

MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup

Maria Vargas-Vera¹, Enrico Motta¹, John Domingue¹, Mattia Lanzoni¹, Arthur Stutt¹ and Fabio Ciravegna²

¹ Knowledge Media Institute
The Open University
Walton Hall, Milton Keynes, MK7 6AA, UK
{m.vargas-vera; e.motta; j.b.domingue; m.lanzoni;
a.stutt}@open.ac.uk

² Department of Computer Science,
University of Sheffield
Regent Court, 211 Portobello Street,
Sheffield S1 4DP, UK
f.ciravegna@dcs.shef.ac.uk

Abstract. An important precondition for realizing the goal of a semantic web is the ability to annotate web resources with semantic information. In order to carry out this task, users need appropriate representation languages, ontologies, and support tools. In this paper we present MnM, an annotation tool which provides both automated and semi-automated support for annotating web pages with semantic contents. MnM integrates a web browser with an ontology editor and provides open APIs to link to ontology servers and for integrating information extraction tools. MnM can be seen as an early example of the next generation of ontology editors, being web-based, oriented to semantic markup and providing mechanisms for large-scale automatic markup of web pages.

1 INTRODUCTION

An important pre-condition for realizing the goal of the semantic web is the ability to annotate web resources with semantic information. In order to carry out this task, users need appropriate *knowledge representation languages*, *ontologies*, and *support tools*. The knowledge representation language provides the semantic interlingua for expressing knowledge precisely. RDF ([14], [20]) and RDFS [2] provide the basic framework for expressing metadata on the web, while current developments in web-based knowledge representation, such as DAML+OIL (reference description of the daml+oil can be found at <http://www.daml.org/2001/03/reference.html>) and the language that will be proposed by the WebOnt group (<http://www.w3.org>), are building on the RDF base framework to provide more sophisticated knowledge representation support. Ontologies [12] provide the mechanism to support interoperability at a con

ceptual level. In a nutshell, the idea of interoperating agents able to exchange information and carrying out complex problem solving on the web is based on the assumption that these agents will share common, explicitly defined, generic conceptualizations. These are typically models of a particular area, such as product catalogues, or taxonomies of medical conditions, although ontologies can also be used to support the specification of reasoning services ([23], [25], [11]), thus allowing not only ‘static’ interoperability through shared domain conceptualizations, but also ‘dynamic’ interoperability through the explicit publication of competence specifications, which can be reasoned about to determine whether a particular web service is appropriate for a particular task.

Ontologies and representation languages provide the basic semantic tools to construct the semantic web. Obviously a lot more is needed; in particular, tool support is needed to facilitate the development of semantic resources, given a particular ontology and representation language. This problem is not a new one, knowledge engineers early on realized that one of the main obstacles to the development of intelligent, knowledge-based systems was the so-called *knowledge acquisition bottleneck* [10]. In a nutshell, the problem is how to acquire and represent knowledge, so that this knowledge can be effectively used by a reasoning system. Although the problem is not a new one, the context provided by the semantic web introduces new aspects to the problem, with respect to the nature of the knowledge and the type of users.

Nature of the knowledge. Traditional knowledge acquisition was concerned with knowledge for problem solving. Semantic markup will primarily focus on ontology population, a far easier knowledge acquisition task.

Type of users. Knowledge-based systems are normally written by skilled knowledge engineers. On the web, it is likely that semantic marking up will become a common activity, carried out by content providers who are not necessarily skilled knowledge engineers. This means that more emphasis will have to be put on facilitating semantic markup by ‘ordinary’ web users (people who are neither experts in language technologies nor ‘power knowledge engineers’). In particular, automated knowledge extraction technologies are likely to play an ever increasing important role, as a crucial technology to tackle the semantic web version of the knowledge acquisition bottleneck.

In this paper we present *MnM*, an annotation tool which provides both automated and semi-automated support for marking up web pages with semantic contents. MnM integrates a web browser with an ontology editor and provides open APIs to link to ontology servers and for integrating information extraction tools. MnM can be seen as an early example of the next generation of ontology editors, being web-based, oriented to semantic markup and providing mechanisms for large-scale automatic markup of web pages.

The rest of the paper is organized as follows: in the next section we will show the process model underlying the design of the tool. Section 3 will show an example of the tool in use. Finally sections 4 and 5 discuss related work and re-state the main tenets and results from our research.

