OntoNERdIE – Mapping and Linking Ontologies to Named Entity Recognition and Information Extraction Resources

Ulrich Schäfer

German Research Center for Artificial Intelligence (DFKI), Language Technology Lab
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
email: ulrich.schaefer@dfki.de

Abstract

We describe an implemented offline procedure that maps OWL/RDF-encoded ontologies with large, dynamically maintained instance data to named entity recognition (NER) and information extraction (IE) engine resources, preserving hierarchical concept information and links back to the ontology concepts and instances. The main motivations are (i) improving NER/IE precision and recall in closed domains, (ii) exploiting linguistic knowledge (context, inflection, anaphora) for identifying ontology instances in texts more robustly, (iii) giving full access to ontology instances and concepts in natural language processing (NLP) results, e.g. for subsequent ontology queries, navigation or inference, (iv) avoiding duplication of work in development and maintenance of similar resources in independent places, namely lingware and ontologies. We show an application in hybrid deep-shallow NLP that is e.g. used for question analysis in closed domains. Further applications could be automatic hyperlinking or other innovative semantic-web related applications.

1. Introduction and Motivation

Ontologies on the one hand and resources for natural language processing (lingware) on the other hand, though closely related, are often maintained independently, thus constituting duplication of work. In this paper, we describe an implemented offline procedure based on XSLT that can be used to map concepts and instance information from ontologies to lingware resources for named entity recognition and information extraction systems.

The advantages of this approach for semantic web and natural language processing-based applications come from a 'cross-fertilisation' effect. While ontology instance data can improve precision and recall of e.g. named entity recognition (NER) and information extraction (IE) in closed domains, linguistic knowledge contained in NER and IE components can help to recognise ontology instances (or concepts) occuring in text, e.g. by taking into account context, inflection and anaphora.

If both resources would be managed jointly at a single place (in the ontology), they could be easily kept up-to-date and in sync, and their maintenance would be less time-consuming. When ontology concepts and instances are recognised in text, their name or ID can be used by applications to support subsequent queries, navigation or inference in the ontology using an ontology query language. The procedure we describe preserves hierarchical concept information and links back to the ontology concepts and instances.

Applications are e.g. hybrid deep-shallow question answering (Frank et al., 2006), automatic typed hyperlinking (Busemann et al., 2003) of instances and concepts occurring in documents, or other innovative applications that combine semantic web and natural language processing technologies.

The approach has been implemented for the ontology on language technology that works at the backend of the LT World web portal (Uszkoreit et al., 2003)[1], but could be easily adapted to other domains and ontologies, because it is already almost fully automated, except for the choice of relevant concepts and properties to map which is a matter of configuration.

The target named entity recognition and information extraction tool we employed is SProUT[2] (Drożdżyński et al., 2004), a shallow multilingual, multi-purpose natural language processor.

The advantage of SProUT in the described approach for named entity recognition and information extraction is that it comes with (1) a type system and typed feature structures as basic data structures[3], (2) a powerful, declarative rule mechanism with regular expressions over typed feature structures, (3) a highly efficient gazetteer module with fine-grained, customisable classification of recognised entities (Piskorski, 2005).

Moreover, SProUT provides additional modules like morphology or a reference resolver, that can be exploited in the rule system, e.g. to use context or morphological variation for improved NER.

The SProUT runtime component has been integrated as NER and IE component into the Heart of Gold (Callmeier et al., 2004), a middleware architecture for the integration of shallow and deep natural language processing components. Through automatically generated mappings, SProUT output enriched with ontology information can be used for robust, hybrid deep-shallow parsing and semantic analysis.

In Section 2., we describe the XSLT-based mapping process. In Section 3., we present an example how the recognised named entities enriched with ontology information can be used in hybrid natural language processing and subsequent applications. Finally, we conclude and give an outlook to future extensions.

