Adaptive Course Generation and Presentation

The ΩMEGA group:
Jörg Siekmann, Christoph Benzmüller, Lassaad Cheikhrouhou,
Armin Fiedler, Andreas Franke, Helmut Horacek
Michael Kohlhase, Andreas Meier, Erica Melis, Martin Pollet,
Volker Sorge, Carsten Ullrich, Jürgen Zimmer
Universität des Saarlandes/DFKI Saarbrücken
Germany

Abstract

Today's interactive mathematics textbooks use a collection of predefined documents, typically a collection of `html` pages. This makes a reuse and re-combination of the encoded knowledge impossible and inhibits a radical adaption of course presentation and content to the user's needs. In order to avoid these drawbacks we have designed a web-based framework for dynamically producing interactive documents (IDs) for learning mathematics. The system design is characterized by the application of a well-known AI-paradigm, the separation of knowledge representation from system functionalities. Remarkable features of our system that realizes this general approach are the individual generation of interactive documents based on user-specific preferences and the user's knowledge as well as on general domain knowledge. The paper describes the distributed web-based architecture of our system and the principles of the session manager.

1 Introduction

Today's interactive mathematics textbooks [5, 6] use a collection of predefined documents (LaTeXdocuments or HTML pages) and including a fixed set of examples and exercises. Neither can the content(e.g., examples and exercises) be tailored to the needs of a particular user nor can the presentation of these documents be adapted flexibly enough.

In order to avoid these drawbacks we have designed a web-based framework for dynamically producing interactive documents (for learning mathematics). The system design is characterized by the application of a well known AI-paradigm, the separation of knowledge representation from system functionalities. For the interactive mathematics document ID [17] the knowledge comprises mathematical facts and methods as well as knowledge about the expertise of a particular user and pedagogical knowledge.

From these knowledge sources the system can compose individuated versions of interactive documents dynamically. In particular, the generation of an interactive textbook or course may include different parts and different presentations that differ, for example, in language, in notational preferences, in the multi-modal display, in the availability of expert services for solving subproblems, or in their depth of explanation.

As opposed to other interactive mathematics textbooks which use a collection of HTML-pages [5] or LaTeX-units [6], our approach requires a representation of standard formalized mathematics in a (possibly distributed) knowledge base. The standardization requires some extra work but provides an ontology for the content of the course which is indispensable for a reuse of teaching or learning material and for a combination of material from different courses. Moreover, the presentation can be far more flexible when based on a unified representation in XML-format.

Another feature of our system is its focus on problem solving rather than on mere content teaching or learning. As opposed to looking up solutions, multiple choice for solutions, or problem

solving without using the computer our approach supports exploratory problem solving. That is, the user can solve problems or discover proofs interactively within the system. The system supports the user by checking the correctness of proof steps and of the overall solution as well as by heuristically guiding the problem solving process. A more detailed discussion of exploratory problem solving supported by expert systems is given in [14].

In this paper we concentrate on the description of the architecture of ID, the representation format, and of the session manager that dynamically assembles interactive mathematics documents, e.g. textbooks and courses, by using information retrieved from a knowledge base, a user model, and a pedagogical module.

The paper is organized as follows: first we give an overview on the architecture of our system. Then, we detail the knowledge represenation format of the underlying mathematical knowledge base. In section 2.4 we explain the functionality of the central component, called session manager, and describe how user-adapted interactive documents are assembled. Finally, we give some examples to illustrate the benefits of our approach.

2 Dynamic Document Generation

Current interactive textbooks and courses are based on a fix collection of predefined HTML or LaTeXpages which are augmented by navigation features and only few allow for a generation of documents from a particular subset of pages. But these few exceptions still use the ready-to-present pages rather than generating them according to the user's content needs and presentation preferences. In this section we describe the principles of our user-adaptive generation of interactive documents based on user-specific preferences and the user's knowledge as well as on general domain knowledge.