2 PROCESS MODEL

Within this work we have focused on creating a *generic process model* for developing semantically enriched web content. The component tools which are used in MnM are ontology servers, Information Extraction (IE) tools and augmented web browsers. During our initial work in this area we found that either the existing tools did not directly support the creation of semantic web content or the mapping between the tasks to be carried out and the toolset was non-trivial. Hence, within MnM, we adopted a *generic process model*, which can be easily understood by web developers who are not necessarily expert ontology engineers or human language technology experts.

Another key feature of our process model is that it is generic with respect to the specific ontology server and IE technologies used.

There are five main activities supported by MnM:

- *Browse*. A specific set of knowledge components is chosen from a library of knowledge models on an ontology server.
- *Markup*. The chosen set of knowledge components is selected to form the basis of an IE mechanism. A corpus of documents are manually marked up.
- *Learn*. A learning algorithm is run over the marked up corpus to learn the extraction rules.
- *Test*. The IE mechanism is run over a test corpus to assess its precision and recall measures.
- *Extract*. An IE mechanism is selected and run over a set of documents

We will now provide more details of each of the above activities in turn.

Browse

In this activity the user browses a library of knowledge models which sit on a web based ontology server. The user can see an overview of the existing models and can select which one to focus on (i.e., which ontology to use to initiate the markup process). Within a selected ontology the user can browse the existing items - for example the classes. Items within an ontology can be selected as the starting point for selecting an IE mechanism. More specifically, the selected class forms the basis for a template which will eventually be matched against a corpus of documents and instantiated in the extraction activity.

Mark-Up

The activity of semantic tagging refers to the activity of annotating text documents (written in plain ASCII or HTML) with a set of tags defined in the ontology, in particular we work with a hand-crafted KMi ontology (ontology describing the knowledge Media Institute- KMi).

MnM provides means to browse the event hierarchy (defined in the KMi ontology). In this hierarchy each event is a class and the annotation component extracts the set of possible tags from the slots defined in each class.

Once a class has been selected a training corpus of manually marked up pages needs to be created. Here the user views appropriate documents within MnM's built-in web browser and annotates segments of text using the tags based on the class's slot as

given in the ontology (i.e., ontology driven mark-up). As the text is selected MnM inserts the relevant SGML/XML tags into the document.

Learning

MnM integrates web browsing, ontology browsing and IE development. It does not have a built-in IE tool but provides a plug-in interface which allows the integration of IE tools easily.

In a previous version of our MnM we integrated Marmot, Badger and Crystal from the University of Massachusetts [26] and our own NLP components (i.e., OCML preprocessor). A full description of this version can be found in ([28], [29]). However, in this paper we will concentrate on the recent integration work that we have carried out with Amilcare, a tool for adaptive information extraction [3].

Amilcare is designed to support active annotation of documents. It performs IE by enriching texts with XML annotations. To use Amilcare in a new domain the user simply has to manually annotate a training set of documents. No knowledge of Natural Language Technologies is necessary.

Amilcare is designed to accommodate the needs of different user types. While naïve users can build new applications without delving into the complexity of Human Language Technology, IE experts are provided with a number of facilities for tuning the final application. Induced rules can be inspected, monitored and edited to obtain some additional accuracy, if required. The interface also allows precision (P) and recall (R) to be balanced. The system can be run on an annotated unseen corpus and users are presented with statistics on accuracy, together with details on correct matches and mistakes. Retuning the P&R balance does not generally require major retraining, facilities for inspecting the effect of different P&R balances are provided. Although the current interface for balancing P&R is designed for IE experts, a future version will provide support for naïve users [6].

At the start of the learning phase Amilcare preprocesses texts using Annie, the shallow IE system included in the Gate package ([22], www.gate.ac.uk). Annie performs text tokenization (segmenting texts into words), sentence splitting (identifying sentences) part of speech tagging (lexical disambiguation), gazetteer lookup (dictionary lookup), named entity recognition (recognition of people and organization names, dates, etc.). Amilcare then induces rules for information extraction. The learning system is based on LP², a covering algorithm for supervised learning of IE rules based on Lazy-NLP ([3], [4]). This is a wrapper induction methodology [19] that, unlike other wrapper induction approaches, uses linguistic information in the rule generalization process. The learning system starts inducing wrapper-like rules that make no use of linguistic information, where rules are sets of conjunctive conditions on adjacent words. Then the linguistic information provided by Annie is used in order to create generalized rules: conditions on words are substituted with conditions on the linguistic information (e.g. condition matching on either the lexical category, or the class provided by the gazetteer, etc. Examples of rules and deep description of the (LP²) algorithm can be found in [4].

