Semantic Annotation Semantically: Using a Shareable Extraction Ontology and a Reasoner

Jan Dědek and Peter Vojtáš

Department of Software Engineering, Charles University,
Prague, Czech Republic
{dedek,vojtas}@ksi.mff.cuni.cz

Abstract. Information extraction (IE) and automated semantic annotation of text are usually done by complex tools and all these tools use some kind of a model that represents the actual task and its solution. The model is usually represented as a set of some kind of extraction rules (e.g. regular expressions), gazetteer lists or it is based on some statistical measurements and probability assertions. In the environment of the Semantic Web it is essential that information is shareable and some ontology based IE tools keep the model in so called extraction ontologies. In practice the extraction ontologies are usually strongly dependent on a particular extraction/annotation tool and cannot be used separately. In this paper we present an extension of the idea of extraction ontologies. The extraction ontologies should not be dependent on the particular extraction/annotation tool. In our solution the extraction/annotation process can be done separately by an ordinary reasoner. We also present a proof of a concept for the idea: a case study with a linguistically based IE engine that exports its extraction rules to an extraction ontology and we demonstrate how this extraction ontology can be applied to a document by a reasoner. The paper also contains an evaluation experiment with several OWL reasoners.

Keywords: Extraction Ontology, Reasoning, Information Extraction, Semantic Annotation

1 Introduction

Information extraction (IE) and automated semantic annotation of text are usually done by complex tools and all these tools use some kind of a model that represents the actual task and its solution. The model is usually represented as a set of some kind of extraction rules (e.g. regular expressions), gazetteer lists or it is based on some statistical measurements and probability assertions (classification algorithms like Support Vector Machines (SVM), Maximum Entropy Models, Decision Trees, Hidden Markov Models (HMM), Conditional Random Fields (CRF), etc.)

In the begging a model is either created by a human user or it is learned from a training dataset. Then, in an actual extraction/annotation process, the

model is used as a configuration or as an input parameter of the particular extraction/annotation tool. These models are usually stored in proprietary formats and they are accessible only by the corresponding tool.

In the environment of the Semantic Web it is essential that information is shareable and some ontology based IE tools keep the model in so called extraction ontologies. Extraction ontologies should serve as a wrapper for documents of a narrow domain of interest. When we apply an extraction ontology to a document, the ontology identifies objects and relationships present in the document and it associates them with the corresponding ontology terms and thus wraps the document so that it is understandable in terms of the ontology [6].

In practice the extraction ontologies are usually strongly dependent on a particular extraction/annotation tool and cannot be used separately. The strong dependency of an extraction ontology on the corresponding tool makes it very difficult to share. When an extraction ontology cannot be used outside the tool there is also no need to keep the ontology in a standard ontology format such as RDF[1] or OWL[2]. The only possibility how to use such extraction ontology is within the corresponding extraction tool. It is not necessary to have the ontology in a 'owl' or 'rdf' file. In a sense such extraction ontology is just a configuration file. For example in [9] (and also in [6]) the so called extraction ontologies are kept in XML files with a proprietary structure and it is absolutely sufficient, there is no need to treat them differently.

1.1 Shareable Extraction Ontologies

In this paper we present an extension of the idea of extraction ontologies. We adopt the point that extraction models are kept in extraction ontologies and we add that the extraction ontologies should not be dependent on the particular extraction/annotation tool. In such case the extraction/annotation process can be done separately by an ordinary reasoner.

In this paper we present a proof of a concept for the idea: a case study with our linguistically based IE engine and an experiment with several OWL reasoners. In the case study (see Section 3) the IE engine exports its extraction rules to the form of an extraction ontology. Third party linguistic tool linguistically annotates an input document and the linguistic annotations are translated to so-called document ontology. After that an ordinary OWL reasoner is used to apply the extraction ontology on the document ontology, which has the same effect as a direct application of the extraction rules on the document. In Section 4 we present an experiment with several OWL reasoner and IE datasets to verify feasibility of the idea.

2 Related Work

Ontology-based Information Extraction (OBIE) [13] or Ontology-driven Information Extraction [14] has recently emerged as a subfield of information extrac-

[1] `http://www.w3.org/RDF/`
[2] `http://www.w3.org/2001/sw/wiki/OWL`

tion. Also Web Information Extraction [2] is a closely related discipline. Many extraction and annotation tools can be found in the above mentioned surveys [13, 2] and many of the tools also use an ontology as an output format, but almost all of them store their extraction models in proprietary formats and the models are accessible only by the corresponding tool.

