WOIDE: Semantic Annotation in MS Word — Scaling Mathematical User Interfaces beyond LaTeX

Aurelius Adrian, Michael Kohlhase

Computer Science, FAU Erlangen-Nürnberg

Abstract

In the last years, we have seen methods from flexiformal mathematics be used for making documents active, i.e. interactive and user-adaptive. The main limitation of e.g. learning assistant systems like ALeA that use active documents is that they depend on the sTeX framework – a semantical variant of LaTeX –. This requires the underlying domain model and the learning objects to be encoded in sTeX specific standards and be suitably annotated semantically to power the knowledge management algorithms that drive active documents.

To accommodate authors of domains other than mathematics, physics, and computer science, where LaTeX is commonplace, systems like ALeA must be scaled beyond those disciplines. We need to provide ways of semantically annotating the respectively prevalent document formats; usually some flavor of office suite.

For this we present the WOIDE plugin for MS Word, which allows to embed sTeX-like semantic annotations into MS Word documents as the most prevalent tool in practice. The main contribution of this paper has been to find a way to persistently store annotations in the DocX file/object format and build a minimally viable plugin.

Keywords

Semantic Authoring, MS Word, Annotation

1. Introduction

In the last years, we have seen methods from flexiformal mathematics be used for making documents active, i.e. interactive and user-adaptive; in particular in the arena of scientific publishing and education.

The main limitation of e.g. learning assistant systems like ALeA (Adaptive Learning Assistant) [Ber+23] is that they require the underlying domain models and learning objects to be encoded in sTeX [CICM22] to power the knowledge management algorithms that enable active documents. And of course sTeX is difficult to learn without a good familiarity of LaTeX.

In subjects like legal sciences, where great terminological precision, rigorous argumentation and a vast knowledge of the legal norms, current interpretations, and precedent is mandated, LaTeX is virtually unknown and the use of MS Word is unopposed. More generally, the use of any formal, logic-based or computational forms of representation are unheard of for which by

MathUI 2024

aurelius.adrian@fau.de (A. Adrian); michael.kohlhase@fau.de (M. Kohlhase)
https://kwarc.info/kohlhase (M. Kohlhase)
0000-0002-9859-6337 (M. Kohlhase)

legal practitioners – in education and practice. So using sTeX as a basis effectively deprives e.g. the legal domain of systems like ALeA.

The ALeA system, therefore, is (currently) very much centered on mathematics, using sTeX– a semantic variant of LaTeX– as the main source representation language that covers semantic annotation of text and formulae, which the ALeA system then uses as the basis for automating learning interventions (adaptive, interactive generated documents). Arguably, mathematical learning is already well-covered in ALeA via sTeX, and a LaTeX-based solution is the adequate tool for mathematical content (including Physics, Computer Science, etc.). The purpose of the WOIDE extension is to open up the LaTeXecosystem to other disciplines (like the humanities, biology/medicine, or – in our example – the law) and (via SHTML) mix them with the already existing sTeX-based mathematical content. Thus, WOIDE is not intended to be applied to mathematical content (we have sTeX for that) but to spread mathematical user interfaces to disciplines outside of the core mathematical disciplines. This might even yield new insights within formal domains which could benefit from the extended knowledge base also by authors within non-formal disciplines. Additionally, opening the possibility to (perhaps in the future) construct and annotate new interdisciplinary relationships of knowledge fragments by the collective use of either sTeX or WOIDE, which could then be used for the same ecosystem (ALeA, etc.).

Indeed, the proximate motivation for the work presented in this paper is that the author, a legal practitioner, of a textbook on the five volumes of the German civil law saw the ALeA system in action and understood its embedded semantic learning support services as a way to make this monster book – ca. 1000 pages – palatable and effective for law students, other practitioners, and eventually potentially the general public.

Nonetheless, the overarching aim of this project is not to complete a specific legal task. Our aim is much rather the extension of an abstract formalization method of knowledge to non-formal domains, which is tried and tested in domains that are already reliant on formal representation. We focus our work specifically on the systems based around the MMT, ALeA and sTeX frameworks, in particular because our research questions focuses on an interdisciplinary approach to knowledge representation. For this reason, the applicability of preexisting other formalization methods for our work (e.g. LegalRuleML[Pal+11]), even if they are specifically intended for the legal domain, is limited. So if we can enable semantic editing without LaTeX, we can port originally mathematical learner/reader interfaces to other subjects.

Generally: To scale the impact of systems like ALeA beyond Mathematics, Physics, and Computer Science, where LaTeX is commonplace we need to provide ways of semantically annotating the respectively prevalent document formats; usually some flavor of office suite.

