Didactics as a First-Class Citizen in Courseware Development *

Khaldoun Ateyeh Birgitta König-Ries
ateyeh@ipd.uni-karlsruhe.de, koenig@informatik.uni-jena.de

Abstract: Since the development of courseware is expensive, reuse is a must. In order to reach the full reuse potential, the individual courseware modules that are stored in a repository should be neutral with respect to as many aspects as possible. In this paper, we examine how didactic-neutral courseware modules can be achieved and in particular how their combination into courses exhibiting a certain didactic method can be supported. This requires a formalized look at didactics and tools that support the weaving of contents and didactics during the course building process.

1 Introduction

Everyone agrees that courseware reuse is a necessity. However, if one regards the current practice, it is obvious that courseware reuse does not really happen. While a number of repositories for course materials exist, they are rather rarely actually used [RHS05]. It is believed that one of the main reasons for the lack of acceptance of these repositories is that their contents are not sufficiently adaptable to different contexts. This lack of adaptability restricts reuse to the reuse of entire courses that happen to be suitable, but hinders the reuse of smaller grained units of material. To support such a wider reuse, what is suggested are repositories that contain fine-grained, individually reusable modules of material. These modules should be as neutral as possible with respect to a number of aspects, in particular with respect to layout, technical environment, and didactics [RHS05, AKKRM03].

While materials might be neutral with respect to these aspects, courses will not. To build a course from the existing material, the instructor needs to impress these aspects onto the material or rather weave them together with the content aspect reflected in the repository. Of course, this weaving can be done manually - however, this is cumbersome and an obstacle to reuse. Instead, the system should support the instructor during this process. There are first approaches that offer this functionality with respect to layout and technical environment. However, the weaving of didactics is not well supported up to now. If one wants to support this aspect, a formal model of didactics and a technique to interleave content and didactics are needed. We present both in this paper.

*This work was partially funded by the *ViKar*-Project [Vik03] within the program *Virtual University* by the state of Baden-Württemberg.

2 Related Work

Approaches found in the literature that consider courseware reuse rarely include the reuse of didactics. For instance, learning technology standards approaches [WG103, ADL04] consider the reuse of content and technical components but ignore the reuse of didactics.

Recently, didactic standardization efforts such as EML (Educational Modeling Language) [Kop04] and based on it the IMS Learning Design standard (IMSDL) [Con04] begin to appear. EML and IMSDL are notational systems that intend to describe a wide variety of instructional models. They are used to model units of study. Instructional or didactic models are modeled as an integral part of a specific unit of study. Consequently, the result of the modeling process is mostly a complete, to some extent monolithic unit of study that mixes the used didactic model with other concerns such as the contents and the technology provided by the environment used. This makes it difficult, if not impossible, to reuse by itself the didactic model or the content. If this is attempted, typically, a considerable effort has to be made that exceeds the costs of a new from-scratch development. They assume pre-structured ready-to-use content objects that happen to match the selected instructional strategy of the unit of study. This is rarely the case in practice.

Courseware reuse along different dimensions has been proposed by Merrill, though [MLJ90, MLJ91]. He introduces the concept of instructional transaction shells to organize existing instructional transactions and knowledge objects into new courseware. However, little is being said about the process of the development of such reusable elements. Rather the focus is on the process of automatic generation of instruction from existing reusable instructional transactions and knowledge objects.

The project K-Med [RHS05] is also concerned with courseware reuse. K-med claims that e-learning materials have to be reauthored according to five different dimensions (didactics, technique, language, content and layout) if they are to be reused. To allow reauthoring, K-Med provides a unified structured data format that can be used to store e-learning content in a neutral format such as XML and that enables the separation between content and navigation information. However, up to now, a format to describe the other dimensions seems to be missing. Also, no information is available on how the reauthoring, i.e. the weaving of the content with the other dimensions is supposed to be done.

3 A Formal Model of Didactics

In order to allow an instructor to select a didactic method when building a course, in addition to content modules, a repository of didactic methods together with information about the kind of material appropriate for each didactic step is needed. In this section, we describe a formal model of didactics as a basis for such a repository.