In the literature we have found only two approaches that use extraction ontologies. The former one was published by D. Embley [6, 5] and the later one IE system Ex [3] published by M. Labský [9]. But in both cases the extraction ontologies are dependent on the particular tool and they are kept in XML files with a proprietary structure.

Also authors of [13] (a recent survey of OBIE systems) do not agree with allowing for extraction rules to be a part of an ontology. They use two arguments against that:

1. Extraction rules are known to contain errors (because they are never 100% accurate), and objections can be raised on their inclusion in ontologies in terms of formality and accuracy.
2. It is hard to argue that linguistic extraction rules should be considered a part of an ontology while information extractors based on other IE techniques (such as SVM, HMM, CRF, etc. classifiers used to identify instances of a class when classification is used as the IE technique) should be kept out of it: all IE techniques perform the same task with comparable effectiveness (generally successful but not 100% accurate). But the techniques advocated for the inclusion of linguistic rules in ontologies cannot accommodate such IE techniques. They assert that either all information extractors (that use different IE techniques) should be included in the ontologies or none should be included.

Concerning the first argument, we have to take into account that extraction ontologies are not ordinary ontologies, it should be agreed that they do not contain 100% accurate knowledge. Also the estimated accuracy of the extraction rules can be saved in the extraction ontology and it can then help potential users to decide how much they will trust the extraction ontology.

Concerning the second argument, we agree that it is not always possible to save an extraction model to an ontology. But on the other hand we think that there are cases when shareable extraction ontologies can be useful. And maybe in the future; if this idea proves its usefulness; new standard ways how to encode models to an ontology will appear.

Also the most widely agreed definitions of an ontology emphasize the shared aspect of ontologies.

> An ontology is a formal specification of a shared conceptualization. [1]

> An ontology is a formal, explicit specification of a shared conceptualization. [12]

[3] `http://eso.vse.cz/~labsky/ex/`

Of course the word 'shareable' has different meaning form 'shared'. Something that is shareable is not necessarily shared, but on the other hand something that is shared should be shareable. We do not think that shareable extraction ontologies will contain shared knowledge about how to extract data form documents in certain domain. This is for example not true for all extraction models artificially leaned from a training corpus. Here shareable simply means that the extraction rules can be shred amongst software agents and can be used separately form the original tool. In time it will turn out if such extraction ontologies are useful or not. But for sure they bring something new that was not possible before.

3 The Main Idea Illustrated – Our Case Study

In this section we will describe the main idea of the paper and we will illustrate it with a case study.

3.1 Document Ontologies

The main idea of this paper assumes that extraction ontologies will be shareable and they can be applied on a document outside of the original extraction/annotation tool. We further assume that the extraction ontologies will be applied by ordinary reasoners. This assumption implies that both extraction ontologies and documents have to be in a reasoner readable format. In the case of contemporary OWL reasoners there are standard reasoner-readable languages: OWL and RDF in a rich variety of possible serializations (XML, Turtle, N-Triples, etc.) Besides that there exists standard ways like GRDDL[4] or RDFa[5] how to transform an ordinary document to a RDF document. We call the output of such transformation a 'document ontology'. A document ontology should contain all relevant data of a document and preferably the document could be reconstructed from the document ontology on demand.

When a reasoner is applying an extraction ontology to a document, it has only "to annotate" the corresponding document ontology. Here "to annotate" means to add new knowledge – new class membership or property assertions. In fact it means just to do the inference tasks prescribed by the extraction ontology on the document ontology.

3.2 Implementation

In this section we will present details abut the case study. We have used our IE engine [4] based on deep linguistic parsing and Inductive Logic Programming. It is a complex system implemented with a great help of the GATE system[6] [3] and it also uses many other third party tools including several linguistic tools a

[4] `http://www.w3.org/TR/grddl/`
[5] `http://www.w3.org/TR/xhtml-rdfa-primer/`
[6] `http://gate.ac.uk/`

Prolog system. Installation and making the system operate is not simple. This case study should demonstrate that the extraction rules produced by the system are not dependent on the system in the sense described above.

