Semantic Web Enabled, Open Source Language Technology

Kalina Bontcheva
University of Sheffield
Regent Crt., 211 Portobello St.
Sheffield S1 4DP, UK
kalina@dcs.shef.ac.uk

Atanas Kiryakov
Ontotext Lab, Sirma AI Ltd.
38A Hristo Botev Blvd.
Sofia 1000, Bulgaria
naso@sirma.bg

Hamish Cunningham
University of Sheffield
Regent Crt., 211 Portobello St.
Sheffield S1 4DP, UK
hamish@dcs.shef.ac.uk

Borislav Popov
Ontotext Lab, Sirma AI Ltd.
38A Hristo Botev Blvd.
Sofia 1000, Bulgaria
borislav@sirma.bg

Marin Dimitrov
Ontotext Lab, Sirma AI Ltd.
38A Hristo Botev Blvd.
Sofia 1000, Bulgaria
marin@sirma.bg

Abstract

This paper motivates the need for Semantic Web enabled language technology tools and introduces a set of freely available, customisable components which integrate data about language with Semantic Web data in the form of ontologies. We also argue for a closer integration between Natural Language Processing (NLP) and Semantic Web tools and infrastructures and present an integrated platform for knowledge and information management, that uses RDF to encode and store language data and resources.

1 Introduction

The Semantic Web aims to add a machine tractable, repurposeable layer to compliment the existing web of natural language hypertext. As part of this, a number of standards like RDF and DAML+OIL have been developed. In this paper we show how they have been used to create Semantic Web-enabled, open source language processing tools that: *(i)* integrate data about language with Semantic Web data, e.g. in the form of ontologies; *(ii)* are robust across genres and domains; *(iii)* can easily be embedded in other applications (via a Java component model and a mature class library); *(iv)* are freely available as open source. This work builds upon and extends GATE, a General Architecture for Text Engineering, which provides a well-established infrastructure for building and maintaining language technology tools.

We also argue for a closer integration between Natural Language Processing (NLP) and Semantic Web tools and infrastructures. In order to lower the integration overhead and allow NLP tools to benefit from ontologies and reasoning services, we have started the development of a common Knowledge and Information Management (KIM) platform, with a specific focus on (semi-)automatic annotation and ontology population for the Semantic Web, using Information Extraction (IE) technology. As part of this effort we have provided support for RDF-encoded language resources and are using Sesame, an RDF repository for storage, maintenance, and reasoning with RDF(S) data.

In Section 2 we introduce the GATE[1] tools used in this work, focusing on the way GATE now provides access to existing ontologies[2] in common formats, visualization/editing of ontologies, and creation of ontology-aware NLP modules. Section 3 shows how language resources can be made ontology aware. They also allow users to make available instance data from their Semantic Web ontologies to the NLP modules (see Section 4. The KIM platform is discussed in Section 5.

[1]Work on GATE has been supported by the Engineering and Physical Sciences Research Council (EPSRC) under grants GR/K25267 and GR/M31699, and by several smaller grants. The third author is currently supported by the EPSRC-funded AKT project (http://www.aktors.org) grant GR/N15764/01.

[2]An ontology is a specification of a conceptualization (Gruber, 1993).

2 Overview of GATE

GATE[3] (a General Architecture for Text Engineering) is a well-established infrastructure for customisation and development of NLP components. In brief, GATE (Cunningham et al., 2002a; Maynard et al., 2002) is a robust and scalable infrastructure for NLP, which allows users to focus on the language processing tasks, while mundane issues like data storage, format analysis, and data visualisation are handled by GATE itself. In order to deal with the problem of handling a variety of linguistic formalisms in a common framework, GATE has adopted a theory-independent annotation format. It is a modified form of the TIPSTER format (Grishman, 1997), is largely isomorphic with the Atlas format (Bird and Liberman, 1999) and successfully supports I/O to/from XCES and TEI (Ide et al., 2000).[4] An annotation has a type, a pair of nodes pointing to positions inside the document content, and a set of attribute-values, encoding further linguistic information. Attributes are strings; values can be any Java object. An annotation layer is organised as a Directed Acyclic Graph on which the nodes are particular locations in the document content and the arcs are made out of annotations. All the markup contained in the text used to create the document content is automatically extracted into a special annotation layer and can be used for processing or for exporting the document back to its original format.

