An Architecture for Recovering Meaning in a LaTeX to OMDoc Conversion

Deyan Ginev

Computer Science
Jacobs University Bremen
Campus Ring 1
28759 Bremen
Germany

Type: Guided Research Thesis
Date: May 11, 2009
Supervisor: Prof. M. Kohlhase

Abstract

Printing-press mathematics has not been left out by the digital era and is now evolving into an extended Web representation. Given the advantages of hypertext over a static paper article, a mathematical document is now enabled to contain not only a presentational representation, but also a semantic one. This is achieved by a second layer of representation that captures the meaning of the mathematics as an explicit formalism. Formats such as OpenMath and Content MathML allow this on the formula level, while document formats such as OMDoc provide a full framework for creating semantic mathematical documents. A lot of effort has been invested into converting the existing sources of printing-press articles, and into creating add-on enhancements in order to use the same typesetting tools for creating digital document equivalents. LaTeX is undoubtedly the most prominent of the printing-age typesetters and there have been a variety of attempts to convert its output to XHTML+MathML. While these attempts focus on the presentational side of mathematics, what we discuss in this paper is an architecture that is converting LaTeX documents to a semantically enhanced OMDoc representation. This framework carries out the task of automatic migration between knowledge representations and provides a generalized mechanism for semantic enrichment and disambiguation during the conversion process, creating state-of-the-art, semantics-oriented, digital mathematical documents.

Contents

1 Introduction

The printing press industry had traditionally found Mathematics to be a very hard target to typeset just until the mid-twentieth century. A satisfiable (and much praised) solution, namely Donald Knuth's TEX [Knu84] typesetting system, finally allowed mathematics to properly integrate into the world of print. A decade ago the same problem was to be solved yet again, only for a new carrier. The digital era and Web 2.0 lead to the birth of the e-book and, consecutively, most printed libraries have now been migrated and extended to online equivalents. Mathematics, inherently different than natural language, required a special treatment for its integration. Emerging solutions offered different approaches with different emphasis. The MATHML [ABC+03] format is now the standard of embedding mathematics in XHTML documents, as the PRESENTATION MATHML standard is almost universally used by browsers for on-screen display. However, the digital era introduced a completely novel realm to mathematical document typesetting. A mathematical article is no longer a static source for human interpretation but is now a dynamic entity that could easily allow online interactions and editing, enabled by hypertext. Having digitalized mathematics, we can transcend the human-oriented output of a static paper page and take the next step - create mathematical documents with additional explicit formal semantics. The latter enables tools such as semantic search and formal verification, that would be of great assistance to mathematicians. A number of competing formats are currently maturing to facilitate such a semantic and computer-oriented representation. The most prominent two, OPENMATH [BCC+04] and CONTENT MATHML, are now heading in a common direction and aim to provide a standardized semantic representation of mathematical formulas.

While MATHML and OPENMATH are formats specifically designed to represent formula semantics, the OMDOC format [Koh06] provides a representation of entire mathematical documents, incorporating the just mentioned formula level content markup as a specific semantic level. OMDOC accommodates three levels of document semantics: theory, statement and formula level, allowing added-on semantic services to benefit from information on all scales, from, say, semantic formula search (formula level), through formal proof verification (statement level), to theory morphism analysis (theory level). This demonstrates the status of OMDOC as a state-of-the-art representation for digital mathematical documents and shows a small glimpse of the exciting new possibilities laying ahead of all varieties of automated document analysis. Mathematical Knowledge Management (MKM) has emerged to study and guide the process of formulating the new representation formats, to solve integrating the classical mathematical typesetting tools and sources with the new standards and services. Furthermore, it is analyzing the

implications and potential of the different knowledge representation approaches, driving the creation and standardization of document analysis tools, closely interacting with the Semantic Web effort.

One of the Holy Grails in the recent efforts in MKM is achieving as explicit, as unambiguous and as complete as possible semantic content representation in the documents of interest, as this is the key to unlocking the full potential of automated computational analyses. What this paper will address is closely related, we develop an approach to obtain and aggregate externally derived semantic content, providing a conversion pipeline from classic LaTeX typeset documents to a semantically rich OMDoc representation. We aim to develop the power to translate such classic documents into their modern, enriched equivalents, as this would allow their integration in the digital era and enable them to benefit from the various services of the Semantic Web. Additionally, we investigate LaTeX as a tool for typesetting OMDoc documents and compare our method to other existing approaches.

We proceed with explaining the motivation behind this project, discussing related work. Next we present two different contexts of applying our approach, presenting a large-scale corpus architecture as well as a self-contained LaTeX to OMDoc conversion. Also, we briefly examine a controlled approach to making this transition, concluding with a summary of the strengths and weaknesses of our design and an outlook to future improvements and applications.

