PlanetOnto: From News Publishing to Integrated Knowledge Management Support

John Domingue and Enrico Motta, Knowledge Media Institute

GIVEN A SCENARIO IN WHICH MEMBERS OF AN ACADEMIC COMMUNITY COLLABORATIVELY CONSTRUCT AND SHARE AN ARCHIVE OF NEWS ITEMS, SEVERAL KNOWLEDGE MANAGEMENT CHALLENGES ARISE. THE AUTHORS' INTEGRATED SUITE OF TOOLS, CALLED PLANETONTO, SUPPORTS A SPEEDY BUT HIGH-QUALITY PUBLISHING PROCESS, ALLOWS ONTOLOGY-DRIVEN DOCUMENT FORMALIZATION, AND AUGMENTS STANDARD BROWSING AND SEARCH FACILITIES WITH DEDUCTIVE KNOWLEDGE RETRIEVAL.

TO KEEP MEMBERS INFORMED, most organizations of a significant size produce and distribute an internal publication. KMI Planet is a Web-based news server that facilitates such communication in our laboratory—the Knowledge Media Institute—and lets the wider community access lab-related items of interest.[1] Unlike most news-of-the-future projects that focus on personalized news feeds, KMI Planet replicates some of a traditional newsroom's functionalities. "Journalists" send stories using the lowest-common-denominator medium, an e-mail message. The Planet news server acts as an intelligent layout editor, formatting the news item and related figures, integrating it into the news archive, and presenting it in several different modes. The most obvious presentation style is the newspaper-like multiple-story page, shown in Figure 1, where the server presents the submitted story in the context of other stories.

This low-overhead approach to news publishing has proven very successful. Today, Planet is often the "front door" to our laboratory, both metaphorically and physically (a dedicated machine running Planet is stationed at the lab's entrance so that KMI's visitors tend to be shown or play with Planet first). Our archive is growing steadily and now contains around 100 stories submitted by 13 journalists. We have 480 registered readers who have accessed our server or subscribed to our alert services. Moreover, the Planet technology has attracted interest from other organizations, both within and outside the Open University, leading to the development of additional, customized versions of the Planet news server.

Despite this success, as the Planet archive and readership grow, we need more sophisticated mechanisms to support semantic searches and individualized presentations and alerts. Having found an interesting news item in Planet, users cannot follow up easily with obvious queries. For instance, having read a story about an award for a paper about visualizing genetic algorithms, a user might want to find out who else works on software visualization, what other projects are going on in this area, and so on. Of course, browsing our Web pages and using standard search mechanisms can also generate this information. However, even in well-organized sites, gaps exist among the many important relations between people, technologies, projects, and organizations, leading to the standard knowledge management problem of finding out who does what and who knows what.

In addition to the need for better search and retrieval facilities, experiences in day-to-day use of Planet over more than two years have highlighted a number of other issues. An obvious one is the need for individualized news feeds and presentations: for now, registered readers periodically get a standard news update message. A less obvious issue is the need for Planet to move away from simply being a passive news archive and become a "real newspaper," where news is not just passively received and edited but proactively identified and requested, in accordance with department events and readers' interests.

To address these issues, we have devel-

1094-7167/00/$10.00 © 2000 IEEE

oped an integrated suite of tools, PlanetOnto, which extends the original Planet news server by providing support for ontology-driven document formalization, integrated browsing and deductive knowledge retrieval, personalized news feeds and alerts, and proactive identification of potentially interesting news items.

Planet-Onto as a knowledge management scenario

Daniel O'Leary proposes a typology of knowledge management activities as specialized connecting and converting processes.[2] Using O'Leary's framework, we view PlanetOnto as providing solutions for the following classes of knowledge management processes:

- *Converting individual knowledge to group knowledge*. In our scenario, this involves providing tools to support journalists who submit stories to a news archive.
- *Converting text to knowledge*. Formalizing the knowledge in a news item and integrating it into a knowledge base associated with the archive.
- *Connecting people to knowledge*. Providing integrated visualization, search, and query-answering facilities so that archive users can quickly find information at different levels of granularity, from collections of stories to specific data (whether they are explicitly included in a document or implied by a collection of documents).
- Connecting *knowledge to people*. Proactively contacting journalists to solicit stories that can plug holes in the archive, and readers when we publish items of interest.

