Course and Exercise Sequencing Using Metadata in Adaptive Hypermedia Learning Systems

STEPHAN FISCHER
Mobile VideoCommunication AG

In the last few years the (semi-) automatic sequencing of course material has become an important research issue, particularly the standardization of metadata for educational resources. Sequencing can help to generate hypermedia documents which, at their best, match the learner's needs. To perform (semi-) automatic course sequencing, a knowledge library as well as modular resources can be used. Both must be described by metadata.

First, metadata standards (IEEE Learning Objects Metadata, Instructional Management Systems Global Learning Consortium, Dublin Core) are analyzed with regard to course sequencing. As an application example, Multibook, an adaptive hypermedia system used to teach multimedia technology, is described. Multibook uses metadata to create course sequences semi-automatically. In this article we explain how a knowledge library can be used to create exercises automatically. We give an example of how courses can be sequenced in general by analyzing the creation of exercises. An evaluation of our system shows the advantages and drawbacks of the automatic course sequencing approach.

Categories and Subject Descriptors: H.5 [Information Systems]: Information Interfaces and Presentation
General Terms: Documentation, Management
Additional Key Words and Phrases: Adaptive hypermedia systems, hypermedia learning, knowledge engineering, sequencing of course material

1. INTRODUCTION
Course sequencing has become an important research issue in the last few years, especially the standardizing of learning metadata (Instructional Management Systems; Learning Technology Standards Committee (LTSC): http://ltsc.ieee.org/; IEEE P1484.6, homepage: http://ltsc.ieee.org/wg6/index.html). The goal of a sequencing approach is to generate a lesson for a
targeted group, e.g., students, that is tailored to the needs of that group. The contents of the lessons are built of hypermedia elements (texts, images, audio, video, animations) and stored in a modularized way.

The emerging standards must define a specification language and an environment for managing sessions in learning technology systems, e.g., computer-aided instruction, intelligent learning environments, or intelligent tutoring systems. In order to achieve this goal, the standards must define the specification language, conceptual models, semantics, and syntax, the control transfer mechanisms and their encodings (e.g., how learning sessions are controlled and conducted), the data transfer mechanisms and their encodings (e.g., how student assessments and lesson plans are exchanged), and an encoding method for storing and transferring session management “programs,” i.e., interactive lesson plans.

An interesting observation can be made by abstracting from the concrete goals of the standardization bodies: Much research is going on in the creation of lessons tailored to the specific user, particularly in adaptive hypermedia systems. And, on the other hand, sophisticated test environments have been described that enable the learner to explore an unknown field with a great degree of freedom (see, for example, ELM-ART [Specht and Weber 1996]). However, an integrated approach containing a lesson tailored to the user as well as interactive tests that can be explored in a self-defined way is still a challenging task. It should be mentioned that some programs do not intend to provide the old-fashioned type of learning document, where exercises follow an initial explanation of the topic. These approaches integrate exercises created manually with the necessary theory. The applications we are working with, however, use a hypermedia document to acquire the necessary skills, which can be tested afterwards.

In our approach, we offer courses and exercises that were created automatically. It can be observed that in many educational books exercises offer the user the possibility of testing his or her understanding of a complex topic. But in most cases these exercises are not in a very interactive form. And even the creation of the old-fashioned, noninteractive exercises is a very time-consuming task. It is our goal to derive very simple exercises from a knowledge base which can, very often, be part of an adaptive hypermedia system. Both the creation of exercises and the integration of background material can be done automatically. But it is not our goal to create an intelligent tutoring system, since the exercises we derive do not adapt to the user by exploiting some kind of rule-based system. Hence we see our approach as an initial step towards a powerful set of exercises that can be built both automatically and manually.

The remainder of the article is structured as follows: In order to develop an approach to course sequencing, more specifically to the creation of exercises, (upcoming) metadata standards are reviewed in Section 2. In Section 3, the architecture of Multibook is explained. Multibook is an adaptive hypermedia system developed in order to teach multimedia technology where our exercise framework has been integrated. In Section 4, the algorithms necessary for creating exercises automatically are described.
Section 5 reports on experiences with course and exercise sequencing. Section 6 summarizes related work. Section 7 concludes the paper and provides an outlook for future work.

2. (UPCOMING) METADATA STANDARDS FOR EDUCATIONAL RESOURCES

Before describing the core course and exercise sequencing using metadata, an outlook for learning environments has to be provided. We concentrate on IEEE’s Learning Technology Systems Architecture (LTSA) because IEEE’s model is generic enough to represent a whole variety of different learning systems from different domains. Figure 1 shows the model used by IEEE’s LTSA. Before describing the components that actually use metadata, we give a short explanation of the model’s components.

