Int. J. Human-Computer Studies (2000) **52,** 1071–1109
doi:10.1006/ijhc.2000.0384
Available online at http://www.idealibrary.com on IDEAL®

Ontology-driven document enrichment: principles, tools and applications

ENRICO MOTTA, SIMON BUCKINGHAM SHUM AND JOHN DOMINGUE
Knowledge Media Institute, The Open University, Walton Hall, Milton Keynes, MK7 6AA, U.K.

(Received 10 November 1999 and accepted 20 January 2000)

In this paper, we present an approach to *document enrichment*, which consists of developing and integrating formal knowledge models with archives of documents, to provide intelligent knowledge retrieval and (possibly) additional knowledge-intensive services, beyond what is currently available using "standard" information retrieval and search facilities. Our approach is *ontology-driven*, in the sense that the construction of the knowledge model is carried out in a top-down fashion, by populating a given *ontology*, rather than in a bottom-up fashion, by annotating a particular document. In this paper, we give an overview of the approach and we examine the various types of issues (e.g. modelling, organizational and user interface issues) which need to be tackled to effectively deploy our approach in the workplace. In addition, we also discuss a number of technologies we have developed to support ontology-driven document enrichment and we illustrate our ideas in the domains of electronic news publishing, scholarly discourse and medical guidelines.

KEYWORDS: semantic web; ontologies; knowledge modelling; digital documents; document retrieval; intelligent news servers; scholarly discourse; medical informatics

1. Introduction

An important activity in knowledge management is "to convert text to knowledge" (O'Leary, 1998). This activity is central to knowledge management for two reasons: (1) work practices and information flow in organizations tend to be *document-centred* and (2) documents themselves do not normally exhibit the amount of structure required to support semantically aware search engines or other forms of intelligent services. For these reasons, there has been much interest in technology to support the specification of structured information in textual documents, especially web pages. The web standardization community has focused on the underlying representational infrastructure: XML (1999) has been proposed as the basic annotation formalism to support the specification of structured information in web pages, while RDF builds on the XML syntax to provide a standard declarative representation, which allows users to express semantic relationships between items on the web. Approaches such as Ontobroker (Fensel, Decker, Erdmann & Studer, 1998) and Shoe (Heflin, Hendler & Luke, 1998) provide formalisms and associated interpreters which make it possible to embed knowledge representation structures in web pages and use them to perform inferences.

1071-5819/00/061071 + 39 $35.00/0

In this paper, we look at the wider issues concerning "the conversion of text to knowledge" and discuss a comprehensive approach to *document enrichment* (Sumner *et al*., 1998), which we are trying out in a number of projects here at the Knowledge Media Institute. The approach is characterized in terms of a set of activities, with associated informal guidelines. In this paper, we also describe a number of technologies, which we have developed to support our approach to document-centred knowledge management. These technologies include a *knowledge modelling* language,† form-based interfaces for adding and retrieving knowledge from a model and a web-based browser/editor, which supports the collaborative development of knowledge models over the World Wide Web. Finally, we discuss the application of our approach to three domains: *electronic news publishing* (Domingue & Motta, 1999), *scholarly discourse* (Buckingham Shum, Motta & Domingue, 1999) and *medical guidelines* (PatMan, 1998).

The paper is organized as follows: in the next section we give an overview of our approach, in terms of the underlying methodological assumptions and the associated process model. In Section 3 we describe the technology we have developed to support the approach. In Sections 4, 5 and 6 we discuss the application of the approach to the three aforementioned domains. Finally, in Sections 7 and 8 we discuss related work and reiterate the main contributions of this paper.

2. Ontology-driven document enrichment

Our approach is *ontology-driven*, in the sense that the construction of the knowledge model is carried out in a top-down fashion, by populating a given *ontology* (Gruber, 1993), rather than in a bottom-up fashion, by annotating a particular document. Figure 1 underlines this point graphically, by emphasizing that the construction of a knowledge model is driven by a pre-existing ontology, a set of documents and other sources of knowledge, such as appropriate (human) experts. Following Gruber, we use the term "ontology" to indicate "a specification of a reusable conceptualization". More simply, an ontology can be seen as providing a vocabulary for describing a range of models. For instance, an ontology for medical guidelines provides a generic set of concepts and relations (e.g. medical condition, diagnostic guideline, guideline user type), which can then be instantiated for particular guidelines to build guideline-specific models, in domains such as stroke management or prevention of pressure ulcer.

An ontology-driven approach to model construction affords several advantages. Instantiating an ontology is usually simpler and speedier than developing a model from scratch. In addition, because an ontology makes explicit the conceptualization underlying a particular model, it becomes easier to maintain, reuse and interoperate the model with other components. Finally, reasoning modules can be associated with an ontology and these are then applicable to all models built by instantiating the ontology in question. For instance, in the case of medical guidelines, one can envisage building

†Here we use the term "knowledge modelling" as a short form for "knowledge-level modelling", an expression introduced by Allen Newell (1982) to describe models of knowledge-intensive behaviour which abstract from the way this behaviour is implemented and focus instead on the knowledge employed by an agent and the goals the agent is trying to achieve.

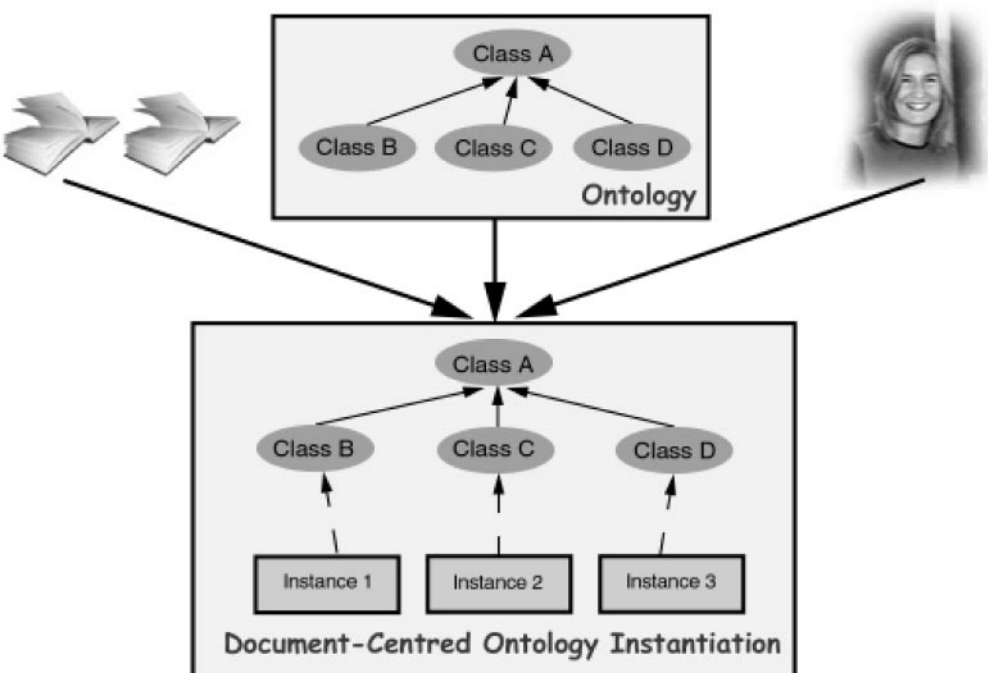

FIGURE 1. Ontology-driven document enrichment.

ontology-specific guideline verification tools, which can then be used to verify individual guidelines developed by instantiating the same generic guideline ontology.

Because our model construction process is ontology-driven, we prefer to use the term "enrichment" (Sumner *et al*., 1998), rather than "conversion" or "annotation", to refer to the process of associating a formal model to a document (or set of documents). In general, a representation, whether formal, graphical or textual, can be enriched in several different ways—e.g. (1) by providing information about the context in which it was created, (2) by linking it to related artifacts of the same nature or (3) by linking it to related artifacts of a different nature. Although in our document-centred knowledge management work, we provide multiple forms of document enrichment, such as associating *discussion spaces* to documents (Sumner & Buckingham Shum, 1998), in this paper we will primarily concentrate on the association of formal knowledge models to documents.‡

Thus, an important facet of an ontology-centred approach to document enrichment is that the formalized knowledge is not meant to be a translation of what is informally specified in the associated document. Hence, the knowledge model typically plays a different role from the associated text. For instance, in the medical guideline scenario the knowledge model helps to verify that all the kinds of knowledge expected to be found

‡Having said so, the medical guideline scenario described in Section 6 does integrate a formal model with a set of discussion spaces, to provide multiple forms of document enrichment.

in a document describing a medical guideline are indeed there. In the scholarly discourse scenario, the knowledge model is meant to capture the meta-knowledge required to structure academic debates (e.g. theory X contradicts theory Y), which is often expressed only implicitly in publications (i.e. acquiring it typically requires some interpretation effort) and is not modelled at all in traditional libraries. In a nutshell, the emphasis in our approach is in identifying the *added value* (in terms of enabling semantic retrieval and document indexing capabilities, or other reasoning services), which can be provided by a formalized knowledge model. Our methodology comprises the following six steps.

1. Identify use scenario.
2. Characterize viewpoint for ontology.
3. Develop the ontology.
4. Perform ontology-driven model construction.
5. Customize query interface for semantic knowledge retrieval.
6. Develop additional reasoning services on top of knowledge model.

These steps are briefly described in the next sub-sections.

2.1. IDENTIFY USE SCENARIO

At this stage, the services to be delivered by the knowledge management system are defined. In particular, issues of feasibility and cost are investigated. Addressing the latter involves answering questions such as: "What is the added value provided by the knowledge model, considering the non-trivial costs associated with the development and instantiation of an ontology?", "Is there the need for a 'full-blown' knowledge model and for going beyond the facilities provided by off-the-shelf search engines?", "What additional reasoning services will be provided, beyond deductive knowledge retrieval?". Addressing feasibility issues requires assessing (among other things) whether or not it is feasible to expect the target user community to perform document enrichment or whether specialized human editors will be needed. This latter solution introduces a significant bottleneck in the process and moreover assumes that to introduce a central editor in the model development is actually feasible. This is definitely not the case in some of our application domains. For instance, in the scholarly discourse scenario our aim is to allow members of academic community to express their own views about academic contributions, whether their own or whether proposed by other scholars. In this scenario, it is not feasible to envisage a central editor mediating between an author and a shared knowledge base. In addition, we also believe that aiming to enable non-experts to take part in the development of knowledge models is a good research objective *per se*, not only on the grounds of cost-effectiveness and sustainability, but also because of the perceived benefits deriving from being engaged in the modelling activity. Specifically, there is already some evidence that involving students in constructing knowledge models improves their understanding of a discipline (Wideman & Owston, 1993). Although no hard data are available, there is no obvious reason why these benefits should not occur in scenarios outside the educational domain.

Of course, successfully enabling non-experts to engage in knowledge modelling is far from a trivial proposition. There are subtle trade-offs involved here, which we address in Section 2.4, when discussing the ontology instantiation task.

2.2. CHARACTERIZE VIEWPOINT FOR ONTOLOGY

The previous step was concerned with assessing the feasibility of the approach and defining the functionalities of the envisaged system. This step focuses on the ontology: having decided on an ontology-based approach, it is important to characterize the particular viewpoint that the envisaged ontology will impose on the documents. For instance, in the electronic publishing domain we focus on the events characterizing academic life, while in the scholarly discourse domain the focus is on academic debate. Clearly, the distinction between this step and the steps immediately before and after can in some cases be fuzzy. For instance, the specification of the viewpoint in the scholarly discourse domain is tightly integrated with the characterization of the use scenario. However, we believe that it is useful to explicitly distinguish a viewpoint specification task for two reasons. The first one is that it is feasible to envisage scenarios in which different viewpoints can be taken as the starting point for the ontology development process. Examples abound in the literature, especially with respect to highly generic ontologies—compare, for example, the *Cyc* ontology (Lenat & Guha, 1990) with the work by Sowa (1995). Another reason is methodological: it is useful to separate the issues concerning the functionalities, scope and purpose of the envisaged system from the specification of the functionalities, scope and purpose of the ontology. For instance, in the electronic publishing scenario the knowledge management system is concerned with providing semantic retrieval capabilities and services supporting story identification and personalized news feeds—see Section 4 for more details. Within this scenario, the chosen viewpoint focuses on modelling academic events and the key "actors" in these events: technologies, people, organizations and projects.

2.3. DEVELOP THE ONTOLOGY

Having defined a particular viewpoint over a domain (in this case, a set of documents), ontology development is largely a technical enterprise, i.e. the issues here concern modelling and formalization, rather than scope and purpose. Several approaches to ontology development have been proposed in the literature, which introduce distinctions along different dimensions—contrast, for instance, the *bottom-up* development style of van der Vet and Mars (1998) with Sowa's (1995) *top-down* approach. Uschold and Gruninger (1996) argue that a *middle-out*, *purpose-driven* approach is most effective, in which the basic concepts in a domain are identified first (e.g. dog), and later generalized (mammal) and/or specialized (cocker spaniel).

In our scenarios, we have followed a task-independent, middle-out approach, and we use the selected viewpoint to help us to identify the *key concepts* in the class of models we want to construct. For instance, in the electronic newsletter scenario, a starting point was the notion of *news item*, which in turn was characterized as relating a number of *events*. Thus, our main modelling effort centred on identifying and modelling the various types of events relevant to our scenario (academic life). In the scholarly discourse domain, the viewpoint is "academic debate" and the ontology then concentrates on characterizing the most common relations which exist between academic theories, methodologies, models, etc.

Another important issue concerns who constructs the ontology? As pointed out earlier, we want to support scenarios in which knowledge models are constructed

collaboratively. But what about the ontology itself? Is this constructed collaboratively? Our answer is negative. In all the projects we have carried out so far, we have centralized the ontology development. The main reason for this choice is that a careful design of the ontology is crucial to ensure the success of any particular document-enrichment initiative. The ontology specifies the selected viewpoint, circumscribes the range of phenomena we want to deal with and defines the terminology used to acquire domain knowledge. In our experience, small errors/inconsistencies in any of these aspects can make the difference between success or failure. Moreover, ontology design requires specialist skills which are normally not possessed by the members of our target user communities.

Our ontology design approach is informed by two main modelling guidelines. The rationale for both of these is user-centred: because we want non-expert users to be able to engage in the model construction process, the ontologies we produce need to be easy to understand and to instantiate. These user-centred considerations have priority over any other design criterion.