Although we find O'Leary's framework useful, his use of the term *converting* is not necessarily the correct way to talk about knowledge transformation activities in our (and maybe any other) scenario. The problem is that the term has a translation-centered connotation, which is misleading. In particular, the formalization process in our scenario is driven by an ontology that defines the concepts needed to describe events related to academic life—for example, projects, products, seminars, publications, and so forth. This means that we ignore the parts of a news story that are not relevant to the ontology, much as in template-driven information extraction approaches. That is, we do not expect all knowledge expressed in textual form in a news item to be formalized. On the other hand, because the formalization is ontology-driven and instantiating an ontology might require the specification of knowledge not given in the story, the knowledge base associated with the news archive would normally contain knowledge that is not present in the archive. In sum, the knowledge base associated with the archive provides neither a subset nor a superset of the knowledge expressed in the stories. This is, of course, not surprising: any translation or encoding process has more to do with reconstructing meaning than with replacing representations. Hence, the metaphor of converting is not the right one when talking about formalizing knowledge in a document, both for fundamental reasons and, in our case, because of the different roles played by the news stories and the associated representations. Thus, we prefer to use the metaphor of *enriching a representation*.[3]

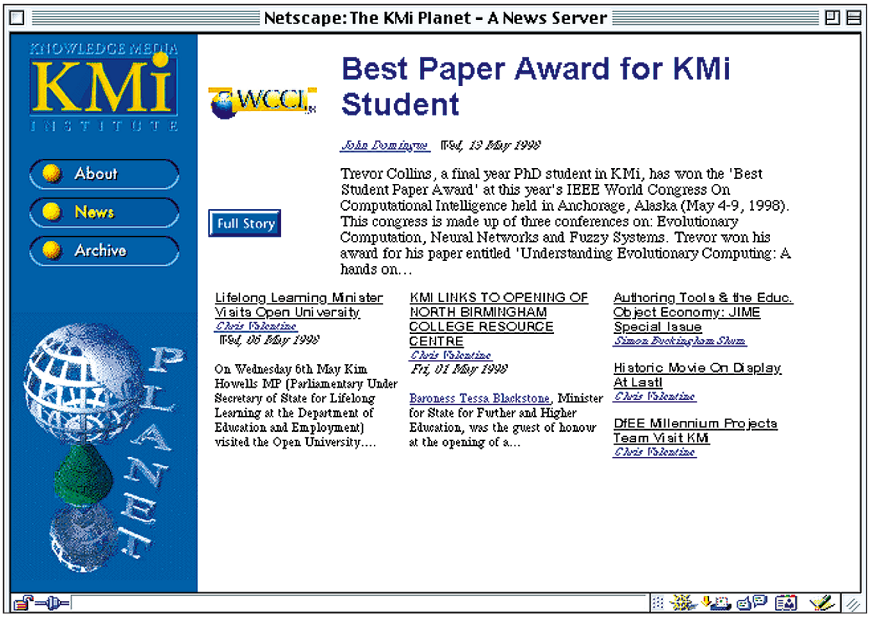

Figure 1. The front page of KMI Planet.

We can enrich a representation, whether formal, graphical, or textual, in several ways. For instance, we can extend it by

- providing information about the context in which it was created,
- linking it to related artifacts of the same nature, or
- linking it to related artifacts of a different nature.

In our scenario, we enrich a document (a news story) by associating it with a formal representation, which supports various forms of reasoning in PlanetOnto.

Architecture

Figure 2 summarizes both the architecture of PlanetOnto and the associated process model. In building PlanetOnto, we augmented the basic publish/find out/read scenario supported by KMI Planet. Three types of users exist: journalists who send stories to KMI Planet, knowledge editors who maintain the Planet ontology and the Planet knowledge base, and readers who read the Planet stories. We developed tools that allow the specification and association of knowledge structures to stories; an end-user interface, Lois, which integrates Web browsing with search and knowledge retrieval; and specialized agents that alert journalists to gaps in the archive and readers to relevant new stories. Specifically, PlanetOnto supports seven main activities (including those supported by the original Planet news server):

1. *Story submissi*on. A journalist submits a story to KMI Planet through plain e-mail. The story is formatted and stored in KMI Planet's story database.
2. *Story reading*. A Planet reader browses through the latest stories using a standard Web browser.
3. *Story annotation*. Either the journalist submitting a story or one of the Planet knowledge editors uses Knote to perform ontology-driven association of knowledge structures to the story.
4. *Provision of customized alerts.* An intelligent agent, Newsboy, builds user profiles from patterns of access to PlanetOnto and then uses these profiles to alert readers about relevant new stories.