3.1 Requirements with Respect to Didactics

This section discusses the requirements that have to be considered by a didactic framework that can be used to design any conceivable didactic strategy and/or method in a way that they can be (re)used by different persons in different learning/teaching contexts.

- *Didactic diversity*: In order to be able to meet different didactic requirements raised in a heterogeneous environment of courseware development and reuse, it is essential to allow pedagogical and didactic diversity. Didactic diversity means that the framework should not bind or restrict the didactic designer to the use of one learning theory or didactic model. It should be possible to define and reuse any didactic strategy or method no matter on which learning theory or model they are based.
- *Reusability of didactic*: There is a need for didactic concepts of reuse that help didactic designers, authors, and instructors to easily identify, access, share, adapt, develop, and reuse didactics. Therefore, units of modularization and reuse for didactics have to be identified.
- *Compatibility with the principle of separation of concern*: The didactic designer must be able to design and develop didactic concepts independently of a concrete learning/teaching context and content. This is also important to fit in an overall development process which requires the application of the principle of separation of concerns in the development process.
- *Flexibility and adaptability*: Instructors and/or authors should be able to adapt the defined didactic strategies and methods to their own needs and requirements. Only then can didactic be reused in a meaningful way.
- *Formality*: The didactic designer must be able to describe the desired didactic concepts in a formal way, so that an effective computer-based processing and (re)use is possible. In particular, similar to the content concern, it should be possible to manage reusable didactic entities in a didactic repository where the instructor, the courseware engineer, or other didactic designers can access, adapt, and reuse the material in their specific learning/teaching contexts.

3.2 Learning Process Didactics

We distinguish between two types of didactics. Learning process didactics and content didactics. This section is concerned with learning process didactics. Content didactics is discussed in Section 3.4.

Learning process didactics describes the phases and activities conducted throughout the learning process to achieve the desired learning objectives. In particular, it is concerned with the definition of *learning/teaching strategies* and *didactic methods*.

The choice and the development of a suitable *learning/teaching strategy* depends very much on the learning theory, instructor belief, and the objectives of the course. However,

despite the differences found between the strategies, especially between those that are based on different learning theories, all split the learning process into different learning phases, with each phase achieving a distinct didactic function or objective. Take as an example the three-step learning/teaching strategy of Figure 1(A) with three main phases:

- Introduction phase: clearly define the starting point and the goals of the learning process.
- Working phase: offer a suitably stimulating environment for the learning process. This phase consists of further sub-phases: set up, work through, apply, transfer, assess and integrate.
- Completion phase: secure the goals of the learning process.

A second example is the problem-based learning/teaching strategy of Merrill [Mer00] (Figure 1(B)). It has its roots in the constructivism learning theory and makes use of the principle that learning is facilitated when the learner is engaged in solving real-world problems. Hence, based on a specific real-world problem the learner should be involved in four distinct learning phases: Activation of prior experience necessary to solve the problem, demonstration of skills necessary to solve the problem,application of learned skills, and integration of skills into real world activities.

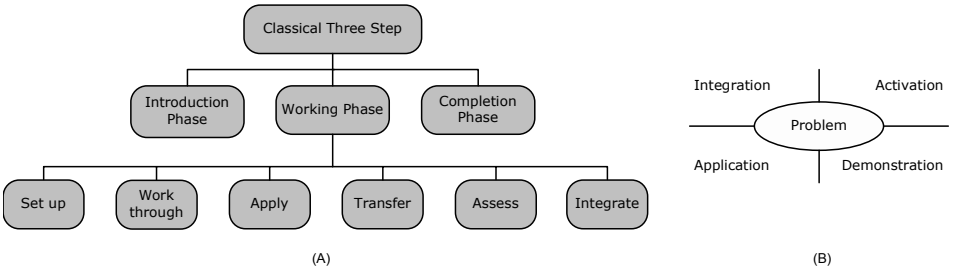

Figure 1: Examples learning/teaching strategy: (A) Classical three steps, (B)Problem-based learning/teaching strategy (based on [Mer00])