The annotations associated with each language resource (e.g., document) are a structure central to GATE, because they encode the language data read and produced by each language processing module. GATE also supports export back to the resource's original format (e.g., SGML/XML/HTML).

2.1 Ontology Support in GATE

Recently, Semantic Web developments have increased the need for NLP applications like Information Extraction, because automatic processing is vital to the task of managing and maintaining access to online information. Such applications, however, require connection to appropriate ontologies and could benefit from language resources and data being represented in a format compatible with the Semantic Web, e.g., RDF.

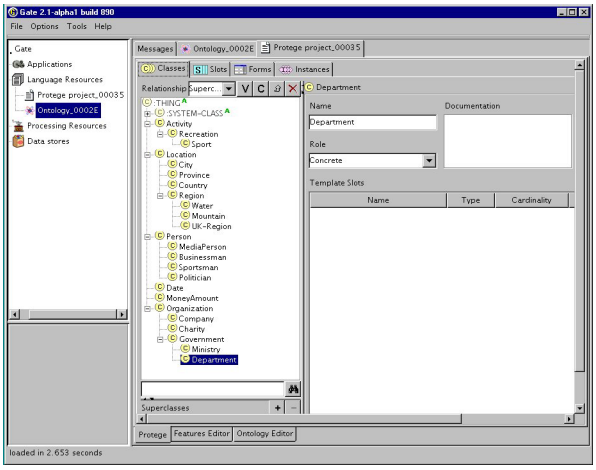

Figure 1: The Protégé ontology editor integrated within the GATE environment

Therefore, GATE has been extended recently to provide support for importing, accessing and visualising ontologies as a new type of resource. Much of this functionality is provided through the integration of the Protégé editor (Noy et al., 2001) within the GATE visual environment (see Figure 1). We also developed GATE-specific ontology-visualisation facilities, because they facilitate the process of developing and testing ontologically-aware NLP components (see Section 3). Ontology import/export is provided from/to DAML+OIL.

The advantage of providing support for ontologies within GATE is that the various NLP modules only need one, uniform way of accessing ontologies (i.e., a common API), regardless of their original formats. This is similar to the way in which GATE transparently supports handling multiple document formats, thus enabling the modules to run on a large variety of texts, without their developers having to implement I/O methods for each data format in each NLP module.

2.2 Generating Semantic Web annotations

In order to allow NLP modules to produce results in a Semantic Web-compatible format, we implemented a DAML+OIL exporter. It takes as input

[3] GATE and its IE tools are freely available, under the GNU Library License, from http://gate.ac.uk.

[4] The American National Corpus is using GATE for a large TEI-based project.

the user's ontology (see Figure 2[5]) and the types of annotations to be exported (e.g., Person, Organization). The exporter is implemented as a GATE processing resource, which means that it can be made part of any GATE NLP application and export its results for the Semantic Web. As shown in the figure, entities with different names (e.g., Mr. Bush and President Bush) are handled using `daml:sameIndividualAs`. The decision whether or not two names refer to the same entity is handled by GATE's coreference resolution modules (Dimitrov et al., 2002). Apart from handling coreference between named entities, GATE also resolves pronoun and nominal mentions (e.g., *he, the president*), using compatibility restrictions from the ontology when choosing the right antecedent. For example, since *bank* is a kind of `Company`, then it cannot refer to entities of kind `Person`.