Course sequencing generally starts with the learner entity, which is an abstraction of a human learner. The learner entity receives the final multimedia presentation, while the learner’s behavior is observed. It also negotiates learning preferences with the teacher. In the next step, the teacher sends queries to the learning resources component to search for learning content that is appropriate for the learner entity. The queries specify search criteria based, in part, on learning preferences and assessment and performance information. The appropriate locators (e.g., lesson plans) are sent to the delivery process. The learning resources component stores “knowledge” as a resource for the learning experience, which may be searched by queries. The matching information is returned as catalog information, i.e., a set of content tags which can also be seen as “card catalog” entries. The locators are then extracted from the catalog information and used by the delivery process to retrieve learning content and deliver that content as a multimedia document to the learner.

The components to the right of the learner entity correspond to performance control. Performance is measured by the evaluation component and stored in the records database. The data in the records database can then be used by the coach when locating new content.

Fig. 1. IEEE’s Learning Technology Systems Architecture.
The advantage of IEEE’s LTSA is its generality. The model applies to a wide variety of application scenarios, and hence is a well-chosen framework for course (and exercise) sequencing.

2.1 General Approaches to Course Sequencing

Course sequencing that results in a multimedia presentation can be done using a knowledge library and a delivery component. The following two approaches can be distinguished versus the knowledge library:

- the use of a standardized general-purpose ontology (an ontology is a set of terms and formal definitions);

- the use of an external component that manages the ontology.

A subgroup of LTSC currently specifies the semantics of a general-purpose, upper-level ontology. This activity will be limited to the upper level, which provides definitions for general-purpose terms and a structure for compliant lower-level domain ontologies. The ontology is planned to contain between 1000 and 2500 terms, plus roughly 10 definitional statements for each term. LTSC also intends to provide the foundation for ontologies of much larger size and more specific scope. To implement such an ontology, LTSC intends to use the Knowledge Interchange Format (KIF): http://www.cs.umbc.edu/kif/; http://logic.stanford.edu/kif/.

To be able to implement a learning resources component and a delivery component, the content resources have to be described by metadata. Two different sorts of metadata can be distinguished:

- metadata that describes knowledge and knowledge networks; and

- metadata that describes modularized content resources.

The first kind of metadata uses relations to describe knowledge ontologies. An ontology can be used by adaptive learning systems to retrieve the context of a course and to structure the content. The ontology does not normally contain modular content, but does contain concepts. Modularized content resources actually contain the content and are linked to the concepts of an ontology.

The IMS Project started its work in 1997. IMS is part of the nonprofit EDUCAUSE consortium (the former EDUCOM). Within EDUCAUSE, US institutions of higher education and their vendor partners develop open, market-based standards for online learning, including specifications for learning content metadata. Also in 1997, the National Institute for Standards and Technology (NIST) and the IEEE P.1484 study group (now the IEEE Learning Technology Standards Committee, LTSC) began a similar effort. After some time, the NIST and IMS began a very close collaboration, while the IMS began to collaborate with the ARIADNE project (Alliance of Remote Instructional Authoring and Distribution Networks for Europe, a European project aiming at a metadata definition for learning documents; ARIADNE homepage: http://ariadne.unil.ch/); [Forte et al. 1999]. In 1998,
IMS and ARIADNE submitted a joint proposal and specification to the IEEE (see the ARIADNE homepage above); [ARIADNE 1999], which formed the basis for the current IEEE Learning Object Metadata (LOM) base document.

To be able to sequence courses and create exercises, the suitability of different metadata approaches for both metadata application areas (relations, description of resources) has to be analyzed. In this article we concentrate on IEEE's Learning Objects Metadata (LOM); we also consider Dublin Core, since it serves as a basis for LOM (it was also extended recently).

Although there are many other metadata standards, there seems to be a consensus that the above-mentioned initiatives are now the most important in the description of educational resources.

2.1 Dublin Core Metadata

The Dublin Core (DC-1) grew out of a workshop in 1995, sponsored by Online Computer Library Center, Inc. (OCLC) and the National Center for Supercomputing Applications (NCSA), and was intended to be a basic collection of metadata elements. The Dublin Core was originally conceived for use by content creators. But nowadays interest in Dublin Core is no longer restricted to that group, many other specialized groups such as museums, libraries, government agencies, and commercial organizations use it as well.

