The Infolayer – A Simple Knowledge Management System Put to Use in Academia

Stefan Haustein[1], Katharina Morik[1], Jörg Pleumann[2]
Univ. Dortmund, Computer Science VIII[1], X[2]
{haustein, morik}@ls8.cs.uni-dortmund.de
pleumann@ls10.cs.uni-dortmund.de

Abstract: In spite of the growing number of KMS emanating from academia, only few of these systems seem to be actively used by their creators. We can only speculate about the reasons, but a lack of usability and support for real-life business processes is likely to be one of them. This paper presents the Infolayer, a KMS developed at the Artificial Intelligence unit at Dortmund university. Originally developed as an integrated information infrastructure for human users and software agents within a specific project, it has grown into a general KMS solution. Where other systems are built around complex reasoning capabilities based on description logics, the Infolayer's focus is ease of use, customizability and integration of industry standards like UML. The system is being actively used to drive the intranet/internet presence of our unit as well as a number of other projects.

1 Introduction

Most Knowledge Management systems (KMS) are technically developed at one site and applied at another site. This is particularly true for academic developments. Are they so hard to implement and is it so cumbersome to fill in content that system developers don't use their own tools? Or aren't there any business processes to be supported at academic institutions? This paper presents the Infolayer, a KMS developed at the Artificial Intelligence unit at Dortmund university. The Infolayer has been developed for integrated information supply to human users and software agents via the internet within the EC-funded COMRIS project [6]. All design decisions have been made favouring easy use by software agents and at the same time by users without any computer scientific background.

Once the system existed, we started to apply it for purposes outside the COMRIS scope, too. First, we used it within research projects. We noticed that the Infolayer is well suited for situations in which there is few or no time available for a knowledge management task. The system is already operational without long preparation, even with an incomplete ontology and content. Hence, we started to use it for our own purposes at the university unit. We realized that, in fact, there are many web services at our unit which ask for immediate attention. We became aware of the many business processes that we actually handled.

Applying the Infolayer to our own knowledge management turned out to very beneficial.

The rest of the paper is organized as follows: In section 2 we present the concepts underlying the Infolayer and give a brief overview of its implementation. Then we describe the business processes at an academic site like our AI unit and show how the Infolayer is used to support these business processes as a KMS.

2 The Infolayer

The Infolayer was originally developed to provide the information infrastructure for an agent-based conference support system. The system collected information on conference topics and participants and helped to organize meetings based on personal intererests. The most important functional requirement was that the infrastructure is accessible by both, human users and software agents. This had to be accomplished without adding redundant information into the system's database, since replicating information for these two worlds usually require duplicate effort.

Further requirements, some of which have a slighly non-functional character, were:

– *Simple Navigation:* The relevant content of the system should be accessed intuitively via hyperlinks, following a browsing paradigm, avoiding explicit queries as the only means to retrieve the systems's content.

– *No Client Side Installation:* The content of the system should be accessible with any regular web browser inside and outside the local network, not requiring any additional installation effort on the client side.

– *Legacy Data Integration:* Different data sources like BibTeX or SQL databases should be integrated seamlessly, without replicating the content.

– *Separation of Content and Presentation:* To allow simple customization of the system's appearance (for example when applying it to different conferences), a separation of these aspects was desired.

– *Simple System Setup:* Initial system setup should be easy, and the ontology definition should not require an extensive background in theoretical computer science.

– *Simple Maintenance:* The conference organizers should be enabled to maintain the content without needing an expert or knowledge engineer. The system should guide the user with input forms and be able to cope with incomplete data.

To meet these goals, we decided to use UML tools to define the ontology. Thus, a basic background in computer science is sufficient for defining the ontology; special skills in description logics are not required.

The meta-information of the ontology is used to generate forms showing only the relevant content. Thus, data input becomes so easy that "everybody" is able to perform content maintenance tasks. The same holds for finding information because of the web-browsing navigation style.