3 Ontology-aware gazetteers

To illustrate how NLP applications can benefit from Semantic Web knowledge, made available through GATE from a given ontology, we will take a gazetteer as an example, show how it has been connected to an ontological resource, and how this enriched information can be used in later processing tasks (e.g., named entity recognition grammars and coreference resolution) to improve their performance. A gazetteer module is frequently used in NLP applications such as Information Extraction (IE), in order to annotate occurrences of phrases in text, given lists of such phrases and their type (e.g., first names, cities, name indicators like Mr. and Ltd.). However, the problem is that typically the types of these phrases is determined in an ad hoc fashion by the system developer, so it is not possible to perform any reasoning about relations between these types e.g., that since companies are organizations, then all phrases annotated as companies are also organizations and can be referred to as such. Therefore, later modules like coreference resolution, cannot directly use this information, but have to rely instead on their own module-specific domain ontology. This leads to duplication of knowledge in the NLP modules and makes their updating and use of third-party ontologies very difficult, since it has to be done for each module separately.

These problems can be solved by having an ontology-aware gazetteer, which treats the phrases as instances of concepts from a given ontology, so later language processing modules can access the same ontology and perform reasoning with the information produced by the earlier gazetteer module, e.g., to obtain the semantic distance between two such concepts. In this way, ontologies become shared resources, just like lexicons.

The new gazetteer is not tied to any specific ontology and could be used with other ontology management systems, since the ontologies management within GATE is separate from the NLP modules which use it. Like all other GATE components, the ontologies and the processing modules which use them can easily be used outside GATE, standalone or as a part of another application (e.g., GATE modules have already been integrated within Protégé to support the knowledge acquisition process by annotating automatically texts with information about companies, person names, dates, etc.).

The main issue is connecting concepts from the ontology and their instances to the gazetteers used for NLP. This is beneficial, because the process of editing gazetteer entries becomes part of ontology maintenance. The other advantage is that it enables the NLP components to annotate automatically in the texts the instances already available in the user's ontology.

In GATE the user can view the ontology and all the instances are shown in a separate window where they can be edited. The user can also specify how they are connected to the gazetteer's semantic classes, which are required by the subsequent components. Figure 3 shows the ontology with all instance names displayed on the right in a list format, which is used to markup automatically their occurrences in the texts. There can be more than one list of instance names per concept, as shown in the figure. In this way, lists provided with the GATE system can be used simultaneously with lists derived from the instances in the user's ontology. In cases where text is annotated as matching a gazetteer list entry coming from the

[5]The ontology used in this example is the one shown in Figure 3.

Figure 2: The DAML+OIL annotations generated by the system

ontology, the annotation also provides information about its ontological class and the URI of the ontology itself. Using this information, subsequent language processing modules can query the ontology for reasoning purposes.

For example, if the user has a set of companies that they work with entered as instances in their ontology (e.g., `mySupplier Inc.`), that information will be used by the ontologically-aware gazetteers to annotate automatically all mentions of these companies in the text as having a class Company. With the default gazetteers the IE tools will recognise `mySupplier Inc.` as an `Organization`, while with the extra information about their ontological class, introduced via the gazetteer, it will be classified correctly as an instance of `Company`. Later NLP modules can then reason about it as either a Company, or Organization, or any of their super-classes.

As mentioned above, the OntoGazetteer also allows lists of entities provided with GATE (e.g., countries, cities) to be mapped to their corresponding class in the user's ontology (see `Mapping definition`, Figure 3). Similar to the above example, the ontological information is used by the later NLP modules and then during DAML+OIL export. For example, if the list of countries is mapped to the `Country` class in the ontology (see Figure 3), then instances like `United States` will be classified as countries, instead of the more general type `Location`. In order to see the differences, compare the DAML+OIL output in Figure 2 (without ontology mapping) and in Figure 4 (with mapping shown in Figure 3).

