In Proceedings of the EdMedia 2001, June 2001

MediBook: Combining Semantic Networks with Metadata for Learning Resources to Build a Web Based Learning System

Achim Steinacker[1], Andreas Faatz[1], Cornelia Seeberg[1,2], Ivica Rimac[1], Stefan Hörmann[1], Abdulmotaleb El Saddik[1] und Ralf Steinmetz[1,2]

{steinacker, seeberg, rimac, hoermann, faatz, el-saddik, steinmetz}@kom.tu-darmstadt.de

1. Introduction

This paper gives an overview of MediBook, a joint project of the Medical Department of University Gießen (Germany) and the Department of Electrical Engineering and Information Technology at Darmstadt University of Technology (Germany) funded by the Hessian Ministry for Science and Arts (HMWK). The goal of the project is the enhancement of education in Medical Schools by means of a flexible system. The flexibility of the system aims at self-studying, independently from space and time. MediBook is a system which provides tools for storing, managing and especially locating and combining learning resources. It offers means for both describing single sources (thereby putting them into a context) and combining single resources to courses. Thus MediBook represents a knowledge base with efficient access and tools to generate arbitrary coherent courses from single information units.

2. Knowledge Base

The basis of MediBook is a formal representation of the medical domain. The formal representation consists of the important medical concepts -- e.g. colon, kidney, bacterium -- which are associated with each other by semantic relations -- e.g. colon 'is_part_of' digestive system. For this formal representation we introduced the notion ConceptSpace. Information units -- in MediBook the notion MediaBrick is used -- are associated with concepts, whereby each MediaBrick is described by metadata. Furthermore MediaBricks are associated with each other by rhetorical-didactic relations in such a way as to put them into a context -- e.g. MediaBrick A 'deepens' MediaBrick B -- thus building the MediaBrickSpace. Since in MediBook we are dealing with educational content we use the Learning Objects Metadata (LOM) [1] for the description of the information units. Using this widespread IEEE proposal for a standardized description of "learning objects" other systems are able to access MediBook resources. MediBook is designed to deal with resources of different formats such as plain text, images, graphics, video, audio or animations which may contain information, theses, motivations, exercises, etc. Existing resources shall be integrated in order to support reuse of multimedia-elements which creation often is very expensive. The basic approach to describe MediaBricks on different levels -- LOM, rhetoric-didactic relations, associations with concepts -- is not limited to the medical domain. It can easily be applied to other knowledge domains by defining the concepts in the ConceptSpace and if necessary adaption of the set of semantic relations.

3. Scenario

In MediBook we define three roles which are supported:

- *Author*: The author is a physician and an expert in the medical domain to be modeled. The problem of building the knowledge base is divided into three sub tasks which are: generating the ConceptSpace, integrating the contents (MediaBricks in the MediaBrickSpace), and associating the MediaBricks with the corresponding concepts.
- *Teacher*: The teacher (physician) selects the MediaBricks which are relevant for his course and puts them into the appropriate order and structure. To achieve better coherence the teacher can create transitions between the single information units. In the process of navigating through the knowledge base the teacher is assisted by the system in such a way that the system is showing the relevant details adaptively according to the teacher's user profile.
- *Learner*: The learner can either follow the way proposed in a stored course or he can navigate through the knowledge base quite like a teacher can, though in a read-only modus. The view on the knowledge base is adapted by means of individual user profiles.

4. Outlook

Currently learners in the context of MediBook are students of Medical Schools but the system can easily be used for advanced education of physicians or as an information system e.g. for patients. Therefore only the knowledge base has to be extended with respect to content -- an extension of the system is not necessary.
In a further step of this project it would make sense to give the learner the possibility to annotate learning material, to define bookmarks or to work through interactive tests. Furthermore the automatic generation of lessons and courses by the system is desirable, particularly regarding a heterogeneous group of users. This necessitates extended user profiles and a rule-based system module which compares the profile entries with the MediaBricks metadata in order to select the relevant MediaBricks and present them as a course including a table of contents.
While the LOM-Editor [10] and the ConceptSpace-Editor described later in this paper are already implemented and the authors are working with the tools, the presentation generator is still work in progress.