2.1 The Open Architecture

The distributed setup of the system employs the MathWeb system [9, 10] developed in our group[1] for distributed automated theorem proving that connects a wide-range of *mathematical services* by a common *mathematical software bus*. The MathWeb system provides the functionality to turn existing theorem proving systems and tools into mathematical services that are homogeneously integrated into a networked proof development environment. The environment thus gains the services from these particular modules, but each module in turn gains from using the features of other plugged-in components. Currently, MathWeb integrates mathematical several services, a proof planner [15], several computer algebra systems, and mediators [1], i.e., mathematical services that transform mathematical knowledge from one format to another, the $\mathcal{LOUI}$ graphical user interface for interactive theorem provers [18], the *Prex* proof presentation system [7], which can transform logic proofs to natural language, and MBase [11, 13], a knowledge base that stores and offers a universal repository of formalized mathematics consisting of theorems, definitions, proof methods, and proofs. Its formal representation allows a semantics-based retrieval of (possibly distributed) mathematical knowledge. For realizing an interactive mathematical course, we add a user model and a session manager to the MathWeb environment.

Figure 1 depicts the architecture of our ID system. It gives an overview of the services used in the context of our framework for web-based user-adaptive interactive textbooks and courses and the interactions of these services. This architecture is centered around the session manager which communicates with the user via a browser and a web-server.

The session manager dynamically creates the actual OMDoc-documents to be presented to the user by the browser. For the document creation, the session manager processes three kinds of information: content information, user information, and pedagogical information. *Content information* comprises the mathematical information to be presented to the user, *user information* consists of the relevant parts of the user model and *session specific information*, whereas *pedagogical information* covers the know-how of how to present content in a way adapted to the user's

[1] The system is available from `http://www.mathweb.org/mathweb`.

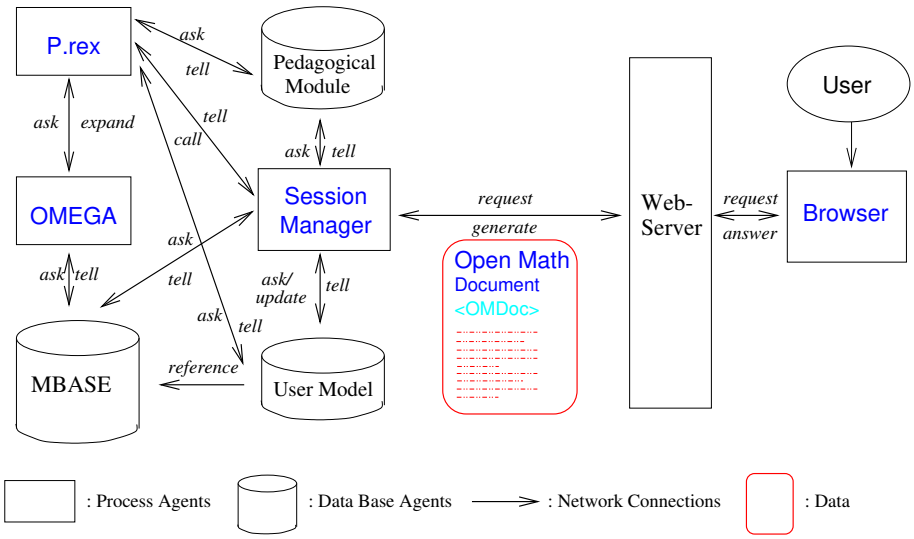


Figure 1: Architecture of ID

capabilities. Some interactive information, such as learned concepts or solved problems, is passed from the browser to the user modeling component where it is used for updating.

In order to answer a request, the session manager taps content information sources and transforms the raw data provided by MBase into a document in OMDoc format. The session manager computes XSL- and CSS-style-sheets [19, 2] according to the intended functionality and the user model. Then it sends the OMDoc-document and style sheets to the browser who displays the content using the style sheets.

Via the browser and session manager or via ΩMEGA and the session manager, some interaction information, such as learned concepts or successfully solved problems, is passed to the user modeling component where it is used for updating.

In the following, we describe the underlying knowledge representation by OMDoc documents and then the session manager and its interaction with the other services in order to dynamically create OMDoc documents.

2.2 Knowledge Representation

All MathWeb services can communicate over the Internet by a standardized protocol based on Xml [3] and Kqml [8]. Our desrciption here will concentrate on the Xml-data representation OMDoc [12] in the communication.