The Dublin Core contains 15 metadata elements:

- **Title**: A name given to the resource.
- **Creator**: An entity primarily responsible for making the content of the resource.
- **Subject**: The topic of the content of the resource.
- **Description**: An account of the content of the resource.
- **Publisher**: An entity responsible for making the resource available
- **Contributor**: An entity responsible for making contributions to the content of the resource.
- **Date**: A date associated with an event in the life cycle of the resource.
- **Type**: The nature or genre of the content of the resource.
- **Format**: The physical or digital manifestation of the resource.
- **Identifier**: An unambiguous reference to the resource within a given context.
- **Source**: A Reference to a resource from which the present resource is derived.
- **Language**: A language of the intellectual content of the resource.
Information that can be stored in Dublin Core metadata does not reflect educational needs specifically (see also [Gilliland-Swetland et al. 2000]). So the description of modular content is too generic. Storing a knowledge library is difficult within Dublin Core because only best-practice relations are used (see Table I). Hence, in order to store the semantic relations needed in a knowledge library (ontology), the Dublin Core element set would have to be extended.

2.2 IEEE’s Learning Objects Metadata

The IEEE’s Learning Objects Metadata (LOM) specifies the syntax and semantics of learning object metadata, defined as the attributes required to fully and adequately describe a learning object. A learning object is defined as any entity (digital as well as nondigital) that can be used, reused, or referenced during technology-supported learning. Examples of learning objects include multimedia content, instructional content, instructional software, and software tools that are referenced during technology-supported learning.

The LOM standard focuses on the minimal set of properties needed to allow learning objects to be managed, located, and evaluated. The standard also allows us to extend the minimal set of properties locally.

Relevant properties of learning objects include type of object, author, owner, terms of distribution, and format (taken from the Dublin Core). Where applicable, learning object metadata may include pedagogical properties, such as teaching or interaction style, grade level, mastery level, and prerequisites. Any given learning object can have more than one description, thus supporting descriptions provided for different application contexts.

The standard also supports security, privacy, commerce, and evaluation, but only to the extent that metadata fields are provided for specifying descriptive tokens related to these areas. The LOM standard references existing open standards and existing work in related areas.

LOM uses the following categories to describe resources. These can be understood as a superset of the elements of the Dublin Core.

• **General**: groups the general information that describes this resource as a whole.

• **LifeCycle**: describes the history and current state of this resource and those that have affected this resource during its evolution.

• **MetaMetaData**: describes the specific information about the metadata record itself (rather than the resource that this record describes), who created this metadata record, how, when, and with what references.
● **Technical**: describes the technical requirements and characteristics of this resource.

● **Educational**: describes the key educational or pedagogic characteristics of this resource. This category stores the pedagogical information essential to those involved in achieving a quality learning experience. The audience includes teachers, managers, authors, and learners.

● **Rights**: describes the intellectual property rights and conditions of use for this resource.

● **Relation**: defines the relationships among this resource and other targeted resources, if there are any. Multiple relationships can be supported.

● **Annotation**: provides comments on the educational use of this resource, who created this annotation and when.

● **Classification**: describes where this resource is placed within a particular classification system. To define multiple classifications, there may be multiple instances of this category.

LOM is a good solution to the problem of describing modular multimedia content. Both the “technical” metadata and pedagogical aspects can be described in detail.

To be able to analyze the capabilities of LOM with regard to knowledge management, the relationships that can be used to reference related resources have to be examined. Since an ontology contains both concepts and links to related concepts, the ability to model these links is important in creating a full-fledged knowledge model. LOM contains the following types of relations (see Table I).

These relations, modeled inside LOM category 7 (category “Relations”), are a best-practice list taken from the Dublin Core. The Dublin Core is intended to describe general resources without taking the special requirements of educational documents into account. Hence, relations such as “isBasisFor” are missing. Such relations can be quite important when picking content modules. The selection of a specific content module, controlled by the user profile, needs information about related modules, which may become necessary in order to understand the actual module.

In conclusion, LOM is well suited for describing modular content. A potential mapping between the elements of the Dublin Core and LOM can increase the acceptance of LOM. However, LOM also has some drawbacks. The necessary description of a knowledge library is only possible to a
limited extent. Important relations that could be used to model a semantic relationship between concepts are missing. These relations must be added to be able to use LOM within the context of a course-sequencing scenario.

In the following, we describe how a course-sequencing approach using an extended LOM metadata scheme can be implemented.

3. ADAPTIVE CREATION OF LESSONS IN MULTIBOOK

To be able to describe the automatic sequencing of courses as well as the creation of exercises, the underlying framework has to be explained. Both goals are considered in the context of the Multibook project, which is currently being developed at the Technical University of Darmstadt.

