering Courseware uses a combination of stereotypes and the overlay technique for student modelling similarly with other systems, such as (Hohl et al., 1996; Virvou & Moundridou, 2000; Virvou & Tsiriga, 2001a; Virvou & Tsiriga, 2001b). Stereotype modelling is used to initialise the student model, in case a new student is registered into the system. The system then updates the student model based on the actions of the student while interacting with the courseware. In order to be able to track the student's mastery of the domain, the student model keeps a *concept-value* pair for each concept of the domain knowledge. The *value* is an estimation of the student's knowledge level on a specific concept.

In particular, each time a new student is registered into the system, s/he is initially assigned to one of four distinct stereotypes, namely novice, beginner, intermediate and expert, according to her/his performance on a preliminary test. The preliminary test is given to students before they have interacted with the courseware. Based on the stereotype that the student belongs to, initial values are set for all the concepts of the domain knowledge. If, for example, the stereotype model indicates that a student is "intermediate", then every concept rated as "easy" and "average" will be considered already known by the student.

In order to monitor the student's progress and knowledge proficiency level, the domain concepts are associated with hypertext pages of the structured theory hyperdocument. For each theory page visited during the student's interaction with the Software Engineering courseware, the corresponding concept is marked as "read" in the student model. Furthermore, if the student has solved correctly exercises related to

that concept in a percentage greater then a threshold, the concept and all its prerequisite concepts are considered “known”.

ADAPTIVE NAVIGATION SUPPORT

One way to support a user while s/he navigates through a hypermedia space is by adapting the links of this space based on the model of each individual user. Link adaptation tries to simplify the rich link structure to reduce orientation problems, while maintaining a lot of navigational freedom, a typical property of hypermedia systems (De Bra et al., 1999). Several methods of adaptive navigation support have been described (Brusilovsky, 1996), such as adaptive link sorting, annotation, hiding, direct guidance and map adaptation. According to the link sorting technique, the available links in the hypermedia space are ordered on the basis of the student model, displaying the most relevant links first. Using adaptive link hiding, the links that are not ready for the student to visit are hidden from the student. With the direct guidance technique, a "next" or "continue" button is shown to the student. This button leads the student to the particular page that the system considers most appropriate for the student to visit. Finally, map adaptation technology comprises various ways of adapting the form of visual hypermedia maps presented to the user.

In the case of the Software Engineering courseware, the method of adaptive link annotation is used to support the student while studying the course material or selects an exercise to solve, alike other systems such as (de La Passadiere & Durfesne, 1992; Weber & Specht, 1997). Evaluations of systems using adaptive link annotation (e.g. Brusilovsky & Pesin, 1998) provide strong evidence that learning using link annotation is faster, more goal-oriented, and significantly reduces the number of steps to cover the hypermedia. The idea of adaptive annotation is to augment links with some form of comments, which can tell the user more about the current state of the nodes behind the annotated links (Brusilovsky, 1996). These annotations can be provided in textual form or in the form of visual cues. The Software Engineering courseware uses different font types and icons to provide adaptive navigation support. Whenever a link appears in the table of contents, exercises, or other pages, the font type and the icon that appears in front of the link are annotated so as to reflect the state of the topic or exercise behind the link, with respect to the student’s current knowledge state.

In particular, when presenting links that point to pages of the theory hyperdocument, there are four possible annotations. The links that are in bold letters and have an arrow pointing to them correspond to topics that are highly recommended by the system. Highly recommended links point to theory pages that the student has visited during a previous interaction but s/he did not perform satisfactorily while solving exercises related to the concepts presented in the topic. Furthermore, links that are in bold and italicised letters and have a sign with an exclamation mark in front of them point to topics that are ready for the student to follow and are recommended by the system. The links that are considered ready and recommended point to theory pages that the student has not visited yet but has read and knows all the concepts that are prerequisite to the concepts associated with the link. Links that are presented in italicised letters and have a check mark icon in front of them correspond to topics that have already been visited and are known to the student. Finally, dimmed links with a stop sign before the link correspond to topics that are not ready for the student to visit.

A topic is considered as not ready when the student does not know some concepts that are prerequisite to those discussed in the topic.

When presenting links to exercises that evaluate concepts of the domain, three types of annotations are used. The links that are in bold letters and have an arrow pointing to them correspond to exercises that are recommended by the system. The exercises that are recommended have not been solved by the student or the student has faced difficulty in solving them and are related to a part of the theory that the student has already visited. Moreover, links that are presented in italicised letters and have a check mark icon in front of them correspond to exercises that are related with topics that the student has mastered to a satisfactory extent. Finally, dimmed links with a stop sign before the link correspond to exercises that are not ready for the student to solve, since the student has not read the corresponding part of the theory related to one or more concepts evaluated by the exercise.

