UIMA Ruta Workbench: Rule-based Text Annotation

Peter Kluegl[12] **Martin Toepfer**[2] **Philip-Daniel Beck**[2] **Georg Fette**[2] **Frank Puppe**[2]

[1]Comprehensive Heart Failure Center
University of Würzburg, Straubmühlweg 2a
Würzburg, Germany
pkluegl@uni-wuerzburg.de

[2]Department of Computer Science VI
University of Würzburg, Am Hubland
Würzburg, Germany
first.last@uni-wuerzburg.de

Abstract

UIMA Ruta is a rule-based system designed for information extraction tasks, but it is also applicable for many natural language processing use cases. This demonstration gives an overview of the UIMA Ruta Workbench, which provides a development environment and tooling for the rule language. It was developed to ease every step in engineering rule-based applications. In addition to the full-featured rule editor, the user is supported by explanation of the rule execution, introspection in results, automatic validation and rule induction. Furthermore, the demonstration covers the usage and combination of arbitrary components for natural language processing.

1 Introduction

Components for natural language processing and information extraction nowadays often rely on statistical methods and their models are trained using machine learning techniques. However, components based on manually written rules still play an important role in real world applications and especially in industry (Chiticariu et al., 2013). The reasons for this are manifold: The necessity for traceable results, the absence or aggravated creation of labeled data, or unclear specifications favor rule-based approaches. When the specification changes, for example, the complete training data potentially needs to be annotated again. In rule-based components, adaptions in a small selection of rules typically suffice. Rule-based approaches are also used in combination with or when developing statistical components. While models are often trained to solve one specific task in an application, the remaining parts need to be implemented as well. Furthermore, rules can be applied for high-level feature extraction and for semi-automatic creation of labeled data sets. It is often faster to define one rule for a specific pattern than to annotate repeating mentions of a specific type.

Unstructured Information Management Architecture (UIMA) (Ferrucci and Lally, 2004) is a framework for analyzing unstructured data and is extensively applied for building natural language processing applications. Two popular systems built upon UIMA are the DeepQA system Watson (Ferrucci et al., 2010) and the clinical Text Analysis and Knowledge Extraction System (cTAKES) (Savova et al., 2010). UIMA allows the definition of scalable pipelines of interoperable components called analysis engines, which incrementally add and modify meta information of documents mostly in form of annotations. The semantics and features of annotations are given by their types, which are specified in type systems.

This demonstration gives an overview on the UIMA Ruta Workbench (Kluegl et al., 2014), a development environment for the UIMA Ruta language. The rule language provides a compact representation of patterns while still supporting high expressivity necessary for solving arbitrary tasks. The UIMA Ruta Workbench includes additional tools that accelerate the efficient creation of components and complete pipelines. The user is supported in all aspects of the development process like specification of rules and type systems, debugging, introspection of the results, and quality assessment. UIMA Ruta is developed by an active community[1] and is released like UIMA under the industry-friendly Apache License 2.0.

[1] http://uima.apache.org/ruta.html

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations, pages 29–33, Dublin, Ireland, August 23-29 2014.

The rest of the paper is structured as follows: Section 2 compares UIMA Ruta to a selection of related systems. The rule language and the tooling support are introduced in Section 3. Section 4 provides an overview of the content of the demonstration and Section 5 concludes with a summary.

2 Related Systems

Rule-based systems for information extraction and natural language processing in general have a long history and thus we can only give a short comparison to a selection of related systems especially based on UIMA. The probably most noted rule-based system is JAPE (Cunningham et al., 2000). It is open source and integrated into the GATE (Cunningham et al., 2011) framework. JAPE propagates a clear separation of condition and action parts, and an aggregated execution of rules of one phase in a finite state transducer whereas UIMA Ruta allows freer positioning of actions for a more compact representation and applies the rules sequentially. Nevertheless, UIMA Ruta is able to compete well concerning performance and provides a high expressiveness. AFST (Boguraev and Neff, 2010) is also based on finite state transduction over annotations and additionally allows vertical patterns, which are also supported in UIMA Ruta. SystemT (Chiticariu et al., 2010) defines rules in declarative statements similar to SQL and applies them using an optimized operator graph. Our system takes first steps in this direction with the concept of variable matching directions, but still provides a compact language for more flexible operations. Other rule-based systems for UIMA are the IBM LanguageWare Resource Workbench[2], Zanzibar[3] and UIMA Regexp[4].