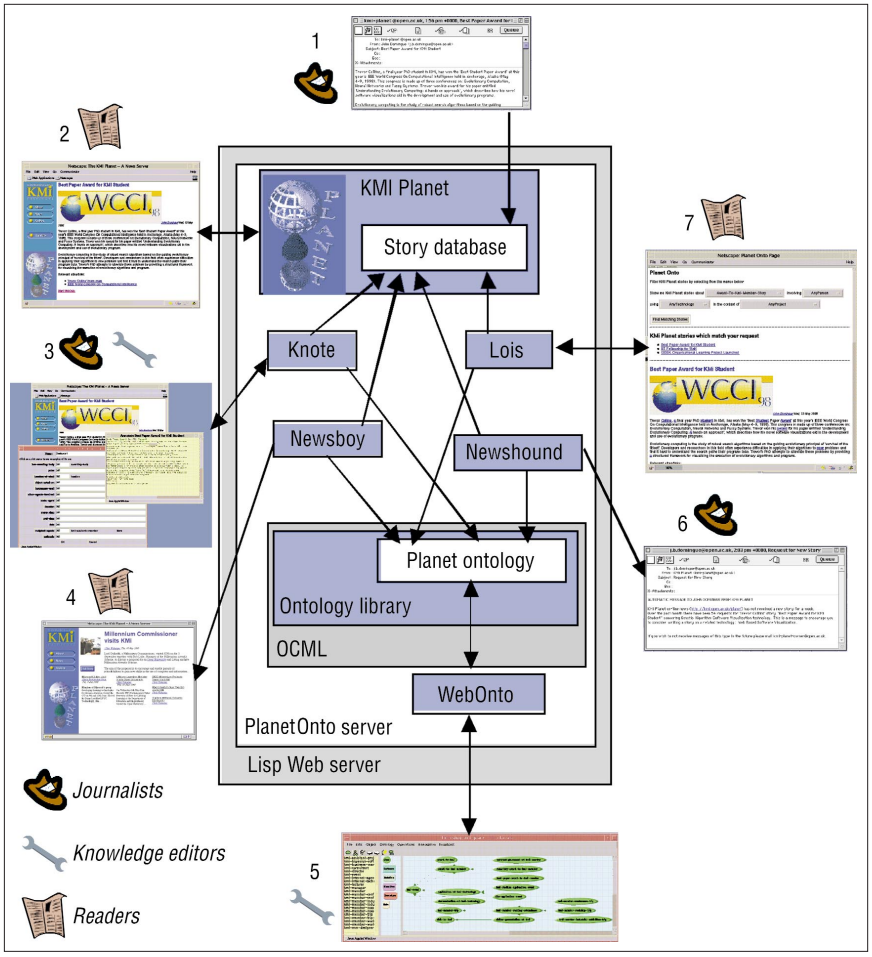

Figure 2. The PlanetOnto architecture.

5. *Ontology editing.* The WebOnto tool[4] provides Web-based visualization, browsing, and editing support for skilled editors to develop and maintain knowledge models specified in the Operational Conceptual Modeling Language (OCML).[5]
6. *Story soliciting.* An intelligent agent, Newshound, periodically gathers data about popular news items and then solicits potentially popular stories from the appropriate journalists. Newshound does this by identifying gaps in the knowledge base—for instance, finding projects in which there is missing information that potentially interesting stories could fill in.
7. *Story retrieval and query answering.* The Lois interface supports integrated access to the story archive and the associated knowledge base by integrating traditional Web browsing and search with knowledge-based query retrieval.

All of PlanetOnto's tools are Web-based; users can access them through ordinary browsers. As a result, a distributed community of users playing specialized roles can develop and use the archive and associated knowledge base. An HTTP server written in Common Lisp provides the underlying technology.

An ontology for characterizing academia-related events

An ontology provides a partial specification of a shared conceptualization, to be used in formulating knowledge-level theories about a domain. Our domain comprises the range of events associated with a university department, the persons who take part in these events, and the entities required to characterize these events and these persons in the context of academic life.

In developing the ontology, we have followed a task-independent, purpose-driven approach. Because the Planet ontology's main role is to drive the annotation of news items relating KMI events, we took the concept of *news item* as our starting point. We characterize each news item in terms of a number of mundane features, such as who wrote it, where and when it was published, and so on. More importantly, the essence of a news item is that of relating one or more *events*. Thus, the notion of event is central to our ontology.

