MnM: A Tool for Automatic Support on Semantic Markup

Maria Vargas-Vera[1], Enrico Motta [1], John Domingue [1], Mattia Lanzoni [1], Arthur Stutt [1] and Fabio Ciravegna [2]

[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, 211Portobello 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.

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 (Hayes (2002), Lassila and Swick (1999)) and RDFS (Brickley and Guha (2000)) 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 OWL the language proposed by the WebOnt group (http://www.w3.org), are building on the RDF base framework to provide more sophisticated knowledge representation support. Ontologies (Gruber (1993)) provide the mechanism to support interoperability at a conceptual 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

(McIIraith, Son and Zeng(2001), Motta (1999), Fensel and Motta (2001)), 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* (Feigenbaum (1977)). 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. Section 4 will present related work. Finally sections 5 and 6 discuss evaluation 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 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 information extraction 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 information extraction 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.
- *Extract*. An information extraction mechanism is selected and run over a set of test documents. Then extracted information is used to populate the selected ontology. 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 exist-

ing items - for example the classes, slots of a specific class and so on. If the user selects the AKT reference ontology then he/she could explore people, organizations, publications, technologies and events. Items within the AKT reference ontology can be selected as the starting point for the information extraction process. More specifically, the selected class forms the basis for a template construction which will eventually be instantiated in the extraction activity.

An example of template could be for example the template for the visiting-a-place-people which consists of the slot visitor, people-or-organization-visited, has-location, other-agents, main-agent, has-duration, start-time and end-time.

The ontology browser window (in MnM version 2) is composed of five viewers: *QSearch viewer*, *Ontology viewer*, *Instance viewer*, *Information* and *Status viewer*. Each of the components are described as follows:

QSearch viewer allows users to perform incremental searches in the ontology viewer (if **on** is selected) or in the instance viewer (if **in** is selected).

Ontology viewer displays the ontology structure as a tree structure (ontologies, classes and slots). A class might have different icons depending on whether or not it has associated a library of information extraction rules.

Instance viewer presents to the user the information of instances belonging to the selected class. A right click on a instance will pop up a menu with the following options import mark-up, rename and remove. A double click on an instance will open a new dialog box. In this box the user could modify the instance manually.

Information viewer shows information about the selected ontology, selected instances. All information provided is in HTML format. A double click on a piece of test means that the user wants more information. It has some basic features such as go back, go forward, home and history management.

Status viewer monitors the progress of background learning and background knowledge.

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 the hand-crafted

AKT reference ontology (ontology describing people, organizations, publication, research areas, technologies and events/news).

MnM provides means to browse the event hierarchy (defined in the AKT reference 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 slot names of a selected class as given in the ontology (i.e., ontology driven mark-up). As the text is selected MnM inserts the relevant SGML tags into the document. MnM also offers the possibility removing tags from a document.

Learning

MnM integrates web browsing, ontology browsing and information extraction development. It does not have a built-in information extraction tool but provides a plug-in interface which allows the integration of information extraction tools easily. In version -1 of our MnM we integrated Marmot, Badger and Crystal from the University of Massachusetts (UMass) (Riloff (1996)) and our own NLP components (i.e., OCML preprocessor). A full description of this version can be found in (Vargas-Vera, Domingue, Kalfoglou, Motta and Buckingham Shum(2001a), Vargas-Vera, Motta Domingue and Buckingham Shum (2001b)). Version-2 of MnM uses as information extraction engine Amilcare of Sheffield University (Ciravegna (2001a)). Both versions of MnM were trained using stories that we had collected in our institution. These stories describe events happening in our institution such as visits, project awards, etc. 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 (Ciravegna (2001a)).

Amilcare is designed to support active annotation of documents. It performs information extraction 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, information extraction 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 information extraction experts, a future version will provide support for naïve users (Ciravegna and Petrelli (2001)).