A short overview of the architecture and implementation shows how we fulfill the requirements.

2.1 Architecture

We designed the Infolayer system as an ontology-based knowledge management that directly operates on an UML model [1] as ontology definition. The reasons for choosing UML were mainly usability considerations. First, UML is a subject of education in computer science and related engineering disciplines, so the language is wide-spread and well-known. Together with wide-spread use comes tool support. UML is supported by a large number of sophisticated graphical tools that can be used for graphical ontology modeling. Second, in UML the (possible) properties of an object are defined in the scope of a class. Thus, all the attributes and possible associations that make sense for an instance can be determined from the class, allowing the generation of adequate input forms based on the ontology. The strong typing schema of UML allows limiting the fields shown to those relevant for a class. The input options for fields can be tailored according to the field type, for example by displaying adequate instance lists when editing associations. Associations are bidirectional in UML, ensuring link consistency automatically. Third, UML is accompanied by the Object Constraint Language (OCL), which allows annotating the class model with formal constraints and dependencies that need to be fulfilled by all instances. These constraints can be exploited in the user interface of the KMS, for example by limiting the possible values for an attribute or association beyond the UML typing constraints.

Figure 1 depicts the fundamental dataflows in an Infolayer-based system. The Infolayer itself is able to read its UML-based ontology directly from an XML Metadata Interchange (XMI) [3] file, as generated by contemporary UML tools. It stores the system's content using an XML-based persistence mechanism. A seamless integration of existing data sources such as SQL or BibTeX databases, is possible by additionally annotating the concepts in the UML diagram with meta-information about the location of the data source. The data is not replicated in the Infolayer system but accessed dynamically to avoid redundancy.

The system is able to generate HTML pages and a hyperlink navigation structure corresponding to the ontology, allowing both, convenient browsing-style and query form based access, to the stored information. For software agents,