Event modeling is an important aspect of research in problem solving and knowledge engineering, and several ontological characterizations are available in the literature.[6] Thus, rather than trying to reinvent a fairly complex wheel, we tried to reuse existing definitions—in particular, the public version of the CYC upper-level ontology, called HPKB (High Performance Knowledge Bases) Upper Level[6]—but this turned out to be a problem. The definition of the `event` class in the HPKB Upper Level ontology contains 94 slots to maximize reusability and account for all the features that could be associated with an event. Given the purpose of our ontology (to let users who are not necessarily knowledge engineers annotate stories), we are mainly interested in usability. In practice, this means adhering to two modeling guidelines:

- *Minimal ontological commitments.* The definition of class `event` in the HPKB upper layer provides an extreme case of a *coverage-centered* approach to reuse. That is, the definition aims to cover all potential attributes that could be relevant to a generic instance of the class. However, typically only a few slots will actually be relevant for any specific instance of the class. An alternative approach consists of minimizing ontological commitments—that is, to provide only the minimal set of attributes needed to define a class.[5,7] When populating the ontology, users don't have to face lots of irrelevant attributes.
- *User-centered definitions.* Users who are not necessarily knowledge engineers must easily understand the terminology used. Avoid heavily technical modeling concepts, such as sophisticated modeling solutions for representing time, and keep the terminology as context-specific as possible. For instance, we can talk about "agents performing events" when describing events in general, but we should use the class-specific terminology "awarding body assigns awards" when talking about an award-giving type of event. This implies that the underlying modeling language should support slot renaming along Isa hierarchies—that is, inherited slots should get subclass-specific names. The importance of domain-specific, user-oriented terminology has been recognized

Table 1. Definition of the `event` class.

Slot name	Documentation	Slot attributes
`main-agent`	The agents causing the event to happen. At least one main agent is assumed but there can be others.	The slot has a minimum cardinality of 1.
`other-agents-involved`	Other agents involved in the event.	
`instrument-used`	The instrument used by the main agent to carry out the action.	
`object-acted-on`	The things affected by the event; for example, in "John broke the stone with a hammer," the stone is the object acted on.	
`recipient-agents`	The agents that are affected by the event.	
`location-at-start`	The location at which an event takes place or starts in the case of events that change the position of something.	The value of the slot must be of type `location`.

in knowledge acquisition for a long time; it arguably provides an important difference between the criteria associated with modeling for knowledge acquisition and those associated with modeling for system development.

Thus, we decided to build our definitions by starting with minimalist concepts and then enriching them when defining specialized subclasses.

The definition of the class `event`, which inherits from the class `temporal-thing`, is shown in Table 1. This definition defines an event's essential aspects. For the sake of compatibility, the terminology reflects that used in the HPKB ontology, but different refinements of this class provide specialized terminology. For example, the class `award-to-kmi-member` renames the slot `main-agent` to `awarding-body` and the slot `object-acted-upon` to `awarded-prize`.

Planet ontology's current version includes about 40 event types, which have so far sufficed to cover the range of stories submitted, and about 450 definitions in all.

Annotating ontology-driven stories using Knote

We aim to enable as wide an audience as possible to annotate stories. Encoding representations of even a moderately growing repository such as KMI Planet can only succeed if the process is distributed as much as possible. Because users will range from regular Planet journalists to experienced ontology engineers, we designed Knote to be low-entry. At the same time, Knote should allow experienced ontology engineers the freedom to create arbitrarily complex OCML expressions.

Stories describe a number of KMI-related events. Story annotation is therefore the description of one or more events occurring within a story. Annotation involves four steps:

- choosing a story,
- selecting a particular event in the story to describe,
- classifying the event in terms of the ontology's hierarchy of event types, and
- filling in an automatically created instance definition form to characterize the new instance of class `event`.

The following scenario shows how Knote supports the annotation process. A Planet journalist decides to annotate the "Best Paper Award for KMI Student" story, shown in Figure 1. After some consideration, she classifies the event in question as an instance of the class `award-to-kmi-member` and hits the Describe Event button. This action creates an instance definition form, partially shown in Figure 3a. As the journalist starts filling in the recipient-agents slot, she can see from the form that the value for this slot must be of type `kmi-member`. She decides to see which more specific types are currently available by clicking on the `kmi-member` menu. This action displays the menu shown on the right side of Figure 3a. After choosing class `kmi-phd-student`, the journalist checks whether there are any defined instances of `kmi-phd-student` by clicking on the rightmost menu of the `recipient-agents` row, shown in Figure 3b. She chooses `collins` from the resulting menu, and the text "collins" is inserted into the recipient-agents value window. Alternatively, the journalist could have chosen to create a new instance of class `kmi-phd-student` by clicking on the menu item `New instance`. This would create a new-instance form and insert the name of the new instance as the value of slot `recipient-agents`.

