A Model of Reusable Educational Components for the Generation of Adaptive Courses

Amel Bouzeghoub[1], Claire Carpentier[1], Bruno Defude[1], Freddy Duitama[1]

[1] GET- Institut National des Télécommunications
Department of Computer Science
9, rue Charles Fourier - 91011 Evry cedex FRANCE
Firstname.Lastname@int-evry.fr

Abstract. In the last few years the problem of developing sequencing of course material has become an important research issue, particularly the standardization of metadata for educational resources. Our work aims at providing an environment of authoring and presentation of pedagogical multimedia contents adapted to the end-user. This environment must increase the productivity of the teacher by supporting the reuse of contents already developed in other contexts (by him/her or others). This goal is achieved by means of a model of educational components. In this paper, we define a set of operators used to build courses by assembly of components. The courses thus defined are then instantiated for each learner according to his/her profile. To classify component our approach uses an ontology to describe the domain model, where each node represents a concept. There exists several kinds of hierarchical and rhetorical relationships between concepts in the domain model.

1 Introduction

With the widespread adoption of Internet and web technologies, new forms of educational applications are appearing. We can easily notice that learning paradigms are changing. Not only learning anytime and anywhere is required, but also flexible learning environments that allow to learners to adapting courses according to their personal preferences and skills.
The dissemination of knowledge on Internet requires the availability of teaching contents adapted to different public. The commercial proposals currently existing facilitate the generation of courses. However, these proposals could not satisfy the keen and pressing demand market. Indeed, with each creation of course, the contents are completely redefined. However, the fast development of technologies in the field of the Internet and the Web allows exchanging information more efficiently. The authors of course reuse increasing external documents and want to effectively reuse [3] their own documents (entirely or partly). Smart-learning project [1] proposes an approach based on the generic course notion to conceive pedagogical sequences. The goal of our work is to provide an environment of authoring and delivering of teach-

ing contents adapted to the end-user [2] by combining components (called teaching objects [9]). These components, available on the Web, could be accessed, and exchanged between teachers belonging to a community. All these components will constitute a common base of knowledge, thus encouraging the sharing of the teaching resources [6].
Our system of training is based on three models: the domain model which represents the concepts covered by the courses, the user model which keeps the profile of learners, the educational component model which describes component contents related to the domain model. The delivering process uses these three models to select the most adapted components and to gather them in a coherent way according to the teaching objectives and learner's profile. In this article, we emphasize on the pedagogical model and the authoring and delivering processes. In section 2, we position our approach to related works. Section 3 gives an overview of our proposal focusing on the domain model. The section 4 describes educational component model and its associated composition mechanisms. Section 5 presents authoring and delivering processes and finally, a conclusion is proposed in section 6.

2 Related Works

A Learning Management System (LMS) is a high-level, strategic solution for planning, delivering, and managing all learning events within an organization, including online, virtual classroom, and instructor-led courses. The focus of a LMS is to manage learners, keeping track of their progress and performance across all types of training activities, but is not generally used to create course contents. In contrast, the focus of a Learning Content Management System (LCMS) is on course contents creation. It gives authors, instructional designers, and subject matter experts the means to create e-learning contents more efficiently. The primary business problem a LCMS solves is to create just enough content just in time, to meet the needs of individual learners or groups of learners. Rather than developing entire courses and adapting them to multiple audiences, instructional designers create reusable content chunks and make them available to course developers throughout the organization. This eliminates duplicate development efforts and allows for the rapid assembly of customized content.
As the very first work in the field of LCMS, the principal goals were fixed: adaptation (to learners), flexibility (rather than precomposition fixed) and scalability (industrial production without proportional cost). An answer can be found with these three goals using the concept of pedagogical component. This concept was already largely studied and certain standards exists (Dublin Core [18], LOM [16], SCORM [19], ARIADNE [20]). All these works are converging to the idea of learning objects, which must specify their contents using both descriptive metadata and more semantic information. The former contains generic information such as authors, type of media used, whereas the last describes pedagogical objectives, what the learner will understand or be able to accomplish upon completion of the learning, some mechanism of evaluation to measure whether or not the goal was achieved and a