All the generalizations are tested in parallel by using a variant of the AQ algorithm [24] and the best -generalizations are kept for IE. The idea is that the linguistic-based generalization is deployed only when the use of NLP information is reliable or effective. The measure of reliability here is not linguistic correctness, but effectiveness in

extracting information using linguistic information as opposed to using shallower approaches. Lazy NLP-based systems learn which is the best strategy for each information/context separately. For example they may decide that using the result of a part of speech tagger is the best strategy for recognizing the speaker in seminar announcements, but not to spot the seminar location. This strategy is quite effective for analyzing documents with mixed genres, a common situation in web documents [5].

The learning system induces two types of rules: tagging rules and correction rules. A tagging rule is composed of a left hand side, containing a pattern of conditions on a connected sequence of words, and a right hand side that is an action inserting an XML tag in the texts. Correction rules shift misplaced annotations (inserted by tagging rules) to the correct position. These are learnt from the errors found whilst attempting to re-annotate the training corpus using the induced tagging rules.

Correction rules are identical to tagging rules, but (1) their patterns also match the tags inserted by the tagging rules and (2) their actions shift misplaced tags rather than adding new ones. The output of the training phase is a collection of rules for IE that are associated with the specific scenario (domain).

Amilcare has been tested on Italian and English but it is easily extendible to cover other languages. It requires to connect a preprocessor for the target language (such as Annie is) including at least a tokenizer and possibly a part of speech tagger and morphological analyzer.

Testing

MnM provides two mechanisms for selecting a test corpus and distinguish this from a training corpora. The user can manually select training and test corpora and these can be in the form of local files or on the web. In addition, it is also possible to simply select a corpus (either locally or on the web) and let the system create test and training corpora randomly.

Extraction

After the training phase Amilcare has a library of induced rules which can be used to extract information from texts.

When working in extraction mode, Amilcare receives as input a (collection of) text(s) with the associated scenario – scenario is the set of tags that the user will insert in the training corpora- (including the rules induced during the training phase). It preprocesses the text(s) by using Annie and then it applies its rules and returns the original text with the added annotations. The Gate annotation schema is used for annotation [22]. Annotation schemas provides means to define types of annotations in Gate. Gate uses the XML schema language supported by W3C for these definitions. However, Gate version 2 supports annotations in SGML/XML.

Once that is done the information extracted is presented to the user for approval. Then the extracted information is sent to the ontology server which will populate the selected ontology.

During the population step the IE mechanism fills predefined slots associated with an extraction template. Each template consists of slots of a particular class as defined in the selected ontology, for instance, the class `visiting-a-place-or-people` has the

slots: visitor, place, etc. More detail about the population phase is given in the following section.

Our goal is to automatically fill as many slots as possible. However, some of the slots may still require manual intervention. There are several reasons for this problem:

- there is information that is not contained in the text,
- none of the rules from our IE libraries match with the sentence that might provide the information (incomplete set of rules). This means that the learning phase needs to be tuned.

The extracted information is also validated using the ontology. This is possible because each slot in each class of the ontology has a type associated with it. Therefore, extracted information which does not match the type definition of the slot in the ontology can be highlighted as incorrect.

Currently our system had been trained using an archive of 200 stories that we had collected in KMi. The training phase was performed using typical examples of stories belonging to each of the different type of events defined in the ontology. We obtained precision 95% and recall 90% using Amilcare on KMi stories.

3 EXAMPLE

We will now explain the process model we described earlier by walking through a specific extraction example. The domain of our example is a web based news letter, KMi Planet [8], that has been running in our lab for five years. The Planet front page, individual story and archive views are generated automatically from stories which are submitted by email or through a web based form. Over the years we have extended Planet to include semantic retrieval, smart layout and personalization services ([9], [17]). Whilst we were happy with the functionality that these services provided we were concerned that the knowledge base was maintained by hand. We have therefore selected this domain to apply MnM. Figure 1 shows the KMi Planet front page.

