The ActiveMath Learning Environment System Description

Paul Libbrecht, Erica Melis, Carsten Ullrich
paul,melis,ullrich@ags.uni-sb.de

Saarland University, Germany

Abstract. ActiveMath is a web-based learning environment that dynamically generates interactive mathematical courses adapted to the student's goals, preferences, capabilities, and knowledge. It integrates several mathematical service systems. The course content is represented in `OMDoc`, an extension of `OpenMath`. For each user, the appropriate content is retrieved from a knowledge base and the actual course is generated individually according to pedagogical rules. The course is presented to the user via a standard web-browser. During a course the learner can interactively practice problem solving by using mathematical services such as computer algebra system or a proof planner. The article provides a brief account of the current state of ActiveMath.

1 Introduction

Web-based learning systems are becoming more predominant and increasingly incorporate intelligent features. In Saarbrücken we are developing the *web-based, user-adaptive, interactive learning environment* ActiveMath.[1] In a nutshell, its major features are user-adapted content, sequencing, and presentation, user-adapted suggestions for learning, support of active and explorative learning by mathematical services, use of proof planning, support of teachers by information about their students, and the reusability of the encoded content via the semantics of the knowledge representation.

We believe that the use of such a system in the maths curriculum is advantageous as it supports active learning of the student. During the last decades, the mathematics pedagogy community recognized that students learn mathematics more effectively, if the traditional rote learning of formulas and procedures is supplemented with the possibility to explore a broad range of problems and problem situations [11]. In particular, the international comparative study of mathematics teaching, TIMSS [1], has shown (1) that teaching with an orientation towards active problem solving yields better learning results in the sense that the acquired knowledge is more readily available and applicable especially in new contexts and (2) that a reflection about the problem solving activities and methods yields a deeper understanding and better performance.

[1] A demonstration of ActiveMath is available at `http://www.activemath.org`.

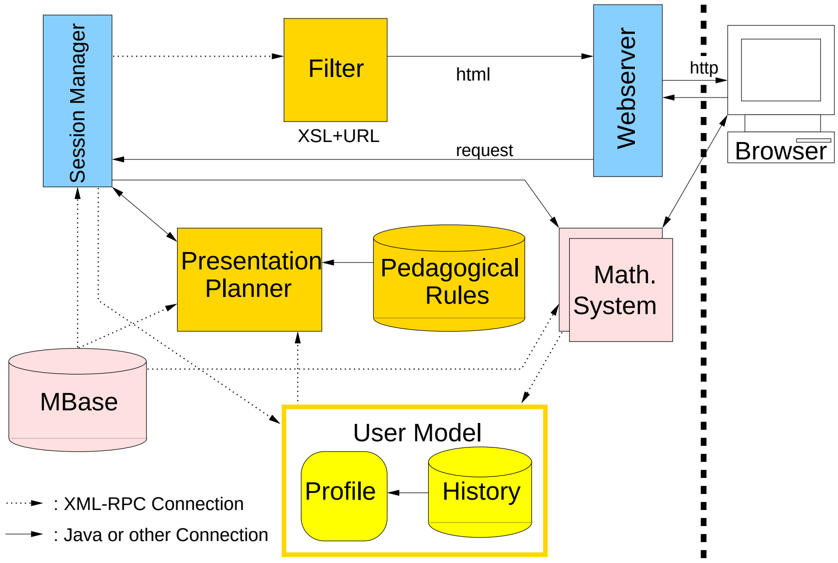

Fig. 1. Architecture of ActiveMath

We also believe that teaching mathematical methods and know-how and know-when has to be introduced into mathematics teaching, apart from the traditional axioms, theorems, and procedure teaching. The knowledge-acquisition work done for proof planning provides a first material that has to be further expanded. First experiments [9] suggest that instruction materials based on descriptions of mathematics methods yield a better subsequent problem solving performance than traditional (textbook like) instruction material.

