KAT: an Annotation Tool for STEM Documents

Tom Wiesing, Deyan Ginev, Sourabh Lal, Michael Kohlhase

Computer Science
Jacobs University Bremen

MathUI Workshop @ CICM
Washington DC, 13. July 2015

Natural Language Processing Systems & Annotatations

- parse natural language and automatically annotate documents
- commonly need a so-called "gold standard"
  - manually annotated corpus
  - used for training and evaluation
- Annotations are meta-data on top of a document
  - do not change the content of the document
  - store information about the content of the document
- need a tool to create and manage annotations

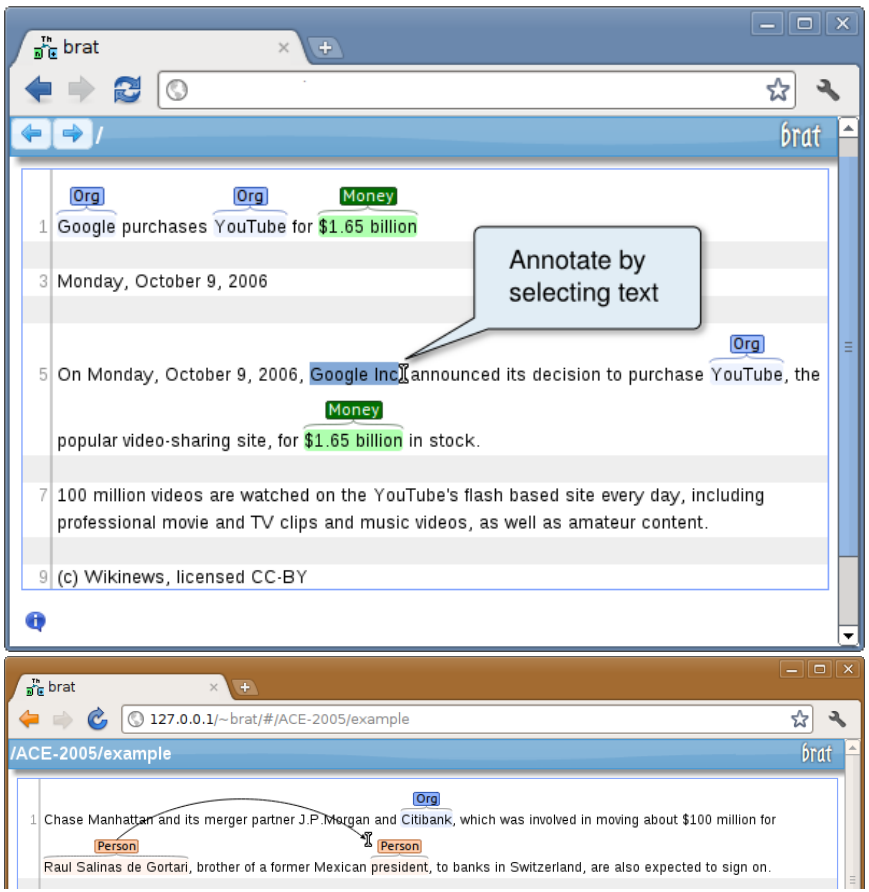
Existing Annotation Tools

- Static Annotators, e.g. *Annotatie*
  - annotate ranges in document "images" (e.g. scans)
- Web Annotators, e.g. *Hypothes.is, YAWAS, ...*
  - annotate online documents as well as PDFs
  - free-form comments
  - can be shared publically or privately
- linguistic annotators, e.g. *brat* (= brat rapid annotation tool)
  - create fix-form annotations to work with automatically
  - text based and relational annotations
  - provides pdf and html export as well as search