Most of the freely available rule-based systems provide only minimal development support. Good development environments and tooling are usually found in at least partially commercial systems. The development environment of SystemT (Chiticariu et al., 2011), for example, provides an editor with syntax highlighting and hyperlink navigation, an annotation provenance viewer, a contextual clue discoverer, regular expression learner, and a rule refiner. One intention of UIMA Ruta consists in providing strong tooling support that facilitates every step in the development process. The UIMA Ruta Workbench includes most features of related systems, but still introduces a few new and useful tools.

3 UIMA Ruta

UIMA Ruta (Rule-based Text Annotation) consists of a rule-based language interpreted by a generic analysis engine and of the UIMA Ruta Workbench, a development environment with additional tooling. Rules are sequentially applied and are composed of regular expressions of rule elements. The rule elements typically define the annotation type to be matched and an optional list of conditions and actions. The language provides a variety of diverse elements to elegantly solve arbitrary tasks. The rule matching supports overlapping alternatives and a coverage-based visibility. The following example illustrates the rule syntax:

```
(ANY{INLIST(MonthsList) -> Month} PERIOD? NUM{REGEXP(".{2,4}") -> Year}){-> Date};
```

This rule matches on any token present in an external dictionary *MonthsList* followed by an optional period and a number that contains two to four characters. If this pattern was recognized, then the actions create new annotations of the types *Month*, *Year* and *Date* for the corresponding matched segments. Following this, the rule identifies and annotates dates in the form of ‘Dec. 2004”, “July 85” or “11.2008”. Rule scripts can additionally include and apply external components, or specify additional types. A detailed description of the UIMA Ruta language can be found in the documentation[5].

The UIMA Ruta Workbench is an Eclipse-based[6] development environment for the rule language. A screenshot of the Workbench's main perspective is depicted in Figure 1. Rule scripts are organized in UIMA Ruta projects and take advantage of the well-known features of Eclipse like version control

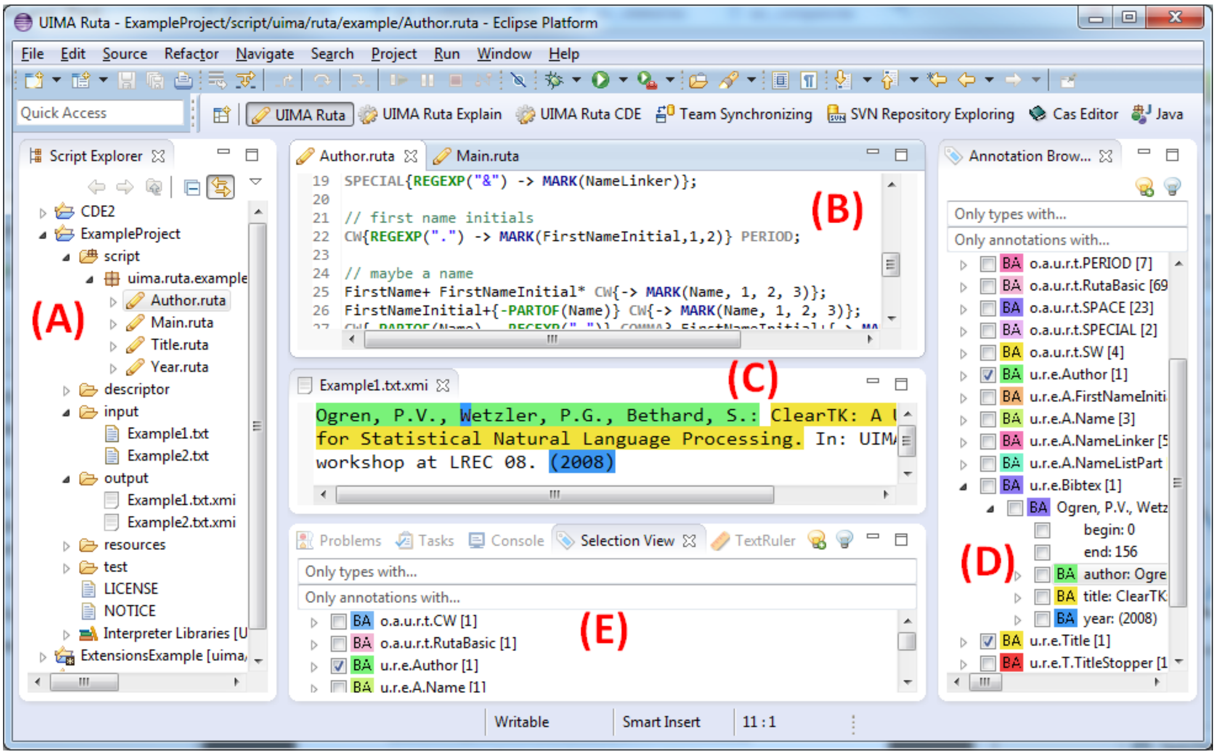
[2] http://www.alphaworks.ibm.com/tech/lrw