4 Use of Ontological Information in Other NLP components

4.1 Anaphora resolution

GATE has light-weight coreference resolution modules that identify coreferring names, pronouns, and nominal phrases. The algorithms are restricted first to identify only phrases which could refer to named entities and then try to resolve them; they are not general purpose anaphora resolution modules. The information from the ontology is used in order to verify whether two entities have compatible ontological types. For example, let us assume that we have the text: "Gordon Brown met president Bush during his 2 day

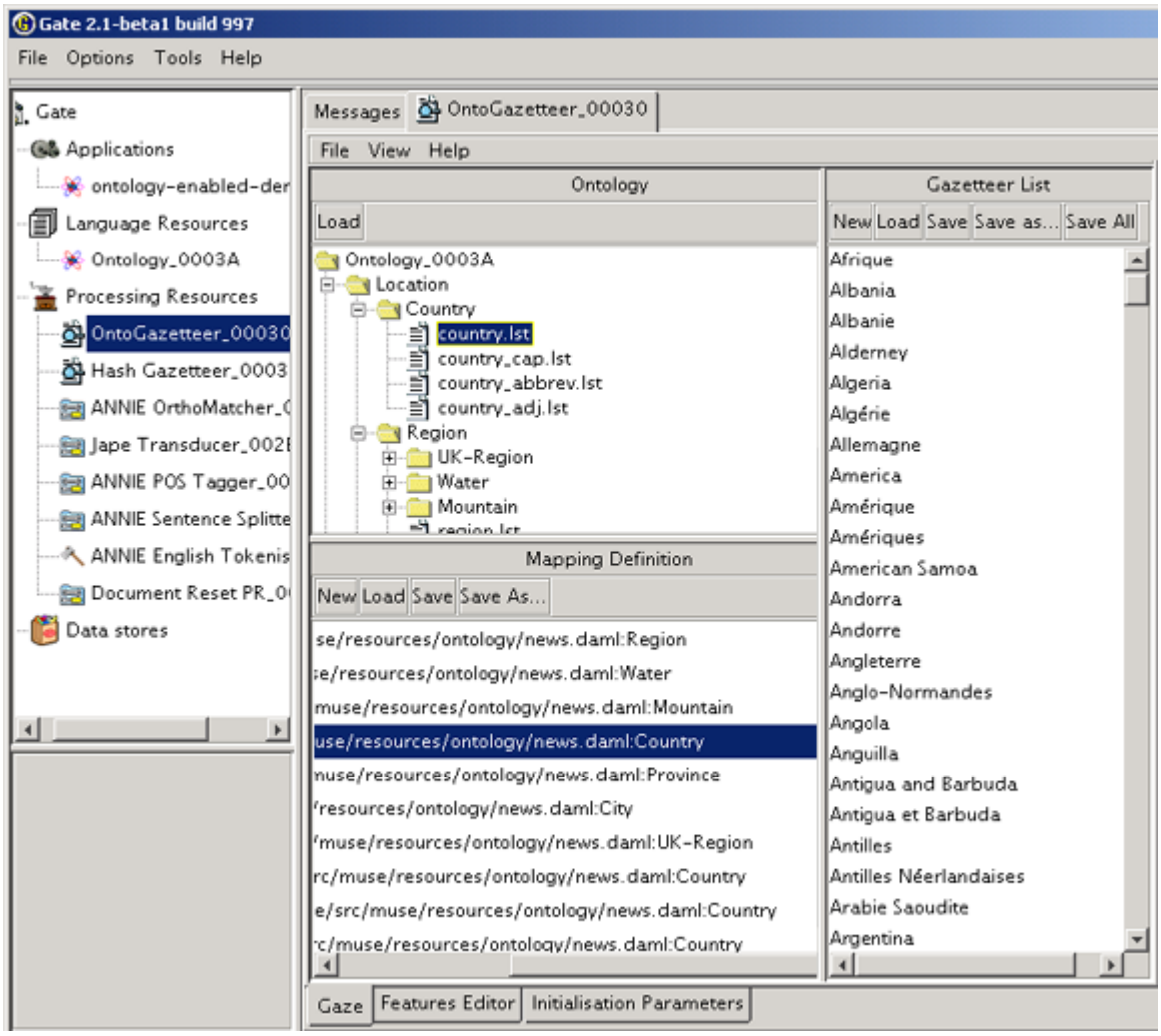

Figure 3: Connecting the user's ontology to the GATE gazetteers

visit. Afterwards the Chancellor said..." In this case, *Gordon Brown* and *Bush* are recognised as persons and puts them as instances in the ontology. Then the nominal resolution module has to resolve *president* and since it appears as a title before a person's name, this is easily resolved as coreferent to *Bush* and the ontology is updated with the corresponding information. Next *Chancellor* has to be resolved. If only recency and ontological compatibility are used, then it will wrongly be resolved as referring to *Bush*, because this is the most recent entity of type Person. However, using the updated information from the ontology it is also possible to check that *Bush* is also a president and presidents cannot be also chancellors (more generally, people have one profession at a time). Therefore, the ontology enables the use of (simple) reasoning to perform anaphora resolution.

4.2 Processing using the ontology