2 Motivation

The project was originally inspired by the the arXMLiv effort [SK08], which strives to convert the arXiv [arx] ePrint collection of scientific articles, a document corpus typeset in LaTeX. In order to achieve a general LaTeX to OMDoc conversion, we place our initial focus on the arXiv collection of documents, containing over half a million scientific articles, which constitute a representative sample of LaTeX in the "wild". We can state that a conversion architecture deployed on the arXiv corpus can easily scale, if not be directly used, to arbitrary LaTeX corpora, due to which we present the following discussion in a corpus-centric view . The arXMLiv conversion is based on Bruce R. Miller's LaTeXML [Mil07] software and targets an XHTML+MathML representation, optionally providing Content MathML or OpenMath content for the mathematical fragments. The main purpose of LaTeXML is to convert LaTeX articles into an XML *equivalent*, i.e. to preserve all semantic and presentation information from the original document, but also not to infer any additional knowledge. This leads to a rather generic

and mainly default semantics in the content mathematics, which is unreliable for further use as most complicated formulas tend to have non-trivial, underspecified and context-dependent semantics. In general, one can not infer a close-to-correct semantics of a given fragment without first performing a real semantic analysis on the document.

To this extent, the "Language and Mathematics Processing and Understanding" (LAMAPUN) project, based at the KWARC research group at Jacobs University, tackles the various challenges with deriving meaning from a corpus of scientific documents. LAMAPUN currently investigates semantic enrichment, structural semantics and ambiguity resolution in mathematical corpora. Long term goals include applications in areas such as Information Retrieval, Document Clustering, Management of Change and Verification. The architecture described here is part of the overall LAMAPUN effort and establishes a basis for its future work.

The LAMAPUN work on the ARXMLIV corpus is currently based on the contributions of a group of Jacobs University graduate students. As such, it is a long-term, distributed effort of alternating developers. Facilitating continuous development demands an intuitive and maintainable abstraction layer allowing for a fast learning curve of new contributors and independence from the specific architecture implementation. The converted nature of the ARXMLIV corpus allows great customizability of its documents, but at the price of a rather involved low-level interaction. Hence, there is a need for a stable backbone which utilizes the power behind the corpus conversion mechanism and automates the different conversion and analysis stages. Furthermore, different applications on top of the corpus demand different emphases on knowledge representation, state of processing and inferred structure. The architecture needs to encapsulate the different representation stages and potential needs. It must also allow easy interaction with external tools, motivating a modular design of stand-alone components, each dealing with a particular intermediate representation and state of the document data. However, developers are not the only source of interaction for such a framework. Many popular techniques in linguistic and semantic analysis use various supervised or semi-supervised approaches and future applications should be allowed to easily facilitate such tasks. Another type of end-user for the framework results would be external automated tools performing formal tasks such as Management of Change or Verification. These types of interactions need fundamentally different representations, having a presentational or a semantic emphasis of the final output format.

The specific focus of this paper is providing a stable architecture backbone that automates all knowledge representation and low-level corpus tasks, while at the same time allowing for a generalized front-end for the development of semantic analysis modules. We enable a set of outputs that can accommodate heteroge-

neous types of follow-up applications, an OMDoc generation procedure in particular. Large-scale corpus analysis exhibits all challenges, prerequisites and features of a general conversion framework, hence we integrate and utilize our LaTeX to OMDoc architecture in the context of the overall LaMaPUn effort.

3 Related work

Existing general-purpose annotation frameworks, such as GATE [CMBT02], Heart of Gold (HoG) [Sch05] or UIMA [FL04], already provide parts of the functionality we need for our system. They focus on providing a setting for creating analysis pipelines, oriented towards linguistic analysis and information extraction. However, none of them is ready for direct deployment on a large body of XML documents, or can be easily accustomed to support various knowledge representations. In the context of analyzing the arXMLiv corpus and the Semantic Web in general, an intuitive and standardized support of hypertext data is vital for a successful and efficient application development and deployment. The LaMaPUn work already focuses on understanding mathematical discourse, demanding support for different XML formats for mathematics and an accessible document representation for our semantic analysis tools.

The Werewolf framework [AJG+09] aims at providing an abstraction layer in which competing semantic modules can analyze and annotate a document, collecting the revealed semantics in an annotation database. Hence, it can accommodate modules that result in semantic enrichment on all document levels, giving a means to potentially infer semantics for the full expressivity of OMDoc. Recent efforts using this framework have already given initial results [AJG+09], yet there is the need of standardization regarding the produced annotations, so that they are readily available for further use, such as aggregation to an OMDoc representation. Alternatively, current LaMaPUn projects employ an RDF annotation database for the same purpose, which could then fully formalize both annotations and inter-document relations via a comprehensive ontology. As we focus on the low-level backbone, we want to be neutral to the particular approach to extracting meaning, yet we base the early implementation on the above-mentioned RDF approach, which we will later discuss in detail.

