Adaptive Help for Webbased Applications

Dorothea Iglezakis

Cath. University Eichstaett-Ingolstadt,
Department for Applied Computer Science,
Ostenstr. 14, 85072 Eichstaett, Germany,
dorothea.iglezakis@ku-eichstaett.de

Abstract. This paper presents an approach that uses the techniques of plan recognition not only to infer short-term plans and goals, but also to infer the long-term procedural knowledge of a user in a non-binary way. The information about the procedural knowledge in terms of activation builds the user model of AdaptHelp, an adaptive help system for web-based systems. AdaptHelp is based onto established adaptive help systems, which are shortly presented and compared on the used mechanism and techniques.

1 Introduction

Although the first contents of the World-Wide-Web consist mainly of static HTML-based information sites, the number of dynamically generated and increasingly complex applications on the Web grows. With increasing complexity of these applications, the need to develop user-friendly support functions grows. The aim of adaptive help systems is to offer the right amount of information in the right moment by adapting the contents of the help system either to the knowledge or to the actual plans and goals of the individual user.

2 Adaptive Help Systems

Existing adaptive help systems use mainly two approaches to adapt the help information to the individual information requirements of different users. They either infer the knowledge of the user by observing the interaction of the user with the help system or consider the actual situation of a user by means of plan recognition. The following section presents the user models, input data, and adaptation mechanisms of the adaptive help systems KNOME [3], ORIMUHS [5], and EPIAIM [4], which infer the knowledge of the user, as well as the systems PUSH [6] and PLUS [2], which use plan recognition to infer the actual plan or information goal of the user.

KNOME is the user modeling component of the Unix Consultant, a natural language help system that generates explanations for UNIX-functions. KNOME uses the questions of the user as input data to calculate the probability that the user belongs to one of four global stereotypes. With the user model and the difficulty of a concept, the system infers the probability that the user knows about a knowledge item. The adaptation happens through output of the version of a help item corresponding to the most probable stereotype.

ORIMUHS is an object-oriented multimedia-help component for graphical systems under a UNIX operating system. Like KNOME, ORIMUHS uses a global stereotype model. ORIMUHS infers the stereotype of the user by protocolling the activities into action graphs and comparing these individual action graphs with predefined action graphs for the stereotypes. The adaptation mechanism uses different versions for each help item to show the appropriate version of the help item.

EPIAIM is a knowledge-based system developed to support health care professionals in epidemiological data analysis. The user model consists in a set of predicates that specify the probability that a user knows about a concept or has practice in a subject area. EPIAIM builds the help contents according to production rules that specify which attributes of a concept are added dependent on the knowledge of the user. The selection of examples depends on the practice a user has in a certain area.

PUSH is a user sensitive hypermedia-help tool, which adapts to the information goal of the user. PUSH uses a short-term model for the current information goal of the user. The system infers this model through a combination of key hole plan recognition and intended plan recognition with the last actions of the user as input data. Each help item consists of different components that support the user in different ways. Dependent on the information goal, the content of a help item is adapted through showing or hiding specific components.

PLUS is a user support system that adapts to the actual plan of the user. The plan recognizer is based on a predefined hierarchically structured plan base. A plan consists of a set of actions with different parameters and an absolute or relative position. The plan recognition processes the low-level inputs (keystrokes, mouse-clicks) to identify the interaction style of the user for a simple user model and to build a dialog history. A spreading activation algorithm then builds hypotheses about the plan of the user. A plan completion component uses this information to generate action sequences for activated but not recognized plan hypotheses in a tutor help.

3 AdaptHelp—A Concept for an Adaptive Help Component for Web-Based Systems

AdaptHelp tries to combine the different approaches presented in the last section by using methods of plan recognition to infer the procedural knowledge of the user, while taking into account the specialties of the web environment.

In contrast to other adaptive web systems like learning environments or recommender systems, adaptive help systems cannot use additional input data like tests or questionnaires. The only possible input data that can be collected non-intrusively is therefore data that logs the activities of the user on the server.

3.1 Knowledge Modeling

All presented systems, which model the knowledge of a user, base on a binary concept of knowledge representation. A user either knows a concept or does not. From cognitive psychology we know that knowledge is not binary. The ACT-theory from Anderson and Lebière [1] delivers an empirically founded theory about learning and forgetting

of memory contents. Each access of a memory item increases the activation of the corresponding memory trace. Without access, the activation fades over time. By logging the activities of the user, it is possible to measure the frequency of occupation with the concepts and tasks within the target application.