5. Architecture

MediBook integrates the access to different information systems, managed at different locations. This scenario requires a broker oriented middleware, to combine the metadata descriptions of the physical resources with the formal representation of the specific domain. With this architecture users have a single point of access not only to different types of informations, but also to information which is managed and stored at different places. Figure 1 gives an overview of the MediBook architecture. Retrieval, tagging and composing requests from all clients are passed to the broker middleware which is responsible to select and query the appropriate database.

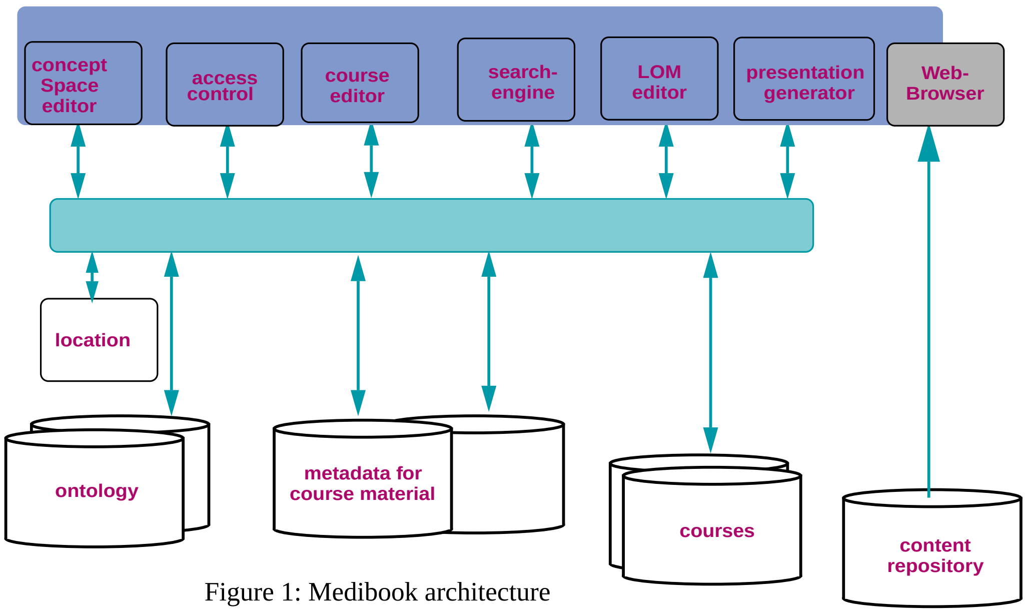

Figure 1: Medibook architecture

Access to the databases is realized by a query based architecture. In opposite to index based techniques, where full indices of the data managed by the different databases are exchanged, a query based system implies a more transparent global data management. The Open Archives Group [2] has specified a simple harvesting protocol to access arbitrary databases. The problem is, that this protocol is limited to a restricted set of metadata. In MediBook we are using the Learning Objects Metadata (LOM) from the IEEE to describe the learning resources and the courses. According to this set of metadata, we have defined a query protocol based on the LOM approach. To integrate a database which manages LOM descriptions the database is registered by sending a message to a central content location service. This announcement includes the kind of metadata the database is responsible for. The content location service is contacted from all retrieval services inside Medibook. Examining the request from the clients, the content location service is choosing the appropriate databases for executing the query. The service collects all the results from the different databases and sends them to the requesting client. Currently we are using the relational database Oracle 8i and the XML database Tamino to store the metadata. Therefore the protocol must be capable of mapping the requests from the clients to an SQL- and also to a XQL query. Regarding the fact, that there are already solutions available to map