[1] http://www.lt-world.org

[2] SProUT stands for Shallow Processing with Unification and Typed feature structures.

[3] The SProUT formalism uses a subset of TDL (Krieger and Schäfer, 1994), but with a closed type world and strict welltypedness and appropriateness conditions.

2. The OntoNERdIE procedure

In this section, we describe the processing steps of the OntoNERdIE approach (the offline part depicted in Figure 1 on the left; the right, online part is described in Section 3.). Following a general motivation presented in

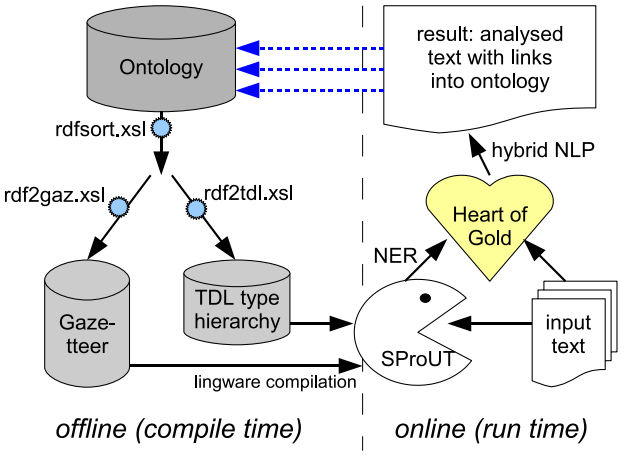

Figure 1: OntoNERdIE flow of information.

(Schäfer, 2003), the approach heavily relies on XSLT transformation of the XML representation formats, both in the offline mapping and in the online application. XSLT (Clark, 1999) is an XML transformation language and W3C standard. It can be used to transform XML documents with known structure to other XML formats or to syntaxes different from XML. In our case, the transformation is an offline mapping from RDF/OWL[4] representation of the ontology to component-specific formats for gazetter entries and type hierarchy.

2.1. RDF preprocessing

Input to the mapping procedure is an OWL ontology file containing both concept and instance descriptions. Figure 2 shows a (shortened) example for the instance `LREC 2006` in the LT World ontology. To ease stylesheet development, the current implementation requires the file to be in the unabbreviated RDF syntax (no QName abbreviations for instances etc.) for the subsequent processing steps. I.e., instead of the abbreviated

```
<Active_Person rdf:ID="obj_72976"> ...
</Active_Person>
```

the full, unabbreviated description syntax has to be used:

```
<rdf:Description rdf:about="http://www.lt-
    world.org/ltw.owl#obj_72976">
 <rdf:type rdf:resource="http://www.lt-
    world.org/ltw.owl#Active_Person"/> ...
</rdf:Description>
```

A further preprocessing step might be necessary that inserts explicit statements where only implicit statements are encoded in the OWL file, e.g. for `rdfs:subClassOf`. This is because for efficiency reasons, the subsequent stylesheets (in the current implementation) will not track implicit information. This could however be done during preprocessing through systems like Sesame[5] that support forward-chaining inference rules generating the missing statements. However, as typically not the full ontology will be mapped to NER/IE resources, a sufficient solution would be typically to enumerate all relevant concepts as part of the configuration of the mapping stylesheets described in Sections 2.3. and 2.4.

2.2. Grouping and sorting rdf:Descriptions

The resulting RDF file is processed with a small but sophisticated XSLT stylesheet (`rdfsort.xsl`; cf. Figure 3). This is a necessary prerequisite for the subsequent extraction steps, and, as it cannot be implemented by a simple XSLT `sort` statement, has to be coded as a proper, dedicated transformation. The stylesheet groups together `rdf:Descriptions` that are distributed over the file but belong together by using the `key` and `sort` statements and the `generate-id()` function.
The next two processing stages take a list of concepts as filter because, depending on the application, it will typically not be desirable to extract all concepts or instances available in the ontology. In both cases, resource files are generated as output that can be used to extend existing named entity recognition resources. E.g., while general rules can recognise domain-independent named entities (e.g. any person name), the extended resource contain specific, and potentially more detailed information on domain-specific entities.