The content of the Multibook system (see http://www.multibook.de) is a book, consisting of about 1200 pages, entitled Multimedia: Computing, Communications & Applications by Ralf Steinmetz and Klara Nahrstedt, and a selection of Java applets [Steinmetz and Nahrstedt 2000]. The aim of Multibook is to have individual views on this material, according to the needs and preferences of the individual users. These views are created on the fly using a sequencing approach [Seeberg 1999]. But first we must examine why such a special setup is necessary to tailor lessons to the intended end-user.

A linear, printed book does not satisfy the above requirements: It does not adapt flexibly to the level of difficulty, the goals and strategy of the learning itself, nor to the media preferences of the specific user. Apart from this, many aspects of multimedia technology can be explained in a better way using motion and interactivity.

A traditional hypertext or hypermedia system that does not adapt the content does not satisfy these requirements either. A hypertext is static, in the sense that the text is heavily linked and there is path for every user, in which case a user might become confused by the number of possibilities and not be able to find his path. There may also be only a few links, in which case the document would probably not satisfy the demands of each individual user.

An Intelligent Tutoring System (ITS) does satisfy these requirements, but systems of this type only work in highly structured areas such as mathematics and (especially) programming languages. Multibook focuses on multimedia technology, which is not highly-structured. Multimedia can address various topics, such as networking, hypermedia, security, content processing, and design. Many learning systems incorporate mechanisms that adapt lessons to the progress of students; the ability to adapt is the primary feature that produces “individualized” instruction—this is what is “intelligent” in Intelligent Tutoring Systems. For example, the purpose of the standard developed by LTSC is to provide a common mechanism for exchanging and developing this kind of information among the users, proctors, and developers of courseware (courseware is the educational content delivered to the students); see working groups P1484.6 and P1484.10 (Learning Technology Standards Committee (LTSC): http://ltsc.ieee.org).
The necessary level of control and detailed knowledge of the user can't be achieved in Multibook's less structured subject domain. To formally model everything that can be said about multimedia in texts or images is not a feasible approach. Also, the extent and the extensibility of our material and the amount of text describing it are constraints that make it impossible to guide the user in such a controlled way. In this context, by “control” we mean the opportunity to automatically analyze why a user did something wrong and the ability to offer a correction.

The general functionality of Multibook—the sequencing of courses and exercises—is knowledge-based [Steinacker et al. 1999; Fischer and Steinmetz 2000]. This approach is similar to IEEE’s Learning Technology System Architecture (LTSA) model, because IEEE proposes using a knowledge library (knowledge base) which is responsible for sequencing a lesson, while the actual compilation of the lesson is performed by a delivery component (see Section 2).

It is essential to understand Multibook’s setup of its knowledge base in order to understand the sequencing of courses and the automatic creation of exercises. Multibook’s knowledge base works with two different spaces: the ConceptSpace and the MediaBrickSpace. Depending on the way an author writes a document, the following order can be specified: (1) an author acquires background knowledge; (2) an author creates an outline for a document; (3) an author fills the outline with content. These steps are modeled in different spaces in Multibook. The ConceptSpace contains an ontology in terms of keywords, which are necessary to create the outline of a lesson. After the sequencing of the outline (corresponding to the creation of a table of contents), it (the outline) is filled with the “real” content—i.e., text, images, audio, video, animation—using elements of the second space, the MediaBrickSpace. A general idea of Multibook is that it is necessary to employ different relations within the ConceptSpace and the MediaBrickSpace to model the different goals that both spaces have. In order to fulfill the requirements of both spaces, we use knowledge management relations in the ConceptSpace and rhetorical didactic relations in the MediaBrickSpace. We describe the necessary relations below.

An advantage in separating concepts from content is that content in the system can be changed without affecting the overall structure stored in the ConceptSpace. It is also very useful to be able to extend parts of a document, for example by deepening explanations or by examples, if additional media bricks are inserted. The process of inserting additional media bricks does not change the ConceptSpace.

In the ConceptSpace we use the set of semantic relations shown in Table II (examples in Section 4). Some of these relations are well known from various approaches to knowledge management. Semantic relations that can be stored as metadata [Mann and Thomson 1987; Fischer 1988] can help to structure a complex domain. It should be mentioned that the set of relations we apply is in no way complete. Multibook uses technical information about multimedia. The semantic relations above are sufficient to model our specific domain. The use of the model for another domain would,
however, suggest a redesign of these relations. For example, in a medical learning system a relation “causes” would have to be introduced to describe diseases whose cause is indexed with a keyword stored within a node.

Thus, sequencing an educational document as well as creating exercises according to a specific learning strategy is a specific navigation of the ConceptSpace. The implementation of a learning strategy has to be done by a pedagogical expert [Seeberg 1999], together with an expert in computer science, in order to match the navigation of the ConceptSpace with a learning strategy.