between XML data structures and relational schemes [3], our query protocol is XML-based. The mapping is realized with the freely available JAVA-based framework Castor [11]. Based on a LOM-XML-DTD published by the IMS Consortium we have developed a LOM Working Draft 4.0 compliant DTD to describe a search request. The clients, described in the next chapters, are using an object model representing the LOM approach.
In addition to the integration of different databases, the MediBook middleware also has to implement the connection between the metadata and the semantic network. The Ontology Inference Layer (OIL) [12] is an approach to standardize the description and exchange of ontologies. However we are using a proprietary approach, the Smalltalk Frame Kit, to combine the different information, as there is an integrated development environment at hand. An integration of the ISO-Standard Topic Maps is not planned.
The amount of information which currently is available through the Web has lead to the development of different metadata standards like Dublin Core [13] for general resources, LOM or MPEG7 [14] which is used to describe multimedia resources. A crucial point for a widespread use of metadata standards is the question how the user has to deal with this amount of information. The problem with metadata information like LOM is mainly the accurateness and the amount of time a user has to invest to describe a resource. Although many of the LOM elements can be generated automatically, there is still a significant number of elements, where the user has to decide about the entries. In the area of semantic networks the situation is even worse. Building an ontology does not only require knowledge about the domain. Regarding the available tools a user also has to be an expert in computer science and data structures to build an ontology.
In the following chapters we want to describe two client tools we have developed for Medibook and we want to show how we deal with these problems in every tool. The next section will give an introduction and introduce the ConceptSpace-Editor which is used to build the ontology. The last tool will be the presentation generator. In this section we will describe how an on-line available course can be presented out of reusable modules described with LOM.

6. The ConceptSpace-Editor

Throughout the following section we focus on how the medical knowledge to be represented in MediBook can be modeled with our editing tool, the ConceptSpace-Editor. We give a brief survey on how the ConceptSpace- Editor works and which requirements it meets in the MediBook project. We are concerned with the logical layer of the ConceptSpace-Editor, which was implemented using the Smalltalk Frame Kit (SFK) [7]. Knowledge Representation in our system MediBook is based on an ontology. An ontology is (formal) conceptualisation of a (knowledge) domain [4]. From our point of view modeling special domains of knowledge always has to fulfill certain tasks. In consequence ontological design as a first step of ontology engineering already has to investigate the later use of the ontology [5]. In our case the ontology is a networked structure which allows browsing and navigating through medical termini. Moreover MediaBricks are attached to the medical knowledge which originally is independent from the media. A logical and consistent ontological design requires types of concepts to reflect the entities within the knowledge domain (i.e. medicine), relation types to model the relations between concepts, and axioms [Staab/Maedche]. Axioms supervise the process of knowledge modeling logically, they come into play while building conceptual and relational instances. For example we use inverse relations, which are automatically triggered, when a relation is drawn between two concepts. Another example is the maintenance of hierarchical relations. We have to formalize rules, which guarantee the establishment of a relation like 'diarrhea is caused by bacterium x' whenever we draw a relation 'bacterium x causes diarrhea'. If we order concepts hierarchically, we want to avoid relations like 'the skeleton is a part of the bones', if we already have the relation 'the bones are a functional part of the skeleton' [5].
The Smalltalk Frame Kit (SFK) is a high-level programming language, which enables us to create an ontological scheme, i.e. a data model for an ontology. Such a scheme contains types of concepts and relations and additionally axioms. It is filled with the actual domain specific concepts by a medical expert. Our approach tries to make this kind of work easier for the expert: he or she should focus on the actual concepts, their classification and relations, the 'weaving of a web' [7]. Formal logic, axiomatic definitions or implementation techniques are due to a software engineering process, which we keep apart from the knowledge engineering process. What we get is a division of labour: the ontology is designed by a software engineer, ontological contents are modeled graphically and intuitively by a medical expert, who relies on the SFK-procedures, which are invisible to him or her, but work as a solid background. Without any idea about implementation, the medical expert is able to use necessary axiomatic support. Of course the software engineer using the SFK and the expert have to keep in touch closely and to discuss, how the knowledge should be organized basically. In the MediBook project we elaborated four concept types: DiseaseOrSymptom, Cell, Substance, Aspects (other concepts). We minimized the number of concept types for the first version, because SFK,

embedded in the object-oriented Smalltalk environment, allows extensions of an ontological scheme rather than changes: we could re-import the instances already modeled by the expert into a more fine grained second version. This means that starting with a more abstract scheme may be followed by instantiating and evaluating it by the expert. This does not have to be followed by a loss of data when implementing a new model. The SFK works with and is based on frame classes. We describe them as semantic entities, which carry attributes called slots. The slots may be restricted concerning their range (defining which frame types are permitted for a slot) or cardinality (how many values might be attached to a slot), whereas other facets of the slots describe its relational properties: in our previous examples this means defining an inverse slot or adding acyclicity and the extinction of short cuts (direct relations to a superconcept if there are other superconcepts from the hierarchy in between). The SFK yields a rich archive of meth-