At the start of the learning phase Amilcare preprocesses texts using Annie, the shallow IE system included in the Gate package (Maynard, Tablan, Cunningham, Ursu and Saggion, Bontcheva and Wilks (2002)), 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 LP^2, a covering algorithm for supervised learning of IE rules based on Lazy-NLP (Ciravegna (2001a), Ciravegna (2001b)). This is a wrapper induction methodology (Kushmerick and Weld and Doorenbos (1997)) 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 (LP^2) algorithm can be found in (Ciravegna (2001b)).

All the generalizations are tested in parallel by using a variant of the AQ algorithm (Michalski and Mozetic and Hong and Lavrack (1986)) 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 an-

nouncements, but not to spot the seminar location. This strategy is quite effective for analyzing documents with mixed genres, a common situation in web documents (Ciravegna (2001c)).

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 information extraction 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.

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 (Mynard et al. (2002)). 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 information extraction 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 information extraction 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 could be 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. However, our current prototype does not provide with this feature yet. After the extraction the user could accept or reject each single extracted information or accept or reject all the extracted information.

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 (Domingue and Scott (1998)), that has been running in our laboratory 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 (Domingue and Mottta (2000), Kalfoglou, Domingue, Motta, Vargas-Vera and Buckingham Shum (2001)). 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. In the Planet stories are indexed using by the measure of ‘’popularity’’ (i.e. how popular the story has been between our readers).