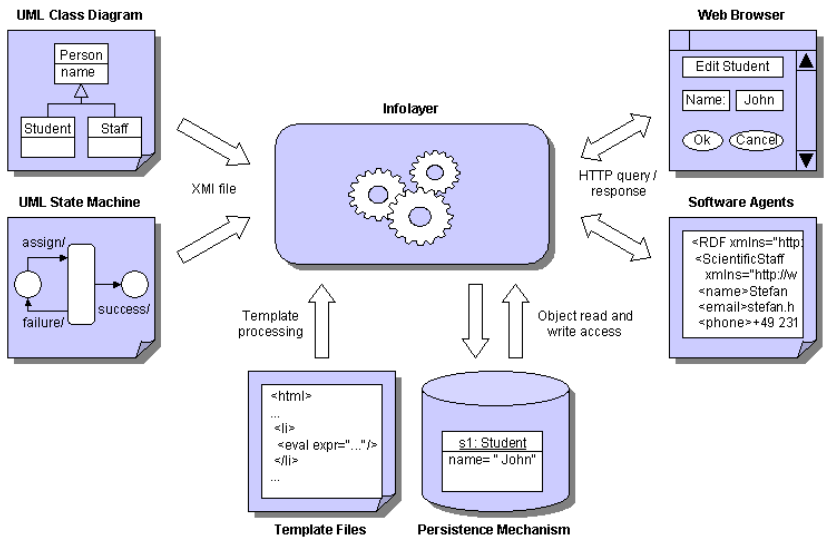


Figure 1: Overview of the Infolayer

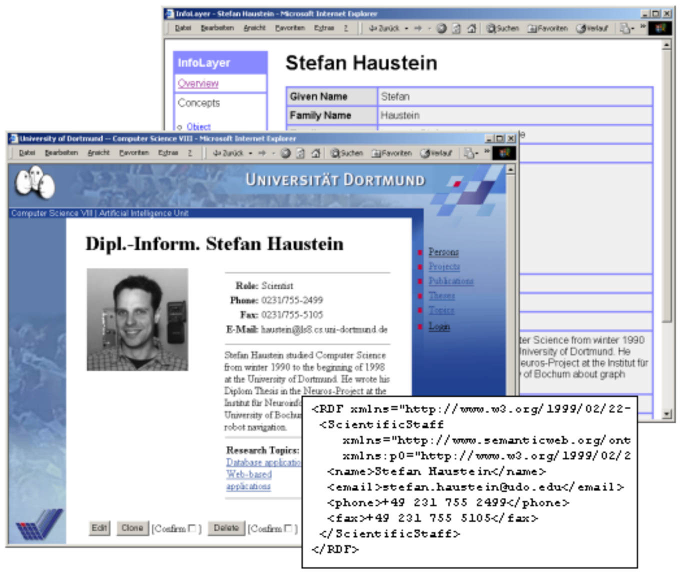
XML code corresponding to the particular needs of the agents (e.g. RDF) is generated using an XML template mechanism. The XML template mechanism also allows to customize the generated HTML code, providing a clean separation of content and layout. Figure 2 shows an example user human user interface with and without templates and an RDF version of the same content.

2.2 Implementation

The Infolayer is implemented in Java. At the heart of the system lies an implementation of selected portions of the UML metamodel, including classes, objects, state machines and OCL. An XMI loader allows to feed the model into the system.

The default persistence mechanism is based on XML files. Another one employs the Java Database Connectivity (JDBC) to integrate existing relational database. As mentioned, it is even possible to have the Infolayer operate on a BibTeX file.

All these components contribute to the system's back-end, which is still largely application-independent. Atop this back-end lies an application front-end based on a servlet. This servlet uses a template mechanism to generate HTML and RDF output based on the model information. Further template sets could be used to address mobile phones using the Wireless Markup Language (WML) or a limited subset of HTML.


InfoLayer - Stefan Haustein - Microsoft Internet Explorer
InfoLayer
Overview
Concepts
Object
University of Dortmund -- Computer Science VIII - Microsoft Internet Explorer
Universität Dortmund
Computer Science VIII | Artificial Intelligence Unit
Persons
Projects
Publications
Theses
Topics
Login
Edit
Clone
[Confirm ☐]
Delete
[Confirm ☐]

Stefan Haustein

Given Name	Stefan
Family Name	Haustein

Dipl.-Inform. Stefan Haustein

Role: Scientist
Phone: 0231/755-2499
Fax: 0231/755-5105
E-Mail: haustein@ls8.cs.uni-dortmund.de

Stefan Haustein studied Computer Science from winter 1990 to the beginning of 1998 at the University of Dortmund. He wrote his Diplom Thesis in the Neuros-Project at the Institut für Neuroinfo University of Bochu robot navigation.

Research Topics:
Database applicatio
Web-based applications

```
<RDF xmlns="http://www.w3.org/1999/02/22-
 <ScientificStaff
    xmlns="http://www.semanticweb.org/ont
    xmlns:p0="http://www.w3.org/1999/02/2
  <name>Stefan Haustein</name>
  <email>stefan.haustein@udo.edu</email>
  <phone>+49 231 755 2499</phone>
  <fax>+49 231 755 5105</fax>
 </ScientificStaff>
</RDF>
```

Figure 2: User interface with and without templates, and RDF generation

In addition the the servlet, other front-ends exist. A command-line client can be used to access the system from a shell or using a telnet protocol. A Swing-based graphical client that adapts its user interface to the model has also been implemented, but is currently not maintained, because the focus lies on the web client. Finally, a client implementing the Simple Object Access Protocol (SOAP) allows to remotely access the Infolayer from third-party applications.