- *Minimal ontological definitions.* That is, to try and provide only the minimal set of attributes needed to define a particular class. This approach has the advantage that, when populating the ontology, users do not have to face lots of irrelevant attributes. For instance, in the electronic news publishing domain we initially reused the definition of class `event` given in the HPKB upper layer (HPKB, 1997), but we then removed about 90% of its slots. The reason for this was that the HPKB definition aims to cover all potential attributes which *can* be relevant to a *generic instance* of class `event`. However, typically only a few slots will *actually* be relevant for any *specific sub-class* of the class. For instance, slot `damages` is only relevant to events which can cause damage. Hence, we would introduce this slot when characterizing a sub-class of class `event`, such as `damaging-event`, rather than associating it with class `event` itself.
- *User-centred definitions.* This guideline requires that the terminology used by the ontology needs to be easy to understand for a user who is not necessarily a knowledge engineer. There are two aspects here: heavily technical modelling concepts—e.g. sophisticated modelling solutions for representing time—ought to be avoided. Moreover, the terminology should be as context-specific as possible. For instance, while we can talk about "agents performing events", when describing events in general, we should use the class-specific terminology, "awarding body assigns awards", when talking about an award-giving type of event. This latter guideline implies that the underlying modelling language should support slot renaming along *isa hierarchies*, i.e. inherited slots should get sub-class-specific names. The importance of domain-specific, user-oriented terminology has been recognized in knowledge acquisition for a long time (Musen, 1989) and arguably provides an important difference between the criteria associated with modelling for knowledge acquisition and those associated with modelling for system development.

2.4. PERFORM ONTOLOGY-DRIVEN MODEL CONSTRUCTION

We are acutely aware that many schemes for registering shared resources and providing structured descriptions founder on the crucial "capture bottleneck"—the envisaged beneficiaries of the system simply do not have the motivation or time to invest in sharing

resources to reach a critical mass of useful material. Sobering lessons on this theme have been drawn for group and organizational memory systems (Selvin, 1999), and indeed, for any system that requires users to formalize information (Shipman & Marshall, 1999). Why should we succeed where others have failed? Our working hypothesis is that our domains have unique characteristics lacking in domains in which collaborative development has failed.

- *Cooperation rather than competition.* We are selecting domains where co-operation, meant here as "willingness to share knowledge", is either a basic work premise or is enforced by external constraints. For instance, academic analysis and publishing require scholars to read, refer to and praise/criticise each other's work. In sum, the dynamics of academic publishing requires cooperation. A similar situation occurs in the medical guidelines scenario. Institutions and individual scientists may compete with each other, but the outcome of consensus conferences (by definition) are shared knowledge resources. In other scenarios, for instance when constructing an organizational memory, other forces (e.g. directives from higher management) may force cooperation, even when competition would be the norm.
- *Benefits outweigh costs.* Motivation is a crucial aspect. Motivation essentially boils down to a cost-benefit analysis. For instance, in the scholarly discourse scenario, we assume that the basic motivation of an academic is to disseminate his/her work. Hence, having completed a new document, the author will want to maximize its "digital presence" on the net by carefully encoding its contributions and connections to the existing literature.
- *Compatibility with organizational work practices.* This requires the seamless integration of our document enrichment model with existing work practices. For instance, in the case of electronic publishing, the ontology is used to enrich news items, which are submitted either through email or through a web-based form. Hence, at least for those users who submit through the latter mechanism, instantiating the ontology becomes an additional form-filling activity, carried out using the same medium (i.e. the web browser) and at the same time. Analogously, in the case of scholarly discourse, at least in some academic communities, authors are used to submitting papers to digital repositories and providing metadata. Filling an ontology-derived form should then be perceived as a small, additional step.§

2.5. CUSTOMIZE QUERY INTERFACE FOR SEMANTIC KNOWLEDGE RETRIEVAL

At this stage the appropriate query interface is designed, in accordance to the use scenario and the expected functionalities. To support this step we have developed a flexible form-based interface, called *Lois*, which can be customized for each specific application domain. Lois is described in Section 3.3.

§ Actually, this is not the case in the medical guideline scenario. In this case, accommodating the guideline management technology described in this paper will normally require a change in organizational work practices, as computerized guideline support is still fairly rare in hospital settings.

2.6. DEVELOP ADDITIONAL REASONING SERVICES ON TOP OF THE KNOWLEDGE MODEL

This final step is where the real benefit of the approach lies. Once a knowledge model has been produced, then it becomes possible to provide additional intelligent functionalities and ensure that the benefits outweigh the costs. These reasoning services tend to be application-specific. For instance, in the scholarly publishing scenario, we are planning to develop specialized agents, whose goal is to identify emerging scholarly perspectives, using heuristic knowledge and machine learning techniques. For instance, an agent could discover a "European perspective" on a particular issue, if a structural pattern in the knowledge model—e.g. use of formal methods—also matched the geographic location of the relevant researchers. In the electronic publishing domain, we have designed two agents, which reason about the contents of the knowledge model to identify new, potentially "hot" stories and to provide personalized news feeds—see Section 4.2.5.

3. Technologies for ontology-driven document enrichment

In this section, we describe the main modelling support technologies we have developed to enable ontology-driven document enrichment. These are as follows.

- *OCML*. An operational knowledge modelling language, which provides the underlying representation for our ontologies and knowledge models.
- *WebOnto*. A tool providing web-based visualization, browsing and editing support for developing and maintaining ontologies and knowledge models specified in OCML.
- *Lois*. A flexible form-based interface for knowledge retrieval.
- *Knote*. A form-based interface for populating an ontology.

In addition to these modelling support technologies, we also have a number of document-centred technologies, such as the *D3E shell* (Sumner & Buckingham Shum, 1998), which supports a tight integration of discourse and debate facilities within documents. Because the focus of this paper is on ontology-driven knowledge management, we circumscribe the technology review in this section to modelling support technologies. Other relevant technologies, such as D3E, will be introduced "opportunistically" in the application-specific sections.

3.1. OCML, AN OPERATIONAL KNOWLEDGE MODELLING LANGUAGE

The development of the OCML language (Motta, 1999) was primarily motivated by dissatisfaction with existing ontology specification languages. Languages such as *Ontolingua* (Gruber, 1993) benefit from a well-defined semantics and extensive libraries, but provide very limited support for model operationalization. No interpreter exists for Ontolingua models and the translation approach proposed for operationalizing Ontolingua models into executable languages suffers from two serious limitations. First of all it does not effectively support rapid prototyping and incremental model development—essentially it proposes a "batch mode" style of operationalization. Secondly, it assumes the availability of smart translators—e.g. for translating Ontolingua models to knowledge representation languages, such as *Loom* (MacGregor, 1991). Unfortunately, these are currently beyond the state of the art of language translation and are "unlikely to be forthcoming in the foreseeable future" (Uschold, Clark, Healy, Williamson & Woods,

```
(def-relation <> (?x ?y)
   "True if ?x and ?y do not denote the same object"
   :iff-def (not (= ?x ?y)))
```

FIGURE 2. Sample OCML definition.

1996). As a result Ontolingua models are typically operationalized through extensive manual effort, an approach which does not lend itself to rapid prototyping. Similar arguments apply of course to non-executable specification languages in general and it does not come as a surprise that the importance of executable specifications has been argued also in mainstream software engineering (Fuchs, 1992).

OCML was therefore conceived as a kind of "operational Ontolingua". Our goal was to develop a language that would be as compatible as possible with Ontolingua, thus benefiting from the advantages in terms of well-formed semantics, availability of libraries and large diffusion enjoyed by Ontolingua, while at the same time providing support for rapid prototyping and incremental development. Hence, consistently with the notion of knowledge-level modelling OCML focuses on logical, rather than implementation-level primitives. It provides constructs for specifying relations, functions, classes, instances and axioms, and, in particular, supports the variety of relation specification keywords used by Ontolingua, e.g. `:iff-def`, `:def`, `:sufficient` and `:axiom-def`.

In order to allow rapid prototyping, OCML provides a function interpreter, a control interpreter and a proof system, which seamlessly integrates inheritance and function evaluation with a backward chaining inference engine. Hence, an important aspect of OCML is that it tries to fulfil a dual role: on the one hand, it aims to provide a rich specification environment, compatible with de facto standards, such as Ontolingua. On the other, it also aims to provide effective operationalization support, by means of interpreters and an inference engine. This dual role is normally supported by OCML in a transparent way. For instance, let us consider the definition given in Figure 2. The `:iff-def` option in the relation specification plays a dual role: it helps to characterize the semantics of the relation and at the same time it is used by the OCML proof system to check whether the relation $\langle\,\rangle$ holds for a pair of arguments.

In other cases, the user may require more control over the operational use of a definition—for instance, i n cases where the given specification will be too inefficient. To this purpose OCML provides extra keywords, which allow a user to associate *procedural attachments* (Weyhrauch, 1980) to a definition, either using OCML itself, or by means of Lisp code.

Figure 3 provides an example of an OCML definition which separates specification and operationalization. The `:iff-def` keyword provides the formal definition of relation `superclass-of`, while the `:prove-by` keyword defines a way to infer whether or not the relation holds for a given pair of arguments. Because a `:prove-by` option has been specified, the OCML interpreter will not use the `:iff-def` option when trying to reason about this particular relationship. Such an integrated use of specification and operationalization in the same language is similar to the way the *epistemological* and *heuristic* levels are amalgamated in Cyc (Lenat & Guha, 1990). Indeed, we found this feature very useful in our knowledge management work, because it allows us to quickly move from ontology specification to a knowledge base and then

```
(def-relation SUPERCLASS-OF (?super ?c)
   "The inverse of subclass-of"
   :iff-def (and (class ?super)
                 (class ?c)
                 (member ?super (all-superclasses ?c)))
   :prove-by (or (and (variable-bound ?c)
                      (member ?super (all-superclasses ?c)))
                 (and (variable-bound ?super)
                      (member ?c (all-subclasses ?super)))
                 (and
                      (not (variable-bound ?super))
                      (not (variable-bound ?c))
                      (class ?super)(class ?c)
                      (superclass-of ?super ?c))))
```

FIGURE 3. OCML definition which separates specification and operationalization.

incrementally improve the efficiency of the knowledge base opportunistically, in response to eventual performance bottlenecks.

OCML modelling is supported by a library of reusable definitions, which is structured according to the basic categories of our application modelling framework: *task*, *method*, *domain* and *application* (Motta, 1999). The library also relies on a number of *base ontologies*, which provide definitions for basic modelling concepts, such as numbers, sets, relations, tasks, methods, roles, etc. The OCML base ontology also includes the constructs specified in the Ontolingua frame ontology, thus ensuring compatibility between frame-based specifications in the two languages. Constructs for which OCML does not provide operational support—e.g. axioms—are simply added to the model, but they are not used in the reasoning process.

Our library of OCML models can be accessed through the WebOnto browser at URL http://webonto.open.ac.uk—see Figure 4. WebOnto is described in the next section.

3.2. WEBONTO: BROWSING AND EDITING KNOWLEDGE MODELS ON THE WEB

WebOnto (Domingue, 1998) enables knowledge engineers to collaboratively browse and edit knowledge models over the web, using a standard web browser. The architecture is composed of a central server and a Java applet. The central server is built on top of a customized web server, *LispWeb* (Riva & Ramoni, 1996). In addition to implementing the standard HTTP protocol, the LispWeb server offers a library of high-level Lisp functions to dynamically generate HTML pages, a facility for dynamically creating image maps and a server-to-server communication method. The WebOnto server interacts with the ontology libraries represented in OCML and is responsible for the following tasks.

- Converting OCML structures into ASCII strings which can be sent via HTTP.
- Sending requested OCML source code to WebOnto clients.
- Updating server source code from WebOnto clients.
- Locking an ontology when it is being edited (sub-ontologies remain unlocked). Locking an ontology ensures that conflicting edit operations, such as two users changing the same OCML structure, cannot occur.
- Managing administrative data about the knowledge models in the library and WebOnto users.

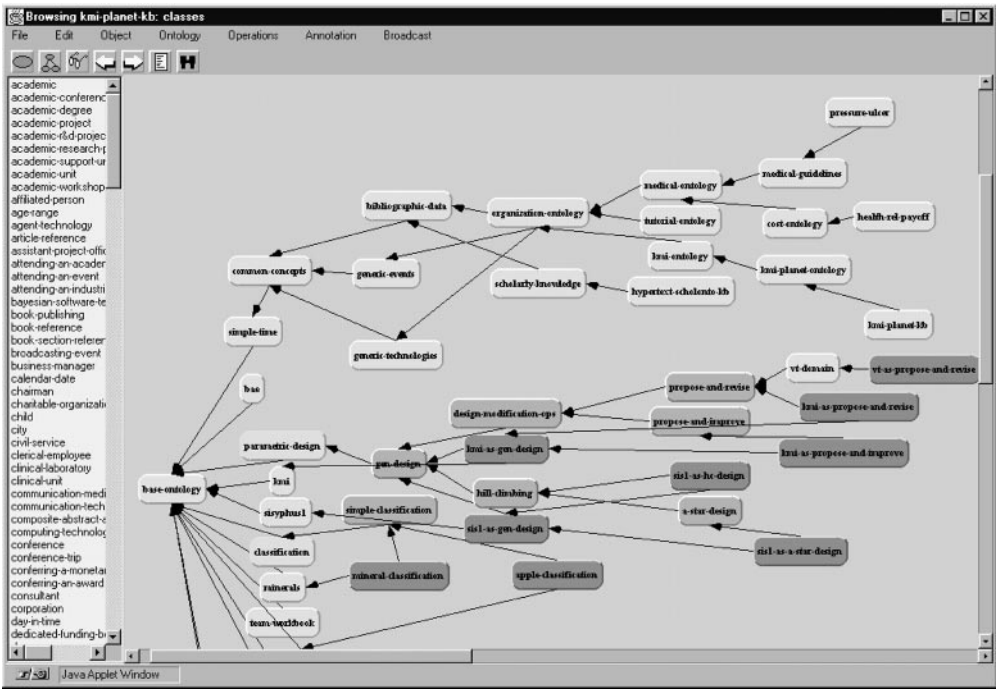

FIGURE 4. WebOnto visualization of part of the KMi library of knowledge models.

The Java applet provides multiple visualizations of OCML knowledge models, a direct manipulation and forms interface for creating new knowledge structures, and a groupware facility which supports both synchronous and asynchronous model building by teams of knowledge engineers. The direct manipulation interface integrates both graphical and textual views of a knowledge model and allows the user to work in either modality, while ensuring that changes are propagated immediately. More specifically, WebOnto gives automatic semantic feedback, in visual form, whenever the user performs an incremental edit of the knowledge model. Tanimoto (1990) describes this as *level 3 liveness*.