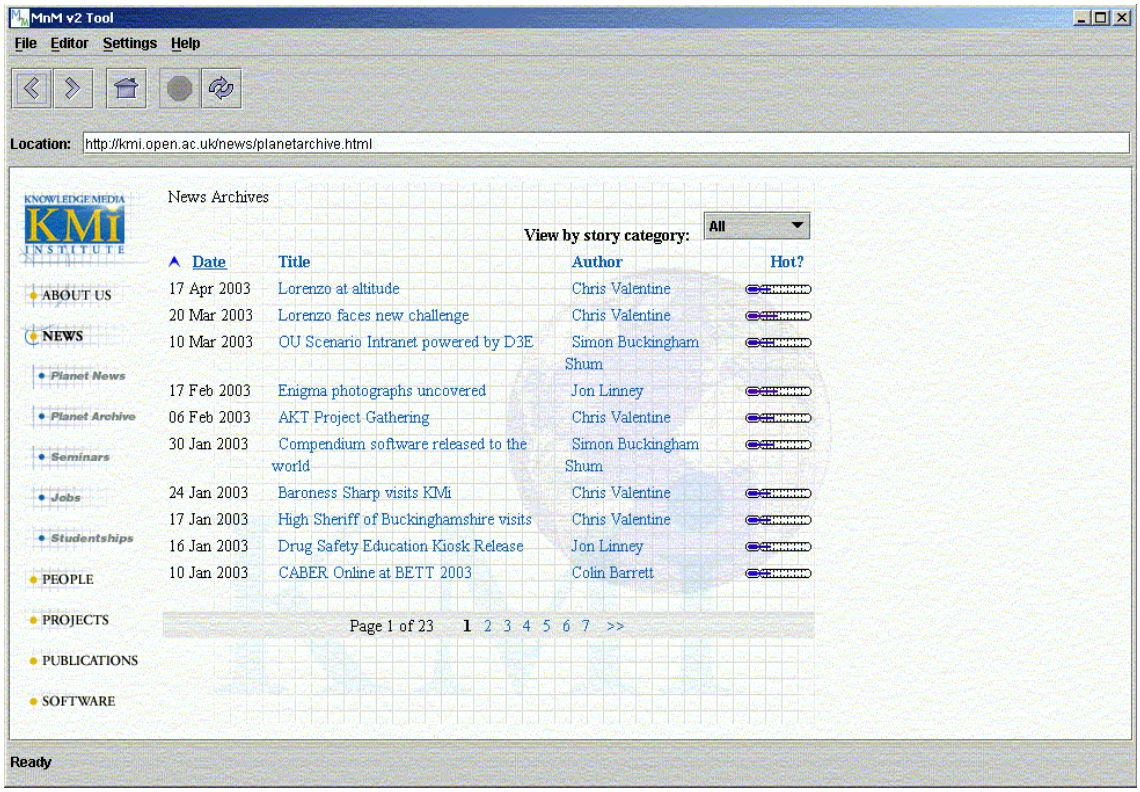

Figure 1. A screen snapshot of the KMi Planet front page

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 (Domingue (1998)) for a description). This knowledge base builds on a dozen ontologies describing domains such as our laboratory, events, organisations research areas, technologies and events/news.

Figures 2-5 show a user setting up an information extraction mechanism for extracting Planet stories about visits to KMi. In figure 2 we can see that MnM consists of five components (QSearch viewer, ontology viewer, instance viewer, information and status viewer). In ontology viewer window we could see that the user has selected the academic conference class. Then, in the instance viewer (middle window) MnM displays instances of academic conference event such as conference *aisb95*. Finally, in the bottom window MnM is displaying detailed structure of the class academic-conference such as superclass and so on..

Figures 2 and 3 show the initial steps in creating the visit event information extraction 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.

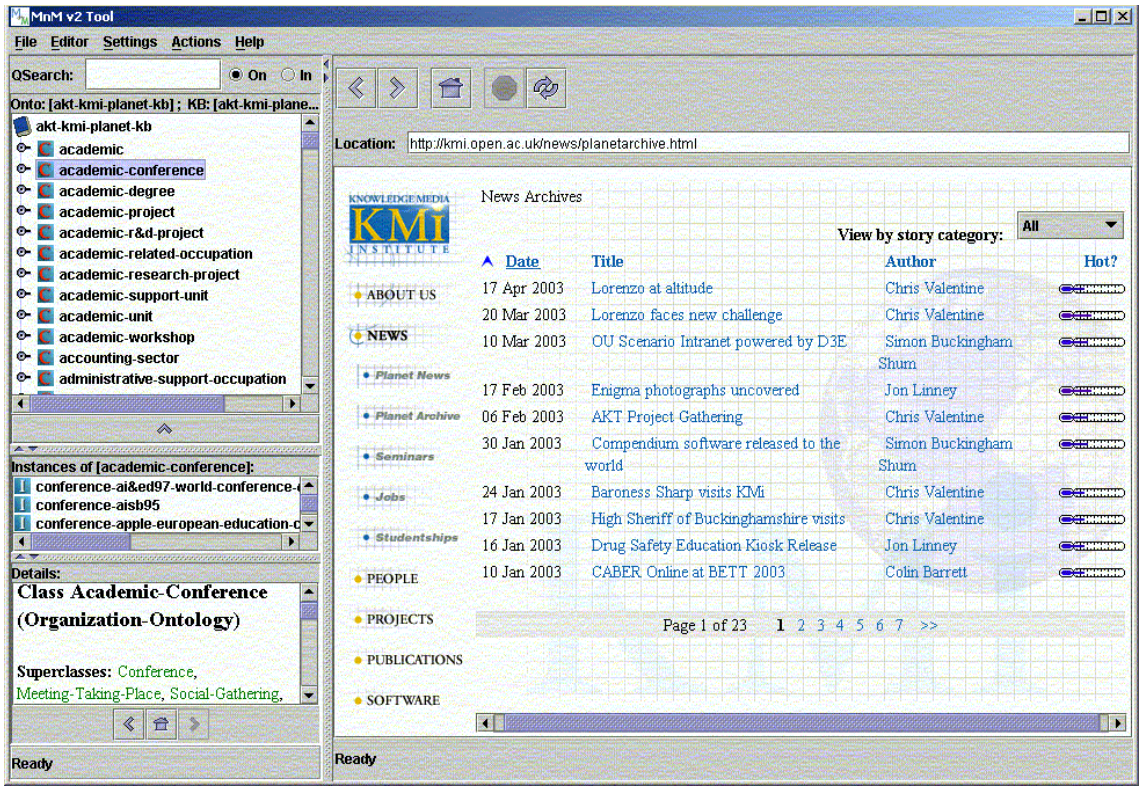

Figure 2. A screen snapshot showing a user browsing the library of knowledge models held on the WebOnto server

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
- double-click on the slot.

