Towards a Wiki for Interactive Educational Mathematics

Christoph Lange

Computer Science, Jacobs University Bremen, ch.lange@jacobs-university.de

Abstract. Two tools addressing different aspects of educational mathematical documents have been developed in our research group and presented to the JEM community on previous occasions: SWiM is a semantic wiki for collaborating on such documents, and JOBAD is a framework that allows for reading documents interactively. In this paper, we outline a vision of an integrated educational environment for interactive learning and collaboration. Starting from an overview of the current state of both SWiM and JOBAD, we show how the vision can be achieved by integrating both systems.

1 Vision

We envision an integrated environment that is potentially open to all aspects of mathematics in education. The system would consist of a database of educational mathematical knowledge, but also be interlinked with external resources on the web. Both learners and teachers would collaborate in that system. Learners would mainly consume educational content. That does not only mean reading static documents, as they have been prepared by the teachers, but we envision interactive documents that invite the reader to explore additional information – inside the knowledge base, or involving external resources or web services –, and to modify the presentation of the knowledge according to their preferences. Teachers would mainly contribute content to the knowledge base, or maintain existing content. But learners would also be able to contribute – their homeworks, for example. All users of the system would be connected to each other by a commenting function: Learners would be able to comment on documents provided by teachers – e. g. giving them feedback on how well they understand them. Conversely, teachers would give learners feedback on their homework. The same feature could also be used for *peer review* – both among learners and among teachers. The system would support users to point out what exactly they consider wrong or improvable in given resources, and, possibly, how they would improve the content. Based on that information, the system would be able to assist authors.

The system that we envision has the potential to blur the boundaries between readers and writers, learners and teachers. While we consider this technically feasible, we are aware of the fact that it may not always be socially desirable. Therefore, the system should have a comprehensive permission management, and we leave the decision on what collaboration features to enable to the teachers setting up the system.

2 Current State

Our current systems realise part of this vision. The SWiM semantic wiki integrates both the author's and the reader's role in a coherent environment. Currently, however, its functionality is targetted at people engineering semiformal mathematical knowledge, such as OpenMath Content Dictionaries [LGP08,Lan09b]. Previous development efforts focused on collaboration on mathematical knowledge rather than on improving their consumption. However, our research group has made progress on investigating mathematical documents in education from a reader's perspective and provided the JOBAD framework (JavaScript API for OMDoc-based Active Documents), which makes mathematical documents interactive by integrating them with web services [GLR09]. This architecture can only reveal its full strength, though, when integrated with a powerful backend – which has not been done yet.

For lack of space, we omit a review of related work but refer the reader to previous publications on SWiM and JOBAD. We aim at providing a system that offers more interactive knowledge-based services than non-semantic websites like the mathematical section of Wikipedia, but at the same time is more agile than existing solutions like MathDox or ActiveMath [MGH+06,CCK+08].

2.1 SWiM, a Wiki for Mathematics

Recent development of SWiM has mainly been motivated by the requirements for maintaining the OpenMath Content Dictionaries on the OpenMath site [Lan09a,Lan09b]. We have some evidence, however, that some of the experiences made in that ongoing case study have a more general impact. Therefore, part of them may be transferable to educational settings, too.

SWiM inherited an elaborate permission management from the underlying IkeWiki system. Both in Content Dictionary maintenance and in education, there are users with different roles in the community. In education, we could distinguish:

Pure consumers who merely read educational contents in the system
Commenting readers who may give feedback on educational resources
Teachers who may edit educational documents and comment on documents containing foundational background knowledge, such as OpenMath Content Dictionaries, which contain formal definitions of mathematical symbols used in educational documents
Full editors who additionally have full access to the Content Dictionaries.

Technically, educational documents and Content Dictionaries are the same to SWiM. They merely differ in their degree of formality. To a learner, SWiM offers various interfaces to browsing documents (see [LGP08]): There can be usual inline links in the document, semantically related documents (e. g. documents imported as prerequisites for the current one, or examples for the definition currently viewed) are accessible via a navigation toolbar, and all mathematical symbols in formulae are linked to the place where they have been defined in a Content Dictionary.