Once the table of contents has been set up, it has to be filled with content. Multibook models the content in the so-called MediaBrickSpace. The difference from the ConceptSpace is that in the MediaBrickSpace we use a second set of relations i.e., rhetorical-didactic relations from natural language processing [Mann and Thomson 1987]. These relations (see Table III) model dependencies between media bricks stored in a modularized form. By exploiting the type of relation, the learner’s preferences can be fulfilled. For instance, using the media brick type, media preferences can be taken into account, e.g., a user on a slow modem link gets only a text version of an educational document. These types of relations can also be used to switch between different levels of difficulty. Since it is a complex task to measure the difficulty of dividing a content module, Multibook uses the fact that the relative difficulty of aggregating bricks can be measured. It is quite obvious that a text $A$ together with two transparent examples is easier to understand than a version containing only text $A$. Also, a text $A$ will become more difficult when presented together with a second text $B$, which is connected to $A$ with the rhetorical relation deepens. The Multibook approach also has another important advantage, in that a lesson can be made more coherent by using a rhetorical relation. The fact that sequential educational documents have the great advantage of coherence should not be ignored. In contrast, a text, when created from isolated content modules, will be very difficult to read because there is no coherence between the

<table>
<thead>
<tr>
<th>Name of relation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superconcept</td>
<td>A node is a superconcept of another node</td>
</tr>
<tr>
<td>AEPart</td>
<td>For all instances of a node there exists a subnode</td>
</tr>
<tr>
<td>EEpartOf</td>
<td>There exists a subnode for an instance of a supernode</td>
</tr>
<tr>
<td>(Inverse)Procedure</td>
<td>A node contains a (inverse) procedure with regard to another node</td>
</tr>
<tr>
<td>Follows/Precedes</td>
<td>A node follows/precedes another node (ordering in a document)</td>
</tr>
<tr>
<td>Formalize/IsFormalizedBy</td>
<td>A node formalizes/ is formalized by another node</td>
</tr>
<tr>
<td>ProblemSolution</td>
<td>A node points to a problem which is a solution node</td>
</tr>
<tr>
<td>Partition</td>
<td>Subnodes partition a domain, for example, images are partitioned as b/ w, grey, and color</td>
</tr>
<tr>
<td>Cost</td>
<td>The cost of another node</td>
</tr>
<tr>
<td>Uses</td>
<td>A node uses another node</td>
</tr>
<tr>
<td>Application</td>
<td>A node is an application of another node</td>
</tr>
<tr>
<td>Instance</td>
<td>A node is an instance of another node</td>
</tr>
</tbody>
</table>

Table II. Relations in Multibook’s ConceptSpace

content modules. A suggestion like "as can be seen in the following figure" cannot be part of such a document. However, a rhetorical relation is well suited to augment coherence—for example, a text A together with an example B. Since A and B are connected by the relation example, it is quite easy to include a little text sample: “In the following, an example will be given which illustrates the problem.” In this context it is interesting that these sample texts do not vary when used several times. In our user evaluations, we first varied the text samples; for example, we used the samples “this will be explained next” and “a more detailed explanation can be found in the next section.” However, the psychologists who evaluated the Multibook project recommended using only one text sample in order to increase the recognition of coherence.

Multibook contains many other important components; for example, a user profile that tracks the actions of a learner. However, these interesting topics are beyond the scope of this article; for details, see Steinacker et al. [1999a; 1999b]; Fischer and Steinmetz [2000] or at http://www.multibook.de.

4. AUTOMATIC DERIVATION OF EXERCISES IN MULTIBOOK

In this section we concentrate on the automatic creation of exercises as part of an adaptive hypermedia system. We chose this application example to explain the functionality of the different sequencing components. Since we understand our paper as a step towards the sequencing of courses and exercises, and because we are also well aware of the fact that the automatic generation of complex exercises is doubtful from the pedagogic viewpoint, we try to address the automatic creation of simple exercises. Many textbooks include a summary of a chapter followed by some simple exercises. It is one of our goals to develop algorithms by which exercises of that type can be created automatically. It will become clear in the remainder of this paper how this specific application can be generalized to perform a (semi-) automatic course sequencing. It should also be mentioned now why the term (semi-) automatic sequencing is used in this article. A general automatic course-sequencing with the goal of replacing a teacher is beyond our scope. We generally prefer solutions where the system creates a course,
which in turn can be checked by the teacher in order to control and correct pedagogic accuracy. Multibook—and maybe such approaches in general—will not be able to create such exercises with a generally sound pedagogical strategy behind the algorithms. Hence, we understand our work as assisting the teacher by enriching the lessons with simple questions, not as replacing the teacher. In no sense is Multibook an “intelligent system” aimed at acting as an educator.