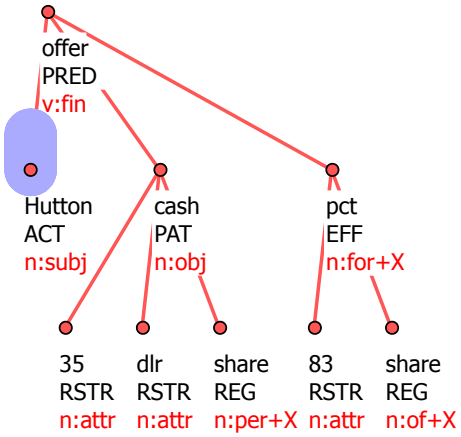

Fig. 1. Tectogrammatical tree of the sentence: "Hutton is offering 35 dlrs cash per share for 83 pct of the shares." Nodes roughly correspond with words of a sentence, edges represent linguistic dependencies between nodes and some linguistic features (tectogrammatical lemma, semantic functor and semantic part of speech) are printed under each node. The node 'Hutton' is docarated as a named entity.

Linguistic Analysis Our IE engine needs a linguistic preprocessing (deep linguistic parsing) of documents on its input. Deep linguistic parsing brings a very complex structure to the text and the structure serves as a footing for construction and application of extraction rules.

We usually use TectoMT system[7] [15] to do the linguistic preprocessing. TectoMT is a Czech project that contains many linguistic analyzers for different languages including Czech and English. We are using a majority of applicable tools from TectoMT: a tokeniser, a sentence splitter, morphological analyzers (including POS tagger), a syntactic parser and the deep syntactic (tectogrammatical) parser. All the tools are based on the dependency based linguistic theory and formalism of the Prague Dependency Treebank project [8] [8].

The output linguistic annotations of the TectoMT system are stored (along with the text of the source document) in XML files in so called Prague Markup Language[9] (PML). PML is a very complex language (or XML schema) that is able to express many linguistic elements and features present in text. For the

[7] `http://ufal.mff.cuni.cz/tectomt/`
[8] `http://ufal.mff.cuni.cz/pdt2.0/`
[9] `http://ufal.mff.cuni.cz/jazz/PML/`

IE engine a tree dependency structure of words in sentences is the most useful one because the edges of the structure guide the extraction rules. An example of such (tectogrammatical) tree structure is in Fig. 1.

In this case study PML files made form source documents by TectoMT are transformed to RDF document ontology by a quite simple GRDDL/XSLT[10] transformation. Such document ontology contains the whole variety of PML in RDF format.

Rule Transformations Extraction rules produced by the IE engine are natively kept in a Prolog format; examples can be seen in Fig. 2. The engine is capable to export them to the OWL/XML[11] syntax for rules in OWL 2 [7] (see in Fig. 4). Such rules can be parsed by OWL API[12] 3.1. Fig. 3 shows the example rules in Protégé[13] 4 – Rules View's format. And the last rule example can be seen in Fig. 5, which shows a rule in the Jena rules format [14]. Conversion to Jena rules was necessary because it is the only format that Jena can parse, see details about our use of Jena in Section4. The Jena rules were obtained using following process: OWL/XML → RDF/SWRL[15] conversion using OWL API and RDF/SWRL → Jena rules conversion using SweetRules[16].

[Rule 1] [Pos cover = 23 Neg cover = 6]
```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), tDependency(C,B), tDependency(C,D),
   formeme(D,'n:in+X'), tDependency(E,C).
```
[Rule 11] [Pos cover = 25 Neg cover = 6]
```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), tDependency(C,B), formeme(C,'n:obj'),
   tDependency(C,D), functor(D,'APP').
```
[Rule 75] [Pos cover = 14 Neg cover = 1]
```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), functor(B,'APP'), tDependency(C,B),
   number(C,pl).
```

Fig. 2. Examples of extraction rules in the native Prolog format.

[10] http://www.w3.org/TR/xslt
[11] http://www.w3.org/TR/owl-xmlsyntax/
[12] http://owlapi.sourceforge.net/
[13] http://protege.stanford.edu/
[14] http://jena.sourceforge.net/inference/#RULEsyntax
[15] http://www.w3.org/Submission/SWRL/
[16] http://sweetrules.semwebcentral.org/