Somewhat surprisingly, while there are many general annotation frameworks for plain text, and some – usually linguistically motivated – frameworks for HTML annotation, there are no general annotation frameworks for word processor formats that we could easily adapt for our purposes. The only notable exception is the CPoint application [Koh08] that supports annotation with text fragment classifications and relations from the OMDoc ontology underlying sTeX within MS PowerPoint and can export the content as OMDoc XML [Koh06] preserving these annotations. This cannot simply be adapted to our purposes, as it is based on the previous PPT object model (which is still supported, but is significantly distinct to that of current MS PowerPoint and MS Word) and moreover is implemented in the (since outdated) Visual Basic environment.

The contribution of this paper is the WOIDE (Word OMDoc IDE) plugin for MS Word, which allows an author to annotate documents with text fragment classifications, relations, and metadata from the OMDoc ontology. For this we had to find a way to persistently store annotations in the Office Open XML file format (.docx) and build an annotation interface that does not break the affordances/features of the host environment. Annotated documents can then be exported into SHTML, an extension of HTML+MathML, which forms the basis of the ALeA system. Indeed, SHTML is what sTeX itself is transformed into for further processing. So the annotated MS Word documents can be transparently integrated into the ALeA ecosystem, but where sTeX comes with an VSCode IDE, WOIDE integrates IDE functionality into MS Word.

This paper is based on [Adr24], which contains many of the details omitted here for space.

2. Invading the MS Word Object Model

The main problem for a client-side, WYSIWYG-document-based[1] annotation system is to find a way of making annotations persistent, i.e., to ensure that when an annotated document is stored to a file and reopened, the annotations are still intact. The fact that .docx files are just zipped collections of XML files suggests that we may use external standoff markup formats (e.g., RDF) and just package them within the zip bundle or add annotations in the XML as custom elements or attributes (e.g., as RDFa). But it seems that MS Office applications "lint" files upon opening, such that any annotations are not persisted.

MS Word is a closed source, but "open API" application, therefore, we can program against the API of the document object model via a variety of supported languages. We have investigated a variety of combinations; in the end, it turned out that the "Content Control Objects" (CCOs) [Mic23] are a suitable basis for an annotation plugin: CCOs are elements that can be inserted in a document which can be referenced by the JavaScript API as well as the Visual Basic for Applications ecosystem. The CCO surrounds the content within and acts like a container with which specific API actions can be performed.

The advantages of using CCOs are that MS Word

1. checks and ensures the persistence of CCO content,
2. provides a somewhat customizable user interface for displaying CCO labels, and
3. automatically tracks changes to the content within and the position of CCOs.

The main disadvantage of using CCOs is that these are not designed specifically for the given annotation task, but rather objects that can be used by various applications and the MS Word user themselves. We must therefore ensure that CCOs that are added to the document for a different purpose and by different means than through WOIDE do not interfere with those added by WOIDE. Additionally, CCOs do not provide a system that enables us to store any arbitrary data structure, which is required for the annotation task. To overcome the lack of configurable data structures, WOIDE implements a component that converts the data structures used to strings which can be saved within the CCOs.

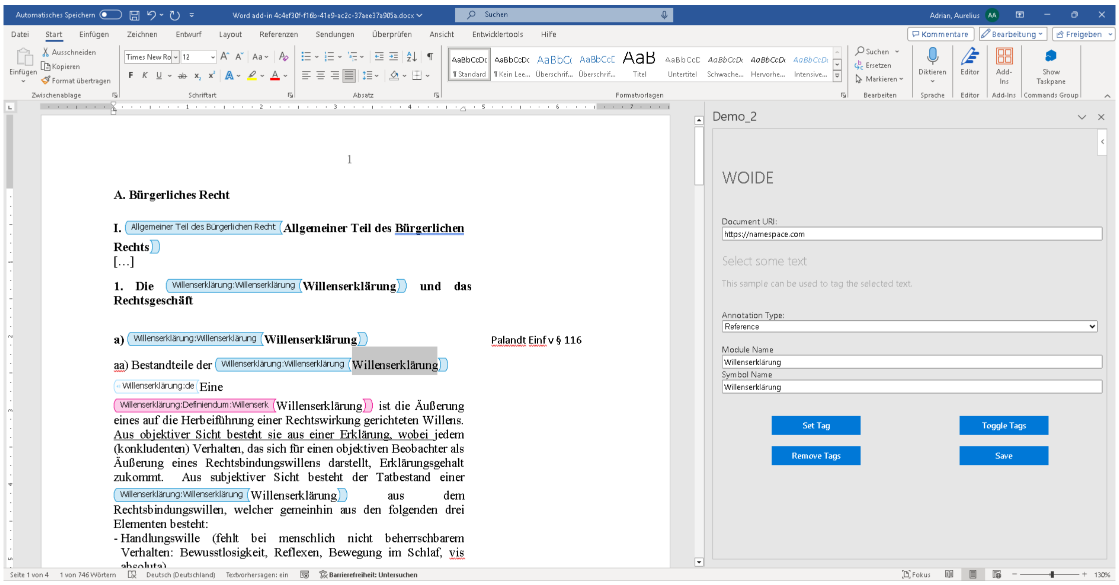
[1]WYSIWG = What you see is what you get