Connecting people to knowledge

An important feature of the PlanetOnto architecture is the integration of traditional Web browsing, including lexical search, with

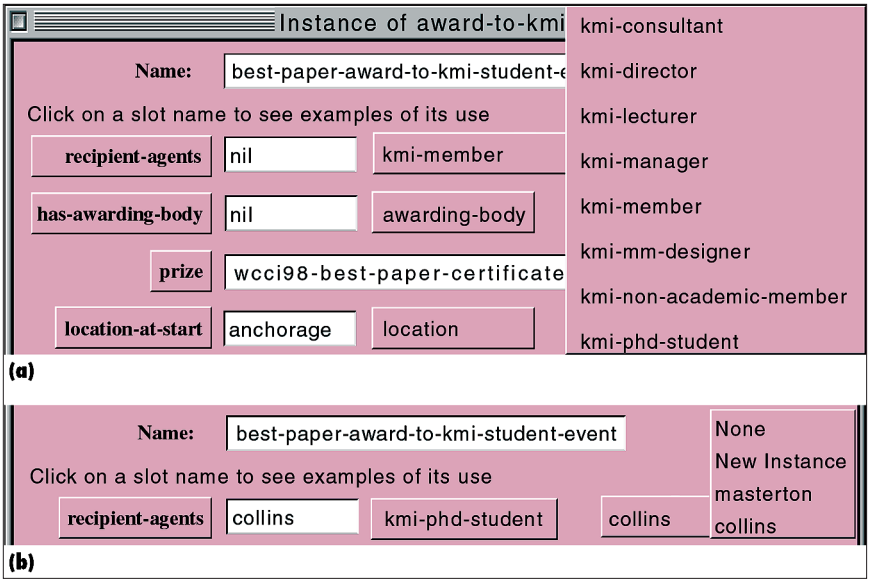

Figure 3. (a) A partially filled-in form for an instance of class `award-to-kmi-member`. (b) The appearance of the `recipient-agents` slot as the journalist fills in the value with the instance `collins`.

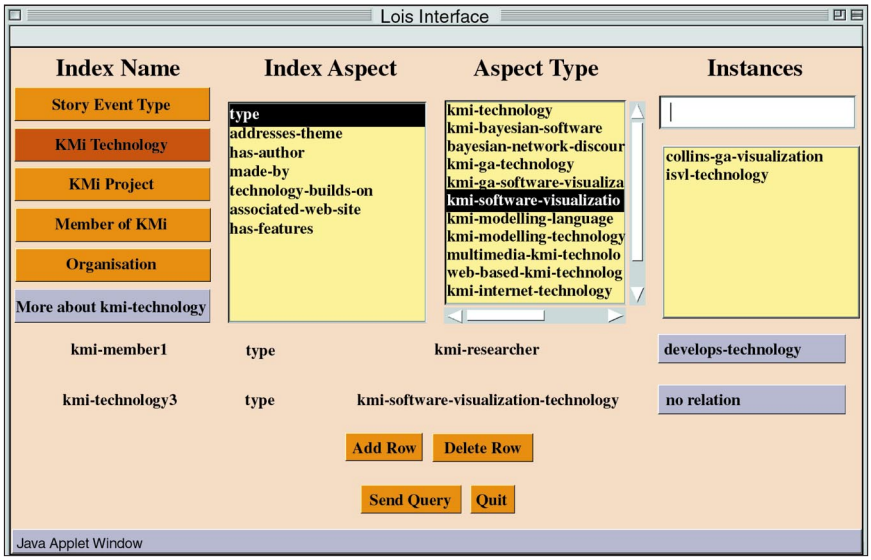

Figure 4. Finding a KMI researcher who works on software visualization.

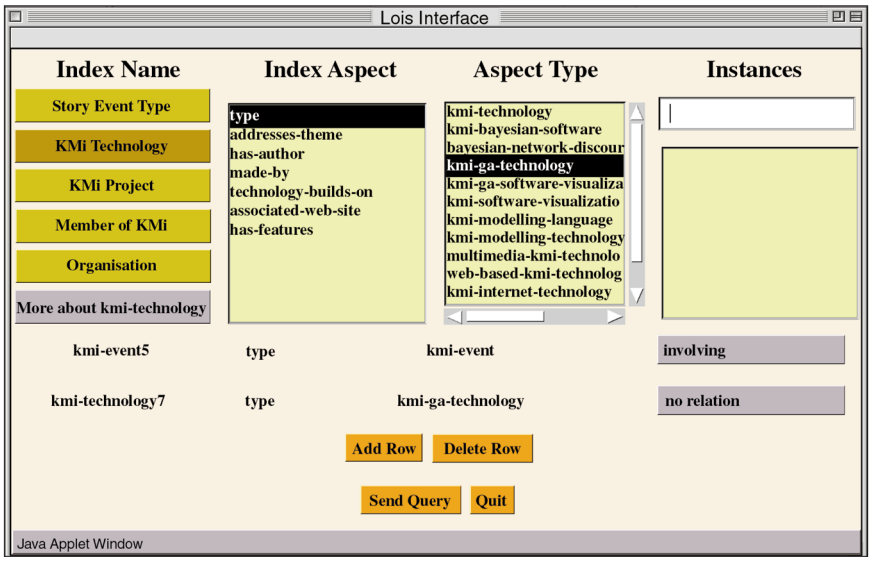

Figure 5. Finding a story that involves a specific KMI technology.