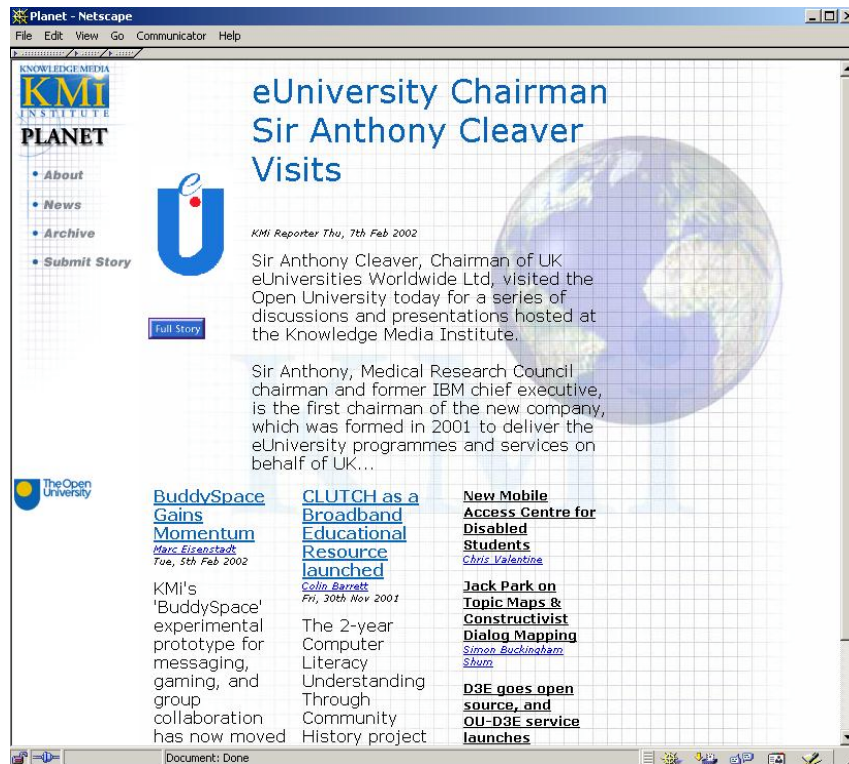


Fig. 1. A screen snapshot of the KMi Planet front page

The Planet services are implemented within the akt-kmi-planet-kb knowledge base/model which sits on our public knowledge model server (at <http://webonto.open.ac.uk> - see [7] for a description). This knowledge base builds on a dozen ontologies describing domains such as our lab, events, organisations and technologies.

Figures 2-5 show a user setting up an IE mechanism for extracting Planet stories about visits to KMi. In figure 2 we can see that MnM consists of three main windows. The window on the right is an augmented web browser. The windows on the left form a mini ontology browser: the top window displaying a high level view and the bottom window displaying detailed structure. Figures 2 and 3 show the initial steps in creating the visit story IE mechanism. In figure 2 the user is looking at a portion of the 200 stories in the story archive. The left top panel shows all the knowledge models on the server (shown in the left panel). The user selects akt-kmi-planet-kb and notes from the documentation that it implements the latest Planet knowledge services. Opening akt-kmi-planet-kb displays all of the classes within the knowledge base – note that the majority of the classes are inherited from the ontologies used by akt-kmi-planet-kb.

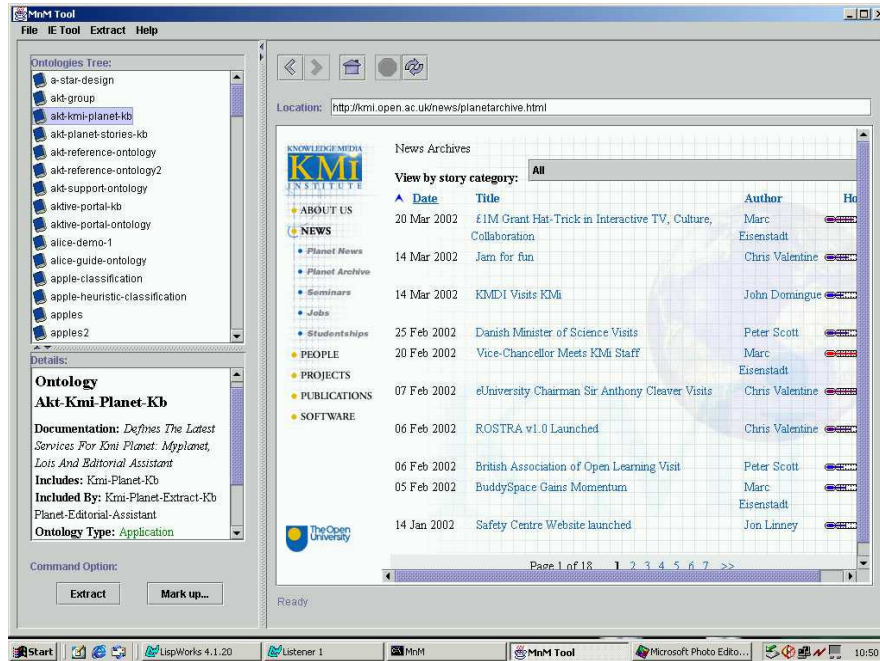


Fig. 2. A screen snapshot showing a user browsing the library of knowledge models held on the WebOnto server

Figure 3 shows the class “visiting-a-place-or-people” from the event hierarchy within the akt-kmi-planet-kb. The names of the slots are used in the markup phase during the annotation process.