When it comes to existing or easy to introduce support of different XML knowledge representations, none of the discussed systems could rise up to meet our needs, thus we contribute to the current state-of-the-art by designing a framework that is quickly deployable, representation-aware and natively supports Semantic Web mathematics.

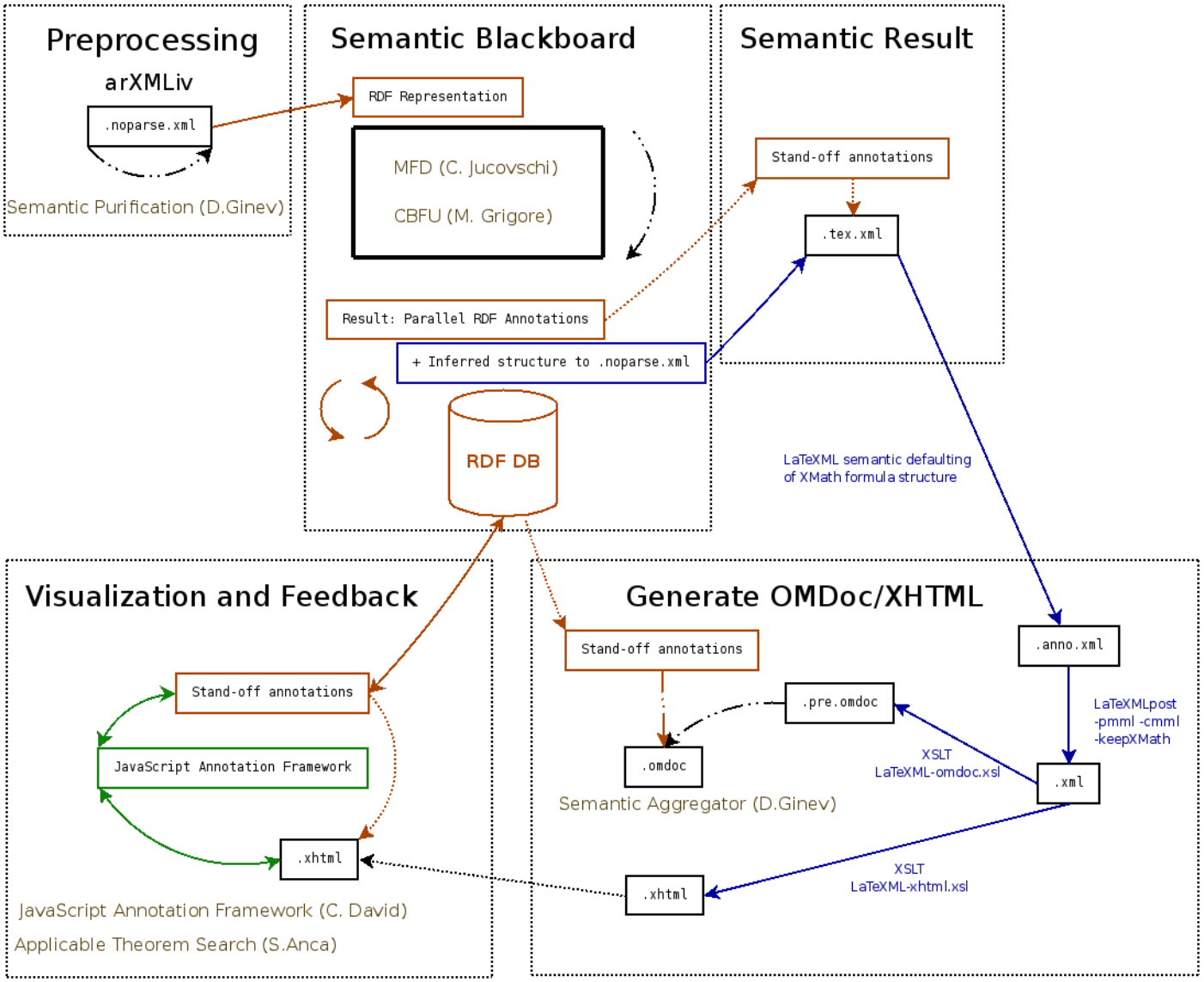

Figure 1: A high-end overview of an ARXMLIV analysis architecture

4 The ARXMLIV analysis architecture

We contribute to implementing a modular architecture that provides a stand-off RDF abstraction of the source documents and automates the migration in between the underlying XML representations, which are essential for the ARXMLIV corpus with an outlook to added-on services. The modules encapsulate *preprocessing*, a "*Semantic Blackboard*" for distributed semantic analysis, a representation of the *semantic results*, appropriate *generation of output formats*, as well as *user interaction and visualization*, as outlined in Fig.1. We proceed with a detailed review of the system components, as described in [GJA+09], followed by an in-depth discussion of the LaTeX to OMDOC system subpart which is our primary contribution.