The classic exercise in a technical book is having the student extend the material already discussed—to prove a lemma or a minor theorem, to generalize a synthetic approach, or to apply an algorithm in an unexpected context. A selection of simple exercises that can be found in technical computer science books is listed below. It should be mentioned that these types are the most interesting when teaching a technical domain, such as multimedia systems. However, the concepts developed to create this type of exercise can be extended to derive simple questions for other education domains. The following question types must be considered—for example:

- **Part-of-questions.** What are the parts of an adaptive hypermedia system?

- **Application-of-questions.** What are the application areas for Intelligent Tutoring Systems?

- **Questions about details.** How many rows and columns constitute a QCIF image?

- **Calculations.** How many bits are needed to store a 400 x 400 pixel image with 24 bit encoding for each pixel?

The ConceptSpace (as part of the knowledge library) is the only location where a derivation of exercises can take place. Since the MediaBrickSpace merely contains rhetorical-didactic questions, there is no information about content dependencies available in the latter space. The goal of deriving exercises automatically from an ontology requires the analysis of the different types of questions.

- **Questions about details** as well as the classic complex questions mentioned above cannot be derived from the ConceptSpace because details are never part of an outline. **Calculations** cannot be derived automatically either, since the terminology of the ConceptSpace does not contain the necessary mechanisms. It is also doubtful that support for the learner could be created automatically, since complex questions can lead to a great variety of different (correct and incorrect) solutions. The (manual) creation of exercises and automatic support for the learner is indeed one of the areas of Intelligent Tutoring Systems.

- **Part-of questions** as well as application-of questions can be created automatically in a multiple-choice style, which will be shown in the remainder of this section. To be able to explain the necessary algorithm it is necessary to explain a part of our ontology as an application example (see Figure 2).
This part of the ontology illustrates the model we use to create courses about the image compression algorithms in JPEG. We repeat that the model is only a part of the whole ontology we use. Some relations were omitted for the sake of clarity. The concept JPEG is the center of the model. The various parts of JPEG—image preparation, discrete cosine transform, quantization, and entropy encoding—are connected by uses-Relations. Applications of JPEG are also indicated, for example the concepts WWW and MPEG. These concepts are connected using the relation applicationOf. In this context we are not interested in MPEG or in the WWW itself, but in MPEG and the WWW as potential applications of JPEG.

To be able to create part-of questions, we use the following algorithm:

1. Create the text of the question: “Which are the parts of the concept <Name of the concept>. Please select the correct answers.”
2. Build a list of the correct answers by following the “uses” or “partOf” relations of the concept JPEG: image preparation, discrete cosine transformation, quantization and entropy encoding.
3. Build a list of wrong answers (explained below).
4. Merge both lists in a random order.
5. Check if correct answers are provided by the learner.
(6) If wrong answers are clicked in the multiple-choice form, offer the option to branch to explanations of wrong concepts as well as to a repetition of the lesson being learned.

To match the background knowledge of the learner, we assume that a selection process using a user profile has already taken place. Multibook contains a user profile, which cannot be described in the scope of this paper.

An obviously important step in the algorithm is the semantic selection of appropriate wrong answers, since these answers cannot be picked randomly out of the ontology. However, the following assumptions can help pick a set of wrong answers:

- Most ontologies contain one or more taxonomies that are tree-like hierarchical structures. An example is the taxonomy Compression => Image Compression => JPEG.

- Wrong answers should originate from a similar context. For the JPEG example, a wrong answer like “compression of motion pictures” is an appropriate choice. A wrong answer like “ATM-cell” is not an inappropriate answer.
We apply the following algorithm to select wrong answers in a semantically meaningful way:

1. Reduce the ontology to a taxonomy using the relations “superconcept” and “partOf”. Traverse “instanceOf”-relations if necessary.

2. Go from the current branch of the hierarchical tree of the taxonomy up to the next superconcept. Traverse “instanceOf”-relations if necessary. Check for other subconcepts (“partOf”).

3. Jump to the new subconcept and select new wrong answers for the question to be created, evaluating the “uses” or “partOf” relations.

4. Repeat the procedure with the next hierarchy level if necessary.

An example of such a selection is illustrated in Figure 3. The algorithm first selects the correct answers by evaluating the “uses” and “partOf” relations connected to the concept JPEG. In the next step it traverses the “instanceOf” relation to the concept “image compression” and from there to the concept “compression” using the relation “superconcept”. Using the path from “video compression” to “MPEG”, new components can be found that are wrong answers with a semantic relationship to the concepts connected to “JPEG”.