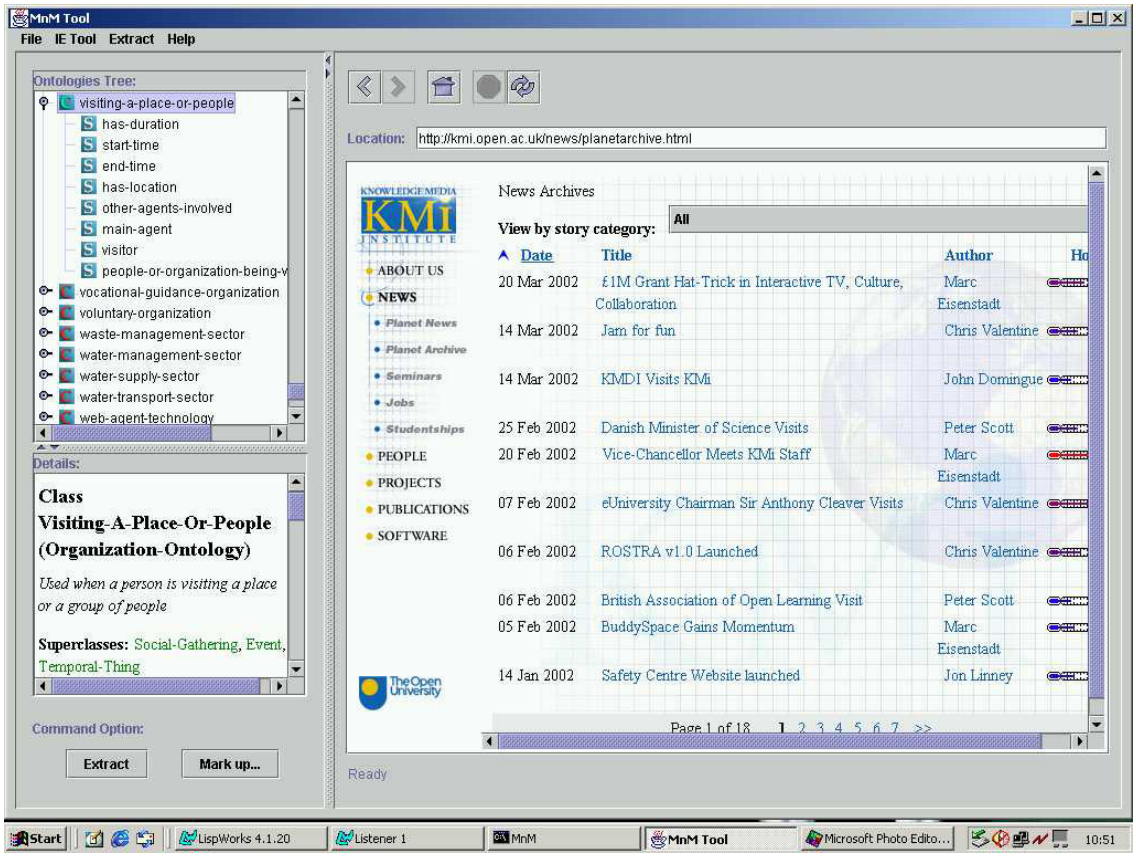

Figure 3. A screen snapshot showing the class visiting-a-place-or-people in the event hierarchy

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 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).

