37. Jahrestagung der Deutschen Gesellschaft für Sprachwissenschaft - Sektion CL, Leipzig, 03.-06. März 2015

seit 1558

linktype.iaa.uni-jena.de/atomic

Atomic: An annotation platform to meet the demands of current and future research

Stephan Druskat[1,2], Volker Gast[2]
[1]Humboldt-Universität zu Berlin, Institut für deutsche Sprache und Linguistik
[2]Friedrich-Schiller-Universität Jena, Institut für Anglistik und Amerikanistik

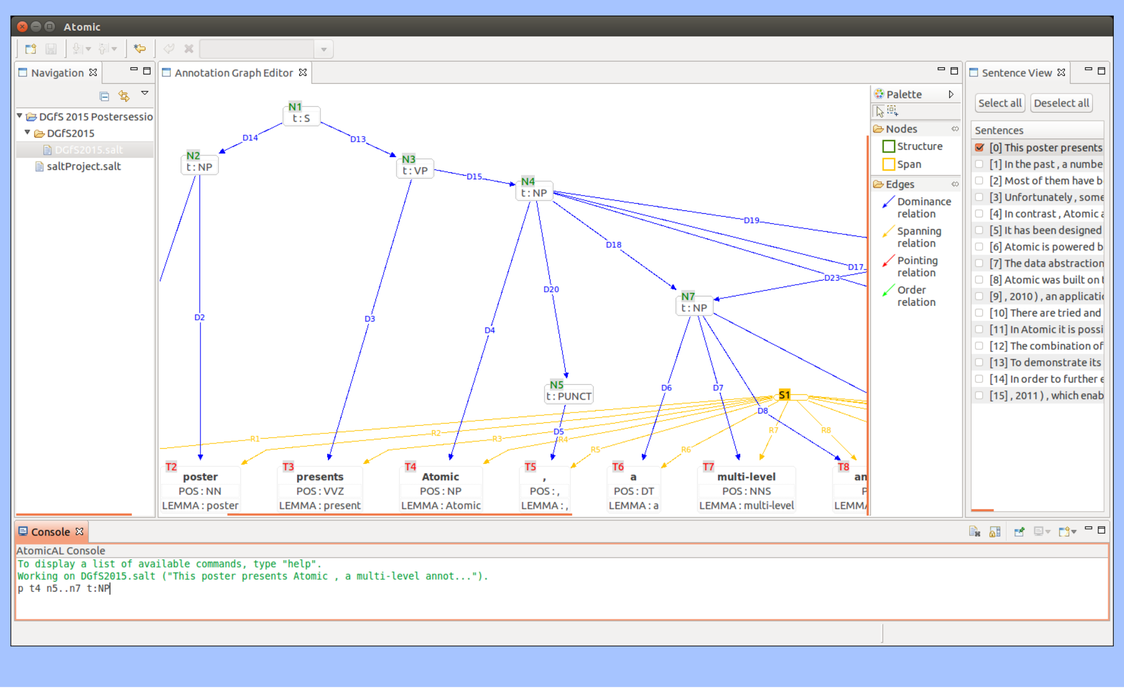
Fork me on GitHub
https://github.com/infraling/atomic

Motivation

- A number of annotation tools have been developed in linguistics in the past (e.g., Synpathy, RSTTool, MMAX2, etc.), usually within research projects, for a specific research question, or a specific annotation type (e.g., syntactic annotations, coreference chains)
- Development is often discontinued after the respective research project duration, which decreases the tools' usability over time
- In order to achieve a higher degree of re-usability and sustainability of tools, they must be *extensible*, *generic*, and *compatible* with other software
  - → **Extensibility** ensures that new functionality (e.g., editors) can be added to an existing tool, so that it can be employed for use cases not envisioned by its original creators
  - → **Genericity** allows for use of the tool for annotation types beyond its original scope, as well as across linguistic theories
  - → **Compatibility** secures the re-usability of a tool's output, as well as the ability to re-use other (legacy) tools' output as input

Atomic

- We are developing Atomic (Druskat et al., 2014; open source under the Apache License, Version 2.0) as both a standalone multi-level annotation tool and an adaptable tooling *platform* around the core specifications of extensibility, genericity, and compatibility