Another important aspect of the WebOnto environment is the tight integration of both *coarse-* and *fine-grained* visualizations, a feature which is essential to support the development of large models (Eisenstadt, Domingue, Rajan & Motta, 1990).

Figure 5 shows a screen snapshot of the main WebOnto Java Applet window. The class hierarchy shown in the figure was created by selecting the class `academic` in the list shown in the left panel and then clicking on the "draw hierarchy" toolbar button ( ). Figure 6 shows a detailed view of one of the classes included in the hierarchy in Figure 5, `kmi-professor`. This view was obtained by selecting the `kmi-professor` node in Figure 5 and then clicking on the "inspect" toolbar button ( ). Any text displayed in the fine-grained view window is automatically parsed and colour-coded, in accordance with the relevant OCML type (e.g. relation, function, instance, etc.). It can then be inspected simply by selecting and clicking on it. As an example, we have highlighted slot `has-duration` in Figure 6.

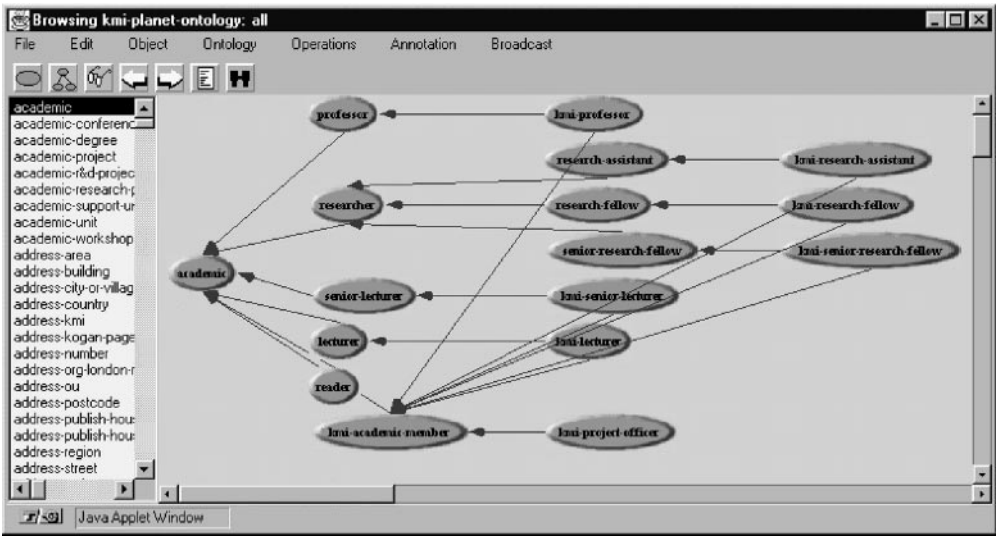

FIGURE 5. The main WebOnto window showing a class hierarchy.

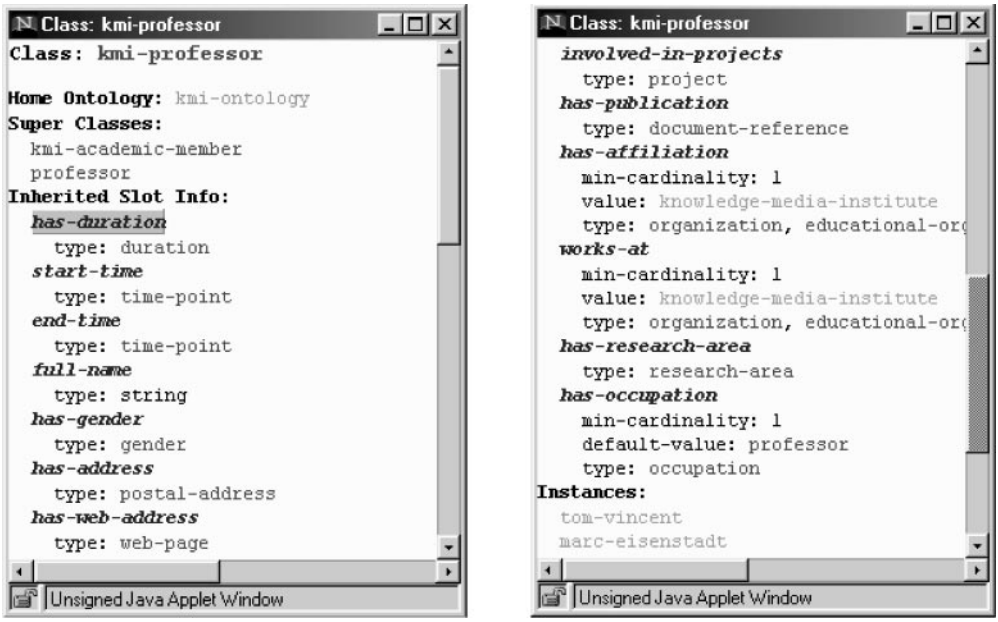

FIGURE 6. The top and bottom halves of a fine-grained view window giving the details of a class.

Since WebOnto was made publicly available on the web in October 1999, we have already had several hundred visitors to the site and the tool has attracted significant interest from both academic and industrial organizations. The university of Amsterdam has carried out a comparative evaluation of several knowledge modelling tools (Duneiveld, Stoter, Weiden, Kenepa & Benjamins, 1999), including WebOnto. This was

evaluated very favourably, in particular being judged as the most user-friendly and as the one requiring the shortest learning curve.

Because WebOnto is primarily aimed at expert model builders, its relationship to knowledge management is indirect: its role is to support the developers of the underlying ontology, rather than the users of the target knowledge management application. However, (1) given that WebOnto has been designed to be as easy to use as possible and (2) in many cases end users want to inspect an ontology directly (for instance, to gain a better understanding of the underlying organization), we usually include pointers to WebOnto in our application interfaces. Hence, in practice we can consider WebOnto as part of our suite of tools for supporting knowledge management applications.

3.3. LOIS, A FLEXIBLE FORM-BASED INTERFACE FOR KNOWLEDGE RETRIEVAL

The aim of Lois is to provide a web-based interface for posing queries to a knowledge model at a level which abstracts from the underlying modelling language. This goal has been accomplished by developing a form-based interface which allows users to select "key concepts" in the ontology and then construct a query by navigating the structure of the ontology (i.e. by following relations between concepts). This navigation leads to the creation of a query as a *list of rows*, which are linked by logical connectives. For instance, Figure 7, which is taken from our electronic publishing domain, shows a query which asks for a member of the KMi academic stuff, who is involved in the development of software visualization technology. This query was constructed by selecting class `kmi-member` first, specializing it to `kmi-academic-member`, selecting the relation `has-research-area` and then circumscribing the range of this relation to `res-area-software-visualisation`. More specifically, the first row of the query was created by (1) selecting the "Member of KMi" button, (2) selecting "concept-sub-type" in the "Attributes" window and (3) selecting "kmi-academic-member" in the "Attribute Types" window. The second row was created by selecting "More about kmi-member1" in the pop-up menu underneath "Organization", then selecting relation "has-research-area" in the "Attributes" window, then "kmi-research-area" in the "Attribute Types" window, and then "res-area-software-visualization" in the "Values" window. The items in each window pane dynamically show the choices relevant to the currently selected item in the pane immediately on the left. For example, the window "Attributes" in Figure 7 shows all the attributes appropriate to the currently selected item under the "Concepts" window, which is "More about kmi-member1". Analogously, once we select item "has-research-area" in the "Attributes" window, then the window "Attribute Types" is automatically updated, to show the types of research areas known to the underlying knowledge base.

The Lois interface is created automatically once the key classes for a knowledge model have been specified. In the example, these are "Story Event Type", "KMi Technology", "KMi Project", "KMi Member" and "Organization". These act as the entry points to the space of possible Lois queries. In other words, Lois can cover any query, as long as this can be specified as a path through the knowledge model, starting from one of the key classes. The usability of the Lois interface also depends on the terminology used by the underlying ontology. The knowledge engineer is therefore required to employ class and slot names which can be understood by the Lois user. If this requirement is satisfied, then the user only needs to learn to construct queries through the accumulation of rows.

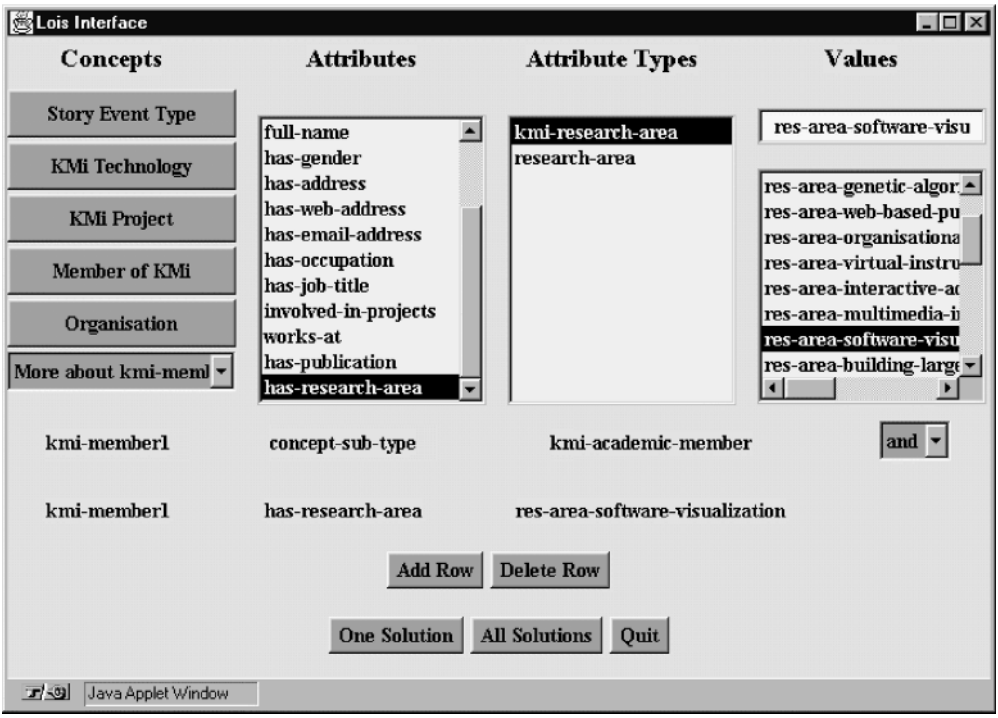

FIGURE 7. Finding a KMi researcher who works on software visualization.

3.4. ONTOLOGY INSTANTIATION USING KNOTE

Our goal is to enable as wide an audience as possible to take part in the ontology-driven construction of a shared knowledge model. Knote was therefore designed to be "low entry", so that users would not necessarily require a background in knowledge modelling. At the same time Knote should allow experienced ontology engineers the freedom to create arbitrarily complex OCML expressions. Knote provides instance forms which are modelled on the *dynamic forms* of Girgensohn, Zimmerman, Lee, Burns and Atwood (1995). The key difference between these two types of forms is that instance forms in Knote are generated directly from the ontology and not from a user description.

The Knote forms are created on the fly from a class definition and are thus domain independent. This approach has the benefit that it is possible to edit classes and instances iteratively, a feature which is often required during model development. This capability is not supported by systems such as Protégé-II (Eriksson, Puerta & Musen, 1994), where the instance forms are created by a distinct meta-tool. Recent work on Protégé-2000 (Eriksson, Fergerson, Shahar & Musen, 1999) seeks to alleviate this problem by allowing classes to be viewed as instances of meta-classes.

An example of a Knote form is presented in Figure 8. The figure shows a form partially instantiating a medical guideline for the prevention of pressure ulcers—see Section 6 for more details on this domain. The structure of the form was derived from the definition of

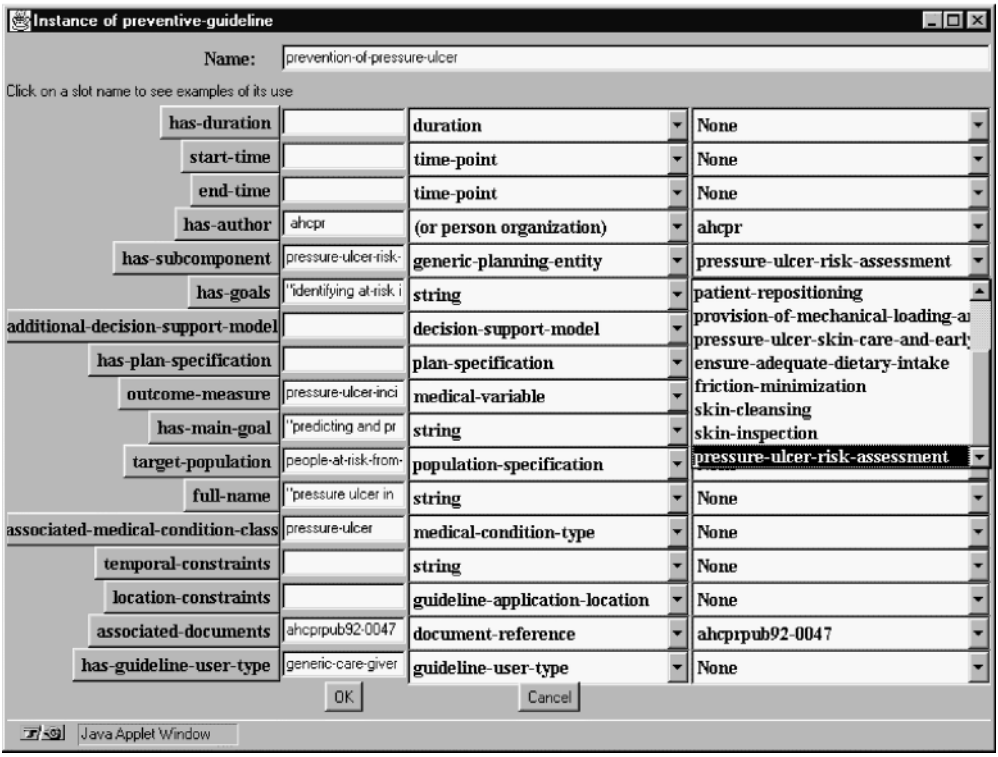

FIGURE 8. Class instantiation with Knote.

class `preventive-guideline` in the ontology `pressure-ulcer`. Knote provides the user with quite a lot of help in filling the form. When a slot is typed, Knote allows the user to navigate the sub-classes and the instances of the given type, to select an existing instance or to create a new one. In the latter case, the appropriate Knote form is then generated. The user can also click on a slot name (listed in the leftmost column of the form) to get examples of the use of the slot. Our experience suggests that this example-centred support tends to be more useful than the generic documentation associated with a slot. Figure 9 shows the help provided when the user clicks on slot `associated-medical-condition-class` in the leftmost column of Figure 8.