2.3. Extracting inheritance statements and converting to TDL type definitions

The second stylesheet (`rdf2tdl.xsl`) converts the RDF `subClassOf` statements from the output of step 2 (Section 2.2.) into a set of TDL type definitions that can be immediatlely imported by the SProUT NER grammar, e.g. currently 1260 type definitions for the same number of `subClassOf` statements in the LT World ontology.
Following are two examples.

```
Active_Conference :=
  Conferences & Backend_Events.

Natural_Language_Parsing :=
  Written_Language & Language_Analysis.
```

This is of course a lossy conversion because not all relations supported by an OWL (DL or full) ontology such as `unionOf`, `disjointWith`, `intersectionOf`, etc. are mapped. However, we think that for named entity (NE) classifications, the `subClassOf` taxonomy mappings will be sufficient. Other relations could be formulated as direct (though slower) ontology queries using the `OBJID` mechanism described in the next step.
If the target of OntoNERdIE would be a NER system different from SProUT and without type hierarchy, then this step can be omitted. The `subClassOf` information can

[4] http://www.w3.org/RDF/, http://www.w3.org/2004/OWL/

[5] `http://www.openrdf.org`; for details, cf. (Frank et al., 2006). Sesame can also be used to produce the unabbreviated RDF format from QName-abbreviated OWL syntax.

```
<rdf:Description rdf:about="http://www.lt-world.org/ltw.owl#obj_89404">
 <rdf:type rdf:resource="http://www.lt-world.org/ltw.owl#Active_Conference"/>
 <dc_keyword rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Machine
    Translation</dc_keyword>
 <dc_keyword rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Semantic Web
    </dc_keyword>
 <dc_keyword rdf:datatype="http://www.w3.org/2001/XMLSchema#string">NLP Tools</dc_keyword>
 <homepageURL rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
    http://www.lrec-conf.org/lrec2006/</homepageURL>
 <dateStart rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2006-05-24</dateStart>
 <dateEnd rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2006-05-26</dateEnd>
 <paperDeadline rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2005-10-14
    </paperDeadline>
 <eventNameVariant rdf:datatype="http://www.w3.org/2001/XMLSchema#string">LREC 2006
    </eventNameVariant>
 <takesPlaceInCountry
    rdf:resource="http://www.lt-world.org/ltw.owl#lt-world_Individual_334"/>
 <eventNameVariant rdf:datatype="http://www.w3.org/2001/XMLSchema#string">5th Conference
    on Language Resources and Evaluation</eventNameVariant>
 <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">5th Conference on Language
    Resources and Evaluation</name>
 <locatedIn rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Magazzini del Cotone
    Conference Center, Genoa</locatedIn>
 <eventName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">5th Conference on
    Language Resources and Evaluation</eventName>
 <eventNameAbbreviation rdf:datatype="http://www.w3.org/2001/XMLSchema#string">LREC 2006
    </eventNameAbbreviation>
</rdf:Description>
```

Figure 2: LT World ontology entry for `LREC 2006` (shortened).

always be gained by querying the ontology appropriately on the basis of the concept name.

2.4. Generating gazetteer entries

The next stylesheet (`rdf2gaz.xsl`) selects statements about instances of relevant concepts via the `rdf:type` information and converts them to structured gazetteer source files for the SProUT gazetteer compiler (or into a different format for other NER systems). In the following example, two of the approx. 20000 converted entries for LT World are shown.

```
Martin Kay | GTYPE: lt_person |SNAME:"Kay"
  | GNAME: "Martin" |CONCEPT: Active_Person
  | OBJID: "obj_65046"

LREC 2006  | GTYPE: lt_event | GABBID:
  "LREC 2006" | CONCEPT: Active_Conference
  | OBJID: "obj_89404"
```

The attribute `CONCEPT` contains the TDL type mapped in step 3 (described in Section 2.3.). For convenience, several ontology concepts are mapped (defined manually as part of the configuration of the stylesheet) to only a few named entity classes (under attribute `GTYPE`). For LT World, these classes are person, organisation, event, project, product and technology. This has the advantage that NER context rules from existing SProUT named entity grammars can be reused[6] for better robustness and disambiguation.

