

# Concept-based Courseware Authoring: An Engineering Perspective

Darina Dicheva  
Winston-Salem State University  
United States of America  
E-mail: [dichevad@wssu.edu](mailto:dichevad@wssu.edu)

Lora Aroyo  
Technische Universiteit Eindhoven  
The Netherlands  
E-mail: [l.m.aroyo@tue.nl](mailto:l.m.aroyo@tue.nl)

## 1. Introduction

With the dramatic change that the Web brought to information access worldwide courseware authoring is acquiring a new sense. Web-based courseware could be perceived as a *gateway* providing personalized and task-oriented access to a variety of Web educational materials. An example of this novel approach is AIMS, an intelligent tool that provides support for instructors to build course libraries and define document links to the syllabus, and for learners to identify information necessary for performing course tasks [2]. It supports more efficient course task performance by linking course tasks to a broad range of on-line structured information and by providing domain-related help. To insure this we employ a powerful approach for knowledge classification and indexing based on conceptualization of the course material and subject domain. The latter involves building subject domain ontology and using it for defining course structures and implementing more efficient information search [1]. This is in line with recent advances in the research related to Semantic Web [4] and ontologies [5]. The use of ontologies in AIMS was inspired by the need for shareable and reusable knowledge bases among different courses and authors (instructors). Our approach met its promises; the implemented tool proved to be useful in supporting students in their information retrieval of task-related information [1]. However, domain and course authoring (i.e. building domain and course ontology and linking learning materials to them) turned out to be a very difficult and time-consuming task and required further support. In this paper we present our view on supporting the process of concept-based courseware authoring from engineering perspective, which comes as an elaboration of our approach to knowledge classification and indexing in a Web-based course environment, aimed at assisting students in retrieving, evaluating, and comprehending information when performing open learning tasks.

## 2. Concept-based Courseware Authoring

Concept-based courseware (CBC) generally consists of two main components – concept-based representation of the subject domain covered by the course and a ‘library’ of educational/learning materials/units connected to the domain concepts. The former can be in a form of ontology or a conceptual structure such as a concept map, knowledge map, conceptual graph, etc. The latter can include various types of indexed learning materials such as lectures, examples, slides, exercises, case studies, conference papers, etc., stored both locally and as links to Web documents. In AIMS the links between the ‘library’ and the domain concepts are explicit. We also have an additional conceptual layer - concept-based representation of courses (course structure) over the conceptual representation of the domain. In our further discussion and analysis of the authoring process we use the courseware model adopted in AIMS, which consists of four main models – Domain model (DM), Course model (CM), Learning material & metadata model (LMM) and User model (UM). Authoring of Web CBC, such as AIMS, involves domain-, course-, and library model authoring, which makes the process rather complicated and labor intensive than the process of a ‘standard’ courseware authoring. Apparently this calls for better authoring support. We envisage such support to include semi-automatic performance of some authoring activities, intelligent assistance to the authors in the form of hints, recommendations, etc., as well as supporting collaboration of different instructors in building, sharing, and reuse of domain and course ontologies/conceptual structures.

A central idea in our approach to CBC authoring support is to use the domain ontology/conceptual structure, which captures the semantics of subject domain terminology used by students when searching course-related information. The same ontology can be used by courseware authors to ask authoring-related questions or by the system to perform automatically some authoring activities. To realize this we introduce an additional ontology-based layer to the AIMS courseware authoring architecture, a semantic layer allowing intelligent

assistance to the courseware authors [3]. Consequently, the automation of courseware authoring tasks can use the domain and course ontologies as a basis for formal semantics and reasoning support. In our work we concentrate on the definition of basic authoring functionality with respect to course and domain ontologies represented as concept maps (CMs). Consequently, we consider authoring support for their design, maintenance, integration, sharing, and re-using in terms of manipulating concept maps [7], for example, creating/modifying CMs, comparing, analyzing, mapping and merging CMs, extracting subsets of CMs, etc.

We consider authoring of CBC as a twofold process concerning both content development and content delivery. In the content development phase, the author has to create the courseware content, i.e. to define the subject domain, course, and learning material & metadata models. In our research context this corresponds to defining the domain and course ontologies and linking library documents to those structures. In the content delivery phase, the author has to configure the final presentation of the courseware content considering issues such as using appropriate instructional strategies and adaptation to individual students. Here, the author applies authoring patterns (design steps sequences), integrating available methods and techniques to provide alternatives within the learning material and, in this way, an appropriate adaptation and instructional delivery to the students.

We are interested primarily in the engineering aspects of authoring concerning its life cycle processes/activities used by the author to build up courseware, i.e. create courseware content in the corresponding models and configure ‘content delivery’ for best content presentation. To support this engineering aspect of authoring we first need to identify, understand, and formalize the key content development-related authoring activities within each of the three models (DM, CM, and LMM). For this purpose we started with identifying a set of *primitive* tasks at all the three information layers in concept-based courseware authoring: subject domain, course, and library [3]. Our intention is to use this set, on one hand, as a semantic-rich basis for defining authoring patterns and design requirements for the content configuration and presentation process. On the other hand, we can also use it as an automation basis to implement a corresponding set of tools to support the course development phase. The latter can include automatic creation of some knowledge structures (ontology and course mapping) and templates for both domain and course ontologies (based on recognizing different information patterns within domain ontologies or instructional and adaptation patterns within course structures), which will support authors in their specific authoring activities. It also can include ontology anagement facilities that help authors keep track of

information sources and their possible change or arising of conflicting situations.