With the description of the Knote form-filling support we have completed our brief overview of our technology for knowledge modelling. In the next sections we will illustrate examples of the approach in three application domains.

4. Planet-Onto: enriching news stories

4.1. THE SCENARIO

KMi Planet (Domingue & Scott, 1998) is a web-based news server which facilitates communication within our laboratory (the Knowledge Media Institute) and allows the

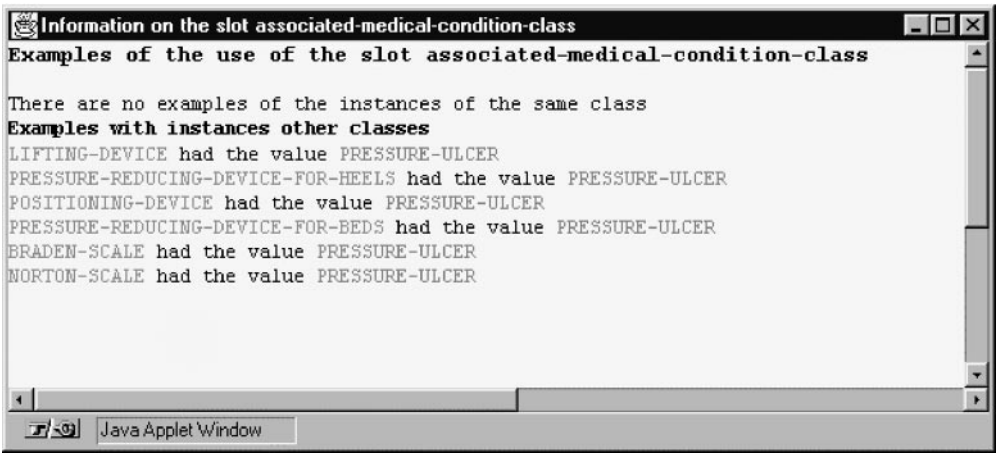

FIGURE 9. Example-based slot documentation.

wider community to access lab-related items of interest. Unlike most news-of-the-future projects, which focus on personalized news feeds, KMi Planet replicates some of the functionalities provided by newsrooms. "Journalists" send stories using the lowest common-denominator medium—an email message—and 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, where the submitted story is presented in the context of other stories, as shown in Figure 10. Planet is used as the "front door" to our laboratory, both metaphorically and physically (a dedicated machine running Planet is stationed at the entrance of the laboratory, so that visitors to KMi tend to be shown or "play with" Planet first). Our archive is growing steadily and now contains well over a hundred stories, submitted by 13 journalists. We currently have 480 registered "readers", i.e. users who subscribe to the Planet alert services. Planet has been a "success story" and numerous versions of the newsletter have now been produced for other organizations, both within and outside the Open University.

For all this success, it is apparent that, as the Planet archive and readership grow, more sophisticated mechanisms supporting semantic searches and individualized presentations and alerts are needed. Users of Planet often come across interesting news items but they cannot easily follow-on with obvious queries. For instance, having read a story about an award for a paper about visualizing genetic algorithms, a user may want to find out who else works on software visualization, what other projects are going on in this area, etc. Of course, many answers can be found by browsing our web pages and through standard search mechanisms. However, even in well-organized sites, many important relations between people, technologies, projects and organizations are often missing, 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, the experience of a day-to-day use of Planet over more than two years has highlighted a number of other

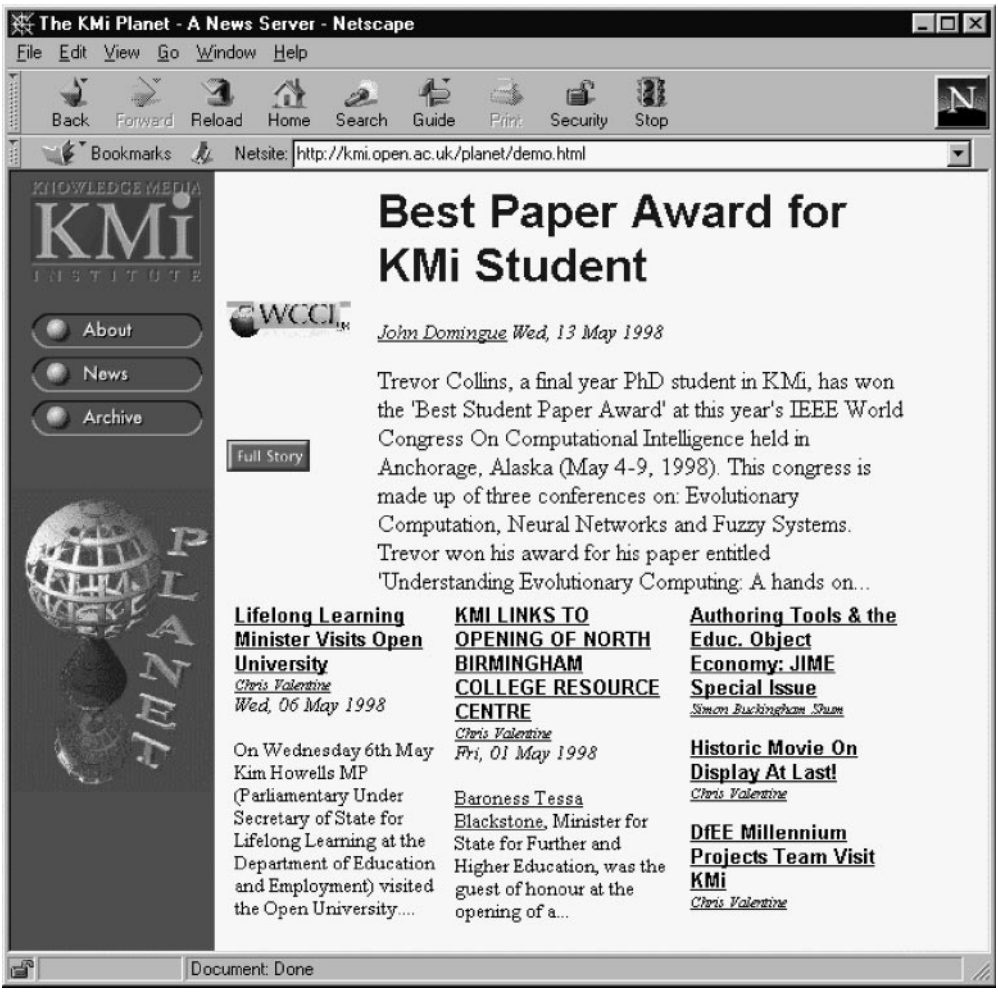

FIGURE 10. The front page of KMi Planet.

issues. An obvious one is the need for individualized news feeds and presentations—currently, 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 but proactively identified and requested, in accordance with events in the department and the observed interests of the readership. To address these issues we have developed an integrated suite of tools, *PlanetOnto*, which extends the original Planet news server by providing support for ontology-driven document enrichment, integrated browsing and deductive knowledge retrieval, personalized news feeds and alerts, and proactive identification of potentially interesting news items.

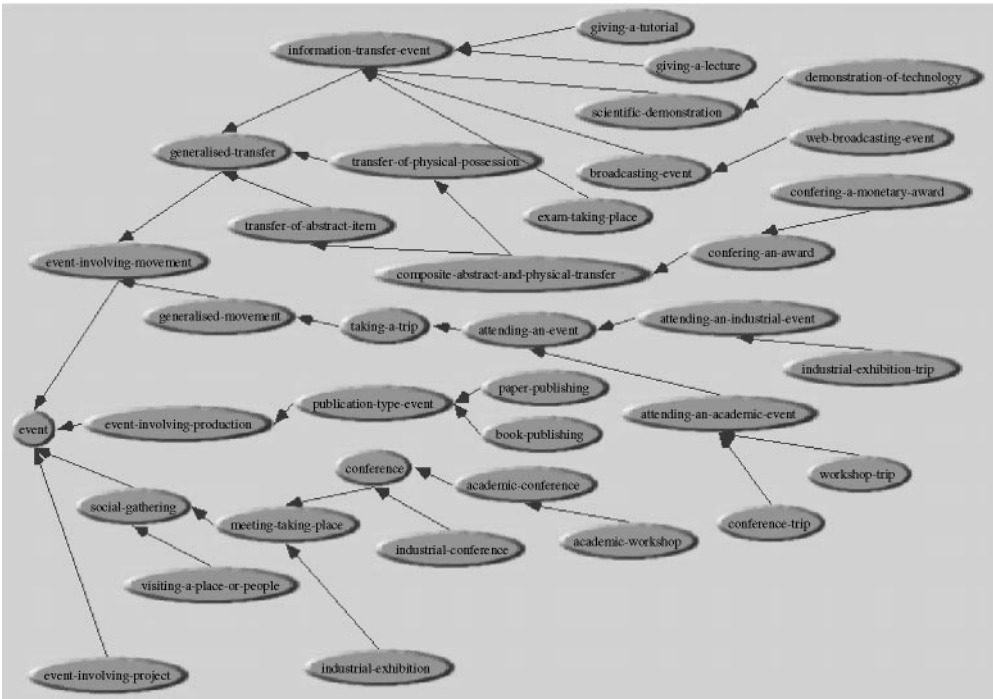

FIGURE 11. Taxonomy of events in the Planet knowledge base.

4.2. INSTANTIATING THE APPROACH

4.2.1. Characterize viewpoint for ontology

The viewpoint is centred on the range of academic events which characterize the life of an academic department. Hence, we have developed a rich taxonomy of events, which is shown in figure 11. This taxonomy includes both very abstract events, such as `event-involving-movement`, which are defined in a highly generic ontology (`event-ontology`), as well as specific events, related to academic life, such as `giving-a-lecture`. In total, we have about 40 different types of events, of which 25 have been defined specifically for the KMi Planet application. In addition, we have identified five key classes, which have driven both the development of the ontology and the design of the Lois interface shown in Figure 7. These are `people`, `organisations`, `stories`, `projects` and `technologies`.

4.2.2. Develop the ontology

The Planet ontology builds on a number of other ontologies included in the library, as shown in Figure 12. Some of these ontologies were reused from the Ontolingua repository at Stanford—e.g. `simple-time` and `bibliographic-data`, others were developed from scratch. It is difficult to quantify the overall development effort, as these ontologies are held in the WebOnto central repository and grow steadily over time. As

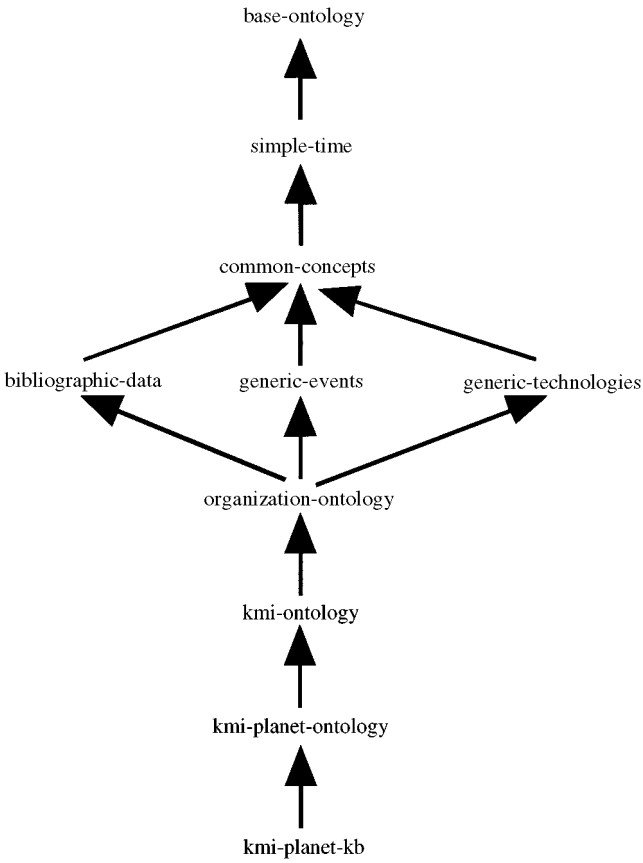

FIGURE 12. Ontology inclusion in the Planet Ontology.

an indication, we can point out that the first version of the Planet ontology took about two weeks to complete and proved adequate to support the range of queries envisaged in the use scenario.

4.2.3. Perform ontology-driven model construction

A prototype model was developed using a subset of the stories. This initial prototype was used to gather initial feedback about the technology and the approach, and used as a showcase demo to publicize the approach to KMi visitors. The experience from the first model construction exercise led to the implementation of slot-renaming mechanisms in OCML. A new, complete version of the Planet knowledge base was developed during the summer of 1999. Full-scale trials of the Planet-Onto system will begin in May 2000.

4.2.4. Customize query interface for semantic knowledge retrieval

This step consisted of developing the Lois form shown in Figure 7. As pointed out earlier, the design of the form was driven by the identification of the key concepts in the ontology. Experience so far shows that the design covers the type of queries which are

FIGURE 13. Solutions to query about academic KMi members active in software visualization.

asked to Planet-Onto, i.e. queries tend to be centred around key concepts. Figure 13 shows part of the web page produced by the Lois server, when answering the query asked in Figure 7. In this case, four solutions were found, i.e. four KMi academic members working in the area of software visualization. The user can click on each item to inspect it, i.e. to find what is known about the item in the PlanetOnto knowledge base and can also try to find stories relevant to the item. Our PlanetOnto knowledge base includes dozens of heuristics to associate items in the knowledge base to stories. For instance, a story is related to a KMi member, say M, if it describes a technology developed by M, or if it talks about a project run by M. Analogous heuristics have been defined to associate stories to technologies, organizations, projects and events. Thus, in this application we use the PlanetOnto knowledge base "to enrich" the Planet newsletter in two main ways: we allow users to find answers to queries which cannot be answered simply by browsing stories and we use the knowledge model as a semantic indexing device for Planet stories.

4.2.5. Develop additional reasoning services on top of knowledge model
We have designed (but not yet implemented) two intelligent agents, whose purpose is to provide additional services by reasoning about the formalized knowledge model. These are described in the next sub-sections.