description of the content. These standards allow right now the definition of platforms for sharing learning resources, thus supporting the creation of a collective intelligence [22]. However, these standards do not fully take into account the semantics of the components and do not precisely describe the composition of the learning objects.
Hypermedia systems also address the issue of adaptation to end-users. Adaptive hypermedia learning systems use the results of the research undertaken in the field of Intelligent Tutoring Systems (ITS) and of the hypermedia [2], keeping the strengths and avoiding the weaknesses of each approach. ITS-style interests look to replace teachers by intelligent computer programs that automatically select and sequence learning objects for students. ITSs control the overall process of training. This global control forces the authors to define courses completely and dedicate to a very limited public. Learners feel very constrained. The hypermedia exploits the multi-media nature of their resources. They propose, on the opposite, to leave learners free navigation during the course. The major disadvantage is that it is impossible to control the process of training. These two approaches are excessive.
The adaptive hypermedia learning systems try to define the medium between constraining users, limited to actions necessary to achieve teaching objectives and the complete freedom of learning. InterBook [21], one of the first examples of adaptive hypermedia, is dedicated to computer programming training. It proposes to navigate in a course using adaptive annotations (showing the educational status for each target: "ready for training", "recommended", "not ready"). It introduces a domain model (ELM-ART)[4]. Multibook [13] concentrates on user adaptivity with a focus on multimedia elements. It proposes to model users using four dimensions (learning aim, background knowledge, teaching method, contents type). These systems have good functionalities but this approach is difficult to scale.

3 Overview of our proposal

This section presents the logical architecture of our system. It is based on the reference model for the adaptive hypermedia applications: AHAM (for Adaptive Hypermedia Application Model) [15], which is an extension of the Dexter hypermedia reference model.
The division into Domain Model (DM) and User Model (UM) provides a clear separation of concerns when developing an adaptive hypermedia application. After a presentation of the DM, we will describe the UM.
To present our approach, we illustrate in Fig. 1 the three levels of modeling we have defined.

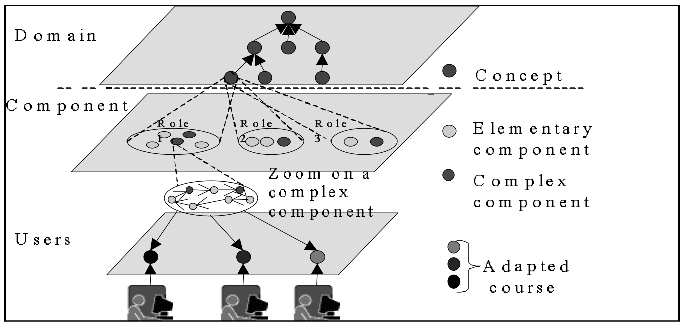

Fig. 1. Modeling levels

3.1 The Domain Model

Our approach uses an ontology to represent all concepts from the knowledge domain. The ontology contains a hierarchical description of important concepts in the domain. Thus, additional relations between concepts may be defined using rhetorical relationships.

Let the domain model be a graph G = <C, A>, where C are nodes representing concept from the domain model and A are arcs representing relationships between two concepts. There are two kinds of possible relationships.

3.1.1 Hierarchical relationship

We define a hierarchy T_i for each specific domain (Fig. 2) (Computer Science, Biology, Electronics, Physics, etc.). This one is defined using the "broader/narrower" relationship. Concept A is broader than concept B whenever the following holds: in any inclusive search for A all items dealing with B should be found. Conversely B is narrower than A [5].

All concepts inside the hierarchy T_i, except root concept, must have a relationship “is-narrower-than” with one and only one concept. All concepts insides the hierarchy T_i can have zero or many "is-broader-than" relationships. The root in each T_i hierarchy is the most general domain concept and the leaves are the most specific domain concepts, i.e. nodes having zero "is-broader-than" relationships.

3.1.2 Rhetorical Relationship

Two domain concepts have a rhetorical relationship if this one exists independently of how both concepts are developed. There is a set of predefined rhetorical relationships took from [10]; however, the system administrator can extend these ones.

a) *Antithesis*: concept A is an antithesis to concept B. Conversely concept B is an antithesis to concept A.

b) *Background*: concept B is a knowledge that provides facilities to understand concept B.

c) *Contrast*: concept A is an alternative theory for concept B. Conversely, concept B is an alternative theory for concept A.

d) *Extend*: concept A develops and adds new elements to concept B. Conversely, concept B is extended by concept A.

e) *Restatement*: concept A is a re-expression of the Concept B. Conversely, concept B is re-expressed by concept A.