[3] https://code.google.com/p/zanzibar/

[4] https://sites.google.com/site/uimaregex/

[5] http://uima.apache.org/d/ruta-current/tools.ruta.book.html

[6] http://www.eclipse.org/



```
SPECIAL{REGEXP("&") -> MARK(NameLinker)};

// first name initials
CW{REGEXP(".") -> MARK(FirstNameInitial,1,2)} PERIOD;

// maybe a name
FirstName+ FirstNameInitial* CW{-> MARK(Name, 1, 2, 3)};
FirstNameInitial+{-PARTOF(Name)} CW{-> MARK(Name, 1, 2, 3)};
```

```
Ogren, P.V., Wetzler, P.G., Bethard, S.: ClearTK: A U
for Statistical Natural Language Processing. In: UIMA
workshop at LREC 08. (2008)
```

Figure 1: A selection of views in the UIMA Ruta Workbench: (A) Script Explorer with UIMA Ruta projects. Script files containing rules are indicated by the pencil icon. (B) Full-featured editor for specifying rules. (C) CAS Editor provided by UIMA for visualizing the results and manual creation of annotations. (D) Overview of annotations sorted by type. (E) Annotations overlapping the selected position in the active CAS Editor.

or search. The full featured editor for writing rules provides most of the characteristics known by editors for programming languages like instant syntax checking, syntactic and semantic highlighting, or context-sensitive auto-completion. An informative explanation of each step of the rule execution extended with profiling information helps to identify unintended behavior of the rules. The user is able to automatically evaluate the quality of rules on labeled documents and also on unlabeled documents using formalized background knowledge. Furthermore, the tools support pattern-based queries in collections of documents, semi-automatic creation of gold standards and different algorithms for supervised rule induction.

The rule language and the tooling are both extensible. The rule language can be enhanced with new actions, conditions, functions and even constructs that adapt aspects of the rule execution. Further functionality can be integrated by implementing supplementary analysis engines. The UIMA Ruta Workbench straightforwardly supports extensions due to its integration in Eclipse, e.g., new views can be added for improving specific development processes. Additional evaluation measures and rule learning algorithms are directly added by extension points. The Workbench also supports workspaces where the user develops interdependent Java and UIMA Ruta analysis engines simultaneously, and it seamlessly integrates components from Maven repositories.

The UIMA Ruta Workbench has been successfully utilized to develop diverse natural language applications. These include template-based information extraction in curricula vitae, segmentation of scientific references, or extraction of characters in novels, but also segmentation, chunking and relation extraction in clinical notes. Furthermore, the system is applied for feature extraction, creation of gold standards, and different pre- and post-processing tasks in combination with statistical models. The effectiveness of the Workbench can hardly be measured, but it is highlighted by the fact that several applications have been engineered in only a few hours of work.

4 Contents of Demonstration

The demonstration of the system concentrates on the general usage of the UIMA Ruta Workbench and how to develop rule-based analysis engines with it. We address the following use cases and aspects:

- **General rule engineering:** Simple examples are given for common tasks in UIMA Ruta, e.g., how to create new projects and script files, and how an initial set of rules is applied on a collection of documents. Several examples highlight the expressivity and effectiveness of the rule language.
- **Debugging erroneous rules:** Manual specification of rules is prone to errors. We will demonstrate the explanation component of the Workbench that facilitates the traceability of the rule matching and allows the identification of unintended behavior.
- **Definition of type systems and pipelines**: The UIMA Ruta language can be applied for textual definition of type systems and rule-based pipelines of arbitrary analysis engines, which enables rapid prototyping without switching tools.
- **Introspection of results:** The usage of the rule language as query statements allows the user to investigate different aspects of documents that have been annotated by arbitrary analysis engines, e.g., also based on statistical methods.
- **Quality assessment:** The automatic assessment of the analysis engines' quality is a central aspect in their development process. We demonstrate test-driven development using gold standard documents, and constraint-driven evaluation for unlabeled documents based on formalized background knowledge.
- **Document preprocessing:** The UIMA Ruta language can be applied for many tasks in the Workbench. These include different preprocessing steps like converting HTML files, anonymization, cutting paragraphs or rule-based document sorting.
- **Rule learning:** The provided rule induction algorithms based on boundary matching, extraction patterns and transformations are illustrated for simple examples.
- **Extension of the language:** Specific projects take advantage of specialized language elements. Extensions of the language and their seamless integration in the Workbench are shown with several examples.
- **Combinations with Java projects:** Some functionality can hardly be specified with rules, even with an extended language specification. Thus, examples provide insights how to make use of additional functionality implemented in Java.
- **Integration of external analysis engines:** Repositories like DKPro (Gurevych et al., 2007) or ClearTK (Ogren et al., 2008) provide a rich selection of well-known components for natural language processing. We demonstrate how these analysis engines and type systems can easily be integrated and how the user is able to specify rules based on the their results, e.g., for improving a part-of-speech tagger or for exploiting their annotations for relation extraction.
- **Semi-automatic annotation:** A semi-automatic process supported by rules can speed up the creation of gold documents. An exemplary use case will illustrate how the user can efficiently accept or reject the annotations created by rules.