3 Applying the System in Academia

The acknowledged work of a university unit is teaching (lectures, seminars, exercises, written and oral examinations) and research (projects, publications). This view hides a lot of administrative tasks, many of which are performed in using the internet today. The web is used by students in order to read the announcement of lectures or seminars among which they can choose, download the teaching material (presentations, scripts, papers, software), the exercises and – a week later – their solutions, become acquainted with possible master/diploma thesis topics, and find current research papers. The web is used by the staff of the unit in order to present themselves together with their projects, publications, and software. Professors use the web in addition for publishing the teaching material. This passive/active use of the web still is the standard at universities. It is already beneficial if this use if supported by more than a linked collection

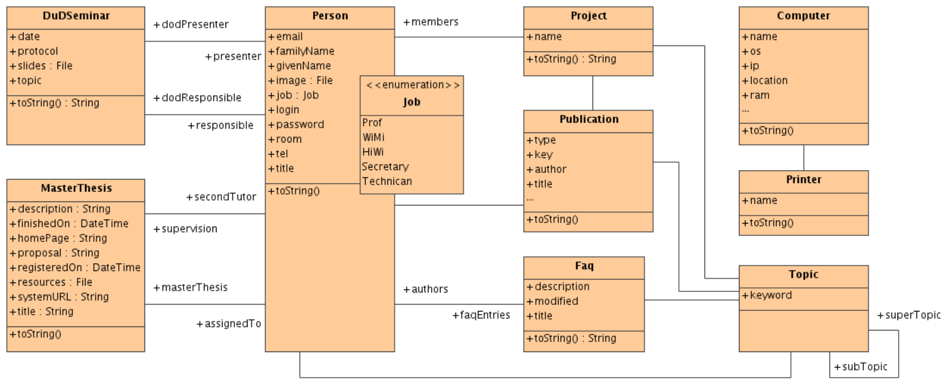


Figure 3: Unit Ontology UML Model (simplified)

of HTML pages. In particular, we want to enter the information once so that changes are to be made only at one place.

However, there are many more tasks that can be supported. Scientists may profit from services for their daily work, and the members of the unit – including the non-scientific employees – may organize themselves using the web. In particular, the *management of the publications* is a crucial part. All scientists have their BibTeX files for citations and a list of their own publications. We have a BibTeX-database where additional optional fields indicate the authors belonging to the unit, the project to which a paper is associated, and the ranking of the publication concerning their impact factor, allowing the automated generation of the publication part of the home pages of unit members. A corresponding database stores the postscript and pdf-files. An artificial agent generates the publication part of the home pages of unit members from the database. Moreover, it generates various reports the unit has to produce for the department, the university, the projects. This eases the task of the secretary.

The publications for seminars are also stored in the two databases. They are a good collection for citations in publications in the respective field. Hence, the task of gathering the citations is eased for all scientists. *Organisation of the unit* includes administrative procedures, frequently asked questions concerning the computers and printers, the arrangement of our weekly meeting where one talk is given and another person takes care of the breakfast, and the time schedule of student employees. The ontology underlying all these business processes is shown in Figure 3.

Since all unit members have access to the Infolayer, they can not only view which time slots are available or how to handle a particular LaTex problem but also book a time slot for a meeting or student job and enter hints for others.

Moreover, they can do so at home, without coming to the university at all. As a result, the whole unit including the secretary is using the system for every-day knowledge management tasks. The system was already operational once we agreed on a first, rudimentary ontology. It was then extended gradually to cover more aspects of the unit's business processes, without invalidating existing content.

As soon as the university allows it, forms for the registration of master theses along with asking the permission for an assistant to supervise it could be handled using the unit's web site. Here, the XML Template mechanism would be used to build XSLT formatting objects (FO), generating a PDF file. Moreover, a UML State Machine can be used to model the "life cycle" of a thesis, automatically generating buttons for state transitions corresponding to the workflow involved in supervising a masters thesis.