OMDoc is an extension of the emerging OpenMath standard [4] (see `http://www.openmath.org`), an Xml-based general framework for encoding mathematical objects. OMDoc extends the OpenMath representation for mathematical objects by representations of mathematical facts like theorems, definitions, proof methods, and proofs. It includes natural language formulations as well as logical forms (as OpenMath) objects. Figure 2 shows an OpenMath representation of the law of commutativity for addition in the reals (the logical formula $\forall a, b.a + b = b + a$). The Xml element `OMBIND` is used for the OpenMath binding construct (in this case used for universal quantification) and the `OMS`, `OMV` and `OMA` are the representations for logical constants,[2] variables and (function) applications. As an example, Figure 3 shows a definition of a monoid. The `CMP` (commented mathematical property) element is used for the natural language representation of the definition (including OpenMath representations for the objects involved) and the `FMP` (formal

[2] OpenMath symbols, including the `cd` reference to the home theory of the mathematical concept represented by this symbol

mathematical property) contains an OpenMath object (... in Figure 3) that encodes a logical formula similar to that in Figure 2 that formalizes the content of the CMP (we have not included it for space reasons). For details on OMDoc, see [12].

```
<OMOBJ>
  <OMBIND>
    <OMS cd="quant1" name="forall"/>
    <OMBVAR>
      <OMV name="a"/>
      <OMV name="b"/>
    </OMBVAR>
    <OMA><OMS cd="logic1" name="implies"/>
      <OMA><OMS cd="logic1" name="and"/>
        <OMA><OMS cd="set1" name="in"/><OMV name="a"/><OMS cd="barshe" name="real"/></OMA>
        <OMA><OMS cd="set1" name="in"/><OMV name="b"/><OMS cd="barshe" name="real"/></OMA>
      </OMA>
      <OMA><OMS cd="relation" name="eq"/>
        <OMA><OMS cd="barshe" name="plus-real"/><OMV name="a"/><OMV name="b"/></OMA>
        <OMA><OMS cd="barshe" name="plus-real"/><OMV name="b"/><OMV name="a"/></OMA>
      </OMA>
    </OMA>
  </OMBIND>
</OMOBJ>
```

Figure 2: An OpenMath representation of $\forall a, b.a + b = b + a$.

```
<definition id="ida.c6s1p4.d1"
            item="monoid" theory="ida.monoid">
  <CMP>
    A structure
    <OMOBJ><OMA>
      <OMS cd="cartesian-products" name="triple"/>
      <OMV name="M"/><OMV name="*"/><OMV name="e"/>
    </OMA></OMOBJ>,
    in which
    <OMOBJ><OMA>
     <OMS cd="cartesian-products" name="pair"/>
     <OMV name="M"/>
     <OMV name="*"/>
    </OMA></OMOBJ>
   is a semi-group with unit <OMOBJ><OMV name="e"/></OMOBJ>, is called a
   <defemph>monoid</defemph>.
  </CMP>
  <FMP><OMOBJ> ... </OMOBJ></FMP>
</definition>
```

Figure 3: A definition of a monoid in OMDoc representation

In the following, the different sources of information are elaborated, the process of the document creation is outlined, and some adaptive functionalities of ID are presented.

2.3 Communicated Information

The realization of these functionalities requires different kind of information from different knowledge sources.

Content Information The content information (in ID: mathematical concepts, theorems, examples, exercises, proof methods, and proofs) to be displayed is received from several information sources, namely (1) from the database MBase containing the knowledge items and relations specifying their structure, and (2) from service systems, such as the proof planner ΩMEGA or the computer algebra system Maple [16].

User Information The user information comprises three aspects of knowledge about the user: her domain knowledge, her learning capabilities, and the knowledge about her current session.

The domain knowledge of the user is maintained by the user modeling component which keeps track of the user's degree of understanding the mathematical concepts and acquiring skills targeted by the pedagogical module.

