Storing and Retrieving OMDoc documents in the ACTIVEMATH learning environmnent

Paul Libbrecht
The ACTIVEMATH group
DFKI and University of Saarland
paul@activemath.org

Abstract: This paper presents the content storage and retrieval of XML-documents for use in the ACTIVEMATH learning environment.

Challenges arise in storing and retrieving document-bits so as to be displayed and to be used within ACTIVEMATH: One, that references encoded in the documents are to be requested not only in the forward way as written in the document but also in backwards way, and from the fact that the document format used, the OMDoc format for semantic encoding of mathematical document, has a reference mechanism which is somewhat more object-oriented than simple URL-references.

This papers presents the challenges, the needs for the ACTIVEMATH learning environment, and the solutions currently envisionned.

Overview of the Paper

After a brief presentation of the OMDoc language and of the ACTIVEMATH learning environment, we explain why a database-oriented approach was needed. We continue by presenting the retrieval methods to be used for ACTIVEMATH. We then describe the first implementation, followed by the possible wrappings that were made. We finally hint towards open-questions.

1 The OMDoc Language for Mathematical Documents

The Open Mathematical Document format (in short OMDoc) is a format that started while building and polishing the OpenMath [CC98] standard. The latter is an XML-based encoding for mathematical objects which addresses only the encoding of formulae. It is based on the notion of *mathematical symbol* which is attached to a semantic humanly specified in a document called a *content-dictionary*. The human-specification character of this semantic came obviously as a drawback: several of the symbol's specifications were clearly defintions. As such a format specifying theories where symbols, their definitions, axioms, theorems and proofs was needed. Michael Kohlhase therefore created the OMDoc format[Kh01], an XML language extending OpenMath in this direction.

The textual content of OMDoc documents is written within atomic elements which we call *items*. They encompass examples, definitions, theorems, motivations, or exercises. Items also contain metadata (using the Dublin-Core [Th98] standard) as well as pedagogically motivated extensionss.

Each item has an identifier, contained within a `theory` element, itself having a identifier. These two identifiers, combined with a notion of *collection* of documents make-up an identifier meant to be globablly unique. Item references occur very commonly within OMDoc documents as any mathematical symbol is a reference to a `symbol` element which can be considered to be a hook to carry the given semantics. We refer to the article [Th03] for more details about the encoding and its usage for ACTIVEMATH.

2 The ACTIVEMATH learning environment

ACTIVEMATH is an open-source web-based, intelligent learning-environment under development since three years. It forsters re-usability of learning content. As a language for semantically represented mathematical documents of the web, OMDoc was a good candidate to fullfill the re-usability, interoperability, presentation independence requirements. This format was thus chosen to encode the learning material and to present it using XSLT-stylesheets (currently in HTML and PDF, soon in SVG and XHTML+mathml).

As `OMDoc` documents carry a rich semantic, it is actually possible to offer more than simple static HTML pages. The paper [Th03] describes an amount of other usage of the semantic to offer such features as symbol-explanations or copy-and-paste of mathematical formulae.

The `OMDoc` content handled by the ACTIVEMATH environment is based on items used as building-blocks of the content. They are such as `motivation`, `proof`, or `exercise`. The books presented to the learner in ACTIVEMATH are composed as a sequence of items which are assembled, enriched, and passed to a presentation engine which converts them to the appropriate viewable format on the fly (the presentation engine, which allows personalization of documents, is described in more details in [GPLU03]).

The ACTIVEMATH learning environment is a web-server, based on Java-servlets. It aims to support at least the serving load of a classroom in a computerized learning-lab, and, if possible, the server of a publically accessible mathematical encyclopedia.

3 ACTIVEMATH's Needs for the Extraction of `OMDoc`-content

This section describes what ACTIVEMATH needs to extract from the `OMDoc` documents. It justifies, thus, the need for a database approach as opposed to the file-based storage with forward `URL`-based reference mechanism currently available on the classical HTML-based world-wide-web.