various levels of deductive knowledge retrieval. Using the WebOnto tool, experienced ontology engineers can directly access the Planet knowledge base and pose arbitrary queries expressed in OCML.[4] However, we assume that most readers either are not experienced knowledge engineers or, even if they are, might not want to interact with PlanetOnto at the OCML level.

Our solution was to use Planet ontology's basic epistemological building blocks (people, organizations, stories, events, projects, and technologies) to organize Lois, a form-based query interface for knowledge retrieval. The rationale for this approach is that, almost without exception, any useful query to PlanetOnto must include one or more subclasses or instances of these six building blocks. Lois lets users express a wide range of queries—ideally, any query that can be expressed directly in OCML—while at the same time shielding them from formalism-related aspects.

For instance, the query shown in Figure 4 asks for KMI researchers involved in software visualization. We constructed this query by first selecting class `kmi-member`, specializing it to `kmi-researcher`, selecting the relation `develops-technology`, and then circumscribing the range of this relation to `kmi-software-visualization-technology`. More specifically, we created the first row of the query by selecting the Member of KMI button, then `type` in the Index Aspects window, then `kmi-researcher` in the Aspect Type window, then the Add Row button, and finally `develops-technology` from the menu at the end of the row. We created the second row in a similar fashion.

To ensure that novice users can construct queries out of these six building blocks, we explicitly included all the obvious binary relations between the six classes. Because stories are interesting only with respect to their associated events, they are not linked to the other main classes. Hence, we had to specify only 24 relations.

Figure 5 shows a simple example of the advantages gained by enriching stories with ontology-based knowledge structures. The query shown asks for stories about genetic algorithms. The answer is the story about Trevor's award, even though the body of the story does not explicitly mention genetic algorithms.

The Lois interface strikes a balance between usability, expressiveness, and ease of construction. To create an interface to a knowledge model, the knowledge engineer need only specify the key classes. The Index Aspect window includes the slots of the selected class. When the user constrains the slot to have values of a certain type, the Aspect Type window shows the type (which is an OCML class) and its descendants. Thus, the usability of the Lois interface depends on the underlying ontology. The knowledge engineer must choose class and slot names that the Lois user can understand; then the user only needs to learn how to construct queries through the accumulation of rows.

Story chasing with Newshound. An important goal in designing KMI Planet was that the system should emulate a newsroom team. We are developing an intelligent agent, Newshound, to emulate a news editor's behavior—identifying potentially popular stories and assigning them to appropriate journalists by e-mail.

To identify potentially interesting stories, Newshound uses two main types of data: statistics on access to individual stories in KMI Planet and records of the queries posed through Lois. Each story in Planet keeps a record of its popularity by counting the number of times users request the full text from the KMI Planet server. Once Newshound identifies a story as "popular," it tries to identify related stories that have the potential to be popular by analyzing the knowledge base, trying to find items of interest that Planet stories have not yet covered. The term *related* is key here. To define relatedness, Newshound uses various heuristics (defined as

Related work

Related research involves a number of areas, including information retrieval, knowledge management, ontological engineering, agent technology, knowledge engineering, and model-based knowledge acquisition. Here, we focus on related approaches to ontology-driven knowledge management.

The SHOE (Simple HTML Ontology Extensions) project[1] proposed an extension to HTML to allow the specification of ontological information. The project team also developed an editor to support the page annotation process. This work is mainly at the infrastructure level: that is, they suggest a mechanism to enable the representation of information and provide tools to edit and retrieve it. In PlanetOnto, we take a holistic approach to the publish-annotate-retrieve process, and we look at the wider issues concerning usability and sustainability. We could easily integrate the technical solutions provided by SHOE within the PlanetOnto framework. For instance, we could represent OCML (Operational Conceptual Modeling Language) structures in terms of relevant SHOE tags.