[6]Alternatively, a fully automatic, but maybe too fine-grained 1:1 mapping of all concepts could be performed.

The rules e.g. recognise name variants with title like Prof. Kay, Dr. Kay, Mr. Kay with or without firstname. Moreover, context (e.g. prepositions with location names, verbs), morphology and reference resolution information can be exploited in these rules.

The following SProUT rule (XTDL syntax) simply copies the slots of a matched gazetteer entry for events (e.g. a conference) to the output as a recognised named entity.

```
lt-event :> gazetteer & [ GTYPE lt_event,
     SURFACE #name, CONCEPT #concept,
     OBJID #objid, GABBID #abbrev ]
-> ne-event &
     [ EVENTNAME #name, CONCEPT #concept,
       OBJID #objid, GABBID #abbrev ].
```

`OBJID` contains the object identifier of the instance in the ontology. It can be used as link back to the full knowledge stored in the ontology, e.g. for subsequent queries, like 'Who else participated in project [with OBJID obj_4789]?' etc.

In case multiple instances with same names but different object IDs occur in the ontology (which actually happens to be the case in LT World), then multiple alternatives are generated as output which is probably the expected and desired behavior (e.g. for frequent names like John Smith). On the other hand, if product names or event name with an abbreviated variant exist in the ontology, they both point to the same object ID (provided they are stored appropriately in the ontology).

The overall processing time (steps 1-4) on a 2.66 GHz Pentium 4 Linux machine is approx. 35 seconds for a 25 MByte

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                          xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <!-- Combine rdf:Descriptions with same rdf:about, rdf:nodeID attributes -->
 <!-- Prerequisites: unabbreviated RDF input syntax (no QName abbreviations) -->
 <!-- Input: unsorted RDF descriptions, Output: grouped,sorted RDF descriptions -->
 <xsl:output method="xml"/>

 <xsl:key name="aboutkeys" match="rdf:Description" use="@rdf:about"/>
 <xsl:key name="nodekeys"  match="rdf:Description" use="@rdf:nodeID"/>

 <xsl:template match="/rdf:RDF">  <!-- root template -->
  <xsl:copy>
   <xsl:copy-of select="@*"/> <!-- copy top attributes -->
   <!-- walk through rdf:Descriptions with rdf:about, rdf:nodeID attributes -->
   <xsl:for-each select="rdf:Description[generate-id(.)=generate-id(key('aboutkeys',
       @rdf:about)[1])]">
    <xsl:sort select="@rdf:about"/>
    <xsl:copy>
     <xsl:copy-of select="@*"/>
     <xsl:for-each select="key('aboutkeys', @rdf:about)">
      <xsl:copy-of select="*"/>
     </xsl:for-each>
    </xsl:copy>
   </xsl:for-each>
   <xsl:for-each select="rdf:Description[generate-id(.)=generate-id(key('nodekeys',
       @rdf:nodeID)[1])]">
    <xsl:sort select="@rdf:nodeID"/>
    <xsl:copy>
     <xsl:copy-of select="@*"/>
     <xsl:for-each select="key('nodekeys', @rdf:nodeID)">
      <xsl:copy-of select="*"/>
     </xsl:for-each>
    </xsl:copy>
   </xsl:for-each>
  </xsl:copy>
  <xsl:apply-templates/>
 </xsl:template>

 <xsl:template match="text()"/> <!-- ignore text here; handled in template above -->