This article provides a brief account of the current state of ActiveMath. In particular, we focus on the architecture, the knowledge representation, the presentation planning, and the user model. For a more detailled description see [8].

2 Architecture

Figure 1 depicts the architecture of ActiveMath, i.e., its components and the communications between them (indicated by arrows). It shows the client-server web-architecture with a browser at the client side. Currently, ActiveMath integrates the following components: a session manager, the knowledge base MBase [4], a presentation planner, a user model, a pedagogical module, and mathematical services such as the proof planner of Ωmega [10] and the Computer Algebra System (CAS) Maple (for a detailled description of the integration of mathematical services see [7]). The components can communicate over the Internet via a standardized xml-rpc protocol. For instance, requests of the user and (in the other direction) HTML-pages are communicated via a web-server to the session manager. The session manager stores the generated courses and translates URL requests into actions (e.g., the request for a new course about the topic *group*) that are passed to the responsible component. The presentation planner generates the learning documents adapted to the user's goals, preferences, and knowledge by requesting and processing information from MBase, from the user model, and from the pedagogical module. Information about the user's actions, such as the history of her actions and the time intervals of her reading a concept or the success of solved problems, is passed from the session manager to the user model (as well as from the proof planner and CAS to the user model), where it is used for updating.

3 Knowledge Representation

Our knowledge representation OMDoc [6], is an extension of the OpenMath [3] standard. OMDoc encodes the semantic of mathematical objects as well as meta-data and mathematical facts such as theorems, definitions, proof methods, and proofs and includes natural language formulations as well as formal (OpenMath) objects. OMDoc uses a semantic XML-based representation of mathematical knowledge that provides an *ontology* for the content of the course which is indispensable for a reuse of teaching and learning material and for a combination of material from different sources.

In addition to the actual conceptual content, our knowledge representation contains meta-data for structures, dependencies, and pedagogical information which can be used for the dynamic generation of interactive documents.

4 Presentation Planning

The central component of ActiveMath is the presentation planner. It generates a personalized course in a three-stage process:
(1) First, the content is retrieved from the knowledge base MBase. Starting from the goal concepts chosen by the user, all concepts they depend upon and corresponding additional information (e.g., elaborations, examples *for* a concept) are collected recursively. This grabbing process uses the dependencies metadata information contained in the OMDoc representation. The result of this retrieval is a collection of all concepts plus additional information about them that need to be known by the learner in order to understand the goal concepts.
(2) Then pedagogical knowledge is applied. According to the information in the user model and the pedagogical module the collection of content items is processed and transformed into a personalized instructional graph of learning materials. This process is detailed below.
(3) Finally, the instructional graph is linearized.

The result of the presentation planning is a linearized instructional graph whose nodes are OMDoc items. Filters (see Figure 1) will transform this collection into HTML pages using XSL-transformations.

The goal of the application of pedagogical knowledge is to select from and transform the collection of items that was gathered in the first stage of presentation planning into a selection of learning material. ACTIVEMATH employs pedagogical information represented in pedagogical rules. It evaluates the rules with the expert system shell JESS [5]. The rules consist of a *condition* and an *action* part. The condition part of a rule specifies the conditions that have to be fulfilled for the rule to be applied, the action part specifies the actions to be taken when the rule is applied.

The presentation planner employs the pedagogical rules to decide:

(1) which information should be presented on a page;

(2) in which order this information should appear on a single page;

(3) how many exercises and examples should be presented and how difficult they should be;

(4) whether or not to include exercises and examples that make use of a particular service system. Since exercises and examples employing service systems require a certain minimal familiarity with the systems, ACTIVEMATH presents those exercises only if the capability is confirmed.

(5) whether to restrict the available functionalities of a service system. For instance, a student learning about mathematical integration and derivation should not use a CAS to solve his exercises completely, whereas using the CAS as a calculator for auxiliary calculation is acceptable.