Example 1 *Consider as a concrete example a subtopic, e.g. nested queries, of a course on SQL. Using the classic three step strategy we proceed as follows: In the introduction phase, existing knowledge about SQL should be recapitulated, and it should be motivated why nested queries are needed. In the working phase, the syntax of nested queries should be introduced (set-up), examples should be shown (work-through and apply), natural language requests for specific information nested queries should be formulated (transfer), nested queries should be compared to alternative ways to obtain the same information (assess) and nested queries should be used together with other SQL constructs (integrate).*

Depending on the learning context, generic learning/teaching strategies can then be specialized and adapted to meet the needs and characteristics of the learning context.

The second didactic entity used to describe the learning process is the *didactic method*. In contrast to learning/teaching strategies, didactic methods are more specific to a learning context and less flexible than learning strategies. They (precisely) describe the activities of the learners and/or instructors necessary to achieve specific learning objectives and assume a more specific learning context and environment for their application. They are used to realize the learning phases of a learning strategy. Examples are mindmapping, spotlight, active structuring, presentation and so on.

Example 2 *Let us return to Example 1. The introduction phase could require students to try and formulate queries with the SQL constructs they already know, revealing the need for nested queries. Then, a presentation of the syntax of nested queries followed by the online presentation of some examples could cover the set up and apply phases, the students could then individually formulate nested queries to gain information, a collaborative discussion comparing nested queries to alternative formulations would be an appropriate method for the assessing phase. A computer session, requiring the usage of nested queries together with other SQL constructs could serve both to integrate and to secure the knowledge gained. In this example the didactic methods brainstorming, impetus presentation, concept mapping, discussion, and active practice are being used.*

In the literature, learning/teaching strategies and didactic methods are mostly expressed using natural language (see, e.g., [Eig00] for an unusually structured but still informal description). This is not suitable in a cooperative environment that aims to increase courseware reuse and that is based on automatic processing of courseware entities. What is needed is a formalism that allows to capture, localize, and separate the didactic concern, so that it can be reused and easily adapted by different persons in different contexts. In the remainder of this section, we provide such a didactic meta model.

3.3 A Formal View at Learning Process Didactics

The notion of learning/teaching strategy introduced above can be formally modeled as depicted in Figure 2 and can be defined as follows:

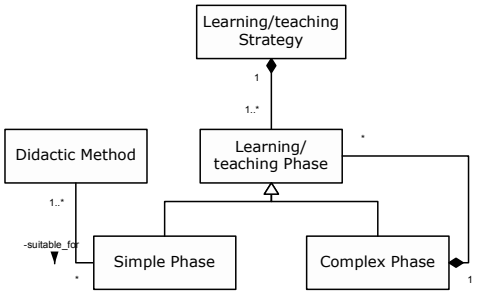

Figure 2: A UML model describing the notion of learning/teaching strategy

Definition 1 (Learning/teaching strategy) *A learning/teaching strategy structures the learning/teaching process into one or more phases, where each phase meets a specific didactic function. Phases can be simple or complex; the latter are comprised of single or complex phases. For every simple learning/teaching phase one or more suitable didactic methods should be used.*