GATE has an easy-to-understand, flexible pattern action language called JAPE (Java Annotations Pattern Engine) (Cunningham et al., 2002b). JAPE rules describe patterns to match and annotations to be created as a result. JAPE provides finite state transduction over annotations based on regular expressions. A JAPE grammar consists of a set of phases, each of which consists of a set of pattern/action rules, and which run sequentially. Patterns can be specified by describing a specific text string, or annotations previously created by modules such as the tokeniser, gazetteer, or document

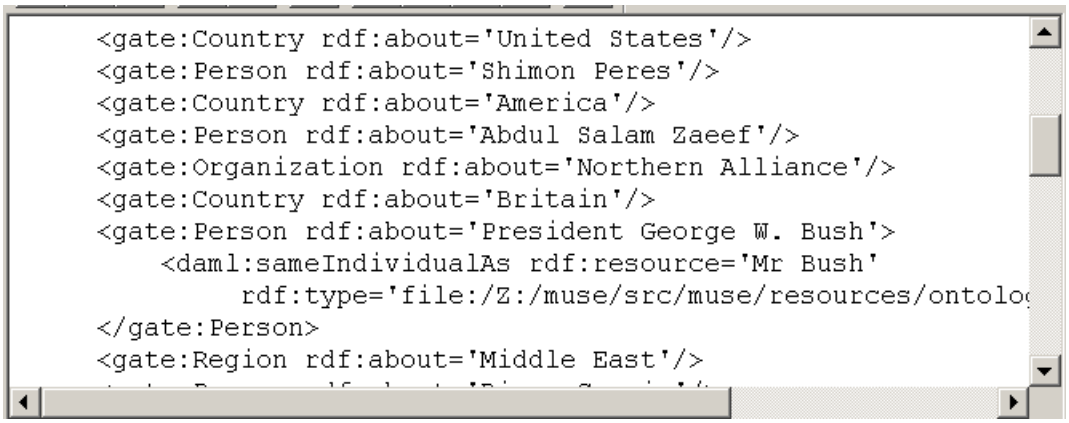

Figure 4: DAML+OIL export using the ontological information

format analysis. Rule prioritisation (if activated) prevents multiple assignment of annotations to the same text string. For example, the following rule specifies that one or more words, starting with an uppercase letter, followed by a company designator, should be annotated as an organisation.

```
Rule: OrgXKey (
  ({Token.kind == word,
       Token.orth == upperInitial})+
  {Lookup.type == cdg}
) :orgName -->
  :orgName.Organization
```

The Lookup annotations are created by the gazetteer lookup module, discussed above. In the case when ontological information is available from the gazetteers, it can be used in order to assign the proper ontological class. For instance, company designators are specified in the ontology as cue words indicating instances of class Company, therefore using this information annotations created by this pattern can be classified more specifically as companies, instead of the more general class `Organization`.

We are planning to extend the JAPE engine to take into account subsumption relations in the ontology when doing the matching on the left-hand side. So for example, a rule might look for an organization followed by a location, in order to create the `locatedAt` relationship between them. If JAPE takes into account subsumption, then the rule will automatically apply to all sub-classes of `Organisation`, e.g., `Company`, `GovernmentOrg`.