4.1 The LATEXML Backbone

The LaTeX to XML conversion that effectively created the ARXMLIV corpus, has been performed by Bruce R. Miller's LATEXML system [Mil07]. LATEXML is a highly customizable tool released in the Public Domain, which supports the conversion from LaTeX to a custom XML format. Consecutively, its postprocessor, LATEXMLPOST, can drive the conversion to XHTML and potentially any other representation via a customized XSLT stylesheet. The chief difference between LATEXML's representations resides in the structural semantics of mathematical fragments. LATEXML is currently able to generate both Presentation and Content MATHML [ABC+03], as well as an OPENMATH [BCC+04] representation of mathematics from its intermediate XMATH format. In this paper, each representation of interest will be distinguished via an appropriate filetype of the document:

- **.noparse.xml** - Contains a representation linguistically equivalent to the LaTeX source document. Mathematical formulas are represented via a linear sequence of atomic components, i.e. tokens, without creating any semantic parse tree (unless explicitly stated otherwise in the LaTeX source). This custom LATEXML XML format is achieved by an explicit demand on LATEXML to not parse any mathematical structures beyond the atomic token level.
- **.tex.xml** - Equivalent to **.noparse.xml** with the exception of parsing mathematical fragments and creating a formula derivation tree. The semantics of the mathematics is changed, as the formula structure is achieved via a predefined grammar and the use of simple heuristics, which implies defaulting of both symbol and structural semantics. This often leads to wrong seman-

tics of the respective augmented fragment. However, the only way to assure a valid conversion of the math fragments to a content representation is via such treatments. This is the case since otherwise any subsequent processor will have to deal with partially linearized mathematics, leftover from the **.noparse.xml** predecessor. Such structure is clearly ambiguous and malformed, hence needing further analysis to be resolved. Still, **.tex.xml** is generated solely to enable postprocessing, which is an analysis-free stage. As a framework default, creating a full derivation tree guarantees a successful pass through the different representation conversions, producing valid XML output.

- **.xml** - Provides additional MATHML and/or OPENMATH representation of the math fragments, optionally using parallel markup and keeping the original XMATH. The rest of the XML DOM is still the same as of the previous **.tex.xml** and **.noparse.xml**.
- **.xhtml** - Achieved via a native XSLT stylesheet which transforms the **.xml** into XHTML.
- **.omdoc** - Not natively supported by LATEXML, achieved via a custom XSLT stylesheet followed by a semantic aggregation procedure which makes explicit any previously derived meaning.

As LATEXML already facilitates the transition between the different intermediate stages, incorporating it as a backbone of the architecture is an obvious choice. The LATEXML developers have contributed to the effort in a fruitful collaboration which gave further power to LATEXML's DOM and postprocessing module. Our discussions lead to an enhanced mechanism for generating "xml:id" attributes needed as annotation-hooks and improving the generationg of parallel mathematics during postprocessing, enabling us to consistently recognize and reconstruct mathematical fragments throughout the different modules. Integrating the different representations together, could now be achieved almost out of the box. A set of low-level Perl scripts and customized XSLT stylesheets manage the consistent transition between intermediate formats, accommodating their proper interpretation by LATEXML and LATEXMLPOST and assuring the preservation of the XML hooks which would be used for stand-off annotations. Remarkably, most of the processing is performed by already existing capabilities of the LATEXML software, which makes the architecture design lighter and provides a very intuitive conversion pipeline.

4.2 Preprocessing Module

The ARXIV corpus contains almost 20 years of good and bad practice of writing TEX and LATEX docucuments. Since TEX/LATEX is presentation-oriented, authors have only been interested in whether the result "looks right", equiping the user with abundant possibilities to achieve their desired presentation. However, some of these techniques are "semantically adequate" while some are not. The preprocessing module tries its best to convert the latter into the former, purging any structures devoid of semantics. For example, a current preprocessor, written by the author as a seminar project, purifies the modularity of mathematics and natural language, improving its semantics.

4.3 Semantic Blackboard Module

The vision behind a "Semantic Blackboard" is essentially to allow distributed document analysis by providing an accessible representation format and the means to store and later use the inferred semantic information from all active analyzers. This module is the semantic core of the architecture, coordinating the analysis process and acting as an interface between the different meaning-extracting applications and the rest of the framework. We discuss one possible solution which we have designed in the context of seminar project work for the LAMAPUN effort. Its implementation is due to Constantin Jucovschi and we have incorporated it in our initial system design.

4.3.1 Knowledge Representation

The idea of introducing semantics into a very large online corpus like the ARXIV is very ambitious. Clearly, it is a long term project and hopefully more research groups will join our efforts (or vice versa) to accomplish this task. In order to ensure a long life to this project we have to use a knowledge representation system that is easy to understand, use, extend, distribute and share. We want other researchers to quickly grasp the basic concepts and spend their time on hunting for new semantic information. We do not want to limit the users in using a certain tool, hence supporting software based on the knowledge representation of choice should already exist. Also, we want to have a system which gives the possibility to fetch only a subset of the available original data and inferred semantics, process it, and then push new semantic data back to a public database. This will make the system faster and more robust to failures, as each user can work with local data.