The list of items contained in a book presented to the learner is stored as set of references, a *table-of-contents*, which the presentation reads. The references are encoded using almost the same theory-based reference mechanism as the `OMS` elements us (see above). Hence, the ACTIVEMATH queries need to translate these references into physical XML objects.

Of importance in the presentation architecture, is to be noted that to present a given page, it is not enough to fetch the textual content of the items. For example, so as to be able to present as a tool-tip the name of each symbols, these names are fetched as well; similarly, the `metadata` elements are fetched so as to allow the XSLT stylesheets to indicate parts of this metadata within the view (an example is the difficulty of an exercise). Conversely, this extraction should avoid to fetch the complete item as the latter may content very large formal-parts being the translation of the textual parts. The content-extraction thus needs to serve about 20 requests for a given web-page request.

ACTIVEMATH also contain a *user-model*, a *course-generator*, and a *suggestion-engine*. The course-generator and suggestion-engine both make use of queries for the references from one item to the other (for example the relationship stating that a definition defines a given symbol). These queries must go in two ways, that is, it must be possible to request all the definitions of a given symbol, or all the items that depend on a given item. This defines another requirement for the database: the two-ways character of the references, a common-problematic partially tackled by the XPOINTER family of specifications. The course-generator process is a typical place where performance issues occur: one of the tasks of this course generator is to walk recursively through the `depends-on` relationships starting at a set of goals. For a moderate content (for example a 30 pages book), a thousand requests is at least needed.

Finally, a dictionary approach is also available im ACTIVEMATH allowing users to inspect an item and its relationships (in two ways as well). This dictionary also allows a text-search.

Being a learning-environment, ACTIVEMATH content has to be authored by mathematics teachers and researchers. The database thus has a duty to be easily updatable. In the update process the database has to check the consistency of the overall link-structure, a consistency that cannot be checked by normal XML validators as it spans several documents. Error-reporting was, thus, an important capability of such databases.

3.1 The MBASEREF Specification of ActiveMath Requests

The set of requests to deserve the ACTIVEMATH needs has been summarized in a Java interface (that is, an object specification) which we describe here.[1] This interface is called MBASEREF (as *reference to an* MBASE), its *functions* can essentially be grouped in three classes:

Simple Extraction Requests : The methods `getCommonName`, `getTextualContent`, `getTypeString`, `getMetadata`, `getAttribute`, `getFormalContent` (in order of frequency usage) are all methods that extract children of the given items in a straightforward way.

Relationship Requests : The relationship methods query forward references (`getDependencies`, `getForWhat`) or backwards references (`getRelated`, `getProofs`, and `getDefinitions`.)

[1]The documentation for the MBASEREF interface can be found at `http://www.activemath.org/~ilo/redirs/MBaseRef.html`.

Search Requests : The `searchText` method is the only one intended to perform a real search, it is not expected to be quick.

4 How to Serve Content for ActiveMath

Our description of the ActiveMath content-retrieval interface allows us now to describe the approaches envisionned. We shall describe all envisionned aproaches thus providing some hints at other solutions.

4.1 Straight URL-based Retrieval

The first candid approach to serve content with a given ID is to use what has been known as the `OMDoc catalog` element: an element which provides filename-URL for the identifier of each theories indicating processors where to locate documents of almost all identifier-based-references given in the document. This approach is the one in use the command-line oriented XSLT-stylesheets one can download for `OMDoc`[2]. This approach, which is very common,[3] has several drawbacks however:

- a processor has to load a complete document to extract a simplest information such as the name of a mathematical symbol
- it does not provide any means to read backwards dependencies

On the positive side, this approach requires no database and is thus web-friendly, allowing a content collection reachable through a web-server to be used. This enables simple individual with a little web-hosting to display their content.

This approach has actually been favoured by the HEΛM project in the University of Bologna, Italy: the intent of the Hypertext Electronic Library of Mathematics (HEΛM) is, at least, to serve a browsable view of the CoQ theories. Within the MoWGLI project,[4] a conversion of this content into the `OMDoc` format is under work. The documents in the HEΛM library are somewhat simpler in terms of addressing: each entity is stored as one file, thus simple URL reference is working and efficient. This character is, however, made only possible by the fact that these documents are generated from the CoQ library (in its own language). An implementation of an MBaseRef type of interface to HEΛM is planned within the MoWGLI project.