4.2.5.1. Story chasing with NewsHound. *NewsHound* will periodically gather data about "popular" news items and will use these data to solicit potentially popular stories from the appropriate journalists. This will be accomplished by identifying "gaps" in the knowledge base, e.g. projects about which there is no information. The design of NewsHound is consistent with one of the main goals in the design of KMi Planet: the system should try and emulate a news room team. One of the tasks that a news editor carries out is to identify potentially popular stories and assign them to one of the journalists in the staff. NewsHound is meant to emulate this behaviour. In order to identify potentially interesting stories, NewsHound will use two main types of data: statistics on access to individual stories in KMi Planet and records of the queries posed through Lois. Each story within Planet keeps a record of its popularity by counting the number of times the full text is requested from the KMi Planet server. Once NewsHound identifies a story as "popular", then it tries to identify related stories which have the potential to be popular. To perform this task, NewsHound analyses the knowledge base trying to find items of interest that have not yet been covered by Planet stories. Typically, these are projects and technologies which (1) are known to NewsHound, (2) are "related" to "popular" projects and technologies, but (3) have not yet been covered by a story. The term "related" is the key here. NewsHound will use various heuristics to define "relatedness". For instance, direct sub-classes of the same class are considered related; technologies are related if they build on the same underlying technology; projects are related if they tackle the same areas. These heuristics are of course completely "soft" and modular and therefore any new one can be added without affecting the existing ones.

An interesting feature of NewsHound has to do with 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 (van Heijst, 1995). In our scenario, incompleteness is defined in pragmatics terms: publications need popular stories.

4.2.5.2. Providing personalized alerts with NewsBoy. Lois has been designed (among other things) to help users to track down Planet stories with very specific characteristics. However, a significant number of users prefer to work with *push technology*, that is they prefer to be automatically notified about potentially interesting stories, rather than having to query Lois about them. We have therefore designed an agent, *NewsBoy*, to provide a mode of use that is complementary to the one supported by Lois. NewsBoy enables users to create a personalized front-page to which interesting stories are "pushed".

When a new story is formalized and added to the Planet-Onto repository, NewsBoy matches the story against the specified interests of registered readers. Readers whose interests match that of the story are notified by email that a new story has been added to their personal Planet page. To make an explicit declaration, a reader simply specifies a number of queries using the Lois interface. The reader is then updated when a new

story matches at least one of the logged queries. Alternatively, a reader can state that she would like NewsBoy (1) to log all the queries she makes using Lois and (2) to create a user profile from the log. The resulting user profile is simply the logical disjunction of the queries contained within the log.

It is interesting to compare NewsBoy to other approaches which attempt to infer user profiles from analysing patterns of access to documents (see e.g. Liebermann, 1995; Krulwich & Burkey, 1997). These approaches try to induce user interests using empirical methods. Our approach is semantic-centred: the user herself specifies the range of documents of potential interest through unambiguous declarative specifications.

4.3. SUMMING UP

The PlanetOnto application historically defined our approach to ontology-driven document enrichment and provides a showcase for both the technology and the approach. It shows that it is possible to use ontologies to circumscribe a range of phenomena in the workplace, that it is feasible for a motivated community of non-expert users to perform ontology-driven collaborative model development and it embeds a number of additional facilities which enrich a textual archive with semantic indexing, deductive knowledge retrieval and intelligent agents. In particular, it shows the advantages derived from adopting a modular approach, which starts from highly generic ontologies to arrive to an application-specific knowledge base, whose main role is to support heuristic knowledge retrieval and semantic indexing of documents.

5. Scholonto: supporting scholarly debate

5.1. THE SCENARIO

Contextualizing ideas in relation to the literature is a fundamental task for authors and readers—are they new, significant and trustworthy? Scholars accomplish this task firstly by bringing to bear their own knowledge of the field. This process leads to *commentary and discourse* of various kinds, which reflect the extent to which peers regard an authors work as authoritative. These can take the form of private exchanges, formal peer review of conference/journal submissions or published reviews of literatures and books. We can think of conventional scholarly publication and debate as a document-centred, text-based process. Text is a rich medium in which to publish and discuss ideas in detail and with subtle nuances, but the corresponding disadvantage is that it takes a long time to read and is hard to analyse computationally.

A complementary approach focuses on the *conceptual models* implicit in textual documents and discourse. The goal is to provide a *summary* representation of ideas and their interconnections, in order to assist literature-wide analysis. We believe that this approach has advantages over textual media for tracing the intellectual lineage of a document's ideas, and for assessing the subsequent impact of those ideas—that is, how they have been challenged, supported and appropriated by others. In addition, the availability of explicit conceptual models opens possibilities for automatic analysis of a community's collective knowledge.

We begin with the idea that an author's goal is to persuade the reader to accept his/her *perspective*, which constitutes a set of claims about the world. Usually, the author has some new ideas that she/he is contributing, and asserts particular relationships between these and existing ideas already published in order to demonstrate both the reliability of the conceptual foundation on which she/he is building, and the innovation and significance of the new ideas. The scholarly reader's task is to understand which ideas are being claimed as new, and assess their significance and reliability. Let us switch from a reading scenario to the scenario of literature search and analysis. In this case, the scholar has some ideas and relationships in mind that she/he is trying to locate in the literature—has anyone written about them, or perhaps these ideas exist but not yet in a single document? The interpretative task includes (1) formulating the ideas of interest in ways that may uncover relevant documents, (2) reading the documents (as just described) and then (3) interpreting them to characterize any patterns that appear to emerge. This is a similar scenario to that of a newcomer to a scholarly community—e.g. a student, librarian, lecturer or researcher from another discipline, who wants to know, for instance, what the seminal papers are, or if there are distinctive perspectives on problems or techniques that define that community.

We contend that scholars are very poorly supported in these tasks by conventional library and technological environments, but that digital libraries open up new possibilities which have yet to be exploited. Consider the document interpretation scenario. In the non-digital world, there is currently no way beyond following citations (only those provided by the author), or using citation indices (to find others citing him for some reason), to ask questions such as: "Has anyone built on the ideas in this paper, and in what way?", "Has anyone challenged this paper?", "Has anyone proposed a similar solution but from a different theoretical perspective?". These are arguably the kinds of phenomena of most interest to scholars when they read or write papers, engage in debate or search the literature. These are also the kinds of questions asked by researchers unfamiliar with a literature. Our goal is to support these kinds of queries by means of our enriched-document approach. Specifically, we have developed an ontology to support scholarly debate and we plan to use this ontology to characterize scholarly relations between documents.

5.2. INSTANTIATING THE APPROACH

5.2.1. Characterise viewpoint for ontology

It might appear paradoxical to propose the use of ontologies to support scholarly communities in managing their knowledge, since conflicting worldviews, evidence and frames of reference lie at the heart of research and debate. Of course, the key issue is what is represented. It is hard to envisage when scholarly communities will no longer need to make claims about, or contest, the nature of a document's *contributions*—e.g. "this is a new theory, model, notation, software, evidence", or its *relationships* to other documents—e.g. "it applies, modifies, predicts, refutes …". Our approach builds on this relatively stable dimension of what are otherwise constantly evolving research fields, by representing scholars' *claims about the significance* of ideas and concepts—a focus on *discourse* and *argumentation* (how scholars support and contest claims) and on *context*

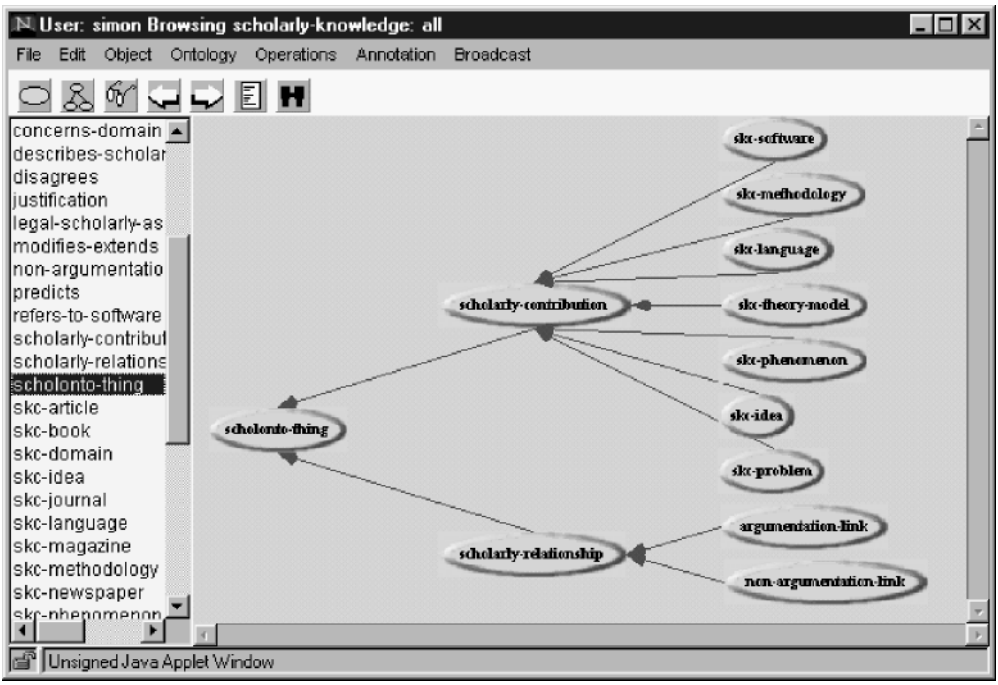

FIGURE 14. Main taxonomy of the ScholOnto ontology.

(the conceptual network in which an idea is embedded). Representing concepts separately from claims about them is critical to supporting multiple perspectives.

5.2.2. Develop ontology

Figure 14 shows the top level structure of the ontology. The semantic network model of the literature is created by scholars' submissions of scholarly contribution elements (nodes) and scholarly relationships (links), the latter being subdivided into *argumentation* and *non-argumentation* links. The ontology is designed to support scholars in making claims by asserting relationships between concepts. Other scholars may for instance *support*, *raise-issues-with*, or refute these claims. A claim is formally defined as a relation between a set of authors, who make a `legal-scholarly-assertion`, with some justification.

The design of the ontology was based on the analysis of scholarly articles from a range of different fields, and took about two person weeks' effort. Once the top-level structure stabilised, it required only two days to develop the first version of the ontology in OCML.

5.2.3. Perform ontology-driven model construction

We are currently seeding a knowledge base with document descriptions to test the ontology's ability to support the scenarios described in this paper. We have also designed a domain-specific version of Knote, which guides users through the modelling schema using dynamic menus, and enables them to browse and search for existing concepts to assist their reuse. This interface is shown in Figure 15.

5.2.4. Customize query interface for semantic knowledge retrieval
A Lois form has been designed, which is shown in Figure 16. In the example the user is trying to assess the motivation behind, and impact of, a theoretical model called the *Dexter Hypertext Model*. The queries specify, respectively, (1) what problems does it analyse? (2) are there any theory-models which modify-extend it? and (3) is there any software which uses or applies it? Analogously to the use of heuristic search patterns in PlanetOnto to support semantic retrieval of news items associated with entities in the knowledge base, we plan to provide links to on-line bibliographic databases, to automatically retrieve papers which exemplify a certain approach/technology/idea/etc. This will provide a much more powerful bibliographic search tool than the conventional, keyword-centred ones, which are currently available in digital libraries.

5.2.5. Develop additional reasoning services on top of knowledge model
A knowledge model enables inference-based searching and alerting. It will be possible to ask the system questions such as "What impact did Theory T have?", since "impact" can be defined, for example, in terms of the number of subsequent documents using or modifying the theory, the number of different domains in which it has been applied, the number of problems addressed which drew on the theory, and so forth. Our knowledge modelling environment makes it simple for us (as system maintainers) to write heuristics that could assist in finding relevant documents—e.g. "if Method Y extends Method X, and Method X is challenged, then Method Y may be challenged". Moreover, as already mentioned, it will be possible to develop specialized agents whose goal is to identify emerging perspectives, using heuristic knowledge and machine learning techniques. As these machine-discovered assertions are added to the knowledge model, software agents effectively become actors in the scholarly debate. This scenario provides an example of a hybrid agent community and therefore raises a whole host of interesting issues, from the dynamics of social interaction to the design of epistemic agents—see Masterton (1998) for an another example and for a detailed discussion of the relevant issues. More in general, the scenario is an example of the general trend towards reducing the boundaries between humans and machines (Stutt & Motta, 1998).

6. Knowledge management of medical guidelines

The area of medical guidelines provides a very good example of document-centred knowledge work. Guidelines are produced as the result of extensive discussions involving large numbers of experts and stakeholders, discussions which are based on both practical experience and existing literature and which result in ever refined and updated versions of collaboratively produced documents. The aim of the EC-funded PatMan project (PatMan, 1998) is to develop a number of technologies to support guideline-centred patient management. These technologies include an editor/interpreter for medical guidelines, which has been produced by the Medical Informatics group at the University of Pavia, as well as a configuration of our knowledge management technology for this domain.