Observation: neither fit our purpose: linguistic annotation of STEM documents

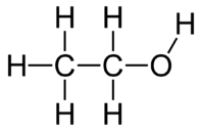
Existing Annotation Tools: Interfaces

brat



YAWAS

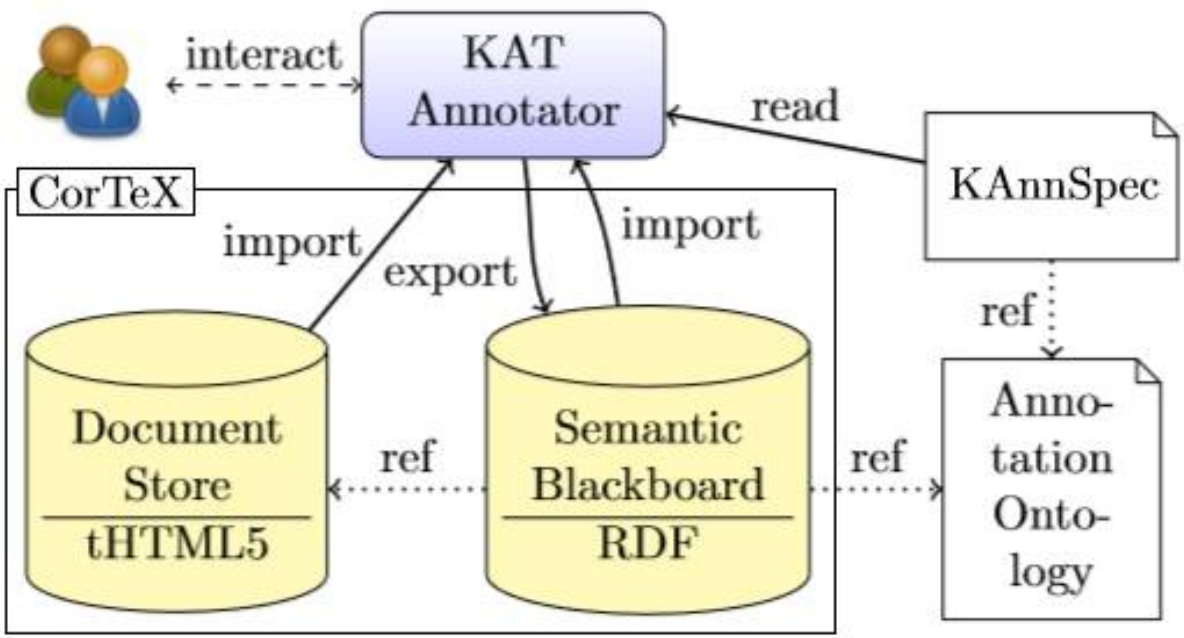
Particularities of STEM documents

- Observation: STEM Documents are multimodal (use more than text)
  - Formulae
    - "For each $\epsilon > 0$"
    - "..., then $\epsilon^2/7 > \delta$..."
    - "The OH group in  (ethanol)..."
  - Tables
  - Diagrams
  - Images
  - Quotes
  - Listings
  - ...
- Observation: XHTML5 (HTML5 in XML serialization) can represent all of them
- KAT Approach: annotate XHTML5 documents

What is KAT?

- KAT = **K**WARC **A**nnotation **T**ool
- web-based annotation tool (= it runs in a browser)
- stores annotations bound to ranges within XHTML5 documents (TEI-tokenized)
- has a flexible ontology specefied by a KAnnSpec