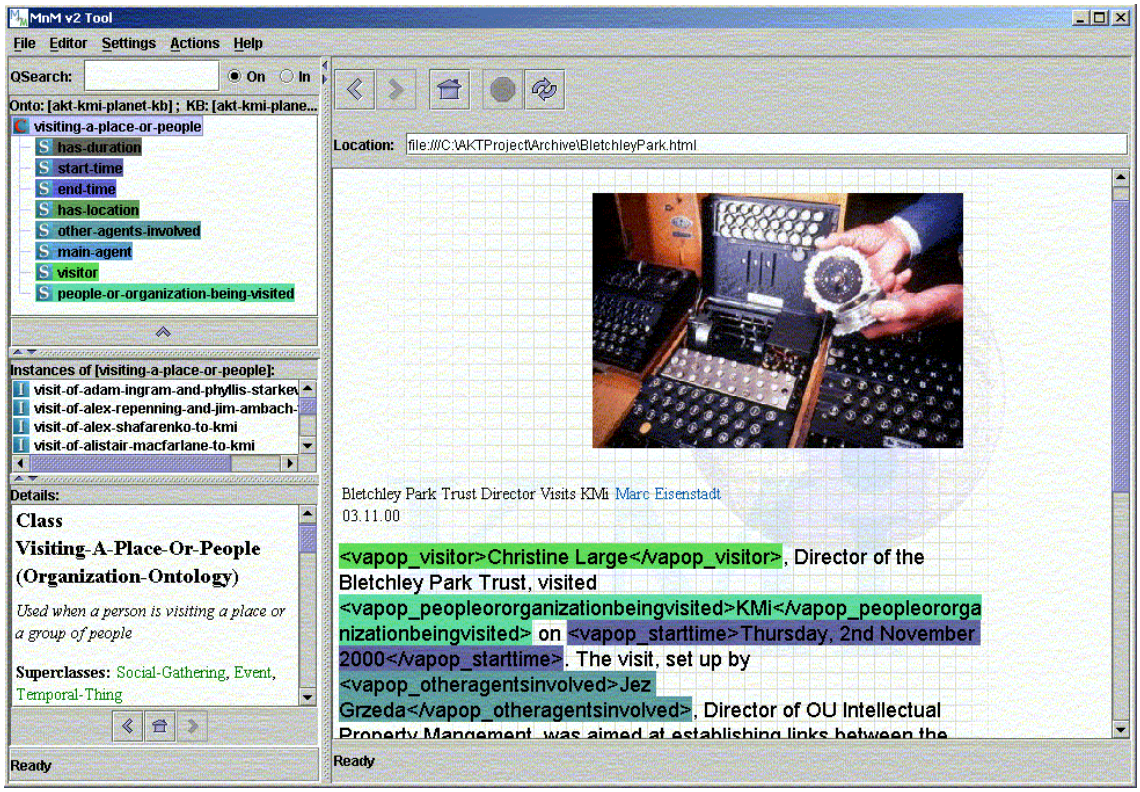

Figure 4. A screen snapshot showing a marked up KMi Planet story and Amilcare

The user initiates the learning phase of the information extraction mechanism to produce rules for visit stories by specifying the location of the corpus of marked up visit stories and selecting the 'Learn' button from the "Actions menu''. 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.