Figure 17 shows the *repository of enriched guidelines*, which we have developed in the course of the PatMan project. This repository integrates the D3E support for discourse and debate (Sumner & Buckingham Shum, 1998) with the ontology-driven approach to

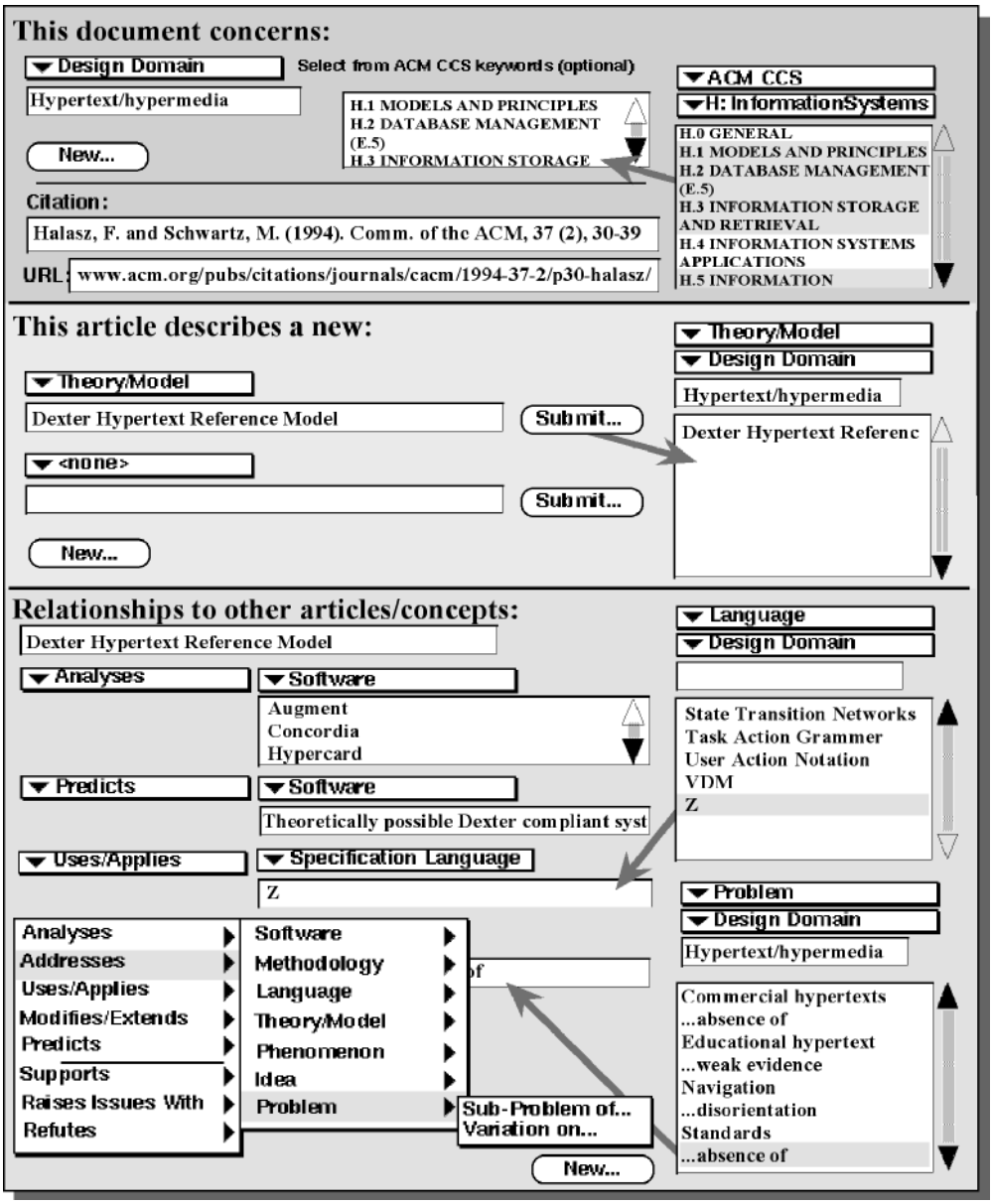

FIGURE 15. Data entry interface to ScholOnto.

document enrichment illustrated in this paper. Thus, we aim to support various knowledge-intensive tasks, both during the design and the application of a guideline. For the sake of explanation, we will start our discussion by illustrating the use of D3E technology in this domain.

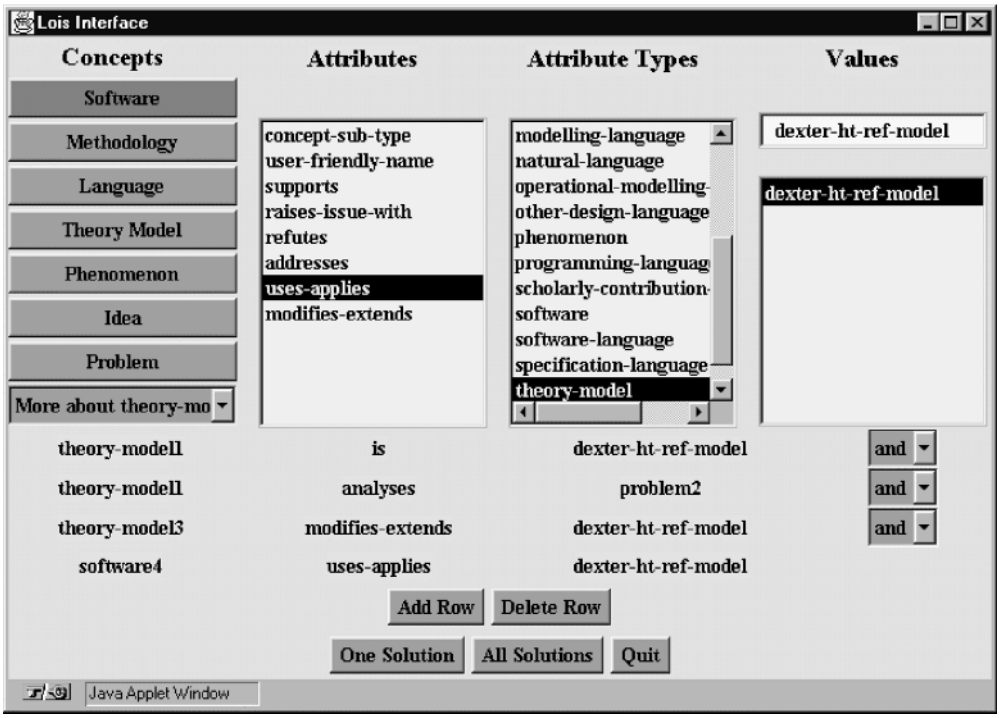

FIGURE 16. Lois interface to ScholOnto.

6.1. SUPPORTING DISCOURSE AND DEBATE IN THE AREA OF MEDICAL GUIDELINES

The D3E technology for discourse and debate can be used to facilitate both the design and the application of a guideline. During the design phase, we can use multiple discussion spaces to structure and record the debate associated with the drafting of a document; in the health-care practice, it can be used (for instance in association with an executable model of the guideline in question) to highlight user problems, modifications, feedback and discussions about those parts of the guideline which are under-specified, either because they are deliberately left open for customization in the local setting, or because of a bug in the specification.

The frame shown in Figure 17 was produced using the CEDAR toolkit, which allows publishers to specify the layout of the target D3E environment using a high-level interface toolkit. The window pane in the middle shows part of the document describing a guideline for the prediction and prevention of pressure ulcer, which has been produced by the Agency for Health Care Policy and Research (AHCPR) in the united states. The window pane on the left provides an outline of the main document, to be used as a navigation device. On the right it is possible to see the various discussion spaces associated with the guideline document. By clicking on a particular discussion space, the user can see the thread of the discussion associated with that particular topic (or document section). Once a discussion space has been selected, the user can add a comment, using a simple set of discussion-structuring primitives—see Figure 18.

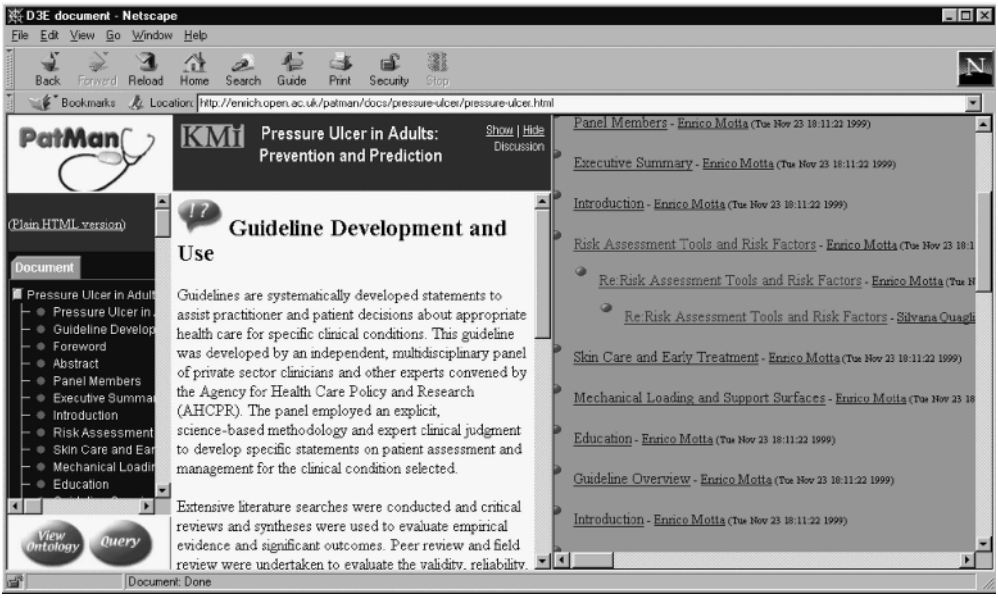

FIGURE 17. The PatMan repository for enriched guidelines.

6.2. ONTOLOGY-DRIVEN DOCUMENT ENRICHMENT IN THE AREA OF MEDICAL GUIDELINES

6.2.1. Characterize use scenario

The use scenario in the medical guideline domain is very similar to the PlanetOnto one. We want to associate a knowledge model to a set of documents, to provide intelligent knowledge retrieval and semantic indexing facilities. In addition, the availability of a formalized knowledge model opens the door to the development of additional tools, such as validation and verification tools customized for medical guidelines. Finally, we want to integrate the document-centred support provided by the D3E technology with the knowledge modelling technology. This last task has been tackled simply by augmenting the D3E-style window frame shown in Figures 17 and 18 with two buttons, which allow the user to browse the knowledge model associated with the current guideline and to query it using the Lois interface.

6.2.2. Characterize viewpoint for ontology

Our viewpoint here is simply "medical knowledge". In other words, when modelling guidelines we focus mainly on the various knowledge types associated with a guideline (such as the associated medical condition, the target population, the classes of users, etc.), rather than the algorithmic aspects. The latter have been addressed in the PatMan project by our colleagues at the University of Pavia, who have developed an editor/interpreter for medical guidelines.

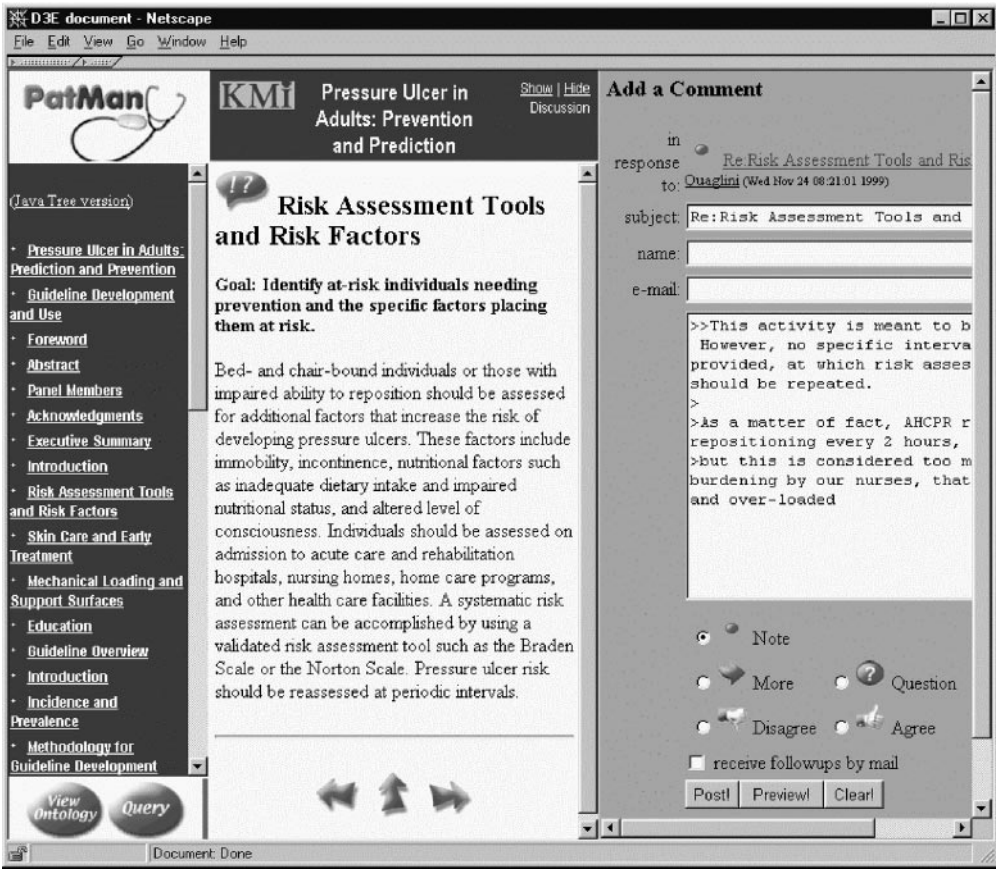

FIGURE 18. Adding a comment to the discussion space associated with the risk assessment task.

6.2.3. Develop ontology

As shown in Figure 19, the `medical-guidelines` ontology builds on eight generic ontologies, seven of which are also used by the PlanetOnto model described in Section 4. The odd one out here is the generic medical ontology developed in an earlier medical informatics project (HC-ReMA, 1997), of which the medical-guidelines ontology is a direct specialization.

When developing the medical-guidelines ontology we took the notion of medical guideline as our starting point and most of the effort was spent in characterizing this class and the associated classes and relations, e.g. *outcome-measure*, *guideline-user-type*, etc. Because the users are expected to be health professionals, we do not envisage the need to provide "deep" medical models, beyond the formalization associated with a medical guideline.

As shown in Figure 20, a medical guideline is modelled as a sub-class of class `plan`. Hence, the slots associated with a plan specification (i.e. the slots typically used by guideline interpreters) are kept separate from additional information about the guideline. We distinguish between three types of guidelines: preventive, therapeutic and diagnostic.

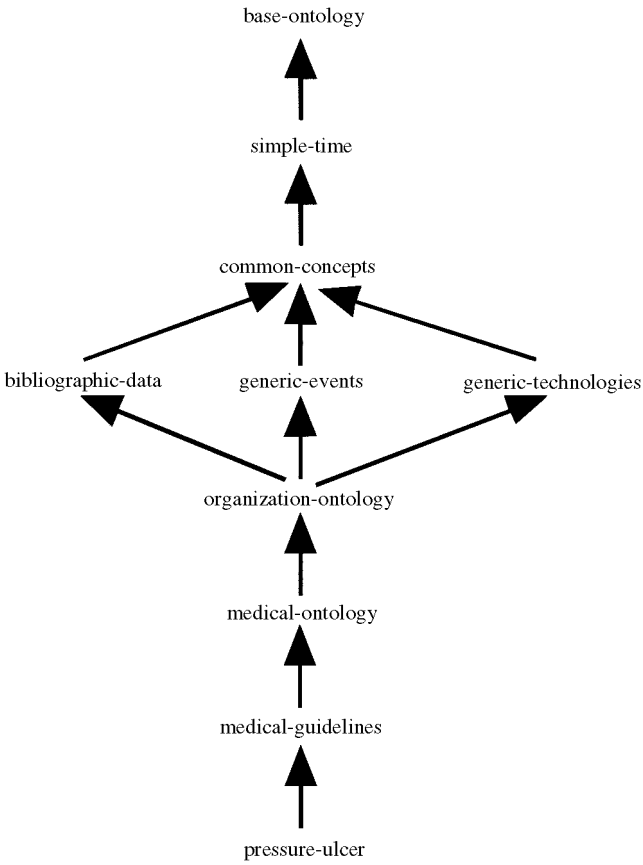

FIGURE 19. Ontology inclusion in the `medical-guidelines` ontology.