5 Towards a Knowledge and Information Management Platform

Tools and infrastructures for the Semantic Web on the one hand and language processing on the other have so far remained largely independent from each other, despite the fact that they share a number of components, namely ontologies and reasoning mechanisms. For example, NLP systems can benefit from new developments like the Ontology Middleware Module (OMM) - an extension of the SESAME RDF(S) repository, see (Klein et al., 2002)) which will enable NLP tools to index and retrieve language data, e.g., annotations, and language resources, e.g. gazetteers, in RDF(S). It will also enable the use of Semantic Web reasoning tools within NLP components.

In order to lower the integration overhead and allow NLP tools to benefit from such ontologies and reasoning services, we have started the development of a common Knowledge and Information Management (KIM) platform, with specific focus on automatic annotation and ontology population for the Semantic Web, using IE technology[6]. KIM combines GATE and Sesame/ OMM and allows the NLP modules to create annotations related to a formal ontology of classes and instances, expressed in RDF(S) (or another compatible language). The annotations associated with each document are stored in Sesame and documents can be browsed based on these annotations, e.g., find-

[6]For further information about KIM see `http://www.ontotext.com/KIM/index.html`.

ing all companies established in 1999 in Delaware, US. Language resources used by the NLP modules can, if chosen, also be stored in Sesame as RDF, for improved inter-operability.

5.1 RDF-based Browsing of (NLP-)Annotated Documents

As discussed in Section 2, the NLP modules in GATE use annotations associated with documents as means to receive input data from previous modules and produce their output results. These annotations can be related to an RDF resource by specifying its URI as part of the annotation, e.g., class and instance information for a given ontology is encoded as features on the annotations such as `class= http://www.ot.com/kim/kimo.rdfs#Person` and `inst = http://www.ot.com/kim/kimo.rdfs #Person.13671`.

KIM has a component called KIM Explorer which enables the browsing of instance information in RDF(S) ontologies from the semantically annotate documents, indexed in Sesame. For each entity the explorer presents (i) the most specific classes it belongs to, (ii) its properties and relations to other entities, and finally (iii) the entities related to it. All the other entities are hyperlinked, so, they can be explored further. The abstractions over the ”native” RDF(S) representation include:

- the resources are presented with their labels, rather than URIs;
- number of “auxiliary” properties are filtered out.

Since the KIM Explorer pane (see Figure 5) pops up when a hyperlink from an annotation in the text is followed, it provides a smooth transition from the text to the formal knowledge available and back. In effect, KIM not only offers (semi-)automatic document annotation with Semantic Web content, using IE, but it also provides a way to navigate the document collection using the RDF annotations.

5.2 Storing Language Resources as RDF

KIM provides means for storing and accessing language resources in RDF and Sesame. Due to space limitations, here we will discuss how the ontology-enabled gazetteers were changed to use only RDF, instead of having some of the lists encoded in GATE’s own format. In order to support language data, the KIM ontology was extended with a special class, called Language Resource, which has a number of subclasses encoding the different types of data that exist in the gazetteer lists. By storing all gazetteer information as RDF we dispense with the need to have lists in a GATE-specific format and make it easier for other applications to use this information, independent of GATE. The mapping definitions specify which classes from the ontology should be used by the gazetteer module - they can be either under LanguageResource or any other class, e.g., Company, which is a kind of Entity. The gazetteer module itself was modified to load all instances of the specified classes and also all instances of their subclasses, by querying Sesame which stores them. Once loaded in memory, the actual gazetteer lookup is performed as before and the appropriate class and instance information is associated with the annotations, exactly in the same way as in the ontology-aware gazetteers discussed in Section 3. The difference between the two gazetteer modules is whether or not all the information is stored as RDF. If the application wants to keep using some already existing lists, without converting them to RDF, then it uses the onto-gazetteers, which provide the ontological information but do not store the entries themselves as RDF. Other applications which want to have fully RDF-based resources use the RDF-aware gazetteer module.