Some knowledge in the user model is rather permanent or at least slowly changing, in particular knowledge about the user's general capabilities, e.g., how abstract a presentation is understood and how much explanation is necessary and general preferences, e.g., whether she prefers natural language over formulas, which modalities to choose in multi-modal presentation.

Finally, the session-specific information (e.g., what is the current/previous/next page) is needed for the on-line generation of ID . From this information, conclusions for the navigation support can be inferred (e.g. indicating the mathematical concepts accessible by the user with her current domain knowledge).

Pedagogical Information Pedagogical information contains the pedagogical know-how about the goals of presenting certain content, e.g., examples and exercises. It also determines under which conditions previously visited lessons should be (partially) repeated or remembered, when to ask questions (e.g., to stimulate the student's self-explanation), and under which conditions which service systems should be available for the user. The pedagogical information may also contain some presentation knowledge.

2.4 The Session Manager

The interaction between the system and the user takes place in form of a session. A session is a completed sequence of actions starting with the identification of the user, followed by a sequence of (mixed-initiative) interactions, and a session closing.

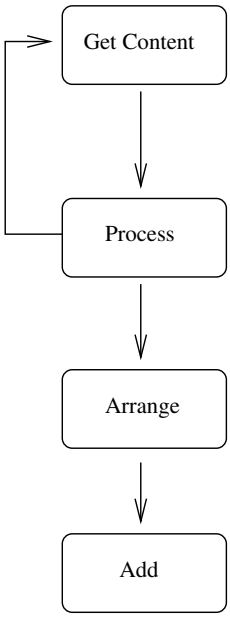
The session manager realizes the composition of the document and display information for a user-adaptive document presentation during a session.

In order to answer a request, the session manager asks for content information and transforms the raw data into a document in OMDoc format that is then adapted according to the intended functionality and the user model. The OMDoc-document is then sent to the browser and displayed using XSL- and CSS-style-sheets. e.g.

2.4.1 Document Creation

ist das schon immer für die ganze session?? was macht process?? wann werden die style sheets erzeugt? wann und wie die buttons selected - durch style sheets? suitable level of abstraction is chosen - ?? generation of omdocs and of style sheets

The session manager dynamically generates a document in five steps.



1. Dependent on the user and pedagogical information, mathematical content is retrieved from information sources like MBase or ΩMEGA. For example, all the MBase items depending on a given concept that are not marked as known in the user model are collected.
2. The collection of content items is then processed according to the information in the user model and the pedagogical module. For example, the suitable level of abstraction is chosen. If more content (such as exercises, remarks and examples) is needed, the first step can be repeated. In this step interaction facilities are chosen too, e.g., the availablity of a service system for solving a task. in omdoc oder in SS?
3. Then this structure is linearized.The user model and the pedagogical module influence the linearization, as the amount to be presented on one page will vary according to the users preferences and capabilities.

After this step, the course is represented as a collection of OMDoc-documents.

4. Finally further functionality is added to the OMDoc-pages, for example, navigation buttons. This step mainly uses information about the session.

5. Style sheets that determine the presentation features are generated from ??

As the result of the document creation one or more documents in OMDoc-format are generated as well as style sheets. The documents can be presented to the user by the browser using the style sheets.

2.5 Adaptive Functionalities

Some possible dimensions of the adaptation performed by the session manager are

- the choice of the content to be presented (e.g., guided tours explained below and adaptive testing)
- the availability of service systems (e.g., a calculator, a computer algebra system, or a proof planner). For example, for routine tasks which are not in the focus of a session a service can be used.
- the additional selection of service functionalities (e.g., which methods can be chosen by the user when using a service system).

They are discussed again in section 2.5. The adaptations depend on the features of the user and of the overall session, for instance on

- the goal of a session,
- the setting (e.g., stand-alone use or classroom setting where the teacher can set the session goal),
- the user's capabilities (e.g., how abstract a presentation must be in order to be understood or how much explanation is necessary),
- the user's understanding of concepts,
- the user's preferences (e.g., language, natural language vs. formulas, modalities in multimodal presentation),
- the user's need for motivation.