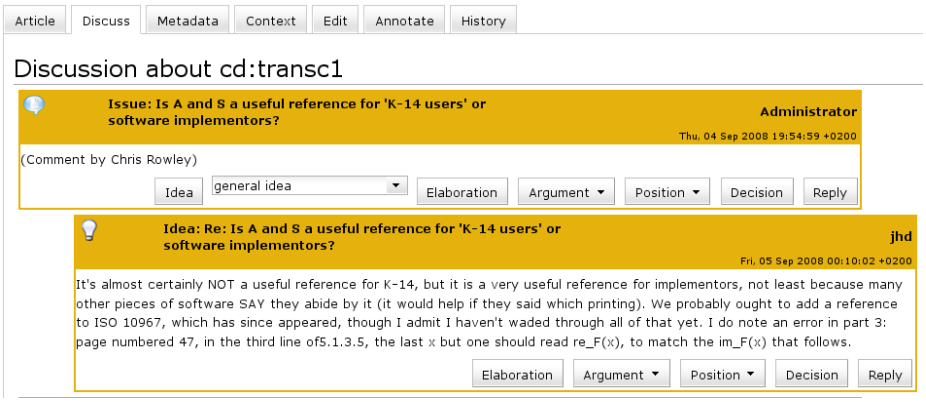

Fig. 1. Part of a discussion page from the OpenMath wiki. Notice the post types and the specialised reply buttons.

Every document has a discussion page. It works like a usual discussion forum but has been extended towards domain-specific problem-solving assistance. Users can give discussion posts and replies exact *types*. We have conducted a survey in the OpenMath and JEM community[1] to elicit patterns of common problem and solution types. For example, a definition of a mathematical concept can be hard to understand, and a common solution would be to provide an example [LHC08]. Once such a standard solution pattern has been suggested, the system can assist the user with implementing it in the knowledge base.

SWiM offers dedicated editing support for all structures of mathematical knowledge (cf. [LGP08]): There is an editor for rich text, extended by a toolbar for adding structural annotations like proof steps. Metadata of documents are editable in a separate form, and finally there is a visual but nevertheless semantic formula editor. In the OpenMath case study, we have found out that SWiM is particularly suited for three use cases that are common in Content Dictionary maintenance: performing minor fixes, discussing and implementing revisions, and editing and verifying notations of mathematical symbols [Lan09b].

2.2 Interactive Documents with JOBAD

JOBAD, the JavaScript API for OMDoc-based Active Documents, is a specification of a MathML-based document format and an HTTP-based communication protocol that enables interactivity in mathematical documents [GLR09,JOB09]. The overall architecture is shown in fig. 2. As shown at the bottom, a JOBAD-enabled document is encoded in XHTML+MathML, where the MathML formulae are annotated semantically by cross-linked parallel markup (i. e. every formula

[1] The survey is still open for participation at `http://tinyurl.com/5qdetd` but likely to be replaced by a more focused survey soon.

is given in a human-readable presentation and a machine-readable structural representation). Interactive services utilise these annotations.

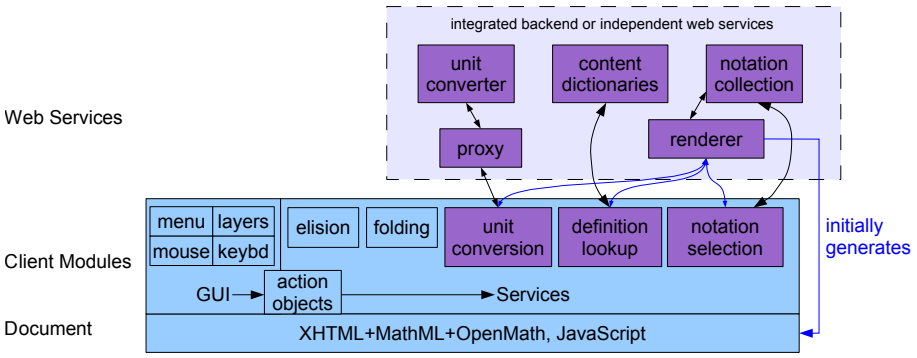

Fig. 2. JOBAD architecture: note the central role of the rendering service, which both generates JOBAD-compliant documents and is used by other services.