5 User Modeling

The user model consists of two subcomponents: the history that stores the data about user's actions (e.g., the reading of a concept at a certain time), and the profile that stores the user's preferences, goals and knowledge mastery data. When a learner registers, she can assert her knowledge mastery values which will be constanly updated according to her actions. The user model is inspectable and modifyable.

The knowledge mastery assessment is represented by values for a subset of the competence features in Bloom's taxonomy [2], namely Knowledge, Comprehension, and Application.

Depending on the type of the user's interaction, different updates of the values are realized. After reading a concept, mainly the Knowledge-value of it is updated. After reading an example for a concept the Comprehension-value of that concept is mainly updated. After acting on an exercise, the success or failure rate mainly updates the Application-value of that concept in a way depending on the abstractness and difficulty levels of the exercise. A dependeny between those values will be introduced later.

The user model also stores a justification for each value, that is, whether it is a direct user input for the user model, indirectly inferred from the history data or other information, or an output of an exercise evaluator.

6 Conclusion and Further Work

This paper gives a short overview on the web-based learning environment ActiveMath that generates mathematical courses adapted to the learner and integrates several mathematical service systems.

We plan to integrate other mathematical systems, such as the CAS MuPad. An interesting question regarding mathematical systems is how to interpret the learners actions and how to calculate the appropriate feedback to the user model, preferable in an abstract way in order to avoid having to build a proper evaluator for each mathematical system.

Other questions concern metadata and presentation planning. We are re-designing the current prototypical version of the proof planner and have to decide what kind of metadata is needed to generate useful learning material.

Last, but not least evaluation is a important issue. We are conducting studies on the design of the user interface and prepare field studies in schools and at university to test whether our system does what it is supposed to do, support learning.

References

1. J. Baumert, R. Lehmann, M. Lehrke, B. Schmitz, M. Clausen, I. Hosenfeld, O. Köller, and J. Neubrand. *Mathematisch-naturwissenschaftlicher Unterricht im internationalen Vergleich*. Leske und Budrich, 1997.
2. B. Bloom, editor. *Taxonomy of educational objectives: The classification of educational goals: Handbook I.* Longmans, Green, New York, Toronto, 1956.
3. O. Caprotti and A. M. Cohen. Draft of the open math standard. Open Math Consortium, `http://www.nag.co.uk/projects/OpenMath/omstd/`, 1998.
4. A. Franke and M. Kohlhase. MBase: Representing mathematical knowledge in a relational data base. In *Proc. 17th International Conference on Automated Deduction*, Lecture Notes on Artificial Intelligence. Springer-Verlag, 2000.
5. E. Friedman-Hill. *Jess, the Java Expert System Shell*, 2000. `http:/herzberg.ca.sandia.gov/jess/`.
6. M. Kohlhase. `OMDoc`: Towards an internet standard of mathematical knowledge. In E. R. Lozano, editor, *Proc. of Artificial Intelligence and Symbolic Computation, AISC'2000*, LNAI. Springer Verlag, 2001.
7. P. Libbrecht, E. Melis, M. Pollet, A. Frischauf, C. Ullrich Integrating mathematical systems with the ActiveMath learning environment. In *Proc. of IAMC Workshop 2001 at ISSAC'2001.* submitted.
8. E. Melis, E. Andrès, A. Franke, A. Frischauf, G. Goguadze, P. Libbrecht, M. Pollet, and C. Ullrich. ActiveMath: A web-based learning environment. *Artificial Intelligence in Education*, 2001. submitted.
9. E. Melis, C. Glasmacher, C. Ullrich, and P. Gerjets Automated Proof Planning for Instructional Design In *Proc. of 23rd Conference of the Cognitive Science Society.*, 2001 submitted.
10. E. Melis and J. Siekmann. Knowledge-based proof planning. *Artificial Intelligence*, 115(1):65–105, 1999.
11. A. Schoenfeld, editor. *A Source Book for College Mathematics Teaching*. Mathematical Association of America, Washington, DC, 1990.