The following examples illustrate what such adaptations may look like. The first example of how different document contents and presentations can be generated from the the same mathematical content is given in Figures 4 and 5. If the concept *monoid* is presented for an expert, then the presentation contains only the desired concept as shown in the screenshot of Figure 4. If it is presented for a student, then the document whose presentation is shown in Figure 5 contains additional exercises, remarks, and examples.

Guided Tour A typical request for a guided tour might be the following:

> I am Jane Doe, and I would like to know about RSA encryption.

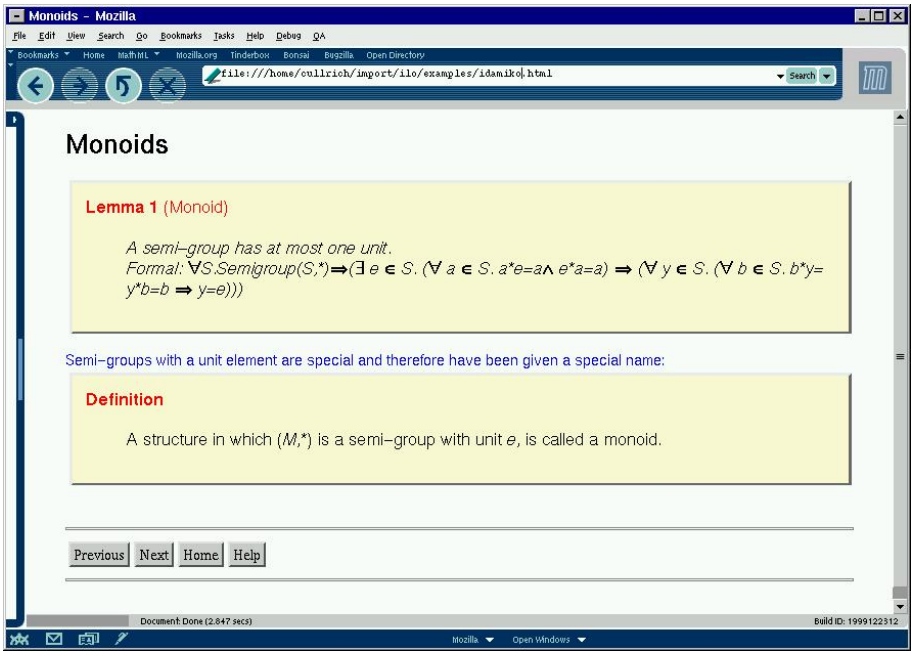


Figure 4: A presentation of Monoids

First, the MBase items the concept *RSA encryption* depends upon are collected: the structure of the knowledge in MBase reveals that RSA encryption presupposes Fermat's so-called "little theorem", so this is added to the collection. MBase contains two proofs of the theorem. One is elementary and uses divisibility and binomial coefficients. The other uses the fact that congruence modulo n is an equivalence relation. To decide which proof to present to the user, the user model of the user "Jane Doe" is queried for her preferences and prior knowledge. In this case it might reveal that arithmetic modulo n is well-known, so the second proof is added to the document.

Then the collection of the retrieved items is linearized. As the user is an mathematical expert his preference for a short proof is inferred. The possibility to call a proof planner is added because Jane is a sceptic user who prefers checking proofs by herself. Finally, navigation buttons are included to complete the document creation. wie?

Guided Proof Assume a student wants to learn the concept *group*. She has already learned the definition of a group and now has to become familiar with the use of this definition. Her user model indicates the general capability of learning by doing.

The session manager retrieves a number of examples she has to check for the group criteria. The choice and number is dependent of her previous performance. Say, the session manager retrieves the two examples
$(\mathbb{Q}, +)$, the rational numbers with the operation $+$, is a group
$(\mathbb{Z}, +)$, the set of integers with the operation $+$, is also a group.