A medical guideline is characterized in terms of 17 slots. Here we describe a subset of them, to illustrate the type of information associated with a medical guideline in our model. The complete ontology can of course be browsed on the web, through the WebOnto editor/browser.

- `Has-Guideline-User-Type`. This specifies the class of users which carry out the procedures specified in the guideline. Guidelines can be specified for specific categories of users, e.g. nurses or physicians, or can include procedures targeted at different categories—e.g. some to be carried out by nurses and some to be carried out by physicians. A guideline user type can be `patient`, `software-agent`, or a sub-class of class `generic-care-giver`, whose `subclass-of` hierarchy is shown in Figure 21. It is important to note that class `guideline-user-type` is in effect a *metaclass*, i.e. a class whose instances are other classes.
- `Associated-Documents`. The documents describing the guideline. These are modelled as instances of class `document-reference`, which is defined in the ontology `bibliographic-data`.

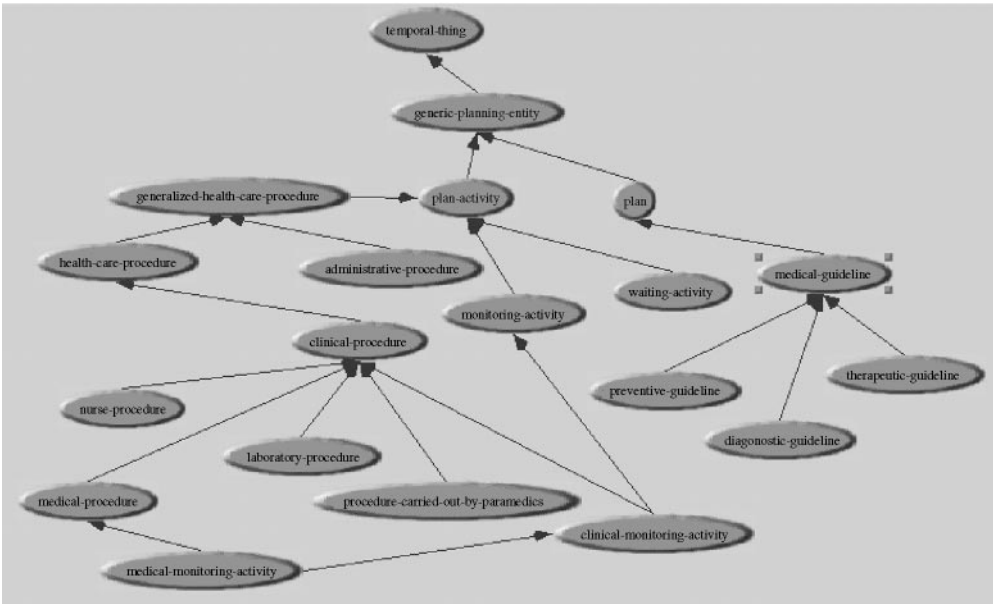

FIGURE 20. Sub-class-of hierarchy for class `generic-planning-entity`.

- `Location-Constraints`. Where to carry out the procedures carried out in the guideline, e.g. whether at home, at the local surgery, at a day hospital, etc.
- `Temporal-Constraints`. Whether the guideline has to be carried out at particular times, or before or after other procedures.
- `Associated-Medical-Condition-Class`. The type of medical condition associated with the guideline. For instance, the guideline "Pressure Ulcer: Prevention and Prediction" is associated with the condition "Pressure Ulcer". In general, a guideline can be associated with more than one type of medical condition.
- `Target-Population`. The population targeted by the guidelines. For instance, the pressure ulcer guideline is targeted at people at risk from pressure ulcers, e.g. people with limited mobility. Instances of this slot should be members of class `Population-Specification`.
- `Has-Plan-Specification`. An executable specification of the guideline. This is the part which is needed by a guideline interpreter to implement computerized workflow support. The class `plan-specification` provides a simple control language to model the algorithmic aspects of a guideline, by means of operators such as `one-of`, `one-or-more-of`, `sequential-all-of`, `non-ordered-all-of`, etc.
- `Has-Subcomponent`. A guideline normally includes a number of procedures, subplans and possibly other guidelines. We use the term "sub-component" to refer to a generic component of the guideline, which is meaningful from a planning point of view. The range of this slot is restricted to instances of class `Generic-Planning-Entity`, whose sub-class-of hierarchy is shown in Figure 20.

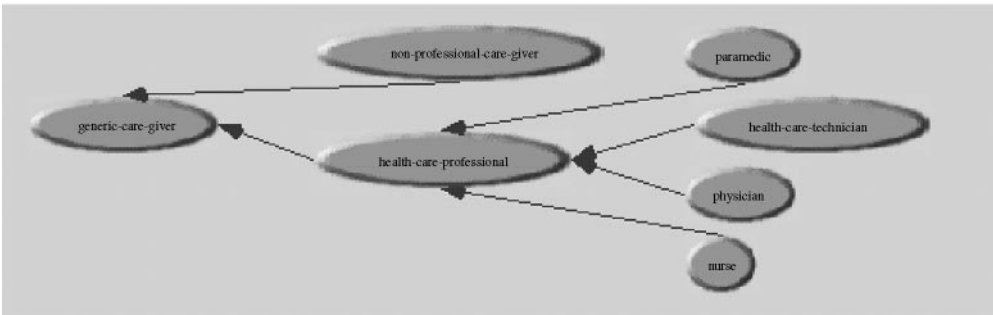

FIGURE 21. Subclass-of hierarchy for class `generic-care-giver`.

The `medical-guideline` ontology contains about 120 definitions, including classes, relations and instances. If we count also the definitions inherited from super-ontologies, then the resulting model comprises over 1000 definitions.

6.2.4. Perform ontology-driven model construction

As already mentioned, we have tested the `medical-guidelines` ontology in the domain of pressure ulcer. The purpose of the pressure ulcer guideline is to help identify adults at risk of pressure ulcers, to define early interventions for prevention and to manage Stage I pressure ulcers, the weakest form of pressure ulcers.

A new ontology, called `pressure-ulcer`, was created on the WebOnto repository and a model of the guideline was created, by instantiating the generic definition of class `medical-guideline` and by defining the additional entities specifically associated with the guideline. These included the guideline components (e.g. "skin inspection" and "risk assessment"), the notion of "pressure ulcer", its various stages, the notion of "people at risk from pressure ulcer", and a number of healthcare techniques and instruments associated with pressure ulcer. In total, about forty new definitions were created. The OCML definition of instance `prevention-of-pressure-ulcer` is shown in Figure 22.¶ The definition states (among other things) that the guideline has been produced by the AHCPR agency, that it has three direct sub-components and that it is targeted at generic care givers, i.e. it is not specific for a class of healthcare professionals.

Finally, we show in Figure 23 the definition of one of the sub-activities specified by the pressure ulcer guideline. In total, about a dozen sub-activities are included in the guideline specification.

6.2.5. Customize query interface for semantic knowledge retrieval

This step consisted of producing the Lois form shown in Figure 24. As usual, the form was generated automatically once the "key classes" in this domain were identified. The assumption here is that typical questions to the knowledge model will centre on some

¶ For the sake of brevity not all slots are shown in Figure 22. Again, the eager reader can check the full specification of this instance on the WebOnto server.

```
(def-instance prevention-of-pressure-ulcer preventive-guideline
  ((has-author ahcpr)
   (has-subcomponent
    pressure-ulcer-risk-assessment
    provision-of-mechanical-loading-and-support-surfaces
    pressure-ulcer-skin-care-and-early-treatment)
   (outcome-measure pressure-ulcer-incidence)
   (has-main-goal "predicting and preventing pressure ulcers")
   (target-population people-at-risk-from-pressure-ulcer)
   (full-name "pressure ulcer in adults: prediction and prevention")
   (associated-medical-condition pressure-ulcer)
   (associated-documents ahcprpub92-0047)
   (has-guideline-user-type generic-care-giver)))
```

FIGURE 22. Definition of the pressure ulcer guideline.

```
(def-instance patient-repositioning health-care-procedure
  ((is-performed-at-periodical-intervals? true)
   (has-min-periodicity 2-hours)
   (activity-of provision-of-mechanical-loading-and-support-surfaces)
   (responsible-agent-type generic-care-giver)
   (uses-healthcare-technique side-lying-position)
   (requires-healthcare-instrument lifting-device
                                   pressure-reducing-device-for-beds
                                   pressure-reducing-device-for-heels
                                   positioning-device)))
```

FIGURE 23. Definition of instance `patient-repositioning`.

medical condition, guideline, or healthcare professional or organization. The "Planning Activity" button makes it possible to ask question directly centred on a particular medical procedure, independently of a medical condition or guideline. Specifically, the query shown in Figure 24 asks for activities in the pressure ulcer guideline, which require the intervention of a physician.

As in the PlanetOnto scenario, heuristic rules associate classes of items in the knowledge model (e.g. medical conditions) to relevant sections of associated guideline documents. For instance, clicking on the sentence "Take me to the relevant guideline" next to the string "pressure-ulcer-risk-assessment" in Figure 25, takes us back to the PatMan enriched repository, configured so that the risk assessment section and associated discussion site are selected.

6.2.6. Develop additional reasoning services on top of knowledge model

Our main priority for the next few months is to integrate the PatMan configuration of our knowledge management technology with a suitable guideline editor/interpreter. Once this goal is accomplished, we plan to develop intelligent validation and verification tools, able to identify errors and omissions in the guideline specification, prior to its transformation to a format suitable for the interpreter.

7. Related work

From a representational and technical perspective, our approach differs from metadata-centred approaches (e.g. RDF, 1999), in that ontology specification languages provide

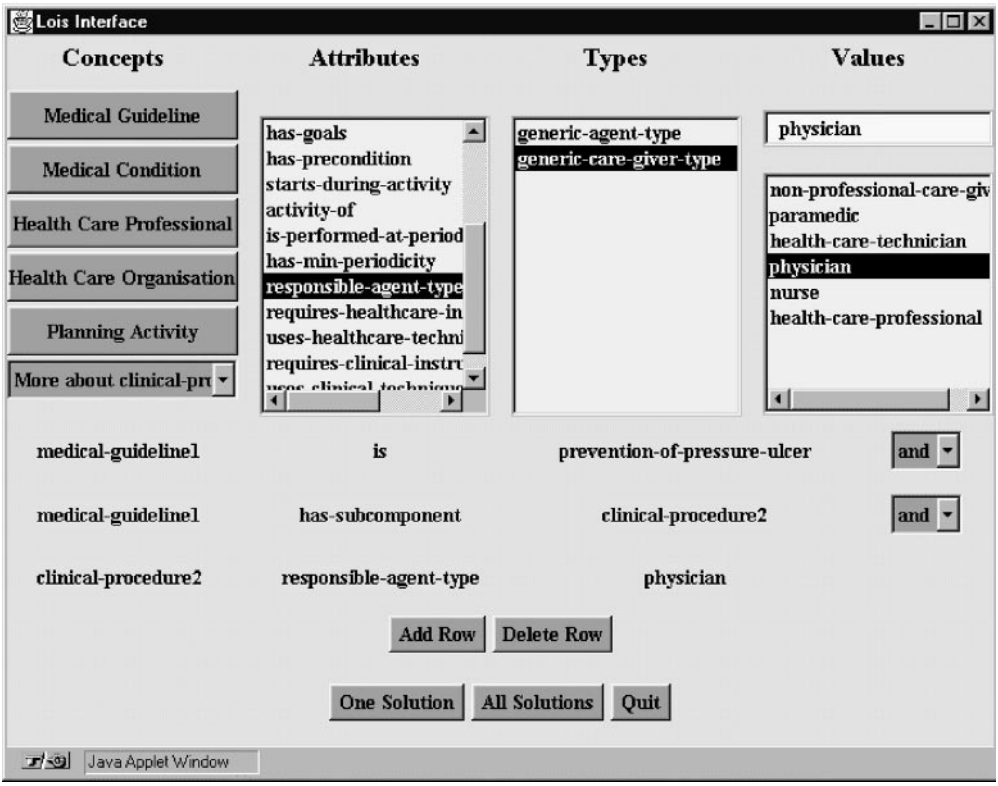

FIGURE 24. Finding activities in the pressure ulcer guideline needing medical intervention.

more powerful modelling capabilities than the formalisms currently available in the metadata community, e.g. for specifying relations. Moreover, languages such as OCML and Ontolingua also provide metalevel modelling support which makes it possible to reason about the ontology itself.

The Shoe project (Heflin *et al.*, 1998) has proposed an extension to HTML to allow the specification of ontological information. The project team has 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 allow the representation of information and provide tools to edit and retrieve it. We take a holistic approach to document enrichment and we look at the wider issues concerning usability and sustainability. Thus, we are not just concerned with providing a mechanism for associating knowledge structures to text but we wish to develop a comprehensive architecture addressing all the relevant issues, from the "right" approach to ontology development to the required visualization and interface tools needed to facilitate the development of knowledge models. Having said so, the technical solutions provided by Shoe could be easily integrated within our framework. For instance, OCML structures could be represented in terms of the relevant Shoe tags.