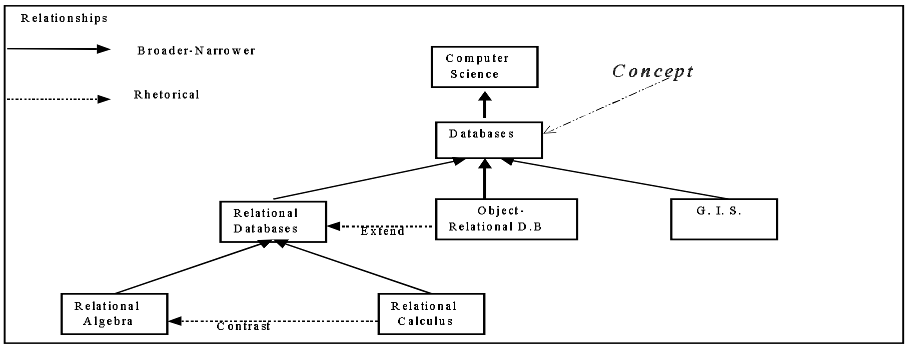

Fig. 2. Domain model fragment for Computer Science

An important point designing our system is to choose an adequate formalism for its representation. We have chosen to use a Description Logic (DL). DL languages are one of the most popular knowledge representation systems [11]. The interest of this formalism is that it offers a good balance between simplicity and expressive power. Besides it offers some reasoning mechanisms such as automatic concept classification or subsumption.

We use DL to represent knowledge at different levels: the domain concepts level, the user model and the component semantic. The principal advantages of this approach is to supply pedagogical helps at authoring time, to facilitate the search of components using a high-level query language and to simplify the add of a new component by giving possible relationships between this one and the existing ones.

3.2 The User Model

Adaptive learning systems may adapt contents to the learners experience, knowledge, goals, or preferences [4, 14]. We have considered the last three aspects in the design of our User Model (UM).

The UM is an overlay model [7], which contains individual information about each learner. Some of this information is given statically by users (e.g. graphical preferences) while some others can be captured or modified dynamically by analyzing their behavior. For each learner we maintain an evaluation of his/her level relative to each concept. A UM can be define with the couple: *UM =<preferences, domain-level>* with *preferences = {<attribute, value>* and *domain-level ={<learner, role, concept , educational-state>}*; where *learner* corresponds to the user Id, the *role* is an optional link between a concept and an educational component (e.g. introduce, define). A *concept* is taken from the domain-model, *educational-state* is one value in ('not-visited', 'visited', knowledge-level) and *knowledge-level* is a level among the set {'very low', 'low', 'medium', 'high', 'very high'}.

If we have no information about the user, stereotypes are used. These ones are predefined by system administrator taking into account the student population characteristics.

4 The Educational Component Model

This section presents the formal definition of an educational component and the different operators that allow the construction of more complex components. The semantic of these operators is described to determine the rules of coherence when the components are combined.

4.1 Educational Component

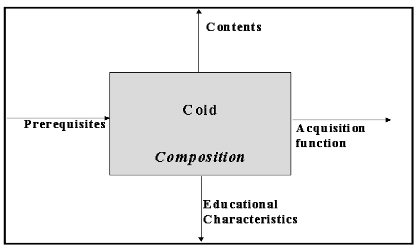

Fig. 3. An Educational Component

An Educational Component is a unit of composition with an interface providing information about requirements for its use, coupling, or replacement during the technology-based learning. This unit can be used independently or for composition by third parties. A component can be primitive (atomic) or composed (structured).

It should be noticed that the component granularity is fixed by its author and that we do not define any constraint on this even if a fine granularity will imply a better reuse of the component.
A course followed by a learner corresponds to the instantiation of a component according to his/her profile.
A component is described by a set of Metadata (Fig. 1):
Component-Metadata=<Coid, Educational-Characteristics, Composition, Semantic> with the following definition:
Coid is the component unique identifier in Database D.
Educational-Characteristics is a set $\{M_i\}$, i = 1...k. where M_i = <tag, value>. This set is used to describe non-functional properties for the components. Our approach uses Learning Object Metadata [16] (e.g. <"title", "Relational Databases">, <"author", "Dupont").
Composition is null if the component is primitive or is an acyclic directed graph denoted by a canonical expression (see Fig. 4) if the component is structured.
Semantic is defined by the tuple <Contents, Input, Output> described in the following.

4.1.1 The Component Contents

A component can play several roles in a concept and it can be attached to several concepts. It means that the component develops one or several aspects of the concept. *Contents = {Rc}*, where *Rc = <Coid, {role}, concept>*. The set of roles is included in the set R={"analyze", "apply", "compare", "define", "demonstrate", "describe", "evaluate", "experiment", "history", "illustrate", "introduce", "summarize"}. These roles are based on the Educational Ontology [12].