A detailed description of these use cases will be available as part of the documentation of the project.

5 Conclusions

This demonstration presents the UIMA Ruta Workbench, a useful general-purpose tool for the UIMA community. The system helps to fill the gap of rule-based support in the UIMA ecosystem and is applicable for many different tasks and use cases. The user is optimally supported during the engineering process and is able to create complex and well-maintained applications as well as rapid prototypes. The UIMA Ruta Workbench is up-to-date unique concerning the combination of the provided features and tools, availability as open source, and integration in UIMA.

An interesting option for future work consists in making the functionality of UIMA Ruta and its tooling also available in web-based systems like the Argo UIMA platform (Rak et al., 2012).

Acknowledgements

This work was supported by the Competence Network Heart Failure, funded by the German Federal Ministry of Education and Research (BMBF01 EO1004), and is used for information extraction from medical reports and discharge letters.

References

Branimir Boguraev and Mary Neff. 2010. A Framework for Traversing dense Annotation Lattices. *Language Resources and Evaluation*, 44(3):183–203.

Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick R. Reiss, and Shivakumar Vaithyanathan. 2010. SystemT: An Algebraic Approach to Declarative Information Extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 128–137, Stroudsburg, PA, USA. Association for Computational Linguistics.

Laura Chiticariu, Vivian Chu, Sajib Dasgupta, Thilo W. Goetz, Howard Ho, Rajasekar Krishnamurthy, Alexander Lang, Yunyao Li, Bin Liu, Sriram Raghavan, Frederick R. Reiss, Shivakumar Vaithyanathan, and Huaiyu Zhu. 2011. The SystemT IDE: An Integrated Development Environment for Information Extraction Rules. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, pages 1291–1294, New York, NY, USA. ACM.

Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. 2013. Rule-based Information Extraction is Dead! Long Live Rule-based Information Extraction Systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 827–832, Seattle, Washington, USA, October. Association for Computational Linguistics.

Hamish Cunningham, Diana Maynard, and Valentin Tablan. 2000. JAPE: a Java Annotation Patterns Engine (Second Edition). Research Memorandum CS–00–10, Department of Computer Science, University of Sheffield, November.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6).*

David Ferrucci and Adam Lally. 2004. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Natural Language Engineering*, 10(3/4):327–348.

David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, 31(3):59–79.

Iryna Gurevych, Max Mühlhäuser, Christof Müller, Jürgen Steimle, Markus Weimer, and Torsten Zesch. 2007. Darmstadt Knowledge Processing Repository Based on UIMA. In *Proceedings of the UIMA Workshop at GLDV*, Tübingen, Germany, April.

Peter Kluegl, Martin Toepfer, Philip-Daniel Beck, Georg Fette, and Frank Puppe. 2014. UIMA Ruta: Rapid Development of Rule-based Information Extraction Applications. *Natural Language Engineering*. submitted.

Philip V. Ogren, Philipp G. Wetzler, and Steven Bethard. 2008. ClearTK: A UIMA Toolkit for statistical Natural Language Processing. In *UIMA for NLP Workshop at LREC.*

Rafal Rak, Andrew Rowley, William Black, and Sophia Ananiadou. 2012. Argo: an Integrative, Interactive, Text Mining-based Workbench Supporting Curation. *Database*, 2012:bas010.

Guergana K. Savova, James J. Masanz, Philip V. Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C. Kipper-Schuler, and Christopher G. Chute. 2010. Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, September.