It is very important to identify wrong answers that are in a close semantic relationship to the correct answers. Several observations can be found that restrict the search:

- An ontology can contain several taxonomies. An example for multimedia systems is when there are two distinct taxonomies for compression and for networking. Although there are semantic relations between compression and networking (compression makes networking more efficient because bandwidth is saved) as part of the overall ontology, there are no such relations between the two taxonomies of networking and compression. An important reason is that a taxonomy is based merely on the relations “superconcept” and “instance” in our model. It can never be that networking is a superconcept of compression in general or vice versa. This observation guarantees that, within very broad limits, totally different concepts cannot become part of the same question.

- It can be that no branches of the taxonomy that are semantically similar to the correct answers can be found where concepts are located. On the other hand, it is quite useless to create a question for which all the answers are correct. Experiments we conducted with students showed that it is better to avoid an automatic generation of exercises in that case.

Having explained the way we generate “part-of”-exercises, we continue our description of a second type of questions, the “is an application for” questions. To generate exercises that verify that a student has understood the general applicability of a concept, we use a similar approach:
(1) Create the text of a question: “What are the applications of <Name of the concept>. Please select the correct answers.”

(2) Build a list of the correct answers by following the “applicationOf” relations, for example of the concept “JPEG”: MPEG is an application for JPEG.

(3) Build a list of wrong answers (explained below).

(4) Merge both lists in a random order.

(5) Check if correct answers are provided by the learner.

(6) If wrong answers are clicked in the multiple choice form, then offer the option of branching out to explanations of wrong concepts as well as to a repetition of the lesson being learned.

With regard to Figure 2, the algorithm will select the correct answers MPEG and WWW where JPEG is applied.

To pick the wrong answers, a similar approach to the one for selecting wrong answers for “part-of” exercises can be used. However, an important observation is the fact that applications of concepts are in most cases instances. JPEG is for example an instance of the abstract concept “image compression”. Hence the selection of wrong answers is less critical compared to the selection of wrong answers for “part-of” exercises. Although it would not make much sense to ask if, for example, Ethernet is an application of JPEG, the learner would not be surprised, but disturbed, if concepts from another taxonomy were used in the “part-of” questions. The algorithm for selecting wrong answers follows:

(1) Reduce the ontology to a taxonomy using the relations “superconcept” and “partOf”.

(2) Leave the current branch of the hierarchical tree of the taxonomy up to the next superconcept. Traverse “instanceOf” relations if necessary. Check for other subconcepts.

(3) Jump to new a subconcept and select new wrong answers for the question to be created, evaluating the “applicationOf” relations.

(4) Repeat procedure with the next hierarchy level if necessary (if no appropriate concepts can be found in one level).

The algorithm is explained in a graphical way in Figure 4.

5. EXPERIENCE

In an evaluation, our students worked with the Multibook exercise framework [Fischer and Steinmetz 2000]. The experimental setup contained both questions generated by hand and questions generated automatically. Part of the evaluation was the question whether the student could identify the exercises created automatically. It turned out that in most cases the students were not able to distinguish between the kinds of exercises. A
The better identification of exercises that were generated automatically became possible once there was a big local displacement of the concepts chosen as correct and incorrect answers in the taxonomy. In other words, each time more than one superconcept-relation was traversed in order to find wrong answers, the semantic similarity of correct and incorrect answers became smaller, and the students were able to identify the difference. However, most students stated that they might not have noticed the difference if we had not told them in advance. However, it should be emphasized that this assumption is only true if the tree of a specific taxonomy is not left and wrong answers are not the result of a different taxonomy.

The evaluation also shows that it is still important to control the results of automatic sequencing by hand. A critical issue for electronic learning systems is their correctness. Many students reported that they would lose confidence in the system if silly exercises were to show up too often. For broad acceptance, the content has to be created in a very sound and correct way. This, again, is an argument for understanding automated systems as a support for teachers. The creation of such an exercise environment gives teachers some time to think about difficult exercises. However, the time for controlling the semantic quality of the resulting exercises should not be saved. Fortunately, the ratio between the time necessary to create exercises by hand and the time necessary to control the output is rather large.
6. RELATED WORK