For the SHTML export we make use of the fact that MS Word already has an HTML export for regular document content. Therefore, we only need to take care of the CCOs themselves; but these directly correspond to the SHTML annotations, so we can resort to custom templates. In lieu of a specification we show an illustrative example:

```
<span shtml:term="OMID" shtml:notationid=""
      shtml:head="https://namespace.com?Willenserklaerung?Willenserklaerung">
   <span shtml:comp="https://namespace.com?Willenserklaerung?Willenserklaerung
      <p class="MsoNormal"><span lang="DE">Willenserklaerung</span></p>
   </span>
</span>
```

The shtml: attributes (and – where necessary – added <span> or <div> elements) constitute the semantic annotations and are constructed from the content of the CCOs. The rest of the HTML/CSS markup (cf. the characteristic class="MsoNormal" attribute) is from the MS Word HTML exporter.

We have implemented WOIDE as a JavaScript add-in for MS Word. This ensures that it runs efficiently/natively both on the desktop application as well as in the hosted/cloud version MS Office 365. The OMDoc annotation ontology, user interface specifics like label colors, and the corresponding SHTML attribute names are implemented as a (currently hard-coded) JSON object that is read to dynamically generate the necessary CCOs, this makes the code much more robust and can pave the way for future configurability of the WOIDE plugin.



1

A. Bürgerliches Recht

I. Allgemeiner Teil des Bürgerlichen Recht **Allgemeiner Teil des Bürgerlichen Rechts**
[…]

1. Die Willenserklärung:Willenserklärung **Willenserklärung und das Rechtsgeschäft**

a) Willenserklärung:Willenserklärung **Willenserklärung** Palandt Einf v § 116

aa) Bestandteile der Willenserklärung:Willenserklärung Willenserklärung

Willenserklärung:de Eine Willenserklärung:Definiendum:Willenserk Willenserklärung ist die Äußerung eines auf die Herbeiführung einer Rechtswirkung gerichteten Willens. Aus objektiver Sicht besteht sie aus einer Erklärung, wobei jedem (konkludenten) Verhalten, das sich für einen objektiven Beobachter als Äußerung eines Rechtsbindungswillens darstellt, Erklärungsgehalt zukommt. Aus subjektiver Sicht besteht der Tatbestand einer Willenserklärung:Willenserklärung Willenserklärung aus dem Rechtsbindungswillen, welcher gemeinhin aus den folgenden drei Elementen besteht:
- Handlungswille (fehlt bei menschlich nicht beherrschbarem Verhalten: Bewusstlosigkeit, Reflexen, Bewegung im Schlaf, vis

Demo_2

WOIDE

Document URI: https://namespace.com

Select some text

This sample can be used to tag the selected text.

Annotation Type: Reference

Module Name: Willenserklärung

Symbol Name: Willenserklärung

Seite 1 von 4 1 von 746 Wörtern Deutsch (Deutschland) Textvorhersagen: ein Barrierefreiheit: Untersuchen Fokus 130%

Figure 1: WOIDE in Action

3. The WOIDE Plugin in Action

Figure 1 shows the WOIDE in action. The user interface to annotate MS Word texts consists of the simple control panel on the right, which can be opened from the ribbon menu after installation. The document URI is necessary for SHTML export – all semantic annotations correspond to URIs. For annotation: the document author – or a dedicated annotator – selects a text fragment for annotation and chooses one of the defined annotation types – here a selection of OMDoc classes or relations – from a drop-down menu. For relational annotations, e.g., a term reference relation that needs a definiendum as a target, a context-sensitive “target chooser” appears. The target chooser provides the author with the appropriate input fields to complete the corresponding annotation; here the symbol with name “Willenserklärung” (declaration of intent) in the module “Willenserklärung”.

The blue buttons allow to commit to an annotation (Set Tag) over a selected range, to remove all annotations within a selection, and to export (Save) the annotated document as SHTML. The blue “Toggle Tags” button cycles through three different display styles offered by the MS Word user interface.

Generally, there are two different approaches. The annotations in fig. 1 are displayed “in edito”, this means they disrupt the formatting and style of the underlying text, as the annotation itself is inserted into the text. Other display types supported by WOIDE are standoff annotation types which do not disrupt the formatting and style of the underlying document; see fig. 2.