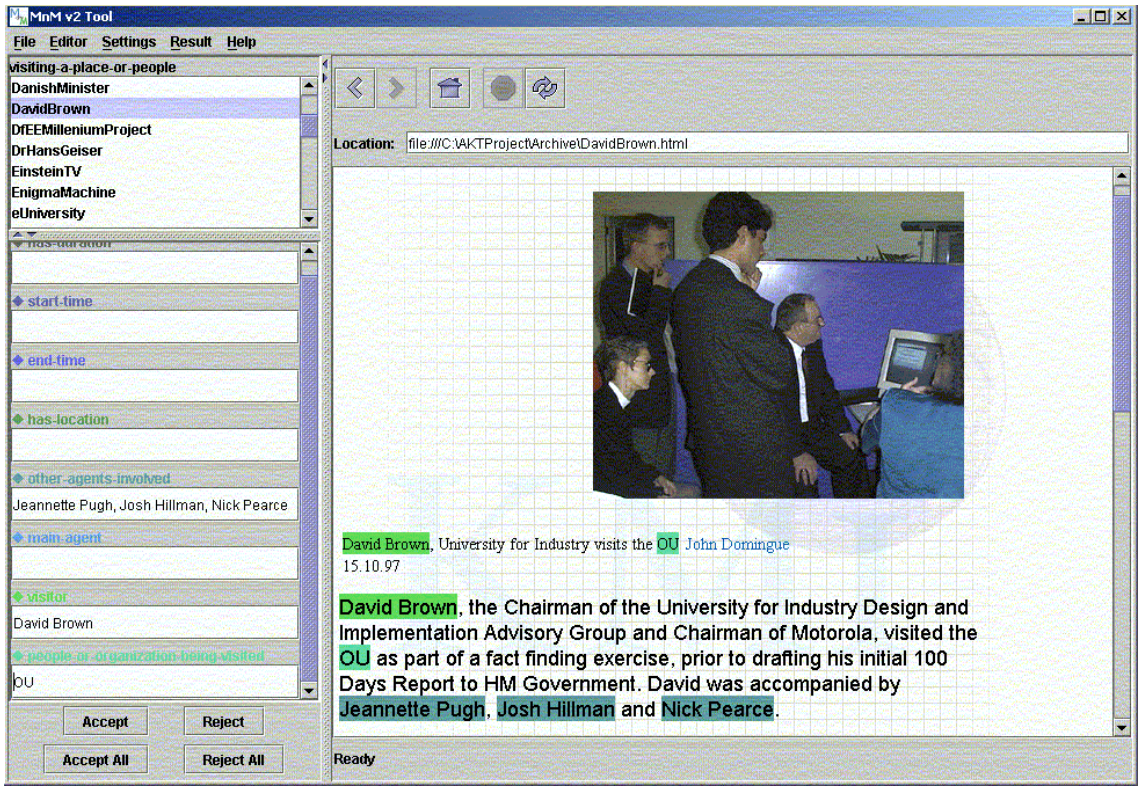

Fig. 5. A screen snapshot showing the result of the extraction phase

4 RELATED WORK

A number of annotation tools for producing semantic markup exist. The most interesting of these are Annotea (Kahan et al., 2001); SHOE Knowledge Annotator (Heflin and Hendler, 2001); the COHSE Mozilla Annotator (Bechhofer and Goble, 2001); AeroDAML (Kogut and Holmes, 2001); Melita (Ciravegna et al 2002) and, OntoMat-Annotizer, a tool being developed using the CREAM annotation framework (Handschuh et al., 2001).

Annotea provides RDF-based markup but 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. SHOE-aware tools such as SHOE Search can query these marked up pages.

The COHSE Mozilla Annotator uses an ontology server to mark up pages in DAML. 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 another web page using a predefined ontology based on WordNet.

Melita adopts an approach similar to MnM in providing information extraction-based semantic annotation. Work on Melita has focused on Human COmputer Interaction issues such as limiting intrusivity of the information extraction system and maximizing proactivity and timeliness in suggestions. Melita does not provide sophisticated access to the ontology, as MnM provides. In this sense Melita explored issues that are complementary to those explored in developing MnM and the two approaches could be integrated.

OntoMat is closest to MnM both in spirit and in functionality. Both allow browsing of predefined ontologies as a means of annotating the web pages displayed using their HTML browsers. Both can save the annotations in the document or as a knowledge base. They differ in that MnM already provides automated extraction currently only planned for Ontomat.

5 EVALUATION

Experience in using MnM with Amilcare suggests that the following issues need to be addressed if information extraction is to be a viable solution to the problem of populating ontologies.