[Rule 1]
```
lex.rf(?b, ?a), t_lemma(?b, "Inc"), tDependency(?c, ?b),
tDependency(?c, ?d), formeme(?d, "n:in+X"), tDependency(?c, ?e)
      -> mention_root(?a, "acquired")
```
[Rule 11]
```
lex.rf(?b, ?a), t_lemma(?b, "Inc"), tDependency(?c, ?b),
formeme(?c, "n:obj"), tDependency(?c, ?d), functor(?d, "APP")
      -> mention_root(?a, "acquired")
```
[Rule 75]
```
lex.rf(?b, ?a), t_lemma(?b, "Inc"), functor(?b, "APP"),
tDependency(?c, ?b), number(?c, "pl")
      -> mention_root(?a, "acquired")
```

Fig. 3. Examples of extraction rules in Protégé 4 – Rules View's format

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Ontology [
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY pml "http://ufal.mff.cuni.cz/pdt/pml/" >
]>
<Ontology xmlns="http://www.w3.org/2002/07/owl#"
  ontologyIRI="http://czsem.berlios.de/onto ... rules.owl">
  <DLSafeRule>
    <Body>
      <ObjectPropertyAtom>
        <ObjectProperty IRI="&pml;lex.rf" />
        <Variable IRI="urn:swrl#b" />
        <Variable IRI="urn:swrl#a" />
      </ObjectPropertyAtom>
...
      <DataPropertyAtom>
        <DataProperty IRI="&pml;number" />
        <Variable IRI="urn:swrl#c" />
        <Literal>pl</Literal>
      </DataPropertyAtom>
    </Body>
    <Head>
      <DataPropertyAtom>
        <DataProperty IRI="&pml;mention_root" />
        <Literal>acquired</Literal>
        <Variable IRI="urn:swrl#a" />
      </DataPropertyAtom>
    </Head>
  </DLSafeRule>
</Ontology>
```

Fig. 4. Rule 75 in the OWL/XML syntax for Rules in OWL 2 [7].