First, there are in-document services (shown in blue) that change the appearance of the document without requiring further interaction with a server – provided, the document has been prepared accordingly. A bracket elision service flexibly shows or hides brackets in mathematical formulae, aiming at making complex formulae more comprehensible. A folding service can replace complex subterms by abbreviations – either author-defined ones, such as W_{pot} for $\frac{-e^2}{4\pi\epsilon_0 R/2}$ in a physical formula, or default ones like "…". Then, there are symbol-based services, where additional information about a mathematical symbol selected by the user is requested from a web service backend. We have currently implemented interactive lookup of a symbol's definition and type from a Content Dictionary. First feedback from users shows that definition lookup is particularly useful in educational practice [Gic09]. Work on a notation selection service, which allows for customizing the appearance of individual mathematical symbols, is currently in progress (cf. [KLM+09]). Finally, there are service that send complex mathematical expressions to a web service, in order to have them computed, verified, or rewritten. As a proof of concept, we have implemented a connection to an OpenMath-based unit conversion web service [SD08]. Both client and server side of the latter two types of services are shown in purple in fig. 2. In our current client-side implementation, interactions can be selected from a context menu or by keyboard shortcuts (shown on the left). They either apply to the mathematical symbol the user clicked on, or to the subterm selected by the user.

As said above, JOBAD is not limited to the services that *we* have implemented, but extensible. On the client, calls to services are represented by generic action objects instead of being hard-wired into the context menu, which paves the road for alternative user interfaces. On the server, any existing mathematical web service can be integrated. The programmer only has to provide a client-side

action object that makes an appropriate request to the web service and processes the response received from the web service. The response can either be used to replace the current mathematical formula, or it can be displayed separately, e. g. in a popup window. JOBAD provides a rich set of utility functions for both.

3 Integration Roadmap

The basic integration of JOBAD with SWiM will be straightforward. For rendering documents, SWiM already relies on the JOMDoc library [JOM], which renders documents in a JOBAD-compliant format. In that stage of integration, JOBAD's lookup services would then access the Content Dictionaries from the SWiM knowledge base, thus making sure that a learner can look up exactly those mathematical definitions that have been provided by his teacher. We are planning to extend definition lookup to definition *expansion*: it will be possible to insert the definition of a symbol in place of the symbol, which theoretically works for all types of definitions except implicit ones.

Not only definition lookup can be realised with SWiM; the discussion support could be made more interactive as well. Currently, questions or problem reports about mathematical concepts can only be posted from the page of the respective concept. In a JOBAD-enhanced system, access to the discussion forums can additionally be given from the place where it is actually needed: from any mathematical formula where the user encounters a symbol that he wants to discuss. Thanks to its semantic web nature, SWiM also has the potential to link resources in its own database to external ones on the web. One can easily establish a link from a mathematical concept to, e. g. its historical background, as given on Wikipedia or Wolfram's MathWorld. By a simple enhancement, JOBAD could then provide such links to the user in his current interaction context.

Users who tried the JOBAD demo and interactive adapted mathematical documents while reading, frequently asked whether they could save their adaptations somewhere [Gic09]. As SWiM manages user accounts anyway, such an enhancement would easily be possible. Not only on explicit request, but also *during* interaction with the system, we could build user profiles – assuming the user's consent. A logging service, immediately notifying a web service about any kind of adaptation requested by the user, would help teachers and developers to find out more about how learners interact with mathematical documents and what adaptation features they consider useful. Given that a lot of mathematical web services exist that have already been successfully evaluated in educational settings, they should also be integrated with JOBAD. Examples known to the JEM community are the services originally developed for the ActiveMath [MGH+06] and MathDox environments [CCK+08], performing tasks such as interactive exercise feedback [HJvLG08]; see [GLR09] for an overview.