The (KA)2 initiative[2] also shares a number of commonalities with our work. Both projects are trying to enable a community to build a knowledge base collectively, by populating a shared ontology. (KA)2's knowledge base is meant to document the knowledge acquisition community's activities. Like the approach used in SHOE, this team is constructing the knowledge base by annotating Web pages with tags that only a specialized search engine cum interpreter can read. In this article, we have emphasized that collective construction of a knowledge base depends crucially on a number of features, including a carefully defined ontology; an underlying modeling language providing user-oriented facilities, such as context-dependent renaming; a user-friendly annotation environment; and the right motivational stimuli for participants. In their article on the (KA)2 initiative, Richard Benjamins and his colleagues[2] mainly focus on the last issue. However, to manage the risks associated with such enterprises, it is important to carefully analyze all the issues associated with collaborative ontology development and instantiation—especially the design of the underlying ontology. Thus, in contrast to (KA)2, our approach keeps ontology design centralized. We do not expect members of the community to develop ontologies, only to populate them.

Related work in knowledge management here in the Knowledge Media Institute has (naturally) many points of contact with PlanetOnto. The Enrich project[3] aims to support organizational learning through the enrichment of Web-based documents, which is carried out through both ontologies and hypertext-based argumentation. Although the Enrich scenario differs greatly from the one PlanetOnto addresses, the underlying assumption is the same for both projects: ontology-driven enrichment of documents plays an important role in knowledge management.

In the ScholOnto project,[4] we are applying the PlanetOnto technologies to support scholarly debate within digital libraries. This approach focuses on representing discourse rather than content, thus ensuring that the underlying ontology is stable and scalable. Once the ontology has been finalized, we will begin constructing a number of services, including searches for significant conceptual structures, sophisticated alerting, and intelligently filtering visualizations of the literature.

References

1. J. Heflin, J. Hendler, and S. Luke, "Reading between the Lines: Using SHOE to Discover Implicit Knowledge from the Web," *AAAI-98 Workshop on AI and Information Integration*, 1998; available in postscript form from www.cs.umd.edu/projects/plus/SHOE/pubs/shoe-aaai98.ps (current May 2000).
2. R. Benjamins, D. Fensel, and A. Gomez Perez, "Knowledge Management through Ontologies," *Proc. Second Int'l Conf. Practical Aspects of Knowledge Management*, Swiss Life Information Systems Research Group, Zurich, Switzerland, 1998, pp. 29–30; SunSITE.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-13 (current May 2000).
3. T. Sumner et al., "Enriching Representations of Work to Support Organizational Learning," *Proc. CEUR Workshop Building, Maintaining, and Using Organizational Memories (OM-98), 13th European Conf. Artificial Intelligence (ECAI-98)*, CEUR-WS/Vol-14, Paper No. 8, 1998; SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-14 (current May 2000).
4. S. Buckingham Shum, E. Motta, and J. Domingue, "ScholOnto: An Ontology-Based Digital Library Server for Research Documents and Discourse," to be published *Int'l J. Digital Libraries*, Vol. 3, No. 3, Aug./Sept. 2000.

Prolog-like rules), including direct subclasses of the same class, technologies that build on the same underlying technology, and projects that tackle the same areas. These heuristics are of course soft and modular, so we can add new ones without affecting existing ones.

However, the most interesting feature of Newshound deals less with specific heuristics than the unique scenario in which it examines a knowledge base for gaps. Typically, completeness in a knowledge base is defined with respect to logical or task-related properties. In our scenario, we define incompleteness in pragmatics terms: publications need popular stories.

Personalized alerts with Newsboy. We designed Lois (among other things) to help users track down Planet stories with specific characteristics. However, many users prefer to work with push technology—being notified automatically about potentially interesting stories rather than having to query Lois about them. We therefore designed an agent, Newsboy, to provide a mode of use that was complementary to the one supported by Lois. Newsboy lets users create a personalized front page to which interesting stories are pushed.

Once a knowledge editor or journalist annotates a new story with Knote, Newsboy matches the story against registered readers' specified interests and then e-mails them that it has added a new story to their personal Planet pages. To make an explicit declaration, a reader simply specifies a number of queries using the Lois interface. Newsboy then updates the reader when a new story matches at least one of the logged queries. Alternatively, readers can have Newsboy log all their queries to Lois and create a user profile from the log. Each resulting user profile is simply the logical disjunction of a log's queries.

It is interesting to compare Newsboy to other approaches that try to infer user profiles from analyzing patterns of access to documents. These approaches try to induce user interests using empirical methods. The main advantage of these systems is that they are largely automatic; the disadvantage is that the quality of the derived relations is limited. Because our approach relies on a handcrafted ontology, the process of matching items to readers is semantic and based on richly described relations.