Listing 1 shows as an example the description of the classic three-step learning/teaching strategy. It also shows how the didactic model can be implemented using XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<learningTeachingStrategy
    name="Classical Three Step" dbi="CTS_DBI" uri="CTS_URI">
    <simplePhase name="Introduction">
        <didacticMethodRef name="Presentation" dbi="M2"/>
        <didacticFunction>is to clearly define the
            starting point and the goals of the learning process.
        </didacticFunction>
    </simplePhase>
    <complexPhase name="Working Phase">
        <didacticFunction>To offer a suitable stimulating
            environment for the learning process.
        </didacticFunction>
        <simplePhase name="Set Up">
            <didacticMethodRef name="Spotlight" dbi="M1"/>
            <didacticMethodRef name="Discussion" dbi="M5"/>
            <didacticMethodRef name="Mind Mapping" dbi="M1"/>
            <didacticFunction>to set up the working phase</didacticFunction>
        </simplePhase>
        ...
        <simplePhase name="Assess">
            <didacticMethodRef name="Mindmapping" dbi="M3"/>
            <didacticFunction>to evaluate the learning process and the achieved
                objectives
            </didacticFunction>
        </simplePhase>
        <simplePhase name="Integrate">
            <didacticMethodRef name="Mindmapping" dbi="M3"/>
            <didacticMethodRef name="Brainstorming" dbi="M2"/>
            <didacticFunction>...</didacticFunction>
        </simplePhase>
    </complexPhase>
    ...
</learningTeachingStrategy>
```

1: An XML document describing the classical three step strategy

Didactic methods on the other hand are used to realize phases of a learning strategy and can be defined as follows:

Definition 2 (Didactic Method) *A didactic method is a system of didactic rules that instructs or guides students or teachers while learning or teaching. A didactic method has one ore more didactic functions and works under certain circumstances to achieve certain learning or teaching goals.*

Figure 3 shows a UML diagram that describes our model that can be used to describe and capture any conceivable didactic method. According to our model, a didactic method can be described by the following properties:

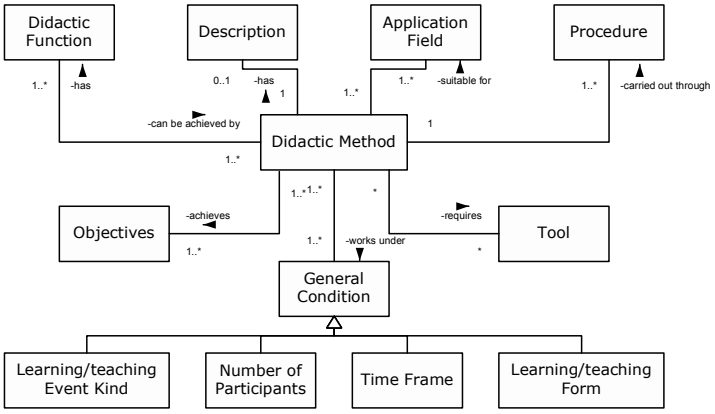

Figure 3: The didactic method model

- **Description** is optional and gives a general description of the didactic method. This is particulary useful when the user wants to get a quick overview of the method.
- **DidacticFunction** captures a didactic function of the didactic method. A didactic method has at least one didactic function.
- **ApplicationField** describes a situation where the method is successfully applied. A method can have one or more application fields.
- **Procedure** describes how the didactic method is conducted. Although most of the time a didactic method would have only one procedure, it is conceivable that a didactic method offers different procedure's alternatives.
- **Tool** is used to assign the didactic method with the tools needed for its conduction. Depending on the didactic method zero or more tools are used.
- **LearningTeachingForm** describes a learning/ teaching form (such as face-to-face, distance,...) where the didactic method can be successfully used. A method can be used as one or more learning/teaching forms.
- **NumberOfParticipants** specifies the minimum and maximum number of participants for which a successful application of the didactic method is feasible. This element is optional.
- **TimeFrame** specifies the minimum and maximum time needed to conduct the method. This element is optional.

It is important to note here that the values of most of the above properties are restricted to stem from predefined ontologies. This is particulary important to enable more control of the semantics of the properties by the courseware system.

Again, listing 2 shows an example implementation in XML. The listing depicts part of the description of the *Spotlight* didactic method.