Examples: <C_1, {"define"}, "Data-Model" > , <C_2, {"define", "evaluate"}, "Data-Model ">, <C_3, {"introduce"}, "Data-Model ">

4.1.2 The Component Input

A prerequisite relationship is a relation between one component and one domain concept. It means that, during the learning process, a user must know this concept before using the component. Each component C_i has a set of prerequisite relationships represented by the tuple:
Rp = <Coid, {role}, concept, knowledge-level> where *knowledge-level* take one of the following values: {"very low", "low", "medium", "high", "very high"}.
Example: The relation <C_2, {"define"}, "first order logic", "high"> defines that component C_2 requires preparation at "high" level on "first order logic" concept with "define" role.

4.1.3 The Component Output

The output is defined by an acquisition function. This function maps for each couple (c, r) concept and role defined in the contents of the component C, two possible values: either new tuple is added in the user model <c, r, new-value> or a FAIL value is returned.
For example, if the component C_{20} is validated, the couple ("Data-Model", "define") belonging to the C_{20} contents is added to the user model with the value "medium" (<"Data-Model", "define", "medium">).

4.2 Composing Educational Components

A structured component is an acyclic directed graph whose nodes represent primitive or structured components and the arcs represent a crossing condition.
A course then becomes a set of components interconnected by didactic relations making it possible to keep coherence, to define the order of the topics and to adapt the course. This composition is obtained by applying operators of composition of learning components.

4.2.1 Component Operators

We formalized different operators of composition allowing to build (possibly recursively) a complex component starting from primitive components (or of structured components in the recursive case).

Canonical operators

(i) sequence: (C_i SEQ C_j) learners have to access components C_i and C_j, C_j can be accessed only if C_i is successful;

(ii) parallel: (C_i PAR C_j) learners have to access components C_i and C_j, C_i and C_j can be accessed independently;

(iii) alternative: (C_i ALT C_j) learners have to access component C_i or C_j.

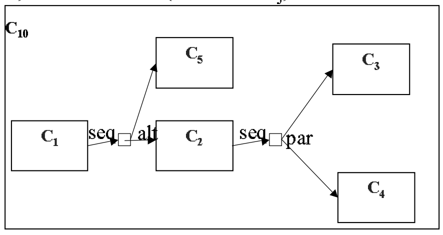

Fig. 4. Example of a structured component

The example shown in Fig. 4 can be expressed with an expression:

C_{10} = C_1 SEQ (C_5 ALT(C_2 SEQ (C_3 PAR C_4)))

In certain cases, there are composition operators, which can not be handled by canonical operators; in consequence we define aggregation operator.

Aggregation operator:

(C_i AGG C_j) is a high-level operation (such as components merging) Aggregation operator is used to merge two components with a merging condition.

Example: if we consider the following component C_{14}=(C_6 SEQ C_7), the expression C_{10} AGG C_{14} ($C_{10}.C_4$ SEQ $C_{14}.C_6$) means that the component C_6 of the structured component C_{14} is attached to the component C_4 of the structured component C_{10}. We then obtain a new component visualized in Fig. 5.

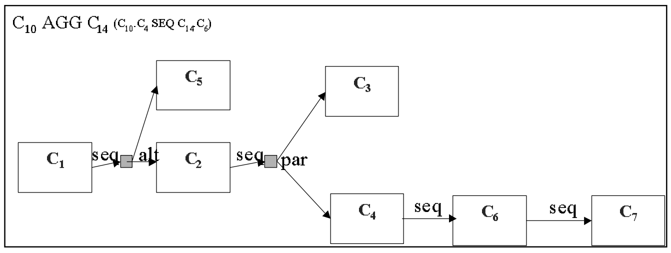

Fig. 5. Example of aggregation

4.2.2 Semantic of the operators

When the components are composed, it is important to process the new entries and exits of the resulting component automatically. The rules that guarantee the coherence of the built components are listed in the following table.

Table 1. Semantic of the operators

	Prerequisite	Acquisition
C_i seq C_j	pre(C_i)	aq(C_i) $\cup$ aq(C_j)
C_i alt C_j	Min(pre(C_i), pre(C_j))	Min(aq(C_i), aq(C_j))
C_i par C_j	pre(C_i) $\cup$ pre(C_j)	aq(C_i) $\cup$aq(C_j)
C_i agg C_j	pre(C_i)	aq(C_i) $\cup$ aq(C_j)

For example, the prerequisite of a sequence of components is equivalent to the first component's pre-requisite.