So far, the design decisions for CCOs and JavaScript add-in have borne out, the plugin is quite usable in practice and (so far) scales well with larger (legal) documents, however, scaling the technology further such that our aim of annotating the "1000-page" book is possible must first be tested. So far we have focused on the terminological/knowledge relations supported by the OMDoc ontology. We expect that the special “legal” referencing system[2] can be mapped to the document-centered iMMT ontology we are developing for/with sTEX, however, we currently have not developed this thoroughly yet.

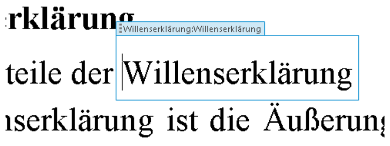

rklärung
Willenserklärung:Willenserklärung
teile der Willenserklärung
ıserklärung ist die Äußerung

Figure 2: Standoff Markup

4. Conclusion

We have presented the WOIDE plugin as a minimally viable proof of concept for a semantic annotation facility for MS Word. The key contribution was to identify a persistent read/writable storage facility in the MS Word object model that could be co-opted for annotations. The WOIDE Plugin is open source (licensed with GPL3) and the code can be found at https://github.com/aurelius-adrian/WOIDE.

The user can simply “install” WOIDE by loading the MS Word Add-In into their MS Office installation. WOIDE could be served with the MS Word Add-In Store. We will consider facilitating that when it is sufficiently stable to be useful beyond the single-book use case.

Currently, the annotation facility in WOIDE is hard-wired to a relatively small subset of the OMDoc ontology that is sufficient for experimenting with the civil law text book in ALeA. But

[2]via named/numbered “books of law” and numbered paragraphs and sentences therein

nothing prevents us from loading annotation ontologies (as JSON structures like the one we hard-code) and thus making WOIDE parametric in it.

Given that the MS Office 365 products now share a joint object model (implemented in JavaScript), it should be possible to extend the methods (and annotation plugins) to other Office 365 applications. The main problem to solve is probably the question of what user interface functionality is compatible with the respective application; for instance it seems unlikely that "in edito" markup will go well with slide presentations in MS PowerPoint.

Finally, it seems reasonable that all the semantic functionality supplied by the MMT system to the VSCode IDE for sTeX should be integrated into WOIDE in some way. A crucial feature of the sTeX VSCode IDE is its language server protocol, which aids the author in making formally correct semantic annotations. Unfortunately, MS Office 365 does not seem to support the underlying language server protocol used in the VSCode IDE directly, which must therefore be fully incorporated in the frontend functionality of WOIDE.

References

[Adr24] Aurelius Adrian. "Building an Annotation Tool for sTeX-Like Annotations in Word - WOIDE". Bachelor Project Report. Computer Science, FAU Erlangen-Nürnberg, 2024. URL: https://gl.kwarc.info/supervision/projectarchive/-/blob/master/2024/Adrian_Aurelius.pdf.

[Ber+23] Marc Berges et al. "Learning Support Systems based on Mathematical Knowledge Managment". In: *Intelligent Computer Mathematics (CICM) 2023.* Ed. by Catherine Dubois and Manfred Kerber. LNAI. Springer, 2023. DOI: 978-3-031-42753-4. URL: https://url.mathhub.info/CICM23ALEA.

[CICM22] Dennis Müller and Michael Kohlhase. "Injecting Formal Mathematics Into LaTeX". In: *Intelligent Computer Mathematics (CICM) 2022.* Ed. by Kevin Buzzard and Temur Kutsia. Vol. 13467. LNAI. Springer, 2022, pp. 168–183. ISBN: 978-3-031-16680-8. URL: https://kwarc.info/people/dmueller/pubs/cicm22stexmmt.pdf.

[Koh06] Michael Kohlhase. *OMDoc – An open markup format for mathematical documents [Version 1.2].* LNAI 4180. Springer Verlag, Aug. 2006. URL: http://omdoc.org/pubs/omdoc1.2.pdf.

[Koh08] Andrea Kohlhase. "Semantic Interaction Design: Composing Knowledge with CPoint". PhD thesis. Computer Science, Universität Bremen, Apr. 2008. URL: https://kwarc.info/ako/pubs/AKo_Promo.pdf.

[Mic23] Microsoft. *ContentControl object (Word).* Nov. 2, 2023. URL: https://learn.microsoft.com/en-us/office/vba/api/word.contentcontrol.

[Pal+11] Monica Palmirani et al. "LegalRuleML: XML-Based Rules and Norms". In: *Rule-Based Modeling and Computing on the Semantic Web.* Ed. by Frank Olken, Monica Palmirani, and Davide Sottara. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 298–312. ISBN: 978-3-642-24908-2.