- Currently, for realistic cases, MnM requires (a) that a single concept at a time is used to mark up training documents and (b) that between 20 and 30 pages are marked up for each learning phase. This means that if there are 100 concepts in an ontology there will need to be up to 3000 manual markups. This is quite a big effort for a user and not many would be willing to do it. Of course, without information extraction , the same annotation would be needed for manual annotation of documents. Moreover it should be done for all the other documents to be annotated afterwards, while a trained information extraction system would just require those 3,000. Anyway it shows the necessity of methodologies able to learn from unannotated corpora (Ciravegna et al 2003).
- Amilcare is able to recognize concept instances and values, but it is not able to establish explicit relations among them. For this reason. if a document contains more than one instance of a concept, then Amilcare will not be able to allocate the correct properties to the correct instance because it is unable to differentiate among them. A typical example is a home page with several

names and phone numbers. Amilcare would not be able to assign phone numbers to persons. This can be avoided by ensuring that no document has more than one instance of a concept. Ideally, however, information extraction systems should make use of concept-property structures when suggesting annotations. Since an ontology contains the conceptual structure of a domain and the populated ontology contains domain specific knowledge, it would seem sensible to make use of this when performing information extraction. This suggests that there should be a two way exchange with information extraction modules - the results of information extraction should be used to populate knowledge bases and the contents of knowledge bases should be available for the information extraction process. Even if IE systems cannot make use of specific domain knowledge they should still be able to make use of its domain-specific structure. This direction of research has already been reported in Vargas-Vera et al. (2001b).

- In our experience in using information extraction in MnM, we have become aware that users need to be able to judge when a particular corpus of texts is suitable for information extraction. For example, in using Amilcare we discovered that the apparently easy task of extracting information from Amazon.com's pages turned out to be quite difficult (if not impossible). There can be multiple reasons for this sort of failure, from limitations in the specific information extraction system capabilities to a misunderstanding on the user's part of what is feasible with the existing technology. We need to investigate how to overcome this opacity problem. One solution may be to provide assistance either by training users or providing sets of guidelines which, for instance, map different information extraction modules to different types of corpus. We also need to indicate how many texts need to be marked up manually. For example, for some specific text types (e.g. free texts) a quite large number of cases are needed in order to train the information extraction system properly. Inexperienced users could think that the texts are not suitable because they do not receive the same early feedback from the information extraction system as they have when using quite structured documents. Again, understanding why the problem arises becomes of fundamental importance for the usability of these technologies.

6 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. 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. It is also planned to include an additional component based on Programming By Example technology which can learn new annotations, store these in a library and use them in critiquing user annotations. It is likely that information extraction could be used as part of this (machine) learning and critiquing process.

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.

REFERENCES

Bechhofer, S. and Goble, C. (2001). Towards Annotation Using DAML+OIL. First International Conference on Knowledge Capture (K-CAP 2001). Workshop on Semantic Markup and Annotation. Victoria, B.C., Canada.

Brickley D. , and Guha R. (2000). 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.

Ciravegna F. (2001a) Adaptive Information Extraction from Text by Rule Induction and Generalisation, Proc. of 17th International Joint Conference on Artificial Intelligence (IJCAI - 2001) .

Ciravegna F. (2001b). LP^2 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).

Ciravegna F. (2001c). Challenges in Information Extraction from Text for Knowledge Management in IEEE Intelligent Systems and Their Applications, November 2001, (Trend and Controversies).

Ciravegna F. and Petrelli D. (2001). 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 17^{th} International Conference on Artificial Intelligence (IJCAI-01).

Ciravegna F., Dingli A., Petrelli D., and Wilks Y. (2002). User-System Cooperation in Document Annotation based on Information Extraction. Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02, publisher Springer Verlag.

Ciravegna F., Dingli A., Guthrie D. and Wilks Y. (2003). Integrating Information to Bootstrap Information Extraction from Web Sites. Proceedings of the {IJCAI} 2003 Workshop on Information Integration

on the Web, workshop in conjunction with the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003). Acapulco, Mexico, August, 9-15.

Domingue J. (19998). Tadzebao and WebOnto Discussing, Browsing, and Editing Ontologies on the Web. Proceedings of the 11th Banff Knowledge Acquisition Workshop, Banff, Alberta, Canada.