Generic data model

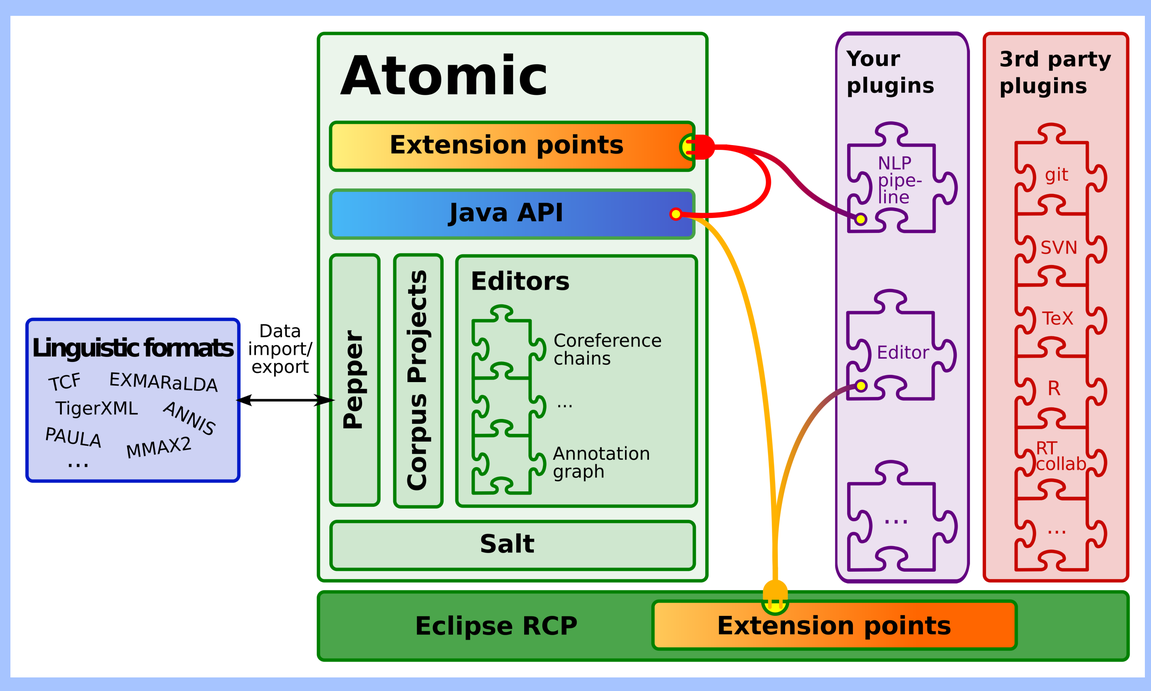
- Atomic works on instances of Salt (Zipser & Romary, 2010), a graph-based, theory-neutral, and semantic-free metamodel for linguistic data
- In Salt, nearly all conceivable kinds of linguistic structures can be modelled as nodes and edges: Tokens, spans, hierarchies and primary texts are represented as nodes, which can be connected by an unlimited number of edges to permit the creation of very diverse types of structures
- The data abstraction via Salt and the inclusion of a generic, graph-based editor working on Salt document graphs enable Atomic to handle potentially all types of annotations

Compatibility with other tools

- In order to establish compatibility with other software, Atomic includes the Pepper universal format conversion framework (Zipser et al., 2011)
- Pepper follows an intermediate model approach, with the Salt meta-model as the intermediate model, and importers and exporters for different formats
- Atomic virtually hooks into the Pepper conversion workflow, in that it works on the intermediate model
- Pepper provides Atomic with compatibility to formats such as EXMARaLDA, TigerXML, tiger2, PAULA, MMAX2, TCF, the ANNIS format, and many more, from which and into which Atomic can import and export

Extensibility

- Atomic is built on top of the Eclipse Rich Client Platform (*RCP*, McAffer et al., 2010), a well-established open-source Java application platform which operates on sets of plugins
- Atomic itself is a set of plugins, integrating with the Eclipse RCP
- Atomic provides extension points (currently for editors and NLP components) and a Java API
- Third parties can easily extend Atomic by creating new plugins, and connecting them with Atomic via the provided extension points (Eclipse plugin creation is well-documented, and additionally a large number of tutorials are available)



- Using these extension mechanisms you can easily write or wrap, e.g., a dependency parser, a morphological analyser, a syntax editor, or an editor for the annotation of historical corpora, and integrate it into Atomic
- Eclipse is tried and tested technology and is supported by a large community, making it a sustainable base for an extensible annotation tool
- Atomic can furthermore make use of the wide range of already existing third-party plugins, e.g.:
  - XML, TeX and R editors
  - version control system interfaces (SVN, Git, etc.: Using a version control system enables collaborative annotation as well as providing change management for corpora)
  - distributed real-time collaboration, etc.

Outlook

- Following feedback and optimization iterations, we plan to enhance Atomic's compatibility by consolidating and completing the integration of the latest Pepper import and export modules, as well as upgrading to the latest API
- Furthermore, and in order to increase the platform's extensibility, we plan to open Atomic's native annotation language *AtomicAL* (cf. Druskat et al., 2014) for extensions (“dialects”) that can be used with new editor types
- Additionally, we will implement further (prototypical) editors for specific annotation types to address a larger portion of the linguistic community

References

- **Druskat, S., L. Bierkandt, V. Gast, C. Rzymski & F. Zipser** (2014). Atomic: an open-source software platform for multi-level corpus annotation. In J. Ruppenhofer & G. Faaß (eds.): Proceedings of the 12th edition of the KONVENS conference, October 2014 (pp. 228–234).
- **McAffer, J., J.-M. Lemieux & C. Aniszczyk** (2010). Eclipse Rich Client Platform. 2nd edn. Addison-Wesley, Boston.
- **Zipser F. & L. Romary** (2010). A model oriented approach to the mapping of annotation formats using standards. In: Proceedings of the Workshop on Language Resource and Language Technology Standards, LREC 2010. Malta.
- **Zipser F., A. Zeldes, J. Ritz, L. Romary & U. Leser** (2011). Pepper: Handling a multiverse of formats. 33. Jahrestagung der Deutschen Gesellschaft für Sprachwissenschaft. Göttingen, 23.- 25. Februar 2011.