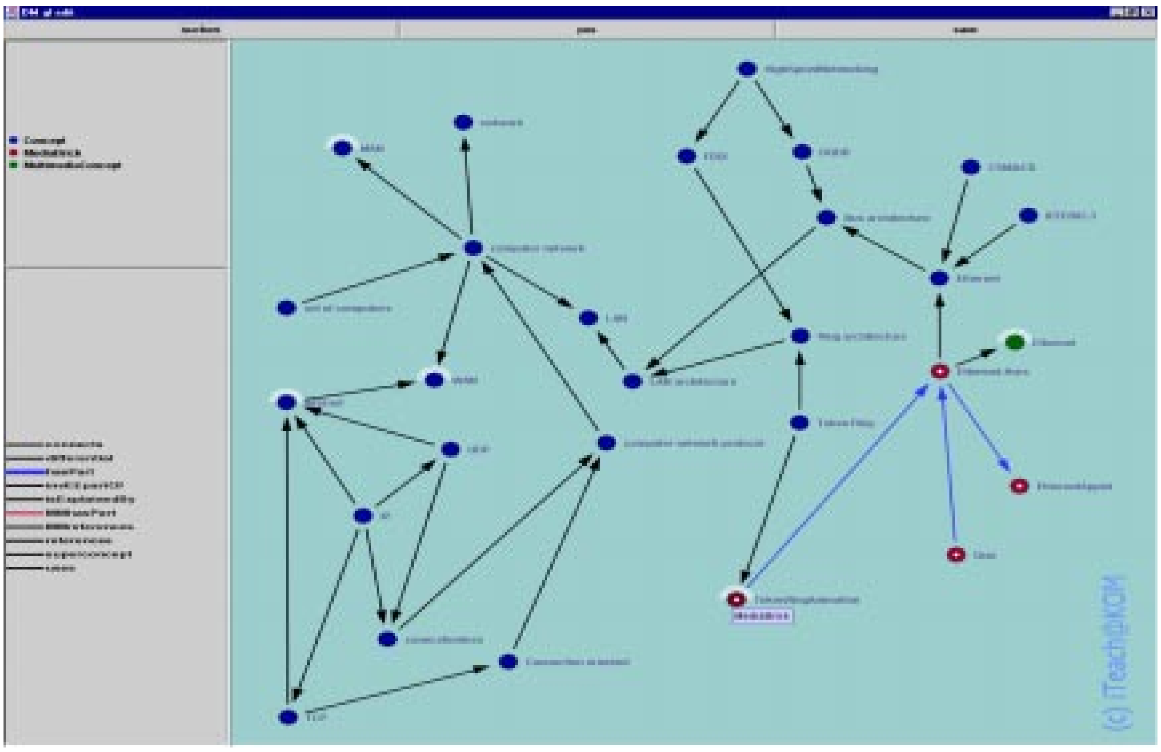

Figure 2: GUI of the ConceptSpace-Editor

ods, which compute additional frame instances and their slot values (or extinct or overwrite frame instances and their corresponding slot values). For example a transitive hull is computed in the background whenever a facet 'transitive' induces triggering new relations automatically. The features reflected by the relational property facets cover the systematic collection of axioms for ontologies described by [8]. An ontological scheme implemented with the SFK-support is a hierarchy of frame classes, which become interconnected by defining slots. This can be done either using SFK as a programming language (in our example: stating that is 'functional part of' is a 'hierarchy-relation-up-slot' in a model method before the classes from hierarchy is generated) or entering the classes from the hierarchy just created to assign 'acyclic' and 'no Short Cuts' to the slot corresponding to 'functional part of'. Inconsistencies like naming conflicts are identified by the system right after the installation of the class hierarchy as an ontological scheme.
The SFK continuously exports the state of the instanciation of a scheme. For this purpose it uses XML, which is the input for a Java client, which is able to visualize the export. On the other hand, all actions (creating a concept by selecting a node of one of the special concept types and naming it, deleting and renaming nodes, drawing or deleting a relation between two concepts) at the client side are sent via XML to the SFK. The SFK tests, if the actions are correct according to the scheme, if so the new state of the ontology is sent to the client, if not a warning is sent and a rollback takes place. This would be the case, if the skeleton would be indicated as a functional part of the bones after the relation 'the bones are a functional part of the skeleton' already had been created. The Java client is the medical

expert's tool, the SFK is the software developer's tool. Together with their XML-communication they form the ConceptSpace-Editor, which combines a visualization of arbitrary graphs with the logical checks necessary in ontology engineering.

7. Presentation Generator