For these reasons we chose to adhere to the standards and best practices of the Semantic Web.

Consequently, we chose a stand-off annotation system. Through that, we avoid having conflicts in between the efforts of different researchers, the resulting system is faster and more stable, and is also easier to share. Also, as prescribed by best practices from Semantic Web, we represent knowledge in the subject-predicate-object paradigm supported by the W3C Resource Description Framework (RDF) [LSWC98]. This will make sharing new semantics easier and tool independent. These decisions represent the only imposed limitations for describing semantics.

We use the openRDF database Sesame [BKH01] to store semantic annotations. It provides fast storage, SPARQL [PS08] query support as well as a friendly user interface (not of least importance). Having a query language enhances developer experience considerably. Firstly, fetching some data does not mean writing yet another program. Secondly, one can specifically download/work with the data from the server in which he/she is interested in. Also, enabling the use of SPARQL query language is a step forward towards more flexibility in choosing the underlying storage database and hence should be adhered to whenever possible.

As we base our work on the intermediate **.noparse.xml** stage in the corpus conversion which is not publicly accessible, we are compelled to keep this data in the public RDF database as subject-predicate-object statements. Storing the corpus documents in this way might sound suboptimal, however it gives us the option of hiding the complexity of the XML representation by ignoring, for example, formatting tags. This also means that we can introduce them back into the database on demand. Another gain is the expressivity to group objects of the same type. For example we are free to define a *followed* relationship between consecutive words, even if they do not appear consecutively in the document.

4.4 Semantic Result Module

The Semantic Result Module is a static module that preserves the semantic analysis results in their original stand-off configuration. The final state of the stand-off annotations produced by the Semantic Blackboard analyzers, after all processing has taken place, is considered the analysis result. The primary **.noparse.xml** document which was the subject of analysis is enhanced with unambiguous and consistent inferred structural semantics, ideally becoming a correct version of LaTeXML's **.tex.xml** representation and in turn changing its own extension to **.tex.xml**.

4.5 Output Generation Module

Over the course of the architecture development, Bruce R. Miller has assisted us in making LaTeXML customizable enough to support the specific needs of the representation migrations for the different architecture modules. The main help of LaTeXML's functionality is in the Output Generation Module, starting with the conversion from **.tex.xml** to **.xml**. At this step we have the option to add parallel MathML (towards **.xhtml**), OpenMath (towards **.omdoc**) and XMath (for annotation visualization and feedback), translating the mathematical fragments into state-of-the-art representations, targeting both human- and computer-oriented applications. Currently there are two supported output formats from this math-enhanced intermediate **.xml** representation, respectively a presentation oriented one and a content oriented one.

As XHTML is the standard for hypertext documents, it is an obvious choice for a presentation-oriented representation. It allows embedding mathematics via the MathML format, which in turn allows for accommodating any alternative representation via annotation-xml elements. In our workflow, we use the global "xml:id" attributes of the ⟨Math⟩ elements as annotation hooks throughout all XML representations, which makes the stand-off annotation process more generalized and maintainable. Preserving these hooks during the **.xml** to **.xhtml** conversion requires a slight deviation from the native LaTeXML to XHTML stylesheet, which is the only change we need to introduce to the existing LaTeXML procedure, giving us the workflow to XHTML at almost zero cost. This facilitates a connection between the XHTML representation and the stand-off annotation database, satisfying the prerequisites for the successive Interaction and Visualization Module.

OMDoc [Koh06] is a state-of-the-art content representation format for mathematical documents and is the second supported output by the architecture. As LaTeXML does not directly support an OMDoc representation at the moment, we had to develop our own LaTeXML to OMDoc stylesheet supporting the transition. Furthermore, as the OMDoc format is capable of expressing semantics on all document levels, it is a target for the aggregation of the inferred stand-off content. This is achieved via a semantic aggregator which performs consistency checks, resolves conflicts and avoids redundancy on the database annotations, embedding the aggregated results into the OMDoc output. The aggregation procedure is still work in progress and would employ a semantic analysis of its own.

4.6 Interaction and Visualization Module

We make use of the **.xhtml** representation generated by the Output Module which allows us to immediately provide online document interaction. The next step in the development utilizes the GREASEMONKEY [Gre09] extension for Firefox, which allows users to customize the way webpages look and function. This method of customization is already widely used and users have developed tools for interaction with websites so that the user would enjoy a better web experience. First of all, this extension allows deep HTML modification in appearance, by allowing user created scripts to modify the original HTML code of the document and add certain types of controls. Secondly, changes could also be functional. Via implementing JAVASCRIPT functions, the script may invoke the refresh of a page at a certain time or other behavior that enhances the experience of the users. Having this setup, the single major add-on left to implement is the safe and correct communication with the RDF database. The implementation of this module is currently under development and its client-side approach promises a distributed, secure and efficient user interactionwith the semantic results of the Semantic Blackboard. This module has been developed by Catalin David in the form of a seminar project as part of the LAMAPUN effort.