```
@prefix pml: <http://ufal.mff.cuni.cz/pdt/pml/>.
[rule-75:
        ( ?b pml:lex.rf ?a )
        ( ?c pml:tDependency ?b )
        ( ?b pml:functor 'APP' )
        ( ?c pml:number 'pl' )
        ( ?b pml:t_lemma 'Inc' )
     ->
        ( ?a pml:mention_root 'acquired' )
]
```

Fig. 5. Rule 75 in the Jena rules syntax.

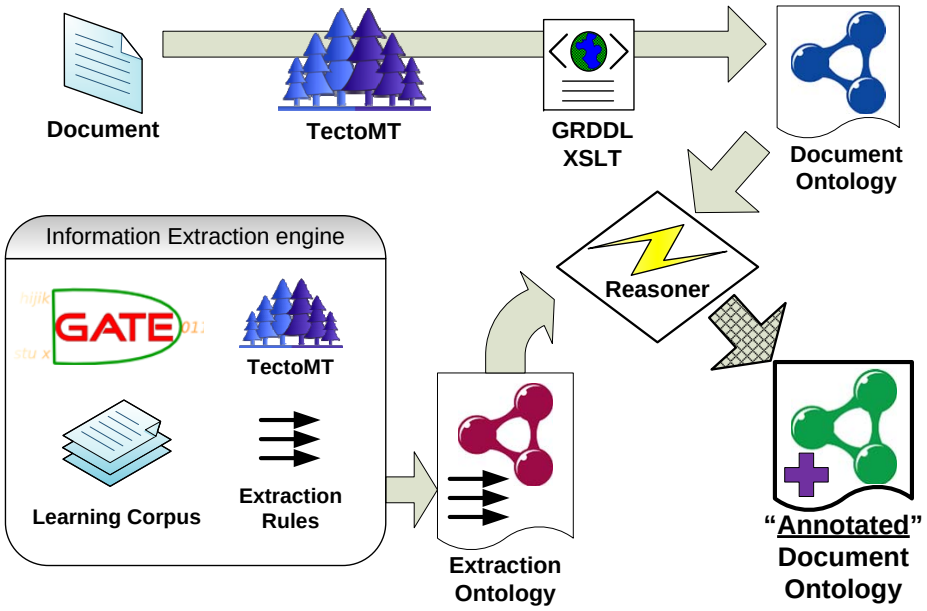

Fig. 6. Semantic annotation driven by an extraction ontology and a reasoner – schema of the process.

Schema of the Case Study Schema of the case study can be seen in Fig. 6.

The top row of the image illustrates how TectoMT (third party linguistic tool) linguistically annotates an input document and the linguistic annotations are translated to so-called document ontology by a GRDDL/XSLT transformation.

In the bottom of the picture our IE engine learns extraction rules and exports them to an extraction ontology. The reasoner in the middle is used to apply the extraction ontology on the document ontology and it produces the "annotated" document ontology, which was described in Section 3.1.

How to Download All the resources (including source codes of the case study and the experiment) mentioned in this demonstration are publically available on the web-page of our project[17] and detailed information can be found there.

4 Experiment

In this section we present an experiment that should serve as a proof of a concept that the proposed idea of independent extraction ontologies is working. We have selected several reasoners (namely Jena, HermiT, Pellet and FaCT++) and tested them on two slightly different datasets from two different domains and languages (see Table 1). This should at least partially demonstrate the universality of the proposed approach.

In both cases the task is to find all instances (corresponding to words in a document) that should be uncovered by the extraction rules. The extraction rules are saved in single extraction ontology for each dataset. The datasets are divided into individual document ontologies (owl files) corresponding to the individual documents of a source IE dataset (see below). During an experiment the individual document ontologies are processed separately (one ontology in a step) by a selected reasoner. The total time taken to process all document ontologies of a dataset is the measured result of the experiment.

The actual reasoning task is more difficult than a simple retrieval of all facts entailed by the extraction rules. Such simple retrieval task took only a few seconds for the Acquisitions v1.1 dataset (including parsing) in the native Prolog environment that the IE engine uses. There were several more inferences needed in the reasoning task because the schema of the input files was a little bit different form the schema used in rules. The mapping of the schemas was captured in another ontology that was included in the reasoning task. The mapping ontology is a part of the publically available project ontologies [18] and a potentially interested reader can find the complete mapping there.

4.1 Datasets

In the experiment we used two slightly different datasets from two different domains and languages. Table 1 summarizes some basic information about them.

[17] `http://czsem.berlios.de/`

[18] See "Data → ontologies" link on the project page `http://czsem.berlios.de/`

dataset	domain	language	number of files	dataset size	number of rules
czech_fireman	accidents	Czech	50	16 MB	2
acquisitions-v1.1	finance	English	600	126 MB	113

Table 1. Description of datasets that we have used.

czech_fireman The fist dataset is called 'czech_fireman'. This dataset was created by ourselves during the development of our IE engine. It is a collection of 50 Czech texts that are reporting on some accidents (car accidents and other actions of fire rescue services). These reports come from the web of Fire rescue service of Czech Republic[19]. The labeled corpus is publically available on the web of our project[20]. The corpus is structured such that each document represents one event (accident) and several attributes of the accident are marked in text. For the experiment we selected the 'damage' task – to find an amount (in CZK - Czech Crowns) of summarized damage arisen during a reported accident.

Acquisitions v1.1 The second dataset is called "Corporate Acquisition Events" corpus and it is described in [10]. More precisely we use the *Acquisitions v1.1* version[21] of the corpus. This is a collection of 600 news articles describing acquisition events taken from the Reuters dataset. News articles are tagged to identify fields related to acquisition events. These fields include 'purchaser' , 'acquired', and 'seller' companies along with their abbreviated names ('purchabr', 'acqabr' and 'sellerabr') Some news articles also mention the field 'deal amount'. For the experiment we selected only the 'acquired' task.

4.2 Reasoners