The user now enters a markup phase. In figure 4 the user has selected the story “Bletchley Park Trust Director visits KMI” to mark up. He/She adds an entry to mark *Christine Large* as the visitor with the following simple steps:

- selects the slot visitor,
- highlights the text “*Christine Large*” and
- presses the ‘Insert’ button.

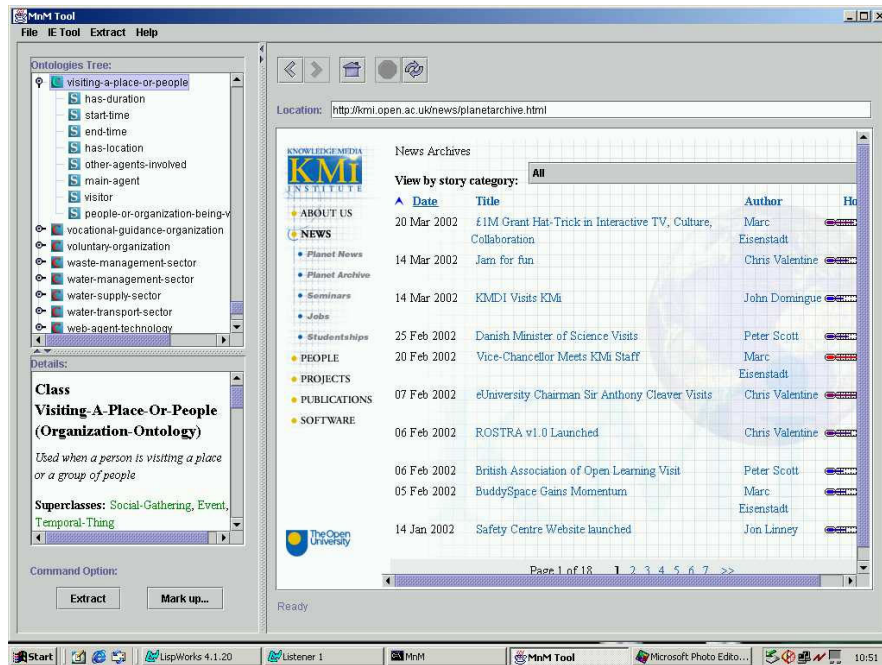


Fig. 3. A screen snapshot showing the class visiting-a-place-or-people in the event hierarchy

The SGML tags `<vapop_visitor>` and `</vapop_visitor>` are inserted into the page. The name of the tag “vapop_visitor” stands for “visiting-a-place-or-people” (vapop) class and “visitor” is the selected slot in the class vapop. The user continues to mark up a number of visit stories in a similar fashion before moving into the learn phase. The marked up stories are stored in a directory (c:\AKTProject\TestCorpus\visiting\)

on the local machine. It is possible to reuse annotated stories. This might be important if we want to use the training set for a different extraction purpose (i.e., we might want to add/remove tags).