Finally, we aim at enabling interactive access to larger-scale mathematical structures than formulae. Based on semantically annotated HTML, we have developed an interactive visualization of rhetorical structures in mathematical documents that are rather text-oriented than formula-heavy, as is frequently the

case in educational documents [Gic08]. We want to enable the same for structured proofs, e. g. allowing for interactively folding proof steps.

Acknowledgments : The author would like to thank Jana Giceva for implementing large parts of the JOBAD framework and collecting first feedback from users. This work was supported by JEM-Thematic-Network ECP-038208.

References

ACR+08. S. Autexier, J. Campbell, J. Rubio, V. Sorge, M. Suzuki, and F. Wiedijk, editors. *Intelligent Computer Mathematics, MKM*, number 5144 in LNAI. Springer, 2008.

CCK+08. H. Cuypers, A. M. Cohen, J. W. Knopper, R. Verrijzer, and M. Spanbroek. MathDox, a system for interactive Mathematics. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications*. AACE, 2008.

Gic08. J. Giceva. Capturing Rhetorical Aspects in Mathematical Documents using OMDoc and SALT. Technical report, Jacobs University, DERI Galway, 2008. Available from: https://svn.kwarc.info/repos/supervision/intern/2008/giceva_jana/project/internship%20report.pdf.

Gic09. J. Giceva. Integrating web services into active mathematical documents. Bachelor thesis, Jacobs University Bremen, 2009.

GLR09. J. Giceva, C. Lange, and F. Rabe. Integrating web services into active mathematical documents. In J. Carette, L. Dixon, C. Sacerdoti Coen, and S. Watt, editors, *MKM/Calculemus*, number 5625 in LNAI. Springer, 2009.

HJvLG08. Bastiaan Heeren, Johan Jeuring, Arthur van Leeuwen, and Alex Gerdes. Specifying Strategies for Exercises. In Autexier et al. [ACR+08].

JOB09. JOBAD framework – JavaScript API for OMDoc-based active documents, 2009. Available from: https://jomdoc.omdoc.org/wiki/JOBAD [cited February].

JOM. JOMDoc Project — Java Library for OMDoc documents. Available from: http://jomdoc.omdoc.org [cited March 2009].

KLM+09. M. Kohlhase, C. Lange, C. Müller, N. Müller, and F. Rabe. Notations for active mathematical documents. KWARC Report 2009-1, Jacobs University Bremen, 2009. Available from: http://kwarc.info/publications/papers/KLMMR_NfAD.pdf.

Lan09a. C. Lange. OpenMath wiki [online]. 2009. Available from: http://wiki.openmath.org [cited February].

Lan09b. C. Lange. wiki.openmath.org – how it works, how you can participate. In James H. Davenport, editor, *22nd OpenMath Workshop*, 2009.

LGP08. C. Lange and A. González Palomo. Easily editing and browsing complex OpenMath markup with SWiM. In P. Libbrecht, editor, *Mathematical User Interfaces Workshop*, 2008.

LHC08. C. Lange, T. Hastrup, and S. Corlosquet. Arguing on issues with mathematical knowledge items in a semantic wiki. In J. Baumeister and M. Atzmüller, editors, *LWA (Lernen, Wissensentdeckung und Adaptivität)*, 2008.

MGH+06. E. Melis, G. Goguadze, M. Homik, P. Libbrecht, C. Ullrich, and S. Winterstein. Semantic-aware components and services of ActiveMath. *British Journal of Educational Technology*, 37(3), 2006.

SD08. J. Stratford and J. H. Davenport. Unit knowledge management. In Autexier et al. [ACR+08].