4.3 Failure Handling

Previous expressions do not handle unsuccessful component access. We introduce the notion of failure to explain what happens when a learner does not validate a component [8]. A component is unsuccessful when its acquisition function returns FAIL. We introduce the negation (noted $\neg$) to write an unsuccessful component. We introduce the following failure expression:
$\neg C_i$ SEQ C_j: after a failure of C_i try C_j;
$\neg C_i^n$: after a failure of C_i try it again n times (almost);
$\neg C_i$ SEQ FAIL: after a failure of C_i propagate the failure to the overall component.
Examples of failure expression:
$\neg C_1^3$: after a failure of C_1 try 3 times again;
$\neg C_5$ SEQ FAIL: after a failure of C_5, propagate failure on C_{10};
$\neg(C_3$ PAR $C_4)$ SEQ C_9: after a failure on (C_3 PAR C_4) try C_9.
If the handling of failure is successful the composition continues as if there was no failure. In the other case, the failure is propagated to the overall component.

5 Authoring and delivering processes

We distinguish two main processes: the authoring process which allows to add new components and the delivering process which allows to select and adapt a course to a specific learner. The delivering process supports two main learning strategies: goals-based and course-based. To the former, the learner chooses a specific concept (or combination of concepts), whereas to the last, the learner chooses a specific component.

5.1 The authoring process

We can distinguish these different cases:
- The addition of a primary component. It consists of describing the different meta-data of the new component (prerequisites, contents, acquisition, learning characteristics);

- The addition of a structured component. The author has to define the canonical expression denoting the composition of the component. The system can infer the description of this structured component using the semantic of the composition operators;
- The search of a component for a specific objective expressed by a conjunction of domain concepts. The system searches if an existing component will satisfy the objective. A component satisfies the objective if its acquisitions are a superset of this one. If the search failed, the author can create a new component satisfying the objective reusing existing components (he/she can use the search tool to select components enforcing a lesser objective) and/or adding new ones.

These tasks are mainly based on inference mechanisms proposed by description logic. For example, a search criterion can be expressed by a DL expression.

Example:
Let suppose that an author wants to find components having concept "Relational Model" with role "define" in their contents: ∃define.Relational-Model
The expressive power of DL allows expressing queries based on concepts and roles.

5.2 The delivering process

In a course-based model, a learner chooses a component in the component base and the system has to adapt this component to the learner's level and preferences. Due to the alternative (ALT) operator, a set S of canonical expressions may be generated from the component definition. The next step consists in removing from S all canonical expressions that do not satisfy the learner level and preferences. If after this last step, S contains several canonical expressions, we can leave the final choice to the user in an interactive way.
Satisfaction can be defined using subsumption in DL. A component satisfies a learner level, if its prerequisites subsume the learner level.

Example:
- ∃define.Relational-Model defines the search expression of the learner and returns the set $\{C_{12}, C_{48}, C_{103}\}$;
- the system expands the component definition of these components and returns the set $\{C_{12}, C_{48}^1, C_{48}^2, C_{103}^1, C_{103}^2, C_{103}^3\}$ where C_i^j stands for canonical expression j of component C_i;
- the system filters this set with the learner level and returns the set $\{C_{48}^2, C_{103}^1, C_{103}^2\}$;
- the system filters with the learner preferences and returns the set $\{C_{103}^1\}$;
- component C_{103}^1 is delivered to the learner.

In a goal-based model, a learner chooses a concept (or a combination of concepts) in the teaching model and the system has to choose in the component base the component, which maps the chosen concept. If this component satisfies the learner level,

the system has just to adapt it to the learner preferences. If the component does not satisfy the learner level, the system has to determine the difference between the learner level and the component prerequisite and then search (recursively) for a component, which corresponds to that difference.
If there is no component corresponding to the chosen concept, the system has to propose some alternatives (for example searching for a more general concept).

6 Conclusion

In this paper, we have presented an educational model for reusable teaching elements. This model allows describing a teaching domain, users and components. We define an ontology to support a general classification by concept, to supply pedagogical helps at authoring time, and to facilitate the search of educational components. Components are described by metadata, which integrate their semantics (prerequisites, acquisitions, contents) and educational characteristics (coming from the LOM standard proposal). Primitive components can be combined using operators, which have a formal definition. Using these models, the system is able to assist the authoring process with a sophisticated search tool and to offer an adaptive delivery of courses.