5 The LaTeX to OMDoc conversion

As a second context for our design, we abstract away from the in-depth description in section 4, focusing on using our approach as a general LaTeX to OMDoc conversion, as can be seen in Fig.2. The pipeline is based on the same modular architecture utilized for the ARXMLIV framework and is neutral to the specific implementation of the Semantic Blackboard module. What is vital, is having a consistent convention for the derived stand-off semantic annotations, allowing a generalized aggregation to OMDoc. However, this is one of the items that still hasn't found a consistent solution, but is work in process and will not be further discussed here. This application was our initial primary goal, but as, the architecture developed it became secondary, as it is essentially a subset of the general functionality presented in section 4.

The system is again based on the LATEXML conversion flow, as described in section 4.1, so we will focus on the challenges in achieving the final steps of the translation, expanding on the introductory description in section 4.5.

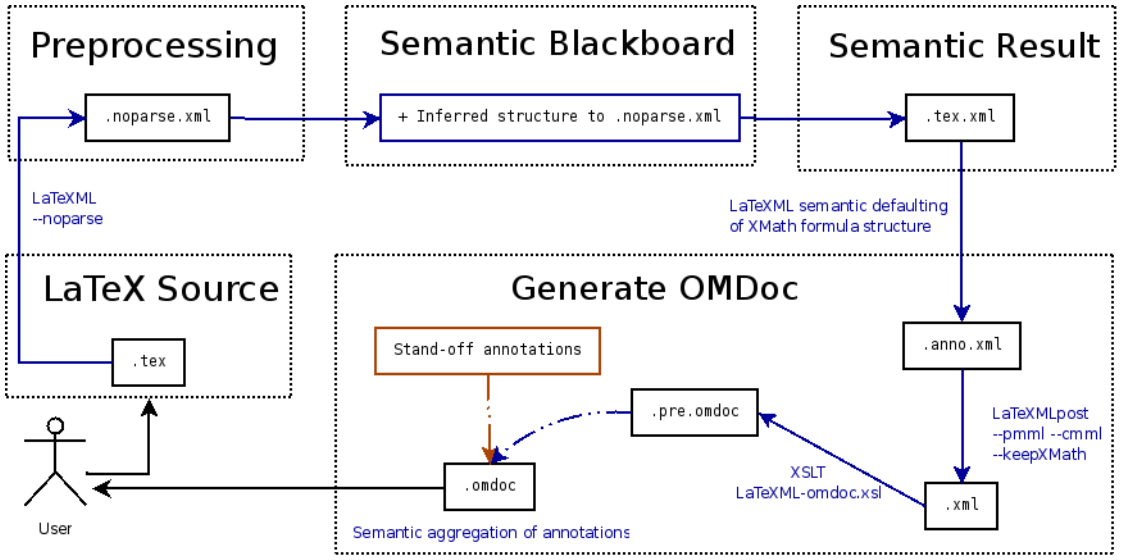

Figure 2: A LaTeX to OMDoc conversion framework

5.1 The translation to OMDoc

Having a semantic result in the **.tex.xml** format, one needs to apply an XSLT style-sheet to obtain a generic OMDoc representation, which is to be further enriched in the following aggregation stage. We took the native LaTeXML "LaTeXML-xhtml.xsl" style-sheet as a starting point, also using the "omdocpost.xsl" style-sheet from sTeX as guidance. LaTeXML tries to represent all of LaTeX in its custom representation, which leads to a ghastly schema and a rather involved XSLT conversion to XHTML. The resulting "LaTeXML-omdoc.xsl" style-sheet, together with its modular subcomponents, is both too long and too unsightly to be directly included in our description, hence we outline the main steps we undertook in the conversion. In the following, all elements without a namespace are in the default OMDoc namespace.

- The logical paragraph elements, ⟨ltx:para⟩,⟨ltx:proof⟩,⟨ltx:theorem⟩ are translated to a ⟨CMP⟩ element, nested in an ⟨omtext⟩
- The figure elements, ⟨ltx:figure⟩, are translated to a ⟨omlet⟩ element.
- The atomic paragraph element, ⟨ltx:p⟩, is translated to its OMDoc equivalent, ⟨p⟩, having the paragraph level encapsulation to a ⟨CMP⟩ defined above.
- All block level elements, such as ⟨ltx:block⟩, are translated to ⟨omgroup⟩.