Knowledge items differ in the type of knowledge. Procedural knowledge covers knowledge about actions and activities, whereas declarative knowledge covers knowledge about facts. Especially for help systems, which have the goal of explaining and supporting the execution of tasks, the procedural knowledge of the user is central.

Therefore, a knowledge item in the context of AdaptHelp is the procedural knowledge about a single task of the target application. For example, if the target application is the configuration menu of an internet provider, one knowledge item could be the knowledge about adding a new e-mail address. Declarative knowledge items can function as prerequisites or as additional information to the procedural knowledge items.

3.2 User Model

Like the user models of the knowledge-modeling approaches presented in Sect. 2, the user model of AdaptHelp contains information about the individual knowledge of items in the target domain. In contrast to the binary concept of knowledge in the presented approaches, the approach of AdaptHelp models the knowledge in terms of current activation of an item.

The systems KNOME, and ORIMUHS infer the knowledge of a user with a general stereotype model. This kind of knowledge modeling works only in domains, where the concepts can be ordered by difficulty. Webbased applications offer mainly a set of services that are independent from each other. Therefore, a global stereotype model makes no sense for theses applications.

The user model of AdaptHelp is an overlay model that assigns an activation value to each knowledge item. The activation of an item is computed by the access time stamps of the corresponding tasks.

3.3 Activation of a Knowledge Item

The systems PLUS, PUSH, and ORIMUHS observe the activities of the user to recognize the plans, the information goal, or the stereotype of the user. This activation data also contains additional data, because it documents the experiences of the user and therefore allows to draw conclusions on the activation of the users' procedural knowledge.

Therefore, AdaptHelp uses extended logfiles of the WebServer as a data source to recognize the executed tasks of the user. The logfiles are in an XML-format and contain the name of the requested site with the used parameters and an identification of the requesting user for each request.

The actions in the logfiles are compared against predefined tasks. Each task consists of a sequence of actions. In the web context, each action corresponds to a web site with its parameters. A parameter represents additional data, which is sent to the site per GET, POST or SESSION variables. An action in a predefined task can set presence- and

absence constraints over the parameters. The tasks are handled as regular expressions over the logged actions resulting in access timestamps for each task. On the basis of these timestamps, the base activation B_i of the knowledge item i is computed using the Base-Level Learning Equation from the ACT-theory ([1])

$$B_i = ln(\sum_{j=1}^{n} t_j^{-d}), \tag{1}$$

where t_j is the time since the jth practice of the corresponding task and $d \in]0;1[$ is a domain dependent parameter, mainly set to 0.5.

3.4 Adaptation

As Höök [6] states, it is very difficult to write and to maintain different versions of the same help items for systems such as KNOME or ORIMUHS. The online assembling of the help items from standardized components like EPIAIM or PUSH is therefore a more promising approach.

The help items of AdaptHelp are written in an XML format that defines the structure of the items. Each item corresponds to one task in the target application and consists of different components. The adaptation happens through online assembling of the components to one help item, dependent of the actual activation of the target process.

3.5 Implementation and Further Work

At this moment, we implemented the process recognition part of AdaptHelp in C++. First performance tests show promising results. Tools to facilitate the writing process of the help items are in the planning phase. The implementation of the adaptation procedure will follow after experimental evaluations that test the helpfulness of different components of the help items for users with different levels of experience in the corresponding tasks.

References

1. John R. Anderson and Christian Lebiere. *The atomic Components of Thought*. Lawrence Erlbaum Associates, Mahwah, NJ, 1998.
2. Frank Berger, Thomas Fehrle, Kristof Klöckner, Volker Schölles, Markus A. Theis, and Wolfgang Wahlster. Plus, plan-based user support. Technical Report RR-93-15, DFKI, March 1993.
3. David N. Chin. *User Models in Dialog Systems*, chapter KNOME: Modeling What the User Knows in UC, pages 74–107. Springer, Berlin, 1989.
4. Fiorella de Rosis, Bernardina de Carolis, and Sebastiano Pizzutilo. User-tailored hypermedia explanations. In *Adaptive hypertext and hypermedia—Workshop held in conjunction with UM'94*, 1994.
5. Miguel Encarnação. Multi-level user support through adaptive hypermedia: A highly application-independent help component. In *Proceedings of the Conference on Intelligent User Interfaces IUI 97*, pages 187 – 194, Orlando Florida USA, 1997. ACM.
6. Kristina Höök. *A Glass Box Approach to Adaptive Hypermedia*. PhD thesis, Stockholm University, Swedish Institute of Computer Science, 1996.