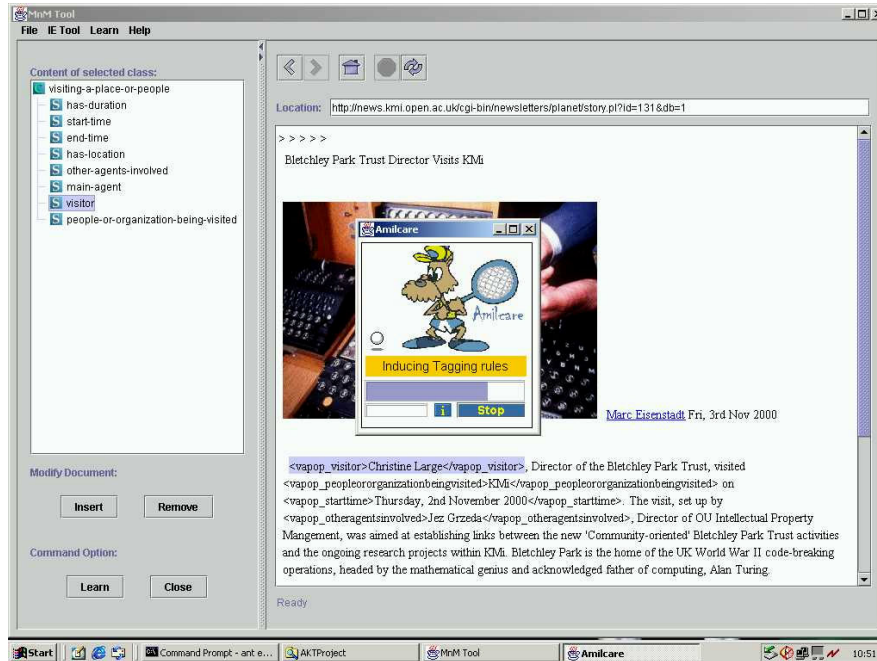


Fig. 4. A screen snapshot showing a marked up KMi Planet story and Amilcare

The user initiates the learning phase of the IE mechanism to produce rules for visit stories by specifying the location of the corpus of marked up visit stories (held in `c:\AKTProject\TestCorpus\visiting\`) and selecting the ‘Learn’ button. This causes Amilcare to start up – the Amilcare status window can be seen in figure 4. At this stage Amilcare learns rules for the event “visiting-a-place-or-people”.

During the extraction phase the user selects a set of rules and the input set of documents. The input set can either be a directory on the local disk or a URL pointing to a directory of documents. In our example the user has selected a local directory containing a set of planet stories. In figure 5 below Amilcare has finished extracting instances from the input set and the user is checking the created instances. In the top left panel the user has selected the third extracted item. The bottom left panel shows the instance slot values extracted and the web browser on the right shows the source KMi Planet story with the matched text segments highlighted. This view enables the user to quickly determine if the extracted data is correct.

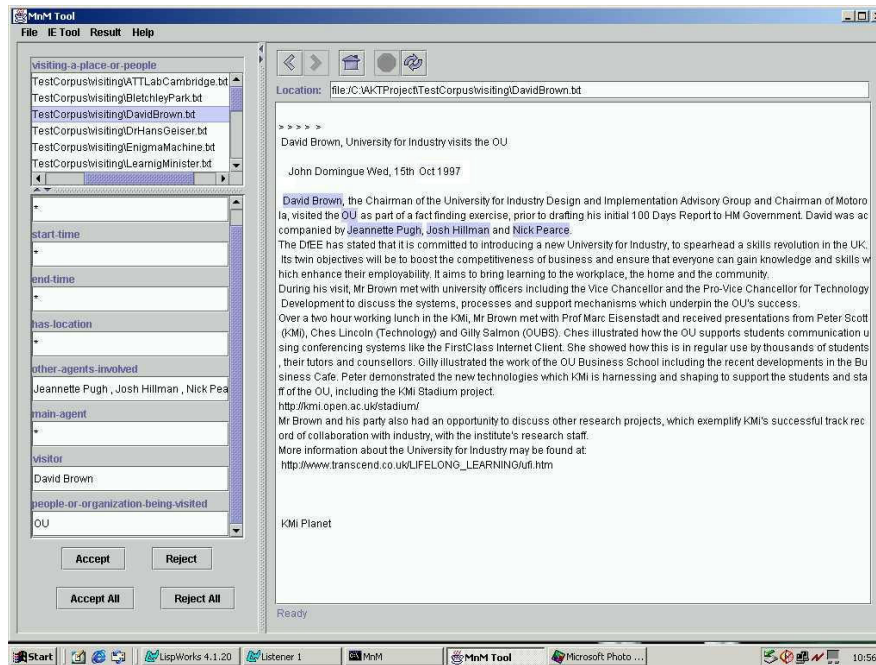


Fig. 5. A screen snapshot showing the result of the extraction phase

4 RELATED WORK

A number of annotation tools for producing semantic markup exist. The most interesting of these are Annotea [16]; SHOE Knowledge Annotator [15]; the COHSE annotator [1]; AeroDAML [18]; and, OntoMat, a tool being developed using the CREAM annotation framework [13]. A commercial version of OntoMat is available as OntoAnnotate (http://www.ontoprise.de/com/co_produ_tool2.htm).