In the experiment we used four OWL reasoners (namely Jena[22] ,HermiT[23] ,Pellet[24] and FaCT++[25]) and measured the time they needed to complete a particular task. The time also includes time spend on parsing the input. HermiT, Pellet and FaCT++ were called through OWLAPI-3.1, so the same parser was used for them. Jena reasoner was used in its native environment and used the Jena parser.

In the early beginning of the experiment we had to exclude the FaCT++ reasoner form all tasks. It tuned out that FaCT++ probably does not work with rules in our setting (We called FaCT++ through OWLAPI-3.1) because it does

[19] `http://www.hzscr.cz/hasicien/`
[20] `http://czsem.berlios.de/`
[21] This version of the corpus comes form the Dot.kom (Designing infOrmation extracTion for KnOwledge Management) project's resources: `http://nlp.shef.ac.uk/dot.kom/resources.html`
[22] `http://jena.sourceforge.net`
[23] `http://hermit-reasoner.com`
[24] `http://clarkparsia.com/pellet`
[25] `http://code.google.com/p/factplusplus`

not returned any result instances. All the remaining reasoners strictly agreed on the results and returned the same sets of instances.

Also HermiT was not fully evaluated on the Acquisitions v1.1 dataset because it was too slow. The reasoner spent 13 hours of running to process only 30 of 600 files of the dataset. And we did not find it useful to let it continue.

4.3 Evaluation Results of the Experiment

reasoner	**czech_fireman**	stdev	**acquisitions-v1.1**	stdev
Jena	161 s	0.226	1259 s	3.579
HermiT	219 s	1.636	$\gg$ 13 hours	
Pellet	11 s	0.062	503 s	4.145
FaCT++	Did not work with rules in our setting.			

Table 2. Time performance of tested reasoners on both datasets. Time is measured in seconds. Average values from 6 measurements. Experiment environment: Intel Core I7-920 CPU 2.67GHz, 3GB of RAM, Java SE 1.6.0_03, Windows XP.

Table2 summarizes results of the experiment. The standard deviations are relatively small when compared to the differences between the average times. So there is no doubt about the order of the tested reasoners. Pellet performed the best and HermiT was the slowest amongst the tested and usable reasoners in this experiment.

From the results we can conclude that similar tasks can be satisfactorily solved by contemporary reasoners because three of four tested reasoners were working in the presented tasks and two reasoners finished in operational time.

4.4 Repeatability

Our implementation is publicly available – source codes and the datasets can be downloaded from our project's web-page[26], so it should be also possible to repeat the experiment in a sense of the SIGMOD Experimental Repeatability Requirements [11].

5 Future Work

In this paper (Section 3.1) we have described a method how to apply an extraction ontology to a document ontology and obtain so called "annotated" document ontology. To have an "annotated" document ontology is almost the same as to have an annotated document. An annotated document is useful (easier

[26] `http://czsem.berlios.de/`

navigation, faster reading and lookup of information, possibility of structured queries on collections of such documents, etc.) but if we are interested in the actual information present in the document, if we want to know the facts that are in a document asserted about the real word things then an annotated document is not sufficient and the conversion of an annotated document to the real world facts is no simple. There are obvious issues concerning data integration and duplicity of information. For example when in a document two mentions of people are annotated as 'injured', what is the number of injured people in the corresponding accident? Are the two annotations in fact linked to the same person or not?

In the beginning of our work on the idea of shareable extraction ontologies we planned to develop it further, we wanted to cover also the step form annotated document ontologies to the real world facts. The extraction process would then end up with so called "fact ontologies". But two main obstacles prevent us to do that.

1. Our IE engine is not yet capable to solve these data integration and duplicity of information issues. The real world facts would be quite imprecise then.
2. There are also technology problems of creating new facts (individuals) during reasoning.

Because of the decidability and finality constraints of the Description Logic Reasoning task it is not possible to create new individuals during the reasoning process. There is no standard way how to do it. But there are some proprietary solutions like `swrlx:createOWLThing`[27] form the Protégé project and `makeTemp(?x)` or `makeInstance(?x, ?p, ?v)`[28] from the Jena project. And these solutions can be used in the future work.

6 Conclusion – the Main Contributions

In the end of the paper we would like to summarize the main contributions of the paper.

- In the beginning of the paper we pointed out the draw back of so called extraction ontologies – in most cases they are dependent on a particular extraction/annotation tool and they cannot be used separately.