</xsl:stylesheet>
```

Figure 3: rdfsort.xsl: XSLT stylesheet that combines distributed rdf:description statements.

OWL LT world ontology input file with mappings for person, project, organisation, event, product and technology concepts and instances, resulting in 1200 TDL type definitions and 20000 structured gazetteer entries.

3. Application to hybrid NLP

We now describe and exemplify how the named entities enriched with ontology information can be employed in a robustness-oriented, hybrid deep-shallow architecture that combines domain-specific shallow NER and deep, domain-independent HPSG parsing for generating a semantics representation of the meaning of parsed sentences.

An application of this scenario is e.g. deep question analysis for question answering on structured knowledge sources. A detailed description of such an application can be found in (Frank et al., 2006).

3.1. Named entity recognition at runtime

The output of SProUT for a recognised named entity is a typed feature structure (e.g. in XML format; cf. (Lee et al., 2004)) containing the RHS of the recognition rule as shown in step 4 (Section 2.4.) with the copied structured gazetteer data plus some additional information like character span, named entity type etc.

The mapping of recognised named entities to generic lexicon entries of an HPSG grammar, in this case the ERG (Flickinger, 2002), for hybrid processing can be performed through an XSLT stylesheet automatically generated from the SProUT type hierarchy. The stylesheet generation facility is part of the freely available Heart of Gold (Callmeier et al., 2004) framework for hybrid deep-shallow processing and described in detail in (Schäfer, 2005). Analogous mappings are currently supported for German, Greek and Japanese HPSG grammars.

To continue the example from the sections above, the gen-

erated stylesheet would at run time produce the following item for `LREC 2006` on the deep parser's input chart (PET XML input chart; the corresponding, mapped HPSG type being `$generic_event`).