Since the user has enough expertise for dealing with the proof planning service ΩMEGA the session manager selects the ΩMEGA button and includes a help button as well. So the user can interactively prove the two conjectures. Clicking such a button in the generated HTML pages will start a servlet that calls the proof planner with parameters, among them the conjecture to be proved. The parameters also include the methods of the proof planner that will be available in the method menu for the interactive proof process for the particular proof task. For instance, the second conjecture could be proved quite easily by using the subgroup criteria and showing that $(\mathbb{Z}, +)$ is a subgroup of $(\mathbb{Q}, +)$. But as the goal of the exercises is to become acquainted with the use of the group definition, ΩMEGA is called by the session manager with a subset of all proof

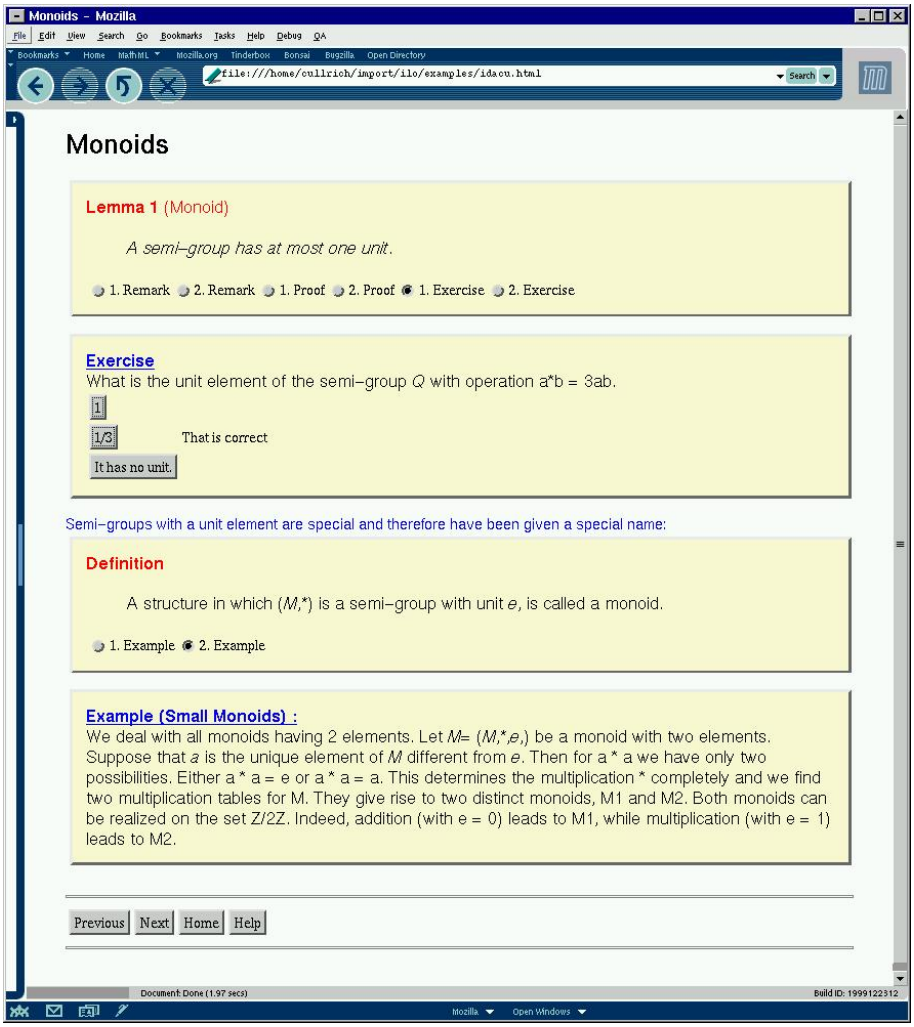


Figure 5: An extended presentation of Monoids

planning operators only, thus forcing the student to use the definition.

During the proof session ΩMEGA sends information to the session manager, about impasses in ther problem solving or on failing applicability of a method the user tried to choose. The session manager passes this and other information to the user model.

3 Conclusion

As opposed to previous tutor systems that use ready-to-present HTML or LaTeXpages rather than generating them according to the user's content needs and presentation preferences, ID individually generate interactive documents based on user-specific preferences and the user's knowledge as well as on general domain knowledge.

The paper describes the design principles of the system ID that dynamically generates interactive documents. The architecture, the representation of knowledge, and the approach of dynamic generation are well-suited not only for an interactive mathematics textbook but more generally for knowledge-intensive learning and tutor systems whose knowledge acquisition and representation is tedious and should therefore be reused, for systems including (expert) services for exploratory learning, and for systems flexibly generating individual versions of documents to be browsed.