```
<?xml version="1.0" encoding="UTF-8"?>

<didacticMethod name="Spotlight" dbid="M1" uri="URi1">
    <didacticFunction>
        learn about expectations.
    </didacticFunction>
    <didacticFunction>
        summarize prior knowledge.
    </didacticFunction>
    <applicationField>
        dealing with conflicts
    </applicationField>
    <applicationField>
        evaluation
    </applicationField>
    <procedure>
        1. Explain the rules: There is always just one person talking,
        2. ...
    </procedure>
    <learningObjective>
        to be able to concisely formulate ones own opinion
    </learningObjective>
    <learningObjective>
        to be able to give feedback
    </learningObjective>
    <learningTeachingForm>any</learningTeachingForm>
    <minNumberOfParticipants>2</minNumberOfParticipants>
    <maxNumberOfParticipants>15</maxNumberOfParticipants>
    <minTimeframe>any</minTimeframe>
    <maxTimeframe>any</maxTimeframe>
</didacticMethod>
```

2: An XML document describing the didactic method ”Spotlight”

3.4 Content Didactics

The learning/teaching strategies and didactic methods discussed above operate on the level of didactic design, that is, they are rather abstract descriptions. In particular, they do not describe what kind of material is useful for each of the steps involved and what alternative realisations are possible. This is the task of the second type of didactics namely the content didactics. The content didactics is dealing with providing so-called *content didactic strategies*. A content didactic strategy describes how different types of material can be combined to implement a certain part of a given didactic method. This is done by offering templates describing possible combinations. The central idea of our approach is to include in the didactic repository a number of such templates. Consequently, a teacher would simply choose templates from the repository to construct the desired course. Each content didactic template should follow one of the didactic methods developed during didactics design. A template, then, represents a possible realisation of a content didactic strategy that prescribes which learning content units are necessary for the achievement of specific learning/teaching objectives, and in which order. Consequently, we define the concept of content didactic strategy as follows:

Definition 3 (Content didactic strategy) *A content didactic strategy structures the con-*

tent of learning modules. It prescribes which learning atoms in which order are necessary for the achievement of specific learning/teaching objectives.

The above definition uses the terms learning modules and learning atoms. These terms represent, in our approach, the concepts of modularization and reuse for the content concern. They are the result of a content engineering phase and are stored in a content repository. Figure 4 shows a simplified UML model for these concepts. A more detailed discussion of these concepts can be found in [Ate04].

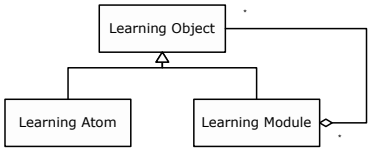

Figure 4: A simplified UML model for the content model

Example 3 *Figure 5 shows a possible didactic template for the* presentation method *in the* set-up phase. *The template states that this phase should begin with an overview followed by a motivation and an activation atom. These learning atoms are mainly used to meet the first learning objective of the set-up phase, namely to activate past knowledge related to the topic. Further, to be able to meet the second learning objective (provide an overview of helpful knowledge), the template states that for each sub-topic of the course an explanation or a definition of the topic followed by an example are needed. Finally the phase should be finished with a concluding example. It is quite possible that there exist other templates for the same phase, offering alternative ways to achieve the learning objective.*

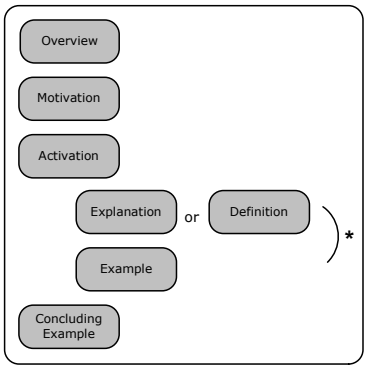

Figure 5: An example for a didactic template

To be able to automatically generate a course from the content repository and a given template, templates should be formulated in a machine-processable language. For this purpose, we have developed DidaScript, a script language with the usual constructs for variable declarations, assignment statements, conditional and loop expressions. DidaScript

