KAT: an Annotation Tool for STEM Documents

Sourabh Lal, Michael Kohlhase, Tom Wiesing

May 26, 2015

Abstract

Contemporary natural language processing (NLP) systems are based on corpora of annotated documents for training and evaluation. To extend NLP to documents from Science, Technology, Engineering, and Mathematics (STEM) we need annotation systems that can deal with structured elements like mathematical formulae, tables, and possibly even diagrams. Current linguistic annotation systems treat documents as word sequences and disregard the structure of complex document elements, and are therefore unsuited for STEM annotation as this very structure carries important syntactic and semantic information.

We present the KAT system, a browser-based annotation tool for linguistic/semantic annotations in structured (XHTML5, i.e. HTML + MathML + SVG in XML serialization) documents. As KAT is parametric in the annotation ontology and represents annotations as RDF, it can easily be integrated into RDF-based corpus management systems; we present an integration into the CorTeX system.

Contents

1 Introduction

[1] EdN:1

An annotation tool is a system that is used to manage annotations on a document. It provides features such as adding, modifying or removing annotations. Annotations are metadata about a document, and do not actually alter the content of the document. They can be thought of as a layer on top of a document which contains information about the text in the document. This can be done by creating annotations either inline or stand-off. Annotations can serve a variety of purposes including:

[1] EdNote: Have a better introductory paragraph

- Posting comments on the content.
- Marking out parts of the document.
- Demarcating relationships between information fragments.
- Discussing the contents of the document (using a comment thread linked to each annotation).

Annotation tools can be categorized according to the type of annotations they make. Generally state of the art anotation tools create one of the following types of annotations:

1. *Dynamic Annotations* - These create annotations that are anchored to the text of the document.
2. *Static Annotations* - These create annotations that are anchored to a particular position in the page of the document.

Annotation tools are of particular interest to the KWARC research group. Digitized, mathematical text lies in the focus of KWARC's research direction[2], and they need an annotation tool that could be used to annotate mathematical documents. The most appropriate technique for this would be the use of a dynamic annotation tool. However, dynamic annotations are fundamentally flawed when handling structured documents as they are equipped to only handle plain-text documents. This meant that KWARChad to build a new tool, which unlike other state of the art annotation tools could create annotations in structured (XHTML) documents. This new system created structured annotations; annotations that are anchored to a node in the document tree. [3] EdN:2 EdN:3

A key component of this project involved development of the frontend for KAT. We aimed to develop a user interface that optimizes system usability and improves the user experience. This involved first identifying which aspects of the user interface maximize usability by conducting design research. Using the results from this design research, we developed each of the main features that an annotation tool should support: creating annotations, modifying annotations and appropriately displaying annotations.

2 State Of The Art

Currently there are several annotations tools available to use for online text annotations including Hypothes.is, brat, Yawas and Annotatie [DM15]. Most state of the art anotation tools create one of the following types of annotations: [4] EdN:4

1. *Dynamic Annotations* - These create annotations that are anchored to the text of the document.
2. *Static Annotations* - These create annotations that are anchored to a particular position in the page of the document.

[2] EdNote: Why are they important in general? See abstract
[3] EdNote: Copy stuff from abstract in the paragraph
[4] EdNote: decide whether to put his in intro or here

2.1 Hypothes.is

Hypothes.is [hyp15] is a tool developed with the aim of "adding a new layer to the web". This tool is an online, dynamic annotation tool that can highlight and annotate pdfs and web pages. It provides additional features such as the ability to make an annotation public or private, and being able to post replies to annotations. A user needs to create an account before they can create an annotation, enabling Hypothes.is to show the user his/her private annotations each time they access the webpage. The tool can be either downloaded or run as a Javascript plugin.

2.2 brat

The brat rapid annotation tool [bra14] is a web based dynamic text annotation tool. It is designed to create annotations that have a fixed form that can be automatically processed and interpreted. brat can handle two basic types of annotations:

1. Text Span Annotations: Creates a simple annotation on a stretch of highlighted text.
2. Relation Annotations: Creates a connection between two text-span annotations.

brat provides several functionalities that make it easy to use. The main ones are:

1. Advanced annotation searching tool.
2. An annotation export interface that can convert the internal storage format to PDF or HTML.
3. Unique address to access each annotation.

2.3 Weaknesses Identified in SotA

Inability to handle structured documents When creating a new annotation, we need to mark the text that the annotation refers to. The state of the art annotation tools are equipped to handle only plain-text documents. While this makes storing annotations very simple as we just store the positions of the start and end characters in a string, it also makes it unsuitable for STEM annotation as the structure carries important syntactic and semantic information

Visually disruptive annotations One key weakness with state of the art annotation tools is the fact that certain annotations break the flow of the underlying text. This is particularly true of relation annotations in brat. If there are several relation annotations connecting two annotations that are separated by a multiple lines, the layout of the document is no longer user friendly.