Hypertext and hypermedia systems exploit the nature of various media like text, pictures, audio, video, and simulations as a way to make differentiated statements and communicating less structured knowledge. In addition, by selecting different nodes in different orders, hypermedia systems offer more than predefined learning paths—individual learners produce a multitude of paths through the material. The drawback is that the learning process cannot be controlled in a well-defined way. This results in insufficient guidance. In particular, when used for educational purposes, hypermedia systems strive for a higher degree of control [Brusilovsky 1998; Greer et al. 1998; Calvi and de Bra 1997; Pilar da Silva et al. 1998]. We regard adding conceptual information on top of hypermedia chunks as the decisive the step, since it is the the basis for intelligent selection and sequencing (see also De Bra [1999]). Connecting the concepts to semantic rather than didactic relations that already imply sequences or dependencies among the concepts provides a yet higher degree of flexibility (“Ontologies for Knowledge Sharing” by J. Sowa, manuscript of the invited talk at TKE'96); [Sowa 2000]. This way, the concept space lends itself to realizing different learning strategies and goals, but also to tasks like information retrieval.

The general setup in Multibook has acquired a great deal of experiences in adaptive hypermedia systems, such as Interbook (InterBook homepage: http://www.contrib.andrew.cmu.edu/~plb/InterBook.html) [Brusilovsky 1998]; KBS Hyperbook (http://www.kbs.uni-hannover.de/hyperbook/) [Nejdl and Wolpers 1999; Henze and Nejdl 2000]; or ELM-ART (http://www.apsymac33.uni-trier.de:8080/lisp-kurs) [Specht and Weber 1996; Weber and Specht 1997; Brusilovsky et al. 1996]. More details on related work in adaptive hypermedia documents with regard to realization and implementation in Multibook can be found in Seeberg [1999].

Multibook is based on a terminological ontology. Related work on ontologies is described in “Ontologies for Knowledge Sharing” by J. Sowa (manuscript of the invited talk at TKE’96) and in Fischer [1988]; the use of ontologies with regard to the rhetorical structure theory is described in Mann and Thomson [1987] and in Nicholas [1995].

The realization and implementation of Multibook has to be seen as being close in context to that of the Learning Technology Standards Committee (LTSC), which also proposes the use of a knowledge base and a delivery component that, in some sense, matches the use of a ConceptSpace and a MediaBrickSpace (Learning Technology Standards Committee (LTSC): http://ltsc.ieee.org).

7. CONCLUSIONS AND OUTLOOK

In this article we examined ways to automate course and exercise sequencing. After a review of metadata standards necessary to store information about a knowledge base and modularized media content, we describe the Multibook system, an adaptive hypermedia learning environment. The creation of exercises is based on a knowledge base (ontology) containing the
curriculum of multimedia technology taught in Darmstadt. We made use of the fact that many ontologies also contain taxonomies, allowing the creation of different types of questions.

The implementation of a knowledge base uses two spaces within the Multibook approach: the ConceptSpace and the MediaBrickSpace. We show that an evaluation of the properties of the ConceptSpace is sufficient to generate questions automatically.

It could be argued that our approach only works on our specific realization of the ontology. In fact, the contrary is true. The generation of exercises may vary from ontology to ontology. However, the algorithms we propose rely on the fact that an ontology contains a set of disjoint taxonomies that is independent from our implementation of the ontology.

The kinds of multiple-choice exercises created automatically are not interactive nor do they allow self-explorative learning—this approach might be restricted to technical domains such as computer science. Again, we stress that it is not our goal to create a sophisticated test environment. The approach described in this article merely matches the state of the art of many educational resources, and can help to reduce the amount of work in producing exercises by hand.

For the future, we will have to integrate the knowledge of the learner into our exercise environment. Problems that have to be solved include the intelligent provision of clues to help a student in the case of wrong answers. These clues can also be derived automatically by exploiting the structure of the ontology and then by using LOM metadata to access the necessary content module. An important aspect is the use of a user profile to record the order in which a learner traverses the educational document. We have not integrated knowledge of the parts of a document a student has already seen, and this can lead to generating questions based on knowledge that has not yet been presented to the learner.

Another research issue we are interested in is the implementation of the ontology model, which will be provided by LTSC. We are convinced that setting-up such an ontology for multimedia cannot be done by a small research group. In such an ontology the exchangeability of parts, which can be used for interactive learning environments, plays an important role.

We understand our approach as a first positive step in the automatic creation of exercises. The goal of this article is the description of the overall ideas. We are well aware that other types of questions could be generated automatically without the use of an Intelligent Tutoring System. For the near future, It is our goal to think about other types of questions that can be derived automatically.

ACKNOWLEDGMENTS
I would like to thank my colleagues Abdulmotaleb El Saddik, Achim Steinacker, Klaus Reichenberger, and Cornelia Seeberg. A large number of fruitful discussions with them has opened the way into this interesting field.
REFERENCES


Received: July 2000; revised: October 2000; accepted: November 2000