```
<w id="SPR3.1" cstart="48" cend="56"
   constant="yes">
  <surface>LREC 2006</surface>
  <typeinfo id="TIN3.1" baseform="no">
   <stem>$generic_event</stem>
  </typeinfo>
</w>
```

I.e., the transformation output then contains only the NER information that is required by the deep parser with its broad-coverage, domain-independent grammar, namely character span and generic HPSG type for a chart item to be generated. A sample output of the semantic representation the deep parsers generates is shown in Figure 4.

How the finer-grained, domain-specific information from the ontology instance is transported to an application, is shown in the next section.

In addition to the basic named entity type mapping for default lexicon entries, the recognised concepts could also be useful for constraining the semantic sort in HPSG in a more fine-grained way (e.g. for disambiguation). The PET input chart format and also the upcoming, similar MAF/SAF format (Waldron et al., 2006) foresee 'injection' of such types into the HPSG structures.

As an alternative to the hybrid deep-shallow processing model, the full output from a SProUT runtime system could be used instead in a shallow-only application framework like automatic typed hyperlinking (Busemann et al., 2003).

3.2. Information extraction at runtime

Similar to the NER mapping from the previous section, Heart of Gold can also automatically generate XSLT stylesheets that produce a richer, robust semantics representation format (RMRS, cf. (Copestake, 2003), example Fig. 5) at runtime from the SProUT named entity recognition analyses.

Here, `OBJID` and other, also structured information like given name and surname, is preserved in the representation. The advantage of the RMRS format is that it can also be combined *ex post* with analyses from other deep or shallow NLP components, e.g. partial analyses when a full parse fails.

It has to be pointed out here that the mapped ontology data is added as a supplement to the standard named entity grammar and resources for proper names, location etc. In case a proper name occurring in text is not in the mapped gazetteer list, it could still be recognized by the normal SProUT named entity grammars as proper name, but then of course without links into the ontology.

The whole process of compiling domain-specific SProUT named entity grammars from extended resources that can be plugged into the Heart of Gold is part of an automation framework called SProUTomat (Schäfer and Beck, 2006).

4. Multi- and Cross-linguality

Some ontologies are multilingual, i.e., for concepts, realisations in different languages are stored together with the language-independent concept, and distinguished by a language attribute (e.g. containing an ISO 639 language code). In non-English (e.g. German) scientific or technology-oriented texts, English terms are used frequently. By simply selecting the appropriate entries as part of mapping configuration, German *and* English entries could be specified as appropriate for German texts but only English entries for English texts.

5. Summary and Outlook

We have described an XSLT-based procedure that maps ontology instances and concepts to named entity recognition and information extraction resources, providing links back for further ontology queries. The process is automatic except for the selection of relevant concepts and properties to map. The possible benefits are (i) improved precision and recall of NER and IE in closed domains, (ii) exploitation of linguistic knowledge for identifying ontology concepts and instances in text, (iii) access to full ontology knowledge through subsequent ontology queries, (iv) reduced workload for managing ontology data and lingware by avoiding duplication of work. An application using hybrid shallow and deep natural language processing on the basis of the mapped ontology data has been successfully implemented for question answering.