Domingue J. and Scott P. (1998). KMi Planet: A Web Based News Server. Asia Pacific Computer Human Interaction Conference (APCHI'98), Shonan Village Center, Hayama-machi, Kanagawa, Japan.

Domingue J. and Motta E. (2000). 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).

Feigenbaum E. A. (1977). The art of artificial intelligence 1: Themes and case studies of knowl edge engineering. Technical report, Pub. no. STAN-SC-77-621, Stanford University, Department of Computer Science.

Fensel D. and Motta E. (2001). Structured Development of Problem Solving Methods. Transactions on Knowledge and Data Engineering 13(6):9131-932, 2001.

Gruber. T. R. (1993). A Translation Approach to Portable Ontology Specifications.Knowledge Adquisition 5(2), 199-220.

Handschuh, S., Staab, S. and Maedche A. (2001). CREAM - Creating relational metadata with a component-based, ontology-driven annotation framework. First International Conference on Knowledge Capture (K-CAP 2001).

Hayes P. (2002). RDF Model Theory, W3C Working Draft, February 2002 URL: http://www.w3.org/TR/rdf-mt/.

Heflin, J. and Hendler, J. (2001). A Portrait of the Semantic Web in Action. IEEE Intelligent Systems, 16(2), 54-59.

Kahan, J., Koivunen, M-J., Prud'Hommeaux , E. and Swick, R. (2001). Annotea: An Open RDF Infrastructure for Shared Web Annotations. In Proc. of the WWW10 International Conference. Hong Kong.

Kalfoglou Y. and Domingue J. and Motta E. .and Vargas-Vera M. and Buckingham Shum S. (2001). MyPlanet: an ontology-driven Web based personalised news service. Proceedings of the IJCAI'01 workshop on Ontologies and Information Sharing, Seattle, WA, USA.

Kogut P. and Holmes W. AeroDAML (2001). 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.

Kushmerick N. and Weld D. and Doorenbos R. (1999). Wrapper induction for information extraction, Proc. of 15th International Conference on Artificial Intelligence, IJCAI-97.

Lassila O. and R. Swick R. (1999). Resource Description Framework (RDF): Model and Syntax Specification. Recommendation, World Wide Web Consortium, 1999. URL: http://www.w3.org/TR/REC-rdf-syntax/.

Riloff E. (1996). An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains. *The AI Journal,* 85, 101-134, 1996.

Maynard D. and Tablan V. and Cunningham H. and Ursu C. and Saggion O. and Bontcheva K. and Wilks Y. (2002). Architectural Elements of Language Engineering Robustness. *Journal o*f Natural Language Engineering – Special Issue on Robust Methods in Analysis of Natural Language Data.

McIlraith S. and Son.T. C. and Zeng H. (2001). Semantic Web Services, IEEE Intelligent Systems, Special Issue on the Semantic Web, Volume 16, No. 2, pp. 46-53.

Michalski R. S. and Mozetic I. and Hong J. and Lavrack H.(1986). 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.

Motta E. (1999). *Reusable Components for Knowledge Models*. IOS Press, Amsterdam.

Riloff E. (1996). An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains. *The AI Journal,* 85, 101-134.

Staab S. and Mädche A. and Handschuh S. (2001) An Annotation Framework for the Semantic Web. In: S. Ishizaki (ed.), *Proc. of The First International Workshop on MultiMedia Annotation.* January, 30 - 31, Tokyo, Japan.

Vargas-Vera M. and Domingue J. and Y. Kalfoglou Y. and Motta E. and Buckingham-Shum S. (2001). Template-driven information extraction for populating ontologies. *Proc of the IJCAI'01 Workshop on Ontology Learning,* Seattle, WA, USA.

Vargas-Vera M. and Motta E. and Domingue J. and Buckingham Shum S. and Lanzoni M.(2001). 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.