Our approach extends works on standardization of learning resources (LOM [16], SCORM [19]) in adding a better description of the semantics of learning resources. Besides, these approaches do not propose a model of composition and consequently cannot handle an adaptive delivering process. Intelligent tutoring systems [2] propose semantic models, which are dedicated to a specific domain and to specific learning models. These approaches are difficult to generalize to handle large corpus of courses and cannot reuse existing courses.

We still have several points to investigate. First, we have to better define our authoring and delivering processes using descriptive based-logic formalism. Some experiments are needed to validate our proposal. We are developing a prototype of our system. The current version already supports the component model with the associated operators and a simplified authoring process.

References

1. Belqasmil Y., Bentaleb Z., Benkiran A., R. Ajhoun R.: An author system for the production of adaptative courses. Proceedings of the International Conference on Information Technology based higher Education and Training, ITHET 2001 July 4-6, Kumamoto, Japan (2001)
2. Brusilovsky P.: Adaptive Hypermedia an Attempt to Analyze and Generalize, Multimedia, Hypermedia and Virtual Reality. Lectures Notes in Computer Science, Vol. 1077, Berlin: Springer-Verlag (1996) 288-304

3. El Saddik A., Fischer S., Steinmetz R.: Reusability and Adaptability of Interactive Resources In Web-Based Educational. ACM Journal of Educational Resources in Computing, Vol 1, n° 1 (2001)
4. Brusilovsky, P., Schwarz, E., and Weber, G. ELM-ART: An intelligent tutoring system on World Wide Web. C. Frasson, G. Gauthier and A. Lesgold (Eds.), Proc. of 3rd International Conference on Intelligent Tutoring Systems, Springer-Verlag, Berlin (1996) 261-269
5. Fischer S., Dietrich H.: From Thesauri towards Ontologies?. W. M. el Hadi, J. Maniez, St. A. Pollit (Eds.): Structures and Relations in Knowledge Organization. Proceedings 5th Int. ISKO-Conference, Lille (1998) 18-30
6. Garlatti S., Iksal S., Kervella P.: Adaptive On-Line Information System. Proc. ISAS'99, Orlando, Floride, July 31 – August 4 (1999)
7. Gerhard W., Marcus S.: User modeling and Adaptive Navigation Support in WWW-Based Tutoring. Proc. of the sixth International Conference on User Modeling (1997)
8. Hérin D., Sala M., and Pompidor P.: Evaluating and Revising Courses from Web Resources. Proc. Educational ITS'2002 (2002) 208-218
9. Kekhia W., Bourda Y.: Implementing Learning Object Metadata using RDF. Proc. Ed-Media, Tampere (2001)
10. Mann W.C., Thomson S.A.: Rhetorical Structure Theory: A Theory of Text Organization.Technical Report RS-87-190, Information Science Institute, USC ISI (1987)
11. Baader F., et al.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2002)
12. Ranwez S., Leidig T., Crampes M.: Pedagogical Ontology and Teaching Strategies: A New Formalization to Improve Life-Long Learning. Journal of Interactive Learning Research, Vol 11 N°3-4, AACE (2000)
13. Seeberg C., Steinacker A., Steinmetz R..: Coherence in Modularly Composed Adaptive Learning Documents. Proc. of AH 2000, Springer-Verlag, Heidelberg (2000)
14. Seeberg C., et al.: From the User's Needs to Adaptive Documents. Proc. Integrated Design & Process Technology IDPT'99 (1999)
15. Wu.,H., Houben, G.J., DeBra, P., AHAM: A Reference Model to Support Adaptative Hypermedia Authoring. Proc. of the Zesde Interdisciplinaire Conferentie Informatiewetenschap, Antwerp (1998) 77-88
16. IEEE Learning Technology Standards Committee (LTSC), Learning Object Metadata (LOM), Draft Document. 2001, IEEE P1484.12.
17. The World Wide Consortium. « Extensible Markup Language » XML 1.0 October 2000. http://www.w3.org/XML/
18. http://www.dublincore.org/documents/education-namespace/
19. http://www.adlnet.org/Scorm/scorm_index.cfm
20. http://ariadne.unil.ch
21. http://www.contrib/andrew.cmu.edu/~plb/InterBook.html
22. http://www.educanext.org/UNIVERSAL/