Future work will include a deeper investigation of adaptability to other ontologies and domains than described here, and extension of the mapping approach to additional relations supported by OWL.

6. Acknowledgements

I would like to thank Hans-Ulrich Krieger for helpful discussions and the LREC reviewers for their comments. This work has been supported by a grant from the German Federal Ministry of Education and Research (FKZ 01IWC02).

7. References

Stephan Busemann, Witold Drożdżyński, Hans-Ulrich Krieger, Jakub Piskorski, Ulrich Schäfer, Hans Uszkoreit, and Feiyu Xu. 2003. Integrating information extraction and automatic hyperlinking. In *Proceedings of the Interactive Posters/Demonstration at ACL-03*, pages 117–120, Sapporo, Japan.

Ulrich Callmeier, Andreas Eisele, Ulrich Schäfer, and Melanie Siegel. 2004. The DeepThought core architecture framework. In *Proceedings of LREC-2004*, pages 1205–1208, Lisbon, Portugal.

James Clark, 1999. *XSL Transformations (XSLT)*. World Wide Web Consortium, http://w3c.org/TR/xslt.

Ann Copestake. 2003. Report on the design of RMRS. Technical Report D1.1b, University of Cambridge, Cambridge, UK.

Witold Drożdżyński, Hans-Ulrich Krieger, Jakub Piskorski, Ulrich Schäfer, and Feiyu Xu. 2004. Shallow processing with unification and typed feature structures – foundations and applications. *Künstliche Intelligenz*, 2004(1):17–23.

Dan Flickinger. 2002. On building a more efficient grammar by exploiting types. In Dan Flickinger, Stephan

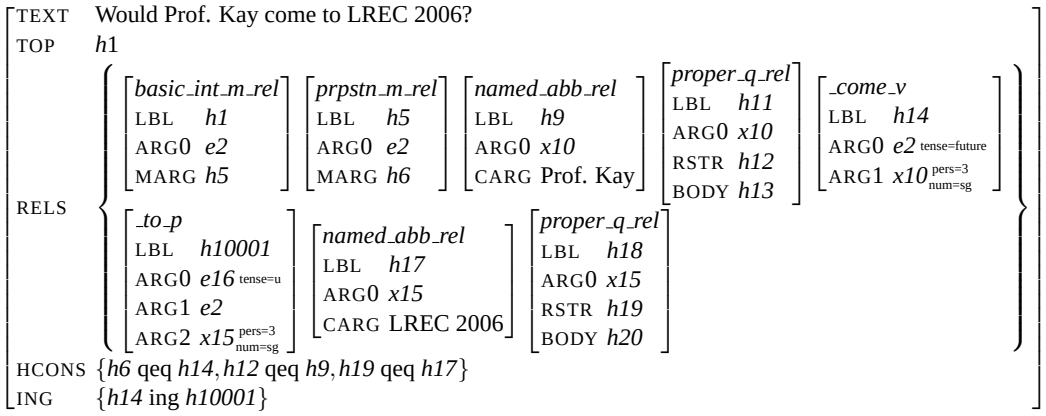

Figure 4: RMRS of deep sentence parsing generated by PET in Heart of Gold.

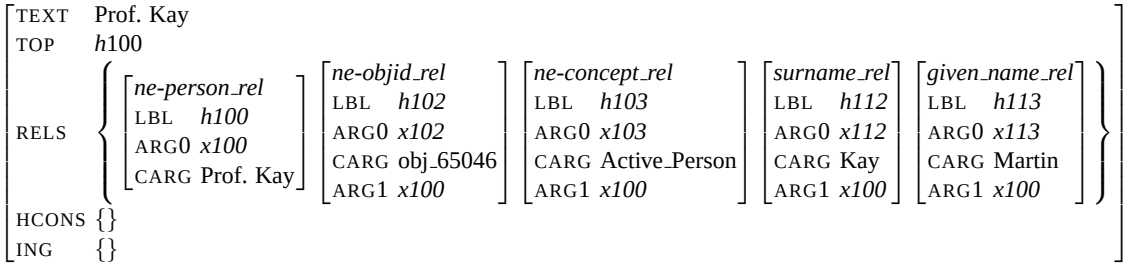

Figure 5: Fine-grained RMRSes of named entities generated from SProUT output in Heart of Gold.

Oepen, Hans Uszkoreit, and Jun-ichi Tsujii, editors, *Collaborative Language Engineering. A Case Study in Efficient Grammar-based Processing*, pages 1–17. CSLI Publications.

Anette Frank, Hans-Ulrich Krieger, Feiyu Xu, Hans Uszkoreit, Berthold Crysmann, Brigitte Jörg, and Ulrich Schäfer. 2006. Question answering from structured knowledge sources. *Journal of Applied Logic*.

Hans-Ulrich Krieger and Ulrich Schäfer. 1994. TDL – a type description language for constraint-based grammars. In *Proceedings of COLING-94*, pages 893–899.

Kiyong Lee, Lou Burnard, Laurent Romary, Eric de la Clergerie, Ulrich Schäfer, Thierry Declerck, Syd Bauman, Harry Bunt, Lionel Clément, Tomaz Erjavec, Azim Roussanaly, and Claude Roux. 2004. Towards an international standard on feature structure representation (2). In *Proceedings of the LREC-2004 workshop on A Registry of Linguistic Data Categories within an Integrated Language Resources Repository Area*, pages 63–70, Lisbon, Portugal.

Jakub Piskorski. 2005. Modelling of a gazetteer lookup component. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (Companion Volume)*, pages 163–168, Jeju Island, Republic of Korea. Asian Federation of Natural Language Processing.

Ulrich Schäfer and Daniel Beck. 2006. Automatic testing and evaluation of multilingual language technology resources and components. In *Proceedings of LREC-2006*, Genoa, Italy.

Ulrich Schäfer. 2003. WHAT: An XSLT-based infrastructure for the integration of natural language processing components. In *Proceedings of the Workshop on the Software Engineering and Architecture of LT Systems (SEALTS), HLT-NAACL03*, pages 9–16, Edmonton, Canada.

Ulrich Schäfer. 2005. Heart of Gold – an XML-based middleware for the integration of deep and shallow natural language processing components, user and developer documentation. http://heartofgold.dfki.de.

Hans Uszkoreit, Brigitte Jörg, and Gregor Erbach. 2003. An ontology-based knowledge portal for language technology. In *Proceedings of ENABLER/ELSNET Workshop*, Paris.

Ben Waldron, Ann Copestake, Ulrich Schäfer, and Bernd Kiefer. 2006. Preprocessing and tokenisation standards in DELPH-IN tools. In *Proceedings of LREC-2006*, Genoa, Italy.