Insufficient help for new users In order to improve user experience and system usability there needs to be more help provided to help new users understand how to use the new system. While brat provided a tutorial, none of the other 3 systems had any sort of help or guide. Yet even in brat, finding the tutorial was not intuitive.

3 The KAT System Architecture & Implementation

As KAT is based on XHTML5, we can employ the XML tool chain and rely on standard libraries for the implementation. In particular, we can use uniform resource identifiers (URI) to identify text fragments and represent annotations in RDF – subject/predicate/object triples where the components are URI references to web resources. The subjects are usually text fragments, the objects are as well (for relational annotations) or alternatively are concepts from an annotation ontology (for classificational annotations). The predicates are always properties and relations defined in the annotation ontology.

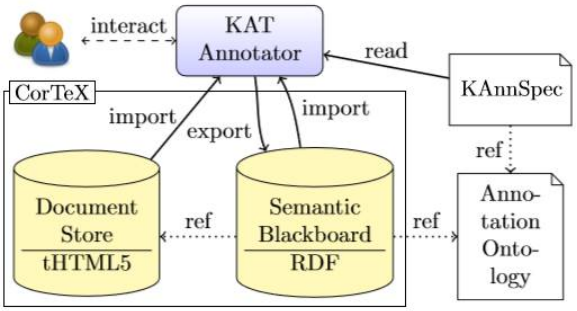

Figure 1: The KAT System Architecture

The KAT system itself is realized as a JavaScript library which instruments an XHTML5 document in a browser. To simplify the URI-based referencing of text ranges (node-sets in the HTML document object model) KAT assumes that the document has been word- and sentence-tokenized; the tokens are wrapped in HTML span elements that carry unique id attributes corresponding to the TEI guidelines. The annotation workflow itself is form-based as shown in Figure 2: the annotator selects a text range, and is then given a modal form to fill classifications and relations as required by the annotation ontology. The annotations are stored as RDF triples in the browser's local storage and can be visualized by special pop-ups and arrows (see Figure 2).

Figure 2: Annotating in KAT: Selection and Form-Filling

KAT is not tied to a particular annotation ontology (or ontology format). At startup, the system is reads a set of *KAT bindings* – custom XML files that describe the annotation interface, the constraints between the components of an annotation frame, and the RDF to be produced. The frame of OMDoc symbols used in Figure 2, is given by the fragment of a KAT binding in Listing 1. It specifies *i*) the fields of the annotation form, their values and validation constraints, *ii*) their display, and *iii*) the RDF subgraph for the frame via a templating mechanism. Note that this frame classifies the annotated word as an OMDoc symbol (via the rdf:type predicate) and relates it to its name via the o:symbolname relation from the OMDoc ontology.

Listing 1: A KAT Frame Specification for OMDoc Symbols

```
<frame name="Symbol">
    <help>An OpenMath/OMDoc Symbol</help>
    <fields>
        <field name="name" type="text">
            <help>The name of the symbol defines it in a theory</help>
            <value>Name</value>
            <default>Symbol</default>
            <validation>[A-Z][a-z]*</validation>
        </field>
    </fields>
    <display> ... </display>
    <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
```

```
        <rdf:Description xmlns:o="http://omdoc.org/ontology#">
            <rdf:type rdf:resource="http://omdoc.org/ontology#Symbol"/>
            <o:symbolname>{name}</o:symbolname>
        </rdf:Description>
    </rdf:RDF>
</frame>
```

[5] EdN:5

Even though KAT can work as a standalone library that can be added to any STEM document in HTML5, it is best used as a component of a corpus management system, such as the CorTEX system [Cor] developed by the second author. In this situation the annotator requests an annotation task from CorTEX, which serves the TEI-tokenized document with KAT and a set of bindings for the intended annotation ontologies. When the annotation is complete, the generated RDF is exported to the semantic blackboard – an RDF triple store maintained by CorTEX. When combined with CorTEX, KAT can be used by multiple annotators; and can be used to review existing annotations, by importing them from the triple store. KAT also has experimental support for inter-annotator agreement reviews via a side-by-side view of the various annotations.

[6] EdN:6

4 Conclusion & Future Work

We have presented the KAT system, an open, parametric, and browser-based annotation system for STEM documents encoded in HTML5. The code base is released under the Gnu Public License and is available at [KAT]. The system is in an early state of development, but can already be used for practical annotation and annotation review work[7]. The user interface both needs and is yet to undergo serious usability testing and polish, to both improve productivity and also to fully stabilize it for production use. The next steps will be to refine our annotation ontologies and integrate more linguistic ontologies to get more coverage. EdN:7

References

[bra14] *brat rapid annotation tool*, 2014. http://brat.nlplab.org/.

[Cor] CorTEX framework.

[DM15] Alex Dumitru and Vlad Merticariu. Current web state of annotation tools. Technical report, Jacobs University Bremen, May 2015.

[hyp15] *Hypothes.is*, 2015. http://hypothes.is/.

[KAT] GitHub repository.

[5] EdNote: Update and redo KAnnSpecs
[6] EdNote: UI Implementaton
[7] EdNote: Nope, it has been changed