includes built-in functions for navigating in a course structure of learning modules , functions for ordering all or selected atoms of a specified module, functions for hiding atoms in a module, and so on [Ate04]. The resulting course structure is represented via XML.

4 Weaving Content and Didactics

Having introduced the concepts learning/teaching strategy, didactic method, and content didactic strategy let us now see how these concepts are related to the process of course engineering. To do that we introduce the concept learning/teaching event, which is a more general concept than a course. In order to implement a concrete learning/teaching event, it is necessary to have an integrated view of the event, taking the contents that are to be taught, the didactics used in teaching these contents, and the constraints that need to be observed into consideration. During the course engineering process, in the process of building a concrete learning event, these aspects will then be configured and merged together to meet the requirements of that particular event. Figure 6 shows a UML diagram that describes the relationships between the learning/teaching event concept and the concepts learning/teaching strategy and didactic method as the modeling units for the didactic aspect, and the learning module and the learning atom concepts as the modeling units for the content aspect. Furthermore, it shows how the didactic and content aspects can be merged together using the concept of content didactic strategy and the concept of *learning resource type*. The intended didactic usage of each learning atom is described by its learning resource type. Examples for learning resource types are: motivation, description, exercise, and example. As depicted in Figure 6, a content didactic strategy is associated

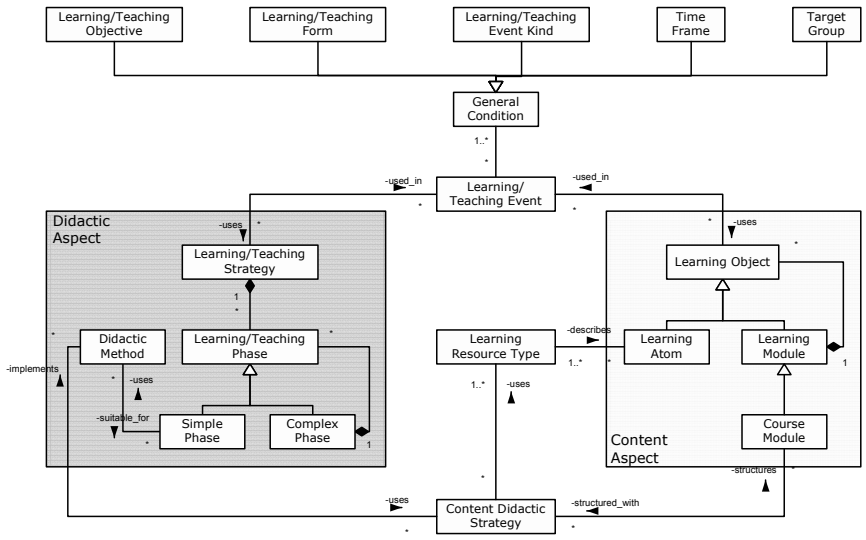

Figure 6: The learning/teaching meta model in some detail

with a specific didactic method and is used to didactically structure the learning modules to meet specific didactic goals given by that didactic method. To do so, content didactic strategies prescribe which learning resource types are necessary for the achievement of the learning/teaching objectives given by the selected didactic method, and in which (didactically meaningful) order. Content didactic strategies are implemented using DidaScript, the didactic script language introduced above.

Let us take a closer look at this weaving of didactics and content. First, the instructor needs to decide which contents to teach and which learning/teaching strategy to use. To decide about the contents, she chooses which parts of the contents model should be covered in her course. The system helps her by providing information about dependencies between contents topics. Let us assume she wants to teach a class about SQL queries. She can then assign each part of the content a learning/teaching strategy. For instance, she could decide that the part about nested queries should be taught according to the classical three-step strategy. She will then be presented with the different phases of this strategy, the steps in each phase and the applicable methods. She might decide to skip or exchange some steps. Also, she needs to choose which of the didactic methods should be used in each step. For instance, she might choose to use a motivating presentation for the set up step of the working phase and a spotlight for the completion phase. She is then given a choice of possible templates for each of these steps. She might choose the template depicted in Figure 5 for the set up step. This information (the contents to be covered and the didactic template to be used) is then fed to a course generator. This program will search for appropriate learning atoms in the repository and combine them into a course structure.

The result of this step will be a semi-finished course. On the one hand, it is possible that the repository does not yet contain suitable learning atoms for certain steps. In this case, appropriate material needs to be created. On the other hand, the automatic configuration will often result in a less than smooth course, requiring the instructor to add some transitionary material etc. However, this is far less effort than creating a course from scratch.