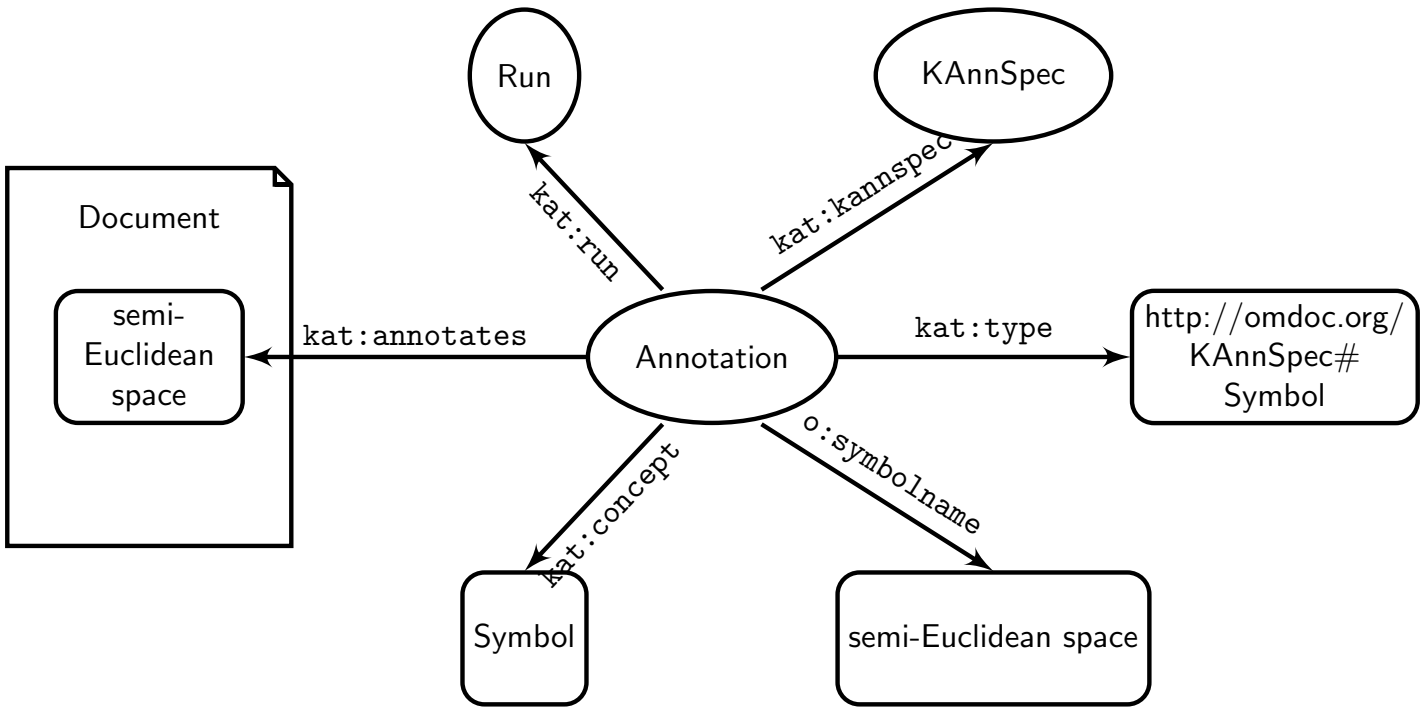
Architecture of KAT



- best used together with a Corpus Management system (e. g. CorTeX)
- annotation procedure:
  1) request annotation task from CorTeX
  2) TEI-tokenised document and KAnnSpec sent to KAT
  3) user annotates document
  4) generated annotations sent back to CorTeX

Representing Annotations in RDF

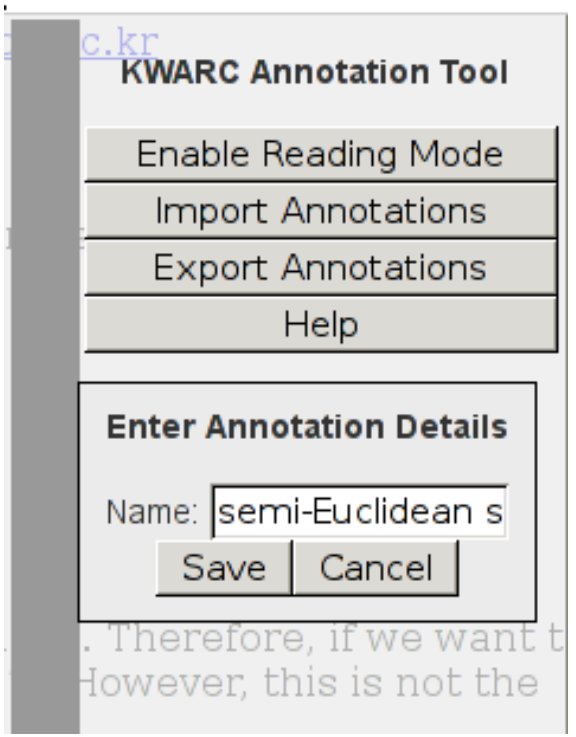
- annotation bound to an XML range in the document
  - XPath to select start and end of range
  - also point to least common ancestor element
  - gives syntax like `doc#cse(con,start,end)`
  - can only select select entire XML elements (words, sentences, subformulae)
- anonymous RDF node for annotation (multiple anontations per range possible)
- RDF triples connect annotation node with annotation information