4 Related Work

Ontology Based KMS: Other Ontology based systems such as Protégé [10] or Ontobroker [2] are based on Description Logics or Frame Logics, providing complex built-in inference services. While the logical foundations of those systems certainly have advantages, they also require additional skills, generating a high entry barrier when using those systems. Also, the requirement that the content of those systems must be logically sound in a formal way tends to cause increasing maintenance costs [4].

Content Management Systems: Content Management Systems (CMS) such as Zope [8] or Hyperwave [9] are designed for handling and publishing large numbers of documents. They support different media types and a separation of content and layout, usually based on templates.

The problem with using a CMS for a department web site is that CMS are document-centric. They are focused on document-oriented meta data such as the author or different states of approval. Usually, their primary document structure is organized hierarchically, and support for semantically rich structures such as typed associations – required for machine readable content – is limited.

Software Agents and OntoWeb: The XML generation capabilities of the Infolayer system allow an easy integeration with Software Agents that are capable of reading RDF, such as the Syndication Systems of the OntoWeb project [11].

RDF Annotation Tools: RDF Annotation Tools such as Ont-O-Mat [5] can be used for semantic annotation of exisiting web pages, but in many cases the annotations will duplicate information already present on the web pages in a machine readable manner.

5 Conclusion

While the Infolayer trades in the ease of use and the evolution of ontology and services for complex built-in inference services that are available in systems based on Description Logics, our experience with the system suggests that this trade is justified for many knowledge management applications. Other web sites utilizing the Information Layer are a database for Java-enabled small devices like cell phones and personal digital assistants[1], a Germany-wide project that develops multimedia teaching material for software engineering education (MuSofT), the MiningMart project, and the training section of KDnet.org. The MuSofT and KDnet installation make use of a content management feature the system provides: Learning material can be uploaded into the system from a Web browser. To allow efficient retrieval of material in MuSofT, LOM [7]-conforming metadata is provided using the system's ontology capabilities.

References

1. S. Cranefield and M. Purvis. Uml as an ontology modelling language. In *Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99*, 1999.
2. Stefan Decker, Michael Erdmann, Dieter Fensel, and Rudi Studer. Ontobroker: Ontology based access to distributed and semi-structured unformation. In *DS-8: Semantic Issues in Multimedia Systems*, pages 351–369. Kluwer Academic Publisher, 1999.
3. Object Management Group. XML Metadata Interchange (XMI) Specification 1.2, 2002. http://www.omg.org/cgi-bin/doc?formal/2002-01-01.
4. U. Hahn and S. Schulz. Towards a broad-coverage biomedical ontology based on description logics. In *Pacific Symposium on Biocomputing (PSB 2003)*, 2003.
5. S. Handschuh, S. Staab, and A. Maedche. CREAM - creating relational metadata with a component-based, ontology-driven annotation framework. In *ACM K-CAP*, Vancouver, October 2001.
6. Stefan Haustein. Information environments for software agents. In Wolfram Burgard, Thomas Christaller, and Armin B. Cremers, editors, *KI-99: Advances in Artificial Intelligence*, volume 1701 of *LNAI*, pages 295 – 298, Bonn, Germany, September 1999. Springer Verlag.
7. IEEE Learning Technology Standards Committee. Final Draft of the IEEE Standard for Learning Objects and Metadata. http://ltsc.ieee.org/wg12, 2002.
8. Amos Latteier and Michel Pelletier. *The Zope Book*. Que, 2001.
9. Hermann Maurer. *Hyper-G (now Hyperwave): The Next Generation Web Solution.* Longman Group United Kingdom, 1996.
10. N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. Creating semantic web contents with protege-2000. *IEEE Intelligent Systems*, 2:60–71, 2001.
11. Peter Spyns, Daniel Oberle, Raphael Volz, Jijuan Zheng, Mustafa Jarrar, York Sure, Rudi Studer, and Robert Meersman. Ontoweb – a semantic web community portal. In *PAKM*, 2002.

[1] http://www.kobjects.org/devicedb