The module named presentation generator generates presentations of the courses, which were produced with the course-editor (see Figure 1 for a detailed overview of the modules). The courses to be presented are composed of lessons, that are an enumeration with a fixed sequence of MediaBricks. Lessons are realized by a LOM object, that references all containing MediaBricks by their metadata objects with a *HasPart* relation. Vice-versa the referenced MediaBricks of a lesson references the lesson by *IsPartOf* relation. Both types of relations used to build lessons are part of the LOM specification. The tree-like structure of the courses, where leaves always contain MediaBricks, results in the possibility to refer to lessons as parts of lessons. This rule defines sublessons in lessons. Without this sublessons there would not be any possibility to compose MediaBricks to a complex structured learning resource like books or lectures. Figure 3a shows a part of a slightly abstracted structure of a course. It should explain the logical context of lessons and courses. The file-formats HTML and PDF are primarily suitable for the automatic generation of presentations of these courses. We assume a learning scenario with a learner working with the computer, but also another scenario, where the leaner uses a portable printed version of courses without any computer. From this, two in principle different requirements emerge for the presentations. That leads to two different solutions for the generation of presentations.
One of the advantages of HTML presentation is, that a learner can directly work through the course at the computer without any further software installation except a web browser. However the most important advantage with the generation of HTML pages is the probability to use continuous media, like sounds and videos, in the presentation. This shapes a multimedia presentation in the sense of [9]. In the presentation generation process the MediaBricks are composed to small units that correspond to HTML files. Working through the course and locating the relevant learning resources should be simplified by it. Figure 3b shows an example for the presentation of the course in HTML files.