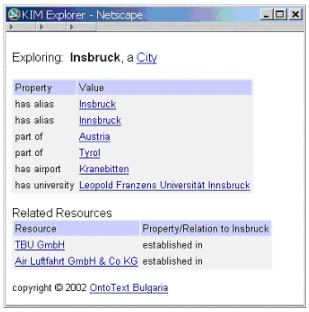

Figure 5: The KIM Explorer showing information about Insbruck

6 Conclusion

In this paper we showed how NLP modules can be made aware of the user's ontology, so they can take into account available instances and also produce annotations which are more directly related to the given ontology. We also discussed the need for closer integration between the language processing and Semantic Web tools in general and proposed a Knowledge and Information Management platform. KIM offers an RDF(S) repository for storage and management of both language and Semantic Web data, reasoning services, ontology editing and browsing, Web crawler, semantic query interface, a browser plug-in for document viewing/annotation. The main goal of KIM is to provide automatic annotation of Web documents with Semantic Web data, based on Information Extraction. Therefore our efforts so far have been mainly focused on providing RDF support for IE modules. Further work will broaden this support towards new types of language resources, e.g., lexicons. We hope that this extension will be made easier by the XML and RDF-based standards for language resources currently being developed (Ide and Romary, 2002).

By making GATE's IE modules – the gazetteers, named entity recognition grammars, and anaphora resolution modules – Semantic Web-enabled, we have allowed the automatic creation of RDF-annotated documents. The annotations contain information about the ontology being used, the class, and instance (where available), thus making it possible to support emerging standards for RDF-based annotation of IE data, such as those being developed for named entities (Collier et al., 2002). Currently the IE results can be exported for the Semantic Web as DAML+OIL, using GATE's exporter module. The exporter will be extended to support the relevant standards when they are finalised.

References

S. Bird and M. Liberman. 1999. A Formal Framework for Linguistic Annotation. Technical Report MS-CIS-99-01, Department of Computer and Information Science, University of Pennsylvania. `http://xxx.lanl.gov/abs/cs.CL/9903003`.

N. Collier, K. Takeuchi, C. Nobata, J. Fukumoto, and N. Ogata. 2002. Progress on multi-lingual named entity annotation guidelines using rdf(s). In *Proceedings of 3rd Language Resources and Evaluation Conference (LREC'2002)*, Gran Canaria, Spain.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002a. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*.

H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. 2002b. *The GATE User Guide*. `http://gate.ac.uk/`.

M. Dimitrov, K. Bontcheva, H. Cunningham, and D. Maynard. 2002. A Light-weight Approach to Coreference Resolution for Named Entities in Text. In *Proceedings of the Fourth Discourse Anaphora and Anaphor Resolution Colloquium (DAARC)*, Lisbon.

R. Grishman. 1997. TIPSTER Architecture Design Document Version 2.3. Technical report, DARPA. `http://www.itl.nist.gov/div894/-894.02/related_projects/tipster/`.

T. R. Gruber. 1993. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220.

N. Ide and L. Romary. 2002. Standards for language resources. In *Proceedings of 3rd Language Resources and Evaluation Conference (LREC'2002)*, Gran Canaria, Spain.

N. Ide, P. Bonhomme, and L. Romary. 2000. XCES: An XML-based Standard for Linguistic Corpora. In *Proceedings of the Second International Language Resources and Evaluation Conference (LREC)*, pages 825–830, Athens, Greece.

M. Klein, D. Fensel, A. Kiryakov, and D. Ognyanov. 2002. Ontology Versioning and Change Detection on the Web. In *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02)*, pages 197–212, Siguenza, Spain.

D. Maynard, V. Tablan, H. Cunningham, C. Ursu, H. Saggion, K. Bontcheva, and Y. Wilks. 2002. Architectural elements of language engineering robustness. *Journal of Natural Language Engineering – Special Issue on Robust Methods in Analysis of Natural Language Data*, 8(2/3):257–274.

N.F. Noy, M. Sintek, S. Decker, M. Crubzy, R.W. Fergerson, and M.A. Musen. 2001. Creating Semantic Web Contents with Protégé-2000. *IEEE Intelligent Systems*, 16(2):60–71.