Annotea provides RDF-based markup but it does not support information extraction nor is it linked to an ontology server. It does, however, have an annotation server which makes annotations publicly available. SHOE Knowledge Annotator allows users to mark up pages in SHOE guided by ontologies available locally or via a URL. These marked up pages can be reasoned about by SHOE-aware tools such as SHOE Search. The COHSE annotator uses an ontology server to mark up pages in DAML+OIL. The results can be saved as RDF. AeroDAML is available as a web page. The user simply enters a URL and the system automatically returns DAML annotations on a web page using a predefined ontology based on WordNet.

Of the systems listed above, OntoMat is closest to MnM both in spirit and in functionality. Both can provide some form of automated extraction. However, while MnM makes it possible to access ontology servers through APIs, such as OKBC, and also

to access ontologies specified in a markup format, such as RDF and DAML+OIL, OntoMat only provides the latter functionality. In contrast with OntoMat, MnM can handle multiple ontologies at the same time, which makes it very easy to switch from one to another, and also allows inherited definitions to be displayed for ontology editing and browsing. On the other hand, OntoMat can store pages annotated in DAML+OIL using OntoBroker as an annotation server. It also provides crawlers which can search the Web for marked up pages for addition to its internal knowledge base.

While both MnM and OntoMat are very similar they illustrate a slight difference of emphasis in providing tools for the Semantic Web. While OntoMat adopts the philosophy that the markup which indicates the knowledge content of a web resources should be included as part of that resource, MnM's annotations are stored both as markup on a page and as items in a knowledge base held on the WebOnto combined ontology and knowledge base server.

5 CONCLUSIONS

In this paper we have described MnM, an ontology-based annotation tool which provides both automated and semi-automated support for annotating web pages with semantic contents. The first prototype of the system has now been completed and tested with both Amilcare and the UMass set of tools. The early results are encouraging in terms of the quality and robustness of our current implementation, however, there is clearly a lot more work needed to make this technology easy to use for our target user base (people who are neither experts in language technologies nor 'power knowledge engineers'). In particular, all the activities associated with automated markup tend to be very sensitive to the quality of markup and to the appropriateness of the chosen corpora. Amilcare already attempts to address some of these issues through its adaptive mechanisms, however, more work is needed in this area. In addition, we also plan to do more work on the user interface, in particular with respect to the integration of markup, ontology browsing and the 'semantic navigation' of web pages. Currently, ontology and web browsing are integrated with respect to contents annotation, but ontologies do not inform the web browsing component of MnM directly. Our vision for the semantic web is one in which new forms of 'conceptual navigation' will emerge, where association between resources will be semantic as well as hypertextual. We plan to experiment with these ideas and extend the interface of MnM to support novel, markup-driven forms of web browsing, as well as the standard HTML based ones.

ACKNOWLEDGEMENTS

This work was funded by the Advanced Knowledge Technologies (AKT) Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. The

AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University. The authors would like to thank Maruf Hasan and Simon Buckingham Shum for their invaluable help in reviewing the first draft of this paper.

REFERENCES

1. S. Bechhofer and C. Goble: Towards Annotation Using DAML+OIL. First International Conference on Knowledge Capture (K-CAP 2001). Workshop on Semantic Markup and Annotation. Victoria, B.C., Canada. October 2001.
2. D. Brickley, and R. Guha: Resource Description Framework(RDF) Schema Specification 1.0. Candidate recommendation, World Wide Web Consortium, 2000. URL: <http://www.w3.org/TR/2000/CR-rdf-schema-20000327>.
3. F. Ciravegna: Adaptive Information Extraction from Text by Rule Induction and Generalisation, Proc. of 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle, August 2001.
4. F. Ciravegna: LP² an Adaptive Algorithm for Information Extraction from Web-related Texts. Proc. of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining held in conjunction with the 17th International Conference on Artificial Intelligence (IJCAI-01), August, 2001.
5. F. Ciravegna: Challenges in Information Extraction from Text for Knowledge Management in IEEE Intelligent Systems and Their Applications, November 2001, (Trend and Controversies).
6. F. Ciravegna and D. Petrelli: User Involvement in Adaptive Information Extraction: Position Paper in Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining held in conjunction with the 17th International Conference on Artificial Intelligence (IJCAI-01), August, 2001.
7. J. Domingue: Tazebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web. Proceedings of the 11th Banff Knowledge Acquisition Workshop, Banff, Alberta, Canada, April 18-23, 1998.
8. J. Domingue and P. Scott: KMi Planet: A Web Based News Server. Asia Pacific Computer Human Interaction Conference (APCHI'98), Shonan Village Center, Hayama-machi, Kanagawa, Japan, 15-17 July, 1998.
9. J. Domingue and E. Motta: Planet-Onto: From News Publishing to Integrated Knowledge Management Support. IEEE Intelligent Systems Special Issue on "Knowledge Management and Knowledge Distribution over the Internet", May/June, 2000, pp. 26-32. (ISSN 1094-7167).
10. E. A. Feigenbaum: The art of artificial intelligence 1: Themes and case studies of knowledge engineering. Technical report, Pub. no. STAN-SC-77-621, Stanford University, Department of Computer Science, 1977.
11. D. Fensel. and E. Motta: Structured Development of Problem Solving Methods. Transactions on Knowledge and Data Engineering 13(6):9131-932, 2001.
12. T. R. Gruber: A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2), 199-220, 1993.
13. S. Handschuh and S. Staab and A. Maedche: CREAM- Creating relational metadata with a component-based, ontology-driven annotation framework. First International Conference on Knowledge Capture (K-CAP 2001), Victoria B.C., October 2001.
14. P. Hayes: RDF Model Theory, W3C Working Draft, February 2002 URL: <http://www.w3.org/TR/rdf-mt/>.