4.2 The MBase Project: A Complete Mathematical Database

As `OMDoc` documents are encoded in a semantic fashion, they can be search with much greater expections than classical text (or TEX) based encoding of mathematical knowledge. The MBase project has exactly this aim: provide a content-storage for `OMDoc`s which can be searched in most details. Typical search facilities that MBase now offered include to search for mathematical expression given a pattern (with joints) along with the data extraction needed to implement MBaseRef.

Just as `OMDoc` documents can contain a formal part to be used by automated-theorem-provers, MBase has as important target to serve the needs of automated provers in searching the applicable theorem within their proof search. A connection of MBase for the ΩMEGA proof-planner has been realized.

MBase is a server program written in the Mozart-Oz programming language, storing its streams within a `MySQL` relational database.

An MBaseRef implementation has been realized for MBase, connecting the servlet-world of ActiveMath to MBase using the XML-RPC protocol. Current experiments have, however, provided disappointing performance results where the `MySQL` database is taking most of the CPU time; about 5 MBaseRef queries could be served per seconds on recent computers.

4.3 Fallbacks: RAM-based Storage

While MBase took time for its development, the database needs for ActiveMath were struggling. Fallback solutions were thus created, based on a simple in-memory loading of the documents. A first implementation was based on a DOM-model. The second and current one, based on the JDOM library appeared.

[2] Available from the `OMDoc` home at `http://www.mathweb.org/omdoc`

[3] This is the approach used by all URL-references of ending with a '#'ed word. For example the reference to an inner anchor in an HTML page

[4] The MoWGLI EU project is documented under `http://mowgli.cs.unibo.it`.

This approach has the clear drawback to be able to store only very little content and impose large memory requirements on the process. Its speed is excellent good (obiously), culminating at more than 500 requests a second currently (with the remote overhead).

It is to be noted that this approach is exactly that of the validators (such as XML-validators). These programs load the whole document in RAM before performing complete checking. Such a program will thus probably have some future life as a validator.

4.4 Combining Multiple Databases

Thanks to the specification-based character of the MBASEREF java interface. It has been relatively easy to provide many implementations, in particular implementations that combine other implementations. We present a little list:

- One of the implementations is built using several MBASEREF objects, for each request, the collection identifier is used to address the request to the appropriate MBASEREF. This repreents a start of *content-distribution*.
- A client and server based on XML-RPC communication allow an MBASEREF to be built which sends MBASEREF requests to an MBASEREF located in a different process or machine. This facility is essential as the startup of some implementations can be relatively long.
- An implementations performs an elementary load-balancing between several others
- An implementation caches the results of the requests to another
- Finally, an implementation allows one of the database to override the content of another. Such a facility is of importance when preparing or revising content allowing the MBASEREF of an author to be relatively small and keep on dedicated servers the need to store large content.

4.5 A First Light-weight Approach

In the effors to remove the tough memory requirements of ACTIVEMATH, which is running by default the in-memory database, a database was built using, as persistence mechanism the Lucene [Fo03] indexing-engine. First experiments seem to provide very good performances similar or exceeding the in-memory solution.

4.6 Alternate Solutions

It is clear that XML-databases are an avenue that has to be explored and we shall do so. However, at the time of writing the first and second in-memory solutions, XML-databases were at their early breath and their speed did not seem to be promising. It is not clear yet that the target usages of XML-databases meet the target usages of ACTIVEMATH as described here.

5 Open Questions

The storage and retrieval effort for the ACTIVEMATH-content has been a side effort since about three years (except the MBASE efforts). This research has, however, opened a set of difficult questions which we allude to here.

Distributed Changing Content : In a world where the world-wide-web dominates, content has now more and more the tendency to stay physically at a central place, where it can be quickly updated. References occur from one place to the other. This *link*-based approach has created (and defined) the web.