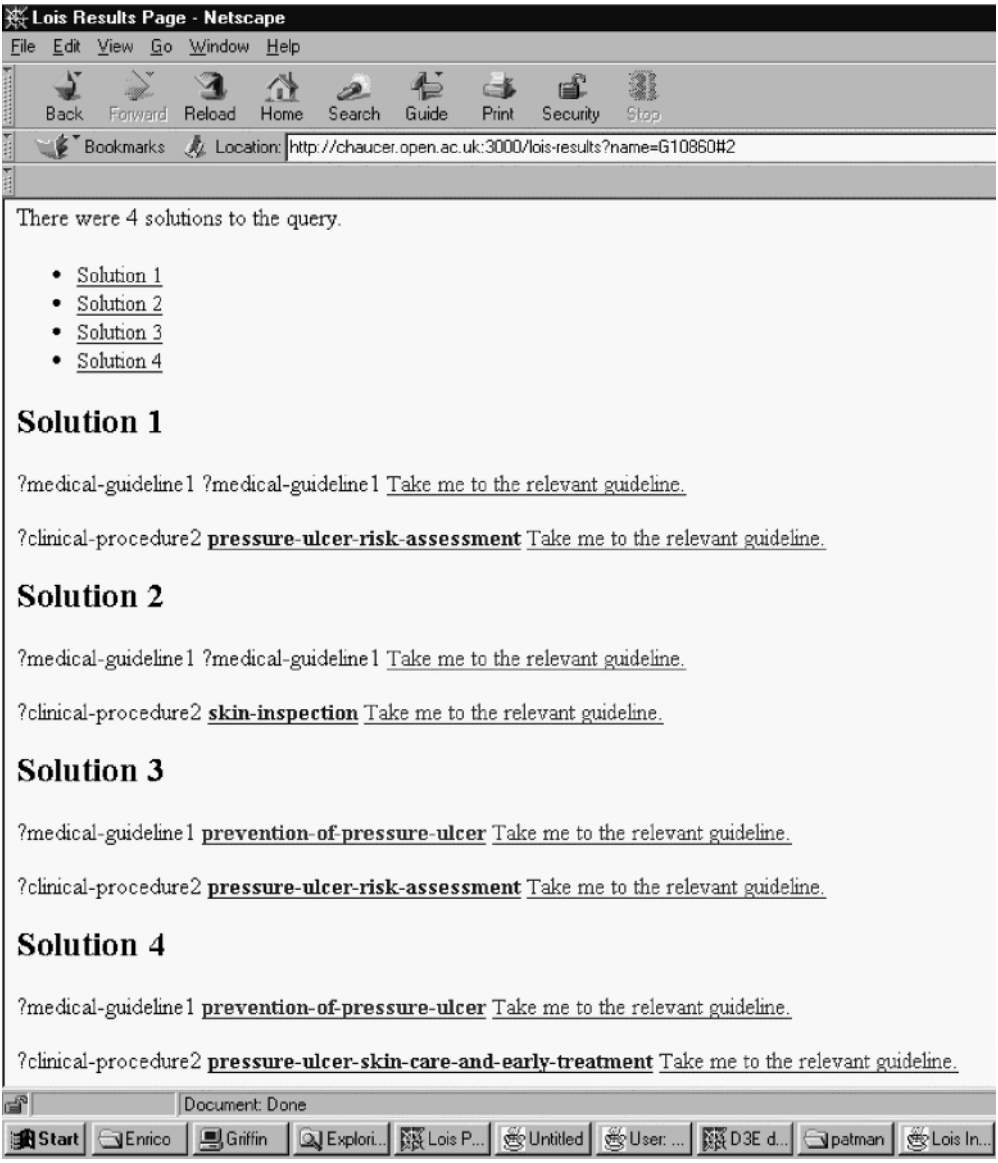

FIGURE 25. Solutions to the query shown in Figure 24.

The (KA)2 initiative (Benjamins, Fensel & Gomez Perez, 1998) shares a number of commonalties with our work. As in most of our scenarios, the aim of (KA)2 is to allow a community to build a knowledge base collectively, by populating a shared ontology. In the case of (KA)2, the knowledge base is meant to document the activities of the knowledge acquisition community. Similarly to the approach used in Shoe, the knowledge base is constructed by annotating web pages with special tags, which can be read by

a specialized search engine cum interpreter, Ontobroker (Fensel *et al.*, 1998). In this paper, we have emphasized that the feasibility of the idea of a collective construction of a knowledge base crucially depends on a number of features, including: (1) a carefully defined ontology; (2) an underlying modelling language providing user-oriented facilities, such as context-dependent renaming; (3) a user-friendly ontology instantiation environment; and (4) the right motivational stimuli for the participants. In their paper on the (KA)2 initiative, Benjamins *et al.* (1998) mainly focus on the latter issue. However, it seems to us that a careful analysis of all the issues associated with collaborative knowledge modelling is required, in order to manage the risks associated with such enterprises. In particular, we believe that a careful design of the underlying ontology is particularly important. For this reason, in contrast with the case of (KA)2, the design of the ontology (but not the ontology population process) is centralized in our approach.

In terms of the underlying architecture, the main difference between our approach and other approaches to adding semantic information to web pages is that we decouple the web pages from the knowledge model. This means that we do not directly annotate web pages but the collaboratively constructed knowledge base is held centrally in a server. As a result, changes made to either the ontology or the knowledge base are immediately available to the users. Moreover, decoupling knowledge models from documents means that the relation does not need to be one-to-one. For instance, we have seen how the PlanetOnto knowledge base extends the underlying Planet ontology by providing heuristics for matching entities in the knowledge base to news items. Our approach allows other users to develop alternative knowledge bases (by adopting different heuristics or by subscribing to a different ontological viewpoint) and therefore providing a different set of semantic indexing mechanisms for the same archive.

Finally, decoupling knowledge models from documents also emphasizes that the latter are not the exclusive source of knowledge for the former. We see formalized knowledge servers as playing a similar role to audio, video and "vanilla web" servers. Each type of server plays a distinct role and provides a distinct set of services. A key to successful knowledge management is to integrate these different media to provide the appropriate services in the relevant scenarios. We refer to the collection of these services as *knowledge media.*

8. Conclusions

In this paper, we have described an ontology-centred approach to knowledge management. The approach and the underlying technology have attracted considerable interest from both governmental and industrial organizations and we have currently several ongoing projects, which are testing this approach in a number of domains, e.g. to support students' help desks, to manage best practice in the aerospace industry and to provide novel forms of educational contents develivery. In this paper, we have tried to highlight the key issues and we have emphasized that the success of these enterprises crucially depends on the successful management of a number of user-, task- and model-centred issues. Our experience to date suggests that our approach appears to provide both the technology and the methodological framework required to minimize risk and ensure the success of knowledge management projects.

The KAW-99 reviewers provided many insightful and stimulating comments on an earlier version of this paper. Silvana Quaglini contributed to the specification of the ontology for medical guidelines; Oscar Corcho Garcia produced the current versions of the ontologies and knowledge bases for the PlanetOnto application; Marek Hatala configured the PatMan repository shown in Figures 17 and 18. Special thanks to Harriett Hansell, who agreed to star in Figure 1. This research was partially supported by the CEC-funded Enrich (P29015) and PatMan (P4017) projects. Enrich is part of the ESPRIT programme on IT for Learning and Training in Industry; PatMan is part of the Healthcare Sector of the Telematics Application Programme.

References

BENJAMINS, R., FENSEL, D. & GOMEZ PEREZ, A. (1998). Knowledge management through ontologies. In U. REIMER, Ed. *Proceedings of the 2nd International Conference on Practical Aspects of Knowledge Management*, Basel, Switzerland, 29–30 October.

BUCKINGHAM SHUM, S., MOTTA, E. & DOMINGUE, J. (1999). Representing scholarly claims in internet digital libraries: a knowledge modelling approach. In S. ABITEBOUL & A.-M. VERCOUSTRE, Eds. *Proceedings of ECDL'99: 3rd European Conference on Research and Advanced Technology for Digital Libraries*. Paris, France, 22–24 September, Lecture Notes in Computer Science. Berlin: Springer-Verlag. Available at: http://kmi.open.ac.uk/projects/scholonto/.

DOMINGUE, J. (1998). Tadzebao and WebOnto: discussing, browsing, and editing ontologies on the web. In B. GAINES & M. MUSEN, Eds. *Proceedings of the 11th Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, 18–23 April. Available on-line at http://kmi.open.ac.uk/people/domingue/banff98-paper/domingue.html.

DOMINGUE, J. B. & MOTTA, E. (1999). A knowledge-based news server supporting ontology-driven story enrichment and knowledge retrieval. In D. FENSEL & R. STUDER Eds. *Proceedings of the 11th European Workshop on Knowledge Acquistion, Modelling, and Management (EKAW '99)*, Lecture Notes in Artificial Intelligence, vol. 621. Berlin: Springer-Verlag.

DOMINGUE, J. & SCOTT, P. (1998). KMi planet: putting the knowledge back into media. In M. EISENSTADT, & T. VINCENT, Eds. *The Knowledge Web: Learning and Collaborating on the Net*, pp. 173–184. Kogan Press, London.

DUINEVELD, A. J., STOTER, R., WEIDEN, M. R., KENEPA, B. & BENJAMINS, V. R. (1999). Wondertools? A comparative study of ontological engineering tools. *12th Workshop on Knowledge Acquisition, Modeling and Management*. Banff, Alberta, Canada, 16–21 October.

EISENSTADT, M., DOMINGUE, J., RAJAN, T. & MOTTA, E. (1990). Visual knowledge engineering. *IEEE Transactions on Software Engineering*, **16**, 1164–1177.

ERIKSSON, H., PUERTA, A. R. & MUSEN, M. A. (1994). Generation of knowledge-acquisition tools from domain ontologies. *International Journal of Human Computer Studies*, **41,** 425–453.

ERIKSSON, H., FERGERSON, R. SHAHAR, Y. & MUSEN, M. A. (1999). Automatic generation of ontology editors. *Proceedings of the 12th Banff Knowledge Acquisition Workshop*. Banff, Alberta, Canada, 16–22 October.

FENSEL, D. DECKER, S., ERDMANN, M. & STUDER, R. (1998). Ontobroker: the very high idea. In D. J. COOK (Ed) *Proceedings of the 11th Annual Florida Artificial Intelligence Research Symposium (FLAIRS-98)*. Sanibel Island, Florida, USA, AAAI Press Menlo Park, California.

FUCHS, N. E. (1992). Specifications are (preferably) executable. *Software Engineering Journal*, **7,** 323–334.

GIRGENSOHN, A., ZIMMERMANN, B., LEE, A., BURNS, B. & ATWOOD, M. E. (1995). Dynamic forms: an enhanced interaction abstraction based on forms. *Fifth International Conference on Human Computer Interaction* (Interact '95) June 25–29, 1995, pp. 362–367.

GRUBER, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, **5,** 199–220.

HEFLIN, J., HENDLER, J. & LUKE, S. (1998). Reading between the lines: using SHOE to discover implicit knowledge from the web. *AAAI-98 Workshop on AI and Information Integration*. Available on-line at http://www.cs.umd.edu/projects/plus/SHOE/shoe-aaai98.ps.

VAN HEIJST, G. (1995). *The role of ontologies in knowledge engineering*. Ph.D. Thesis, University of Amsterdam.

HC-ReMa (1997). Health-care resource management. Telematics Applications Project HC 3103. Project Description available at http://aim.unipv.it/projects/hcrema/.

HPKB (1997). High performance knowledge bases. Darpa Project. Project Description available from http://www.teknowledge.com:80/HPKB/.

KRULWICH, B. & BURKEY, C. (1997). The Infofinder agent: learning user interests through Heuristic phrase extraction. *IEEE Expert Intelligent Systems and their Applications*, **12,** 22–27.

LENAT, D. B. & GUHA, R. V. (1990). *Building Large Knowledge-Based Systems*: *Representation and Inference in the Cyc Project*. Reading, MA: Addison-Wesley.

LIEBERMAN, H. (1995). Letizia: an agent that assists web browsing. *International Joint Conference on Artificial Intelligence*, *IJCAI'95*. Montreal, August.

MACGREGOR, R. (1991). Using a description classifier to enhance deductive inference. *Proceedings of the 7th IEEE Conference on AI Applications*. Miami, FL, February.

MASTERTON, S. (1998). The virtual participant: a tutor's assistant for electronic conferencing. In M. EISENSTADT & T. VINCENT, Eds. *The Knowledge Web*. Kogan Press, London.

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

MUSEN, M. A. (1989). *Automated Generation of Model-Based Knowledge Acquisition Tools*, Research Notes in Artificial Intelligence. London: Pitman.

O'LEARY, D. E. (1998). Knowledge management systems: converting and connecting. *IEEE Intelligent Systems*, **13,** 30–33.

PATMAN (1998). Patient management workflow systems. Telematics Applications Project HC 4017. Project Description available at http://aim.unipv.it/projects/patman/.

RDF (1999). http://www.w3.org/RDF/.

RIVA, A. & RAMONI, M. (1996). LispWeb: a specialized HTTP server for distributed AI applications. *Computer Networks and ISDN Systems*, **28,** 953–961. (also available at http://kmi.open.ac.uk/~marco/papers/www96/www96.html).

SELVIN, A. (1999). Supporting collaborative analysis and design with hypertext functionality. *Journal of Digital Information*, **1**. Available at: http://jodi.ecs.soton.ac.uk/Articles/v01/i04/Selvin/.

SHIPMAN, F. M. & MARSHALL, C. C. (1999). Formality considered harmful: experiences, emerging themes, and directions on the use of formal representations in interactive systems. *Computer Supported Cooperative Work* **8,** 333–352.

SOWA, J. F. (1995). Top-level ontological categories. *International Journal on Human-Computer Studies*, **43**, 669–685.

STUTT, A. & MOTTA, E. (1998). Knowledge modelling: an organic technology for the knowledge age. In M. EISENSTADT & T. VINCENT, Eds. *The Knowledge Web*. Kogan Press, London.

SUMNER, T. & BUCKINGHAM SHUM, S. (1998). From documents to discourse: shifting conceptions of scholarly publishing. *Proceedings of CHI 98: Human Factors in Computing Systems*, Los Angeles, CA, pp. 95–102. New York: ACM Press. Available at: http://kmi.open.ac.*uk*/techreports/papers/kmi-tr-50.pdf.

SUMNER, T., DOMINGUE, J., ZDRAHAL, Z., HATALA, M. MILLICAN, A., MURRAY, J., HINKELMANN, K., BERNARDI, A., WEISS, S. & TRAPHONER, R. (1998). Enriching representations of work to support organizational learning. In *Proceedings of the Interdisciplinary Workshop on Building, Maintaining, and Using Organizational Memories (OM-98)*. *13th European Conference on Artificial Intelligence (ECAI-98)*, Brighton, UK, 23–28 August. Available at: http://kmi.open.ac.uk/projects/enrich/enrich-oms98-paper.html.

TANIMOTO, S. (1990). VIVA: a visual language for image processing. *Journal of Visual Languages and Computing* **2,** 127–139.

USCHOLD, M., CLARK, P., HEALY, M., WILLIAMSON, W. & WOODS, S. (1998). An experiment in ontology reuse. In B. GAINES & M. MUSEN, Eds. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based System Workshop (KAW'96)*. Banff, Canada, April 18–23.

USCHOLD, M. & GRUNINGER, M. (1996). Ontologies: principles, methods and applications. *Knowledge Engineering Review*, **11,** 93–136.

VAN DER VET, P. E. & MARS, N. J. I. (1998). Bottom-up construction of ontologies. *IEEE Transactions on Knowledge and Data Engineering*, **10,** 513–526.
WEYHRAUCH, R. W. (1980). Prolegomena to a theory of mechanized formal reasoning, *Artificial Intelligence*, **13,** 133–170.
WIDEMAN, H. H. & OWSTON, R. D. (1993). Knowledge base construction as a pedagogical activity. *Journal of Educational Computing Research*, **9,** 165–196.
XML (1999). http://www.w3.org/XML/.

Paper accepted for publication by Editor, Dr B. Gaines