5 Summary and Conclusion

The reuse of teaching material can be enabled by the provision of repositories where this material is stored in a neutral form with respect to both technical and didactic aspects accompanied by tools that allow to weave these aspects together with the contents during course creation. In this paper we have concentrated on, maybe, the most challenging prerequisite for such tools: the existence of a formal model of didactics.

We have introduced in this paper a formal model of didactics. It is implemented in a tool that allows to formulate didactic templates using a machine-processable script language. These templates can then be automatically combined with content from a repository to create initial versions of courses. The ideas presented in this paper have been implemented in a prototype and have been evaluated on a small scale. For the future, we hope to integrate our ideas into a complete e-learning platform.

Also, from a theoretical point of view, the didactic model should be augmented by the pos-

sibility to express learning objectives not only in terms of contents but also in terms of the associated abilities, i.e. an objective might be that a student understands SQL queries another objective would be that she is able to formulate and run queries on her own. Also, it would be desirable to give better support towards the definition of more flexible ”course”structures following the constructivistic paradigm. Right now, building such courses requires a certain amount of manual adjusting by the instructor.

Finally, it is important to note that the didactic framework presented in this paper is part of a complete courseware engineering framework that divides the process of courseware development into two major processes, a domain engineering process, *engineering for reuse*, and a course engineering process, *engineering with reuse*. The domain engineering process is conducted by a domain-specific community and reflects all tasks associated with building an infrastructure of reuse and a common understanding among the community members for a specific knowledge domain. The course engineering process, on the other hand, includes all tasks related to building context-specific courses using the infrastructure of reuse established in the domain engineering process by the domain specific community. For more detail on the complete framework we refer to [Ate04].

References

[ADL04] Advanced Distributed Learning Initiative ADL. Sharable Content Object Reference Model (SCORM): SCORM 2004 Overview Version 1.3 SCORM_2004_Overview.pdf. World wide web pages: http://www.adlnet.org/, February 2004.

[AKKRM03] Khaldoun Ateyeh, Michael Klein, Birgitta König-Ries, and Jutta Mülle. A Practical Strategy for the Modularization of Courseware. In *Professionelles Wissensmanagement - Erfahrungen und Visionen: Adaptive E-Learning and Metadata*, Luzern, 2003.

[Ate04] Khaldoun Ateyeh. *Reuse-Driven Courseware Engineering*. PhD thesis, Universität Karlsruhe, 2004.

[Con04] IMS Global Learning Consortium. IMS Learning Design v 1.0 Final Specification. World Wide Web Pages: http://www.imsglobal.org/learningdesign/index.cfm, 2004.

[Eig00] Gunther Eigler. *Besser Lehren: Heft 2 Methodensammlung und Heft 5 Lehrstrategien*. Beltz Verlag, 2000. In German.

[Kop04] Rob Koper. Modeling units of study from a pedagogical prespective. World Wide Web Pages: http://eml.ou.nl/introduction/articles.htm, 2004.

[Mer00] M. D. Merrill. First Principles of Instruction. http://www.id2.usu.edu/Papers/5FirstPrinciples.PDF, 2000.

[MLJ90] M.D. Merrill, Z. Li, and M.K. Jones. Second Generation Instructional Design. *Educational Technology*, 30(2):7–14, 1990.

[MLJ91] M.D. Merrill, Z. Li, and M.K. Jones. Instructional Transaction Theory. *Educational Technology*, 31(6):7–12, 1991.

[RHS05] Christoph Rensing, Stefan Hoermann, and Ralf Steinmetz. Wiederverwendung und Autorenunterstützung im E-Learning auf Basis von Digitalen Repositories. *thema Forschung: E-Learning*, pages 84–91, February 2005. In German.

[Vik03] Virtueller Hochschulverbund Karlsruhe Vikar. World wide web pages: http://www.vikar.de, 2003. In German.

[WG103] LTSC WG12. LOM meta data standard. World wide web pages: http://ltsc.ieee.org/wg12/, 2003.