THE PLANETONTO ARCHITECTURE provides an example of an ontology-centered approach to knowledge management (for more information, see the sidebar "Related work"). With the exception of Newsboy and Newshound, which are still at an initial implementation stage, all components of the architecture are in place and are undergoing preliminary user testing. Once we have com-

pleted them, PlanetOnto will become fully operational. We also plan to apply the PlanetOnto technology to support semantic access to medical guidelines.

Obviously, several research issues remain open. The move from Planet to PlanetOnto means that what we had originally conceived as an internal newsletter is now rapidly becoming an integrated knowledge management system. The next step will be to integrate other departmental resources into the overall knowledge management system. In particular, we plan to investigate information extraction methods to automatically integrate information from departmental Web pages into the Planet knowledge base. Actually, the more PlanetOnto grows, the clearer the need for clever information extraction techniques. The main bottleneck so far was the process of collaboratively constructing the knowledge base. Despite our efforts to maximize the ontology's usability, we realize that this approach works only in restricted scenarios, where users are reasonably motivated and skilled and the annotation process is relatively lightweight. Thus, the major challenge for this and other similar enterprises remains to develop tools that help to automate the semantic annotation of documents. ■

Acknowledgments

Several European-funded projects provided partial support for this research, including Enrich (p29015) under the Esprit program on IT for learning and training in industry, and HC ReMa (p4124) and PatMan (p4017) under the Telematics Applications program. Thanks to Simon Buckingham-Shum and Paul Mulholland for insightful discussions on PlanetOnto and especially to Simon for his important feedback on its usability. We also thank Anthony Seminara and Harriett Ansell for their extensive work on the figures in this article.

References

1. J. Domingue and P. Scott, "KMI Planet: Putting the Knowledge Back into Media," *The Knowledge Web: Learning and Collaborating on the Net*, M. Eisenstadt and T. Vincent, eds., Kogan Press, London, 1998, pp. 173–184.

2. D.E. O'Leary, "Knowledge Management Systems: Converting and Connecting," *IEEE Intelligent Systems*, Vol. 13, No. 3, May/June 1998, pp. 30–33.

3. T. Sumner et al., "Enriching Representations of Work to Support Organizational Learning," *Proc. CEUR Workshop on Building, Maintaining, and Using Organizational Memories (OM-98)*, 13th European Conf. Artificial Intelligence (ECAI-98), CEUR-WS/Vol-14, Paper No. 8, 1998; SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-14 (current May 2000).

4. J. Domingue, "Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web," *Proc. 11th Knowledge Acquisition for Knowledge-Based Systems Workshop*, Vol. 2, KM-4, 1998; SRDG Publications, Dept. of Computer Science, Univ. of Calgary, Calgary, Alta, Canada, 1998; ksi.cpsc.ucalgary.ca/KAW/KAW98/domingue or kmi.open.ac.uk/people/domingue/banff98paper/domingue.html (current May 2000).

5. E. Motta, *Reusable Components for Knowledge Modelling: Principles and Case Studies in Parametric Design*, IOS Press, Amsterdam, 1999.

6. D.B. Lenat and R.V. Guha, *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*, Addison-Wesley, Reading, Mass., 1990.

7. T.R. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing," *Int'l J. Human-Computer Studies*, Vol. 43, Nos. 5–6, 1995, pp. 907–928.

John Domingue is a senior research fellow in artificial intelligence and an assistant director at the Knowledge Media Institute of the Open University. He is also a principle investigator on the EU-funded Esprit IV Enrich project (p29015), where he is exploring how the ontological enrichment of Web documents can support organizational learning in various industrial settings. He chaired the Practical Applications of Knowledge Management Conference in 1999 and coedited *Software Visualization: Programming as a Multimedia Experience* (MIT Press, 1998). He received his first degree in computer science from the University of Warwick, UK, and a PhD in artificial intelligence from the Open University. Contact him at the Knowledge Media Inst., The Open Univ., Walton Hall, Milton Keynes, UK MK7 6AA; j.b.domingue@open.ac.uk.

Enrico Motta is a senior research fellow and assistant director of the Knowledge Media Institute of the Open University. His research and publications deal with knowledge acquisition, knowledge sharing and reuse, knowledge management, formal languages, and multiagent systems. His current research focuses on reusable components for knowledge modeling—ontologies and problem-solving methods—and the application of knowledge modeling techniques to knowledge management. He received his first degree in computer science from the University of Pisa, Italy, and a PhD in artificial intelligence from the Open University. He is the author of *Reusable Components for Knowledge Modelling* (IOS Press, 1999). Contact him at the Knowledge Media Institute, The Open Univ., Walton Hall, Milton Keynes, UK MK7 6AA; e.motta@open.ac.uk.