- We extended the concept of extraction ontologies by adding the shareable aspect and we introduced a new principle of making extraction ontologies independent of the original tool: the possibility of application of an extraction ontology to a document by an ordinary reasoner.
- In Section 3 we presented a case study that shows that the idea of shareable extraction ontologies is realizable. We presented implementation of an

[27] `http://protege.cim3.net/cgi-bin/wiki.pl?action=browse&id=SWRLExtensionsBuiltIns`

[28] `http://jena.sourceforge.net/inference/#RULEbuiltins`

IE tool that exports its extraction rules to an extraction ontology and we demonstrated how this extraction ontology can be applied to a document by a reasoner.
- More over in Section 4 an experiment with several OWL reasoners was presented. The experiment evaluated the performance of contemporary OWL reasoners on tasks of application of extraction ontologies.
- A new publically available benchmark for OWL reasoning was created together with the experiment. Other reasoner can be tested this way.

We would like to conclude the paper by stating that even if the fundamental idea of the paper will not set up in the community it is at least a new use case for both: usage of IE tools and reasoners.

Acknowledgments This work was partially supported by Czech projects: GACR P202/10/0761, GACR-201/09/H057, GAUK 31009 and MSM-0021620838.

References

1. Borst, W.N.: Construction of Engineering Ontologies for Knowledge Sharing and Reuse. Ph.D. thesis, Universiteit Twente, Enschede (September 1997), http://doc.utwente.nl/17864/
2. Chang, C.H., Kayed, M., Girgis, M.R., Shaalan, K.F.: A survey of web information extraction systems. IEEE Trans. Knowl. Data Eng. 18(10), 1411–1428 (2006)
3. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proceedings of the 40th Anniversary Meeting of the ACL (2002)
4. Dědek, J.: Towards semantic annotation supported by dependency linguistics and ILP. In: Proceedings of the 9th International Semantic Web Conference (ISWC2010), Part II. Lecture Notes in Computer Science, vol. 6497, pp. 297–304. Springer-Verlag Berlin Heidelberg, Shanghai / China (2010), http://iswc2010.semanticweb.org/accepted-papers/219
5. Embley, D.W.: Toward semantic understanding: an approach based on information extraction ontologies. In: Proceedings of the 15th Australasian database conference - Volume 27. pp. 3–12. ADC '04, Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2004), http://portal.acm.org/citation.cfm?id=1012294.1012295
6. Embley, D.W., Tao, C., Liddle, S.W.: Automatically extracting ontologically specified data from html tables of unknown structure. In: Spaccapietra, S., March, S.T., Kambayashi, Y. (eds.) ER. Lecture Notes in Computer Science, vol. 2503, pp. 322–337. Springer (2002)
7. Glimm, B., Horridge, M., Parsia, B., Patel-Schneider, P.F.: A Syntax for Rules in OWL 2. In: Proceedings of the 6th International Workshop on OWL: Experiences and Directions (OWLED 2009). vol. 529. CEUR (2009)
8. Hajič, J., Hajičová, E., Hlaváčová, J., Klimeš, V., Mírovský, J., Pajas, P., Štěpánek, J., Vidová-Hladká, B., Žabokrtský, Z.: Prague dependency treebank 2.0 cd–rom. Linguistic Data Consortium LDC2006T01, Philadelphia 2006 (2006)

9. Labský, M., Svátek, V., Nekvasil, M., Rak, D.: The Ex Project: Web Information Extraction Using Extraction Ontologies. In: Berendt, B., Mladenic, D., de Gemmis, M., Semeraro, G., Spiliopoulou, M., Stumme, G., Svátek, V., Železný, F. (eds.) Knowledge Discovery Enhanced with Semantic and Social Information, Studies in Computational Intelligence, vol. 220, pp. 71–88. Springer Berlin / Heidelberg (2009), http://dx.doi.org/10.1007/978-3-642-01891-6_5
10. Lewis, D.: Representation and learning in information retrieval. Ph.D. thesis, University of Massachusetts (1992)
11. Manolescu, I., Afanasiev, L., Arion, A., Dittrich, J., Manegold, S., Polyzotis, N., Schnaitter, K., Senellart, P., Zoupanos, S., Shasha, D.: The repeatability experiment of sigmod 2008. SIGMOD Rec. 37(1), 39–45 (2008)
12. Studer, R., Benjamins, V.R., Fensel, D.: Knowledge engineering: Principles and methods. Data & Knowledge Engineering 25(1-2), 161 – 197 (1998), http://www.sciencedirect.com/science/article/B6TYX-3SYXJ6S-G/2/67ea511f5600d90a74999a9fef47ac98
13. Wimalasuriya, D.C., Dou, D.: Ontology-based information extraction: An introduction and a survey of current approaches. Journal of Information Science 36(3), 306–323 (June 2010), http://dx.doi.org/10.1177/0165551509360123
14. Yildiz, B., Miksch, S.: ontoX - a method for ontology-driven information extraction. In: Proceedings of the 2007 international conference on Computational science and its applications - Volume Part III. pp. 660–673. ICCSA'07, Springer-Verlag, Berlin, Heidelberg (2007), http://portal.acm.org/citation.cfm?id=1793154.1793216
15. Žabokrtský, Z., Ptáček, J., Pajas, P.: TectoMT: Highly modular MT system with tectogrammatics used as transfer layer. In: Proceedings of the 3rd Workshop on Statistical Machine Translation. pp. 167–170. ACL, Columbus, OH, USA (2008)