OMDoc content would enjoy the same advantages: being distributed where the authors or publishers manage it. However the problem of the validity of the links to an external resource remain. Moreover, it should be possible to compute the set of pointers that point to this external resource. These problems are made worse when publishers actually change their content: there is, technically, no way, currently, for a publisher A to mention to a publisher B that a resource in the domain of A points to a resource in the domain of B and that it would be wished that this resource in the domain of B should not be removed (or not modified, or that this version be kept servable).[5]

[5]A quick answer to this problem could be a local copy of the desired resource. This is however undersirable to allow the complete control of the publisher B over his content. Copyright-conditions can actually go relatively as far as prohibiting local copies as does `http://www.mathgoodies.com/`.

This raises a general problem of the management of distributed consistency which seems surprisingly not tackled.

Performing MBASEREF **Requests over** HTTP As indicated earlier, it would be wished to be able to serve the MBASEREF queries using a simple web-server. Two ingredients appear to be needed to reach such a facility:

- while the forward links are already encoded in the `OMDoc`-documents. The backwards link would need to be added to the content
- as collections and documents can be of a huge size whereas requests are expected to serve very small information, a quick access to the precise elements pointed to by a request would be needed. the separation into elementary files for each request would be a possibility, it would be preferrable though to be able to parse XML-documents-children by extracting the appropriate byte-sequences. It is worth noting that this feature of serving indicated byte-sequences of a given file is part of the HTTP 1.1 specifications.

Such a database has not yet been realized. It could be expected, however, that it would be of more general interest.

Thanks

The author would like to thank Andreas Franke and Michael Kohlhase for (respectively) the implementation and first design of the MBASE-project which triggered this research. Thanks also go to George Goguadze which has agreed reformulating large content for the changes in the `OMDoc`-language and for the needs of the database implementations. Finally, thanks go to Adrian Frischauf for having supported the MBASEREF implementation connecting to MBASE, Martin Fuchs for having restructured the MBASEREF interface, and Shahid Manzoor for having implemented the MBASEREF-cache and the Lucene-based MBASEREF,

Conclusions

We have presented the needs for content storage and extraction within the ACTIVEMATH learning environment and the envisionned solutions.

The ACTIVEMATH environment seems to have needs of XML-extractions which are way beyond the content-management simple retrieval facilities. We believe however that more and more systems will face similar needs and that high-speeds solutions will come.

Within the perfomance measures, it appears that the Lucene-based database will be the one recommended in the distribution of the learning environment. It is remarkable to note that this implementation is actually only XML-aware at loading-time and is otherwise, purely text- (and index-) based.

References

[CC98] Caprotti, O. und Cohen, A. M.: Draft of the open math standard. Open Math Consortium, `http://www.nag.co.uk/projects/OpenMath/omstd/`. 1998.

[Fo03] Foundation, T. A. S. Jakarta lucene. July 2003. See `http://jakarta.apache.org/lucene/docs/index.html`.

[GPLU03] Gonzalez-Palomo, A., Libbrecht, P., und Ullrich, C.: A presentation architecture for individualized content. In: de Bra, P. (Hrsg.), *AH-2003 Workshop on Adaptive Web-Based Hypermedia at the The Twelfth International World Wide Web Conference*. 2003. See also about `http://wwwis.win.tue.nl/ah2003/`.

[Kh01] Kohlhase, M.: `OMDoc`: Towards an internet standard for mathematical knowledge. In: Lozano, E. R. (Hrsg.), *Proceedings of Artificial Intelligence and Symbolic Computation, AISC'2000*. LNAI. Springer Verlag. 2001. See also `http://www.mathweb.org/omdoc`.

[Th98] The Dublin Core Metadata Initiative. Dublin core metadata initiative - home page. 1998. `http://purl.org/DC/`.

[Th03] The ActiveMath Group: Knowledge representation and management in ACTIVEMATH. To appear in the Proceedings of MKM'01 in Annals of Mathematics and Artificial Intelligence. 2003.