CONCLUSIONS AND FUTURE WORK

In this paper we have introduced a Web-based educational system for the domain of Software Engineering. The adaptation decisions of the courseware are made based on two factors: whether a concept has been "read" by the student, and how well s/he performs on exercises related to that concept. Based on the domain and student models, the Software Engineering courseware provides adaptive navigation support using the adaptive link annotation technique while the student navigates through the course material or selects an exercise to solve.

Within the future plans of this research, is the evaluation of the system in real educational settings in order to determine the effect that it could have in students' motivation and learning in the domain. Furthermore, we plan to produce alternative versions of the same courseware with an authoring tool (Moundridou & Virvou, 2001). This will involve more than one subject matter experts to help us explore how different instructors or experts may produce different courses by providing their own input to an authoring tool.

REFERENCES

Boyle, C., & Encarnacion, O. (1994). Metadoc: An adaptive hypertext reading system. User Modeling and User Adapted Interaction, 4, 1-19.

Brusilovsky, P. (1996). Methods and Techniques of Adaptive Hypermedia. User Modeling and User Adapted Interaction, 6(2/3), 87-129.

Brusilovsky, P., & Pesin, L. (1998). Adaptive navigation support in educational hypermedia: An evaluation of the ISIS-Tutor. Journal of Computing and Information Technology, 6(1), 27-38.

De Bra, P., Houben, G. J., & Wu, H. (1999). AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In Proceedings of the 10th ACM Conference on Hypertext and Hypermedia (pp. 147-156). New York: ACM Inc.

De Bra, P. (1996). Teaching Hypertext and Hypermedia through the Web. In Proceedings of WebNet'96, World Conference of the Web Society (pp. 130—135). Charlottesville: AACE.

de La Passadiere, B., & Durfesne, A. (1992). Adaptive Navigational Tools for Educational Hypermedia. In Tomek, I. (Ed.) Computer Assisted Learning (pp. 555-567). Vienna New York: Springer-Verlag.

Ebner, T., Magele, C., & Dietinger, T. (1999). Design and Implementation of Interactive, Web-Based Courses. In de Bra, P. & Leggett, J. (Eds.) Proceedings of WebNet 99 World Conference on the WWW and Internet (pp. 319-324). Charlottesville: AACE.

Eklund, J., & Brusilovsky, P. (1998). Individualising Interaction in Web-Based Instructional Systems in Higher Education. In Proceedings of the Apple University Consortium Academic Conference.
Available: http://auc.uow.edu.au/conf/conf98/papers/eklund.html. (March 10, 2001)

Hohl, H., Böcker, H., & Gunzenhäuser, R. (1996). Hypadapter: An Adaptive Hypertext System for Exploratory Learning and Programming. User Modeling and User Adapted Interaction, 6(2/3), 131-155.

Moundridou, M., & Virvou, M. (2001). Authoring and Delivering Adaptive Web-Based Textbooks Using WEAR. In Proceedings of ICALT-2001, IEEE International Conference on Advanced Learning Technologies, to appear.

Murray, T., Piemonte, J., Khan, S., Shen, T., & Condit, C. (2000). Evaluating the Need for Intelligence in an Adaptive Hypermedia System. In Gauthier, G., Frasson, C. and VanLehn, K. (Eds.) Intelligent Tutoring Systems, Proceedings of the 5th International Conference on Intelligent Tutoring Systems, Lecture Notes in Computer Science, 1839 (pp. 373-382). Vienna New York: Springer-Verlag.

Virvou, M., & Moundridou, M. (2000). A Web-Based Authoring Tool for Algebra-Related Intelligent Tutoring Systems. Educational Technology & Society, 3(2), 61-70.

Virvou, M., & Tsiriga, V. (2001a). Web Passive Voice Tutor: an Intelligent Computer Assisted Language Learning System over the WWW. In Proceedings of ICALT-2001, IEEE International Conference on Advanced Learning Technologies, to appear.

Virvou, M., & Tsiriga, V. (2001b). Adaptive Tutoring based on the Student Model of a Web-based ICALL. In Proceedings of TELEMATICA 2001 International Conference on Telematics and Web-Based Education, to appear.

Weber, G., & Specht, M. (1997). User Modeling and Adaptive Navigation Support in WWW-based Tutoring Systems. In Jameson, A., Paris, C. and Tasso, C. (Eds.) Proceedings of the Sixth International Conference on User Modeling (pp. 289-300). Vienna New York: Springer-Verlag.