KAnnSpecs and Ontologies (1)

- KAT is not tied to a particular ontology
- we use so-called KAnnSpecs (= **K**AT **Ann**otation **Spec**ifiction) that describe
  - annotation interface
  - constraints between annotation components and
  - RDF to be produced
- in particular we need to specify:

  1) fields of the annotation form (with values + constraints)
  2) their display
  3) RDF attributes required to create RDF triples



KAnnSpecs and Ontologies (2)

- KAnnSpec XML example
- concept declaration for an OMDoc Symbol:

```
<concept name="Symbol" rdftype="o:Symbol">
  <documentation>An OpenMath/OMDoc Symbol</documentation>
  <field name="name" type="text" rdfpred="o:symbolname">
    <documentation>
       The name of the symbol defines it in a theory.
    </documentation>
    <value>Name</value>
    <default>Symbol</default>
    <validation>.*</validation>
    <number atleast="1" atmost="1"/>
  </field>
  <display>
    <template><b>Symbol:</b> <br/> {name}</template>
  </display>
</concept>
```

Generating RDF

- we need to be able to export the annotations
- we use RDF triples
- automatically generated from properties given in the KAnnSpec
- represent the annotation and how it is linked to the document

```
<rdf:RDF xmlns:o="http://omdoc.org/KAnnSpec#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:kat="https://github.com/KWARC/KAT/">

  <!-- omitted a lot of meta-information here -->

  <rdf:Description rdf:nodeID="KAT_1433087821332_4477">
    <kat:annotates
      rdf:resource="https://kwarc.github.io/KAT/content/sample1.html#cse(%2F%2F*%5B%
    %5D%2C%2F%2F*%5B%40id%3D'word.202'%5D%2C%2F%2F*%5B%40id%3D'word.203
    <kat:run rdf:nodeID="kat_run" />
    <kat:kannspec rdf:nodeID="KAT_1433087757661_OMDoc" />
    <kat:concept>Symbol</kat:concept>
    <kat:type rdf:resource="http://omdoc.org/KAnnSpec#Symbol" />
    <o:symbolname>semi-Euclidean space</o:symbolname>
  </rdf:Description>

</rdf:RDF>
```

Demo

Time for a Demo

`https://kwarc.github.io/KAT/`

Conclusion & Future Work

- KAT is a browser-based linguistic annotation tool for STEM documents in XHTML5 form
- UI can be instantiated to a given annotation ontology via a KAnnSpec
- tested with KAnnSpecs for OMDoc, POS, Stanford grammar
- Available under GPL3 at `http://github.com/KWARC/KAT`

Conclusion & Future Work

- KAT is a browser-based linguistic annotation tool for STEM documents in XHTML5 form
- UI can be instantiated to a given annotation ontology via a KAnnSpec
- tested with KAnnSpecs for OMDoc, POS, Stanford grammar
- Available under GPL3 at `http://github.com/KWARC/KAT`
- Future: improve implemenation
  - Better visualisation of annotations (in particular relations)
  - Optimize interactions/workflows (save clicks mouse moves)
  - Integration with CorTeX (for corpus management)
  - Test with more ontologies (KAnnSpecs) and documents
  - Stresstest in a multi-user setting

Conclusion & Future Work

- KAT is a browser-based linguistic annotation tool for STEM documents in XHTML5 form
- UI can be instantiated to a given annotation ontology via a KAnnSpec
- tested with KAnnSpecs for OMDoc, POS, Stanford grammar
- Available under GPL3 at `http://github.com/KWARC/KAT`
- Future: improve implemenation
  - Better visualisation of annotations (in particular relations)
  - Optimize interactions/workflows (save clicks mouse moves)
  - Integration with CorTeX (for corpus management)
  - Test with more ontologies (KAnnSpecs) and documents
  - Stresstest in a multi-user setting
- Perspective: use KAT as a lightweight OMDoc editor (annotate and harvest)