- Mathematics (represented as MATHML and/or OPENMATH) only needs to change its namespace to the default OMDOC one and is already natively supported.
- Structure level elements, such as ⟨ltx:section⟩,⟨ltx:chapter⟩, are translated to omgroup.
- Metadata is mapped to its respective element in the Dublin Core specification, e.g. ⟨ltx:creator⟩ and ⟨ltx:abstract⟩ are respectively translated to ⟨dc:creator⟩ and ⟨dc:description⟩.
- We preserved the original **xml:id** generation mechanism, which creates semantic id attributes, based on the location of the fragment in the document tree.
- Due to the fact that a significant subset of the XHTML expressivity is also utilized in the OMDOC format, we could preserve all XHTML target elements that were in the intersection from the original "LaTeXML-xhtml.xsl" style-sheet (e.g. ⟨span⟩,⟨table⟩,⟨emph⟩,...).

5.2 The aggregation procedure

In order to produce the OMDOC representation that we ultimately desire, one would need to handpick and process the annotations, aggregating them in a consistent and conflict-free routine. The semantic aggregator has the task of performing this analysis and document manipulation, resolving conflicts and giving meaningful defaults when appropriate. The aggregation is to be as generic as possible, so that it could facilitate diverse and yet to be created annotation types. The annotations are grouped into types, depending on which level of OMDOC semantics they address and are subjected to conflict tests, both with respect to OMDOC validity and semantic consistency. In the cases where conflicts occur, a generic semantic analysis[1] would resolve them and select a (possibly empty) subset of non-conflict annotation semantics. Underspecification and context ambiguity embody a lot of potential pitfalls for such analysis, so the techniques implemented are to be well justified and tested against a set of use cases. Due to the lack of annotation conventions, however, proper testing of this module is yet to be undertaken, as such having such conventions is a vital prerequisite for a consistent aggregation.

An early implementation of this aggregator is based on the JOMDOC API for OMDOC. It analyses the annotations via first integrating them as ⟨ignore⟩ ele-

[1] Can a mathematical fragment be both a matrix and a sequence? Can a segment be a theory and the proof if the annotations target the same scope?

ments via a script preceding the XSLT processing. This allows to work entirely in the context of an OMDoc document, which is the very playground of JOMDoc, allowing for a convenient abstraction for further semantic analysis and a rapid development process.

6 A controlled conversion approach

The sTEX project [Koh08] is employing semantic preloading of the LATEX sources and offers a translation library for LATEXML that explicitly converts the in-built sTEX semantics to OMDoc. While sTEX approaches document conversion from the inside-out, our architecture implements an inverse technique which strives to infer the semantics via a parallel analysis system and then aggregate and disambiguate it back to a single document. This puts us in the serious disadvantage of dealing with the full set of challenges of underspecification, ambiguity and context-specific semantics. However, it also gives us a broader target, as our architecture could eventually facilitate any LATEX document[2] in the wild.

Interestingly, both approaches are based on the LATEXML software to perform the transition to OMDoc, yet while our architecture employs a meaning-revealing postprocessing on a document, sTEX introduces semantics, as explicit OMDoc elements, in specific LATEXML libraries (or bindings). This gives power to preload a document with the intended semantics, placing the burden of semantic enrichment on the author. While this defines the two approaches as fundamentally different, it also allows for future collaboration. As an example, sTEX could be used for creating ground truth OMDoc equivalents of a LATEX source, which could be later used as a comparison basis of our general approach, providing a ground truth goal for the enrichment and aggregation mechanisms.

7 Conclusion and Outlook

We have introduced a conversion architecture from LATEX to OMDoc, providing a stable backbone based on the LATEXML software and enabling an OMDoc output, aggregating derived semantics in the form of stand-off annotations.

Additionally, we have integrated our work with the LAMAPUN project to achieve a large-scale analysis framework working on top of the ARXMLIV corpus. The well-motivated modular design promises scalability and easy maintainability in

[2] Provided it is convertible by LATEXML

the long-term, while harnessing the power of existing semantic tools and platforms. Based on the LaTeXML system, the process of migration in between knowledge representations, while simultaneously preserving inferred semantics, becomes stable, fully-automated and encapsulated from the rest of the system. A data abstraction of the corpus documents, which stores them in the context of an online database of stand-off annotations in the W3C RDF format, provides an intuitive and distributed platform for potential developers, at a very small learning curve, as well as a rapid implementation and deployment timeframe. In the Interaction and Visualization module we facilitate multiple purpose user interaction for various supervised techniques and, as a means to ease the development process, display inferred annotations and enable their creation and editing.

The preprocessing and postprocessing modules are currently being further developed and improved and there are plans for novel applications utilizing the power of our framework. We envision the design of an ontology for mathematical discourse relations, formalizing the RDF representation employed by the document abstraction layer. Furthermore, we are looking for collaborations in creating analysis modules that infer semantics from the arXMLiv corpus, using the techniques from Computational Linguistics and Computational Semantics.