15. J. Heflin and J. Hendler: A Portrait of the Semantic Web in Action. IEEE Intelligent Systems, 16(2), 2001.
16. J. Kahan and M. Koivunen and E. Prud'Hommeaux and R. Swick: Annotea: Open RDF Infrastructure for Shared Web Annotations. In Proc. of the WWW10 International Conference. Hong Kong, 2001.
17. Y. Kalfoglou and J. Domingue and E. Motta and M. Vargas-Vera and S. Buckingham Shum: MyPlanet: an ontology-driven Web based personalised news service. Proceedings of the IJCAI'01 workshop on Ontologies and Information Sharing, Seattle, WA, USA 2001.
18. P. Kogut and W. Holmes: AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages. First International Conference on Knowledge Capture (K-CAP 2001). Workshop on Knowledge Markup and Semantic Annotation, Victoria, B.C., Canada, October 2001.
19. N. Kushmerick and D. Weld and R. Doorenbos: Wrapper induction for information extraction, Proc. of 15th International Conference on Artificial Intelligence, IJCAI-97.
20. O. Lassila and R. Swick: Resource Description Framework (RDF): Model and Syntax Specification. Recommendation, World Wide Web Consortium, 1999. URL: <http://www.w3.org/TR/REC-rdf-syntax/>.
21. E. Riloff: An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains. *The AI Journal*, 85, 101-134, 1996.
22. D. Maynard and V. Tablan and H. Cunningham and C. Ursu and O. Saggion and K. Bontcheva and Y. Wilks: Architectural Elements of Language Engineering Robustness. *Journal of Natural Language Engineering – Special Issue on Robust Methods in Analysis of Natural Language Data*, forthcoming, 2002.
23. S. McIlraith and T. C. Son and H. Zeng: Semantic Web Services, IEEE Intelligent Systems, Special Issue on the Semantic Web, Volume 16, No. 2, pp. 46-53, March/April, 2001.
24. R. S. Mickalski and I. Mozetic and J. Hong and H. Lavarack: The multi purpose incremental learning system AQ15 and its testing application to three medical domains', in Proceedings of the 5th National Conference on Artificial Intelligence, Philadelphia. Morgan Kaufmann publisher, 1986.
25. E. Motta: *Reusable Components for Knowledge Models*. IOS Press, Amsterdam, 1999.
26. E. Riloff: An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains. *The AI Journal*, 85, 101-134, 1996.
27. S. Staab and A. Mädche and S. Handschuh: An Annotation Framework for the Semantic Web. In: S. Ishizaki (ed.), *Proc. of The First International Workshop on MultiMedia Annotation*. January, 30 - 31, 2001. Tokyo, Japan.
28. M. Vargas-Vera and J. Domingue and Y. Kalfoglou and E. Motta and S. Buckingham-Shum: Template-driven information extraction for populating ontologies. *Proc of the IJCAI'01 Workshop on Ontology Learning*, Seattle, WA, USA 2001.
29. M. Vargas-Vera and E. Motta and J. Domingue and S. Buckingham Shum and M. Lanzoni: Knowledge Extraction by using an Ontology-bases Annotation Tool. First International Conference on Knowledge Capture (K-CAP 2001). *Workshop on Knowledge Markup and Semantic Annotation*, Victoria B.C., Canada, October 2001.