Currently, we are implementing the described architecture and are semi-automatically recoding the content of an existing interactive textbook in OMDOC format. Until summer 2000 a first version of the session manager and other components will be implemented.

References

[1] A.Meier.

[2] B. Bos, H. Lie, C. Lilley, and I. Jacobs. Cascading Style Sheets, level 2 CSS2 Specification. W3C Recommendation. Technical report, May 12, 1998.

[3] Extensible Markup Language (XML). W3C Recommendation TR-XML, World Wide Web Consortium, December 1997. Available at `http://www.w3.org/TR/PR-xml.html`.

[4] Draft of the Open Math standard. The Open Math Society, `http://www.nag.co.uk/projects/OpenMath/omstd/`, 1998.

[5] A. Cohen, H. Cuypers, and H. Sterk. *Algebra Interactive!* Springer-Verlag, 1999.

[6] B.I. Dahn and H. Wolters. *Analysis Individuell.* Springer, Berlin, Heidelberg, 2000.

[7] Armin Fiedler. Using a cognitive architecture to plan dialogs for the adaptive explanation of proofs. In Thomas Dean, editor, *Proceedings of the 16th International Joint Conference on A rtificial Intelligence (IJCAI)*, pages 358–363, Stockholm, Sweden, 1999. Morgan Kaufmann.

[8] T. Finin and R. Fritzson. KQML — a language and protocol for knowledge and information exchange. In *Proceedings of the 13th Intl. Distributed Artificial Intelligence Workshop*, pages 127–136, Seattle, WA, USA, 1994.

[9] A. Franke and M. Kohlhase. System description: MATHWEB, an agent-based communication layer for distributed automated theorem proving. In H. Ganzinger, editor, *16th International Conference on Automated Deduction (CADE-99)*, volume 397, pages 217–221. Springer, 1999.

[10] Andreas Franke, Stephan M. Hess, Christoph G. Jung, Michael Kohlhase, and Volker Sorge. Agent-oriented integration of distributed mathematical services. *Journal of Universal Computer Science*, 5:156–187, 1999.

[11] Andreas Franke and Michael Kohlhase. System description: MBASE, an open mathematical knowledge base. Submitted to the Conference on Automated Deduction, CADE17, 2000.

[12] Michael Kohlhase. OMDoc: Towards an OPENMATH representation of mathe matical documents. Seki Report SR-00-02, Fachbereich Informatik, Universität des Saarlandes, 2000. http://www.mathweb.org/ilo/omdoc.

[13] Michael Kohlhase and Andreas Franke. Mbase: Representing knowledge and context for the integration of mathematical software systems. *Journal of Symbolic Computation, special Issue on the integration of computer algebra and deduction systems*, 11:1–37, 2000.

[14] E. Melis and A. Fiedler. On the benefit of expert services in mathematics education systems. Submitted to the ITS-2000 Workshop on Modeling Human Teaching Tactics and Strategies, 2000.

[15] E. Melis and J.H. Siekmann. Knowledge-based proof planning. *Artificial Intelligence*, 115(1):65–105, November 1999.

[16] Darren Redfern. *The Maple Handbook: Maple V Release 5*. Springer Verlag, Berlin, Germany, 1998.

[17] J. Siekmann. Interaktives Lehrbuch. Project Proposal, July 1999.

[18] Jöorg Siekmann, Stephan M. Hess, Christoph Benzmüller, Lassaad Cheikhrouhou, Armin Fiedler, Helmut Horacek, Michael Kohlhase, Karsten Konrad, Andreas Meier, Erica Melis, Martin Pollet, and Volker Sorge. $\mathcal{LOUI}$: $\mathcal{L}$ovely ΩMEGA $\mathcal{U}$ser $\mathcal{I}$nterface. *Formal Aspects of Computing*, 11:326–342, 1999.

[19] Extensible stylesheet language (xsl) specification. W3c working draft, W3C, 1999. Available at http://www.w3.org/TR/WD-xsl.