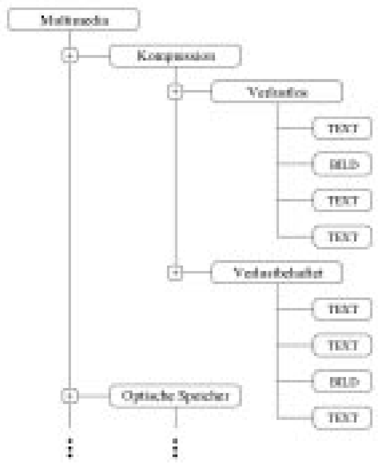

Figure 3a: Course structure

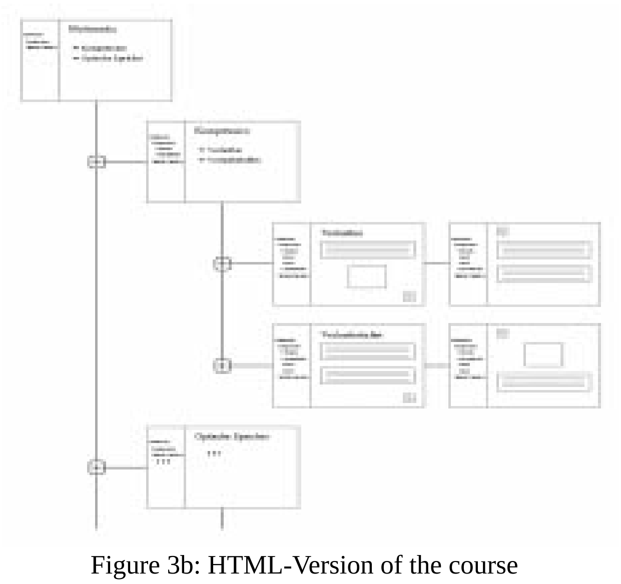

Figure 3b: HTML-Version of the course

The presentation of the MediaBricks assigned to a course as a PDF file is predestinated for print outs, but can also be viewed easily and comfortably at the computer. However the course representations as PDF files is mainly suitable for printing them on paper. For this reason, the presentation of the courses is restricted to MediaBricks with static content, like texts and pictures. Unlike HTML presentations in the presentation of the course as a PDF file the MediaBricks are not composed to small units. Rather the MediaBricks have to take a linear order, in which they are printed sequentially in the PDF file. The proper order of the MediaBricks results in a pre-order traversal of the course tree. However before a presentation is generated by the presentation generator a description of the course is generated.

This description can be stored as an XML file to speed up later generations of presentations. Together with the path within the course, that points to the lesson from which a presentation should be generated, and the XSL stylesheet that should be used for the generation of the presentation, the presentation can be generated automatically from this description. There is no difference in generating PDF presentations of courses when distributing them over the web or storing them on CD-ROM, but there is a slight difference in generating presentations of courses in HTML files because of the storage and access to the file. While all resources of HTML presentations, that are distributed over the web, may be accessed via web servers, all resources of a local presentation on a CD-ROM have to be stored on this CD-ROM. Therefore there have to be different HTML files with different links in the presentations depending of the way the courses are distribution or stored. Additionally to an HTML presentation of a course the structure of the course is displayed as a table of contents. This table of contents is presented by a JAVA Applet and should support the learners in the navigation through the course and give them the opportunity to get an overview of the course. The lesson actually shown is always highlighted in the table of contents to ensure that the learners do not lose track of the course by the well-known lost in hyperspace syndrome.

References

[1] Learning Objects Metadata, *http://ltsc.ieee.org/wg12*

[2] The Open Archives Group, *http://www.openarchives.org*

[3] XML Database Products, Ron Bourret, *http://www.rpbourret.com/xml/XMLDatabaseProds.htm*

[4] T.R. Gruber: *Towards principles for the design of ontologies used for knowledge sharing*. International Journal of Human-Computer Studies, 43, 1995

[5] Mike Uschold, Martin King: *The Enterprise Ontology.* The Knowledge Engineering Review Vol.13, Special Issue on Putting Ontologies to Use, 1998

[6] Aldo Gagnemi, Domenico Pisanelli, Geri Steve: *An overview of the ONIONS project: Applying ontologies to the integration of medical terminologies*. Elsevier's Journal on Data and Knowledge Engineering, 31, 1999

[7] L. Rostek, D. Fischer, W.Möhr: *Weaving A Web: Structure and Creation of an Object Network Representing an Electronic Reference Framework.* Electronic Publishing 6, 1994

[8] Steffen Staab, Alexander Mädche: *Ontology Engineering beyond the Modeling of Concepts and Axioms.* Proceedings of the Ontology Learning Workshop at the ECAI 2000, Berlin

[9] Steinmetz R., Nahrstedt K., *Multimedia: Computing Communications & Applications* 2nd edition, to be published, Prentice Hall, 1998

[10] iTeach-Lom-Editor: *http://www.multibook.de/lom*

[11] CASTOR: *http://castor.exolab.org*

[12] D. Fensel et al., OIL in a nutshell, *Proceedings of the European Knowledge Acquisition Conference (EKAW-2000),* Lecture Notes in Artificial Intelligence, LNAI, Springer-Verlag, October 2000.

[13] Dublin Core Metadata Initiative: http://dublincore.org

[14] Rob Koenen (ed.), MPEG-7: Context, Objectives and Technical Roadmap, V.12, *ISO/IEC JTCl/SC29/WGll/N2861*, Vancouver, July 1999