The aggregation procedure to OMDoc needs to be further extended, following the creation of a consistent anntation convention. The JOMDoc API is a powerful aid with performing analysis on top of an OMDoc target and could easily facilitate aggregation of both structural and symbol semantics, achieving a fully enriched OMDoc output. The promise of this architecture and the future work of the LaMaPUn group looks to achieve a large-scale formalization pipeline which performs full semantic enrichment of informal mathematical discourse, creating a consistent, unambiguous, formal OMDoc representation.

8 Acknowledgements

The original architecture design was adopted from Prof. Dr. Michael Kohlhase, whose continuous supervision and guidance helped to make his Semantic Blackboard vision take form. Much insight was gained during the in-depth discussions at the LaMaPUn project meetings - the author thanks Stefan Anca, Catalin David, Mihai Grigore and Constantin Jucovschi for their helpful feedback and close collaboration. Bruce R. Miller contributed with invaluable guidance to the LaTeXML components and interplay and made possible the rapid and stable development of the framework. We are also grateful to Christop Lange (soon to be PhD), who oversaw the creation of the XSLT style-sheet from **.tex.xml** to **.omdoc**.

References

[ABC+03] Ron Ausbrooks, Stephen Buswell, David Carlisle, Stéphane Dalmas, Stan Devitt, Angel Diaz, Max Froumentin, Roger Hunter, Patrick Ion, Michael Kohlhase, Robert Miner, Nico Poppelier, Bruce Smith, Neil Soiffer, Robert Sutor, and Stephen Watt. Mathematical Markup Language (MathML) version 2.0 (second edition). W3C recommendation, World Wide Web Consortium, 2003.

[AJG+09] Andrei Aiordăchioaie, Constantin Jucovschi, Deyan Ginev, Elena Agapie, Elena Digor, Kristina Sojakova, Siarhei Kuryla, Mihai Grigore, Michael Kohlhase, Milena Makaveeva, Razvan Pascanu, Stefan Anca, and Soenke Holsten. Computational semantics of natural langauge lab results and findings, 2009.

[arx] Arxiv e-print archive. http://arxiv.org.

[BCC+04] Stephen Buswell, Olga Caprotti, David P. Carlisle, Michael C. Dewar, Marc Gaetano, and Michael Kohlhase. The Open Math standard, version 2.0. Technical report, The Open Math Society, 2004.

[BKH01] Jeen Broekstra, Arjohn Kampman, and Frank Van Harmelen. Sesame: An architecture for storing and querying rdf data and schema information. In *Semantics for the WWW*. MIT Press, 2001.

[CMBT02] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. Gate: A framework and graphical development environment for robust nlp tools and applications. In *Proceedings of the 40th Annual Meeting of the ACL*, 2002.

[FL04] David Ferrucci and Adam Lally. Uima: an architectural approach to unstructured information processing in the corporate research environment. *Nat. Lang. Eng.*, 10(3-4):327–348, 2004.

[GJA+09] Deyan Ginev, Constantin Jucovschi, Ştefan Anca, Mihai Grigore, Cătălin David, and Michael Kohlhase. An architecture for linguistic and semantic analysis on the arxmliv corpus. Submitted to AST'09, 2009.

[Gre09] Greasemonkey - a firefox extension for customizing web pages. http://diveintogreasemonkey.org, seen March 2009.

[Knu84] Donald E. Knuth. *The TEXbook*. Addison-Wesley, 1984.

[Koh06] Michael Kohlhase. OMDoc – *An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, 2006.

[Koh08] Michael Kohlhase. Using LaTeX as a semantic markup format. *Mathematics in Computer Science*, 2008.

[LSWC98] Ora Lassila, Ralph R. Swick, World Wide, and Web Consortium. Resource description framework (rdf) model and syntax specification, 1998.

[Mil07] Bruce Miller. `LaTeXML`: A LaTeX to xml converter. Web Manual at `http://dlmf.nist.gov/LaTeXML/`, seen September2007.

[PS08] Eric Prud'hommeaux and Andy Seaborne. Sparql query language for rdf. Technical report, W3C, January 2008.

[Sch05] Ulrich Schäfer. *Heart of Gold – an XML-based middleware for the integration of deep and shallow natural language processing components, User and Developer Documentation*. DFKI Language Technology Lab, Saarbrücken, Germany, 2005.

[SK08] Heinrich Stamerjohanns and Michael Kohlhase. Transforming the arχiv to xml. In Serge Autexier, John Campbell, J. Rubio, Volker. Sorge, Masakasu Suzuki, and Freek Wiedijk, editors, *Intelligent Computer Mathematics, 9th International Conference, AISC 2008 15th Symposium, Calculemus 2008 7th International Conference, MKM 2008 Birmingham, UK, July 28 - August 1, 2008, Proceedings*, number 5144 in LNAI, pages 574–582. Springer Verlag, 2008.