### 3. Basic Courseware Authoring Tasks

In this section we present basic domain-related authoring tasks in CBC authoring. Due to space restriction we cannot elaborate on basic authoring tasks related to course and library model authoring. Some abbreviations used throughout the section are given below.

DCM = Domain Concept Map,  
 CS = Course Structure,  
 EML = Educational Metadata Library,  
 DirL<sub>Co</sub> = A set of Directly Linked Concepts,  
 Rel<sub>Co</sub> = A set of Related Concepts,  
 Rel<sub>Doc</sub> = A set of Related Documents.

Common domain authoring tasks include creating a domain, editing a domain, copying a domain, merging domains, etc. These tasks involve primitive concept-maintenance related tasks such as *add/delete/edit* a concept, *create/delete/edit* a link or link type between concepts, etc. Table 1 gives an overview of the primitive domain authoring tasks. At a coarser granularity level, domain authoring tasks include removing all direct links to a concept, removing all segments of a path between two concepts, editing/creating the domain map (ontological domain structure), creating links between the domain structure and the library, etc. Such tasks can be implemented with repetitive calls to atomic operations. For example, the composite tasks ‘*delete all direct links of a given concept*’ or ‘*delete all segments of a path between two concepts*’ can be implemented with a repetitive call to the atomic operation ‘*delete a link in the DCM*’.

It is imperative that the basic authoring operations ensure data consistency. This can be done by performing domain specific checks for conflicts. For instance, when a course author adds a new concept *Co* to the domain concept map DCM, the task *Add (Co, DCM)* is executed. It checks whether the concept *Co* is not already in the map. In case the concept is found in DCM, the result consists of the following: (a) a list of all *Co* synonyms, (b) a list of all concepts connected directly to *Co*, and (c) notification to user(s). These results are fed back to the author who is provided with the opportunity to select manually an ‘add’ or ‘delete’ option for each of the results, as well as to specify the preferred presentation form (textual or graphical). Here are some examples of selections:

1. V (Text, Synonyms);
2. V (Graph, DCM, Matched-Synonyms);
3. V (Text, DirRelco, Relevance %);

**Table 1. Domain authoring tasks.**

Task	Description	Result
Add ( $C_0$ , DCM)	<ul style="list-style-type: none"> <li>▪ Chk (<math>C_0, \exists</math> in DCM) = true</li> <li>▪ perform keyword search on <math>C_0</math> within DCM</li> <li>▪ U (DCM, <math>C_0</math>), notify other authors of adding <math>C_0</math> to DCM</li> </ul>	<ul style="list-style-type: none"> <li>▪ L (synonyms, <math>C_0</math>, DCM)</li> <li>▪ L (DirRel<math>C_0</math>, <math>C_0</math>, DCM)</li> <li>▪ L (NonDirLinks, <math>C_0</math>, DCM)</li> </ul>
Add (L, DCM)	<ul style="list-style-type: none"> <li>▪ Chk (<math>C_1, \exists</math> in DCM) = true, Chk (<math>C_2, \exists</math> in DCM) = true</li> <li>▪ Chk (L, <math>\exists</math> in DCM) = true</li> <li>▪ Chk (<math>C_1</math>-<math>x</math>-<math>C_2, \exists</math> in DCM) = true (indirectly linked with path 'x')</li> <li>▪ U (DCM, L), notify other authors of adding L to DCM</li> </ul>	<ul style="list-style-type: none"> <li>▪ L (x, <math>C_1</math>-<math>C_2</math>, DCM)</li> </ul>
Edit ( $C_0$ , DCM)	<ul style="list-style-type: none"> <li>▪ U (<math>C_0</math>, Weight), U (<math>C_0</math>, Description), U (<math>C_0</math>, Keywords) within different system modules (CS, DCM, EML)</li> <li>▪ Notify other authors of editing C in DCM</li> </ul>	<ul style="list-style-type: none"> <li>▪ V (options to choose): <ul style="list-style-type: none"> <li>- Change (<math>C_0</math>, weight)</li> <li>- Edit (<math>C_0</math>, Description)</li> <li>- Del (<math>C_0</math>, Keywords)</li> <li>- Del (<math>C_0</math>, DCM)</li> </ul> </li> <li>▪ L (DirRelC, <math>C_0</math>, DCM)</li> </ul>
Del (L, DCM)	<ul style="list-style-type: none"> <li>▪ Chk (<math>C_1</math>-<math>x</math>-<math>C_2, \exists</math> for L in DCM) = true</li> <li>▪ Del (L) and U (DCM)</li> </ul>	<ul style="list-style-type: none"> <li>▪ L (x, <math>C_1</math>-<math>C_2</math>, DCM)</li> </ul>
Del ( $C_0$ , DCM)	<ul style="list-style-type: none"> <li>▪ Chk (L, <math>\exists</math> for <math>C_0</math> in DCM) = true</li> <li>▪ Del (<math>C_0</math>, DCM) and U (DCM)</li> </ul>	<ul style="list-style-type: none"> <li>▪ L (DirRel<math>C_0</math>, <math>C_0</math>, DCM)</li> </ul>

4. V (Graph, DCM, Matched\_DirRelco);
5. V (Text, NonDirLinks, Relevance %);
6. V (Graph, DCM, Matched-NonDirLinks).

In [2, 4 and 6] the matching objects are highlighted. This way we aim at supporting author's task activities most efficiently.

#### 4. Conclusion

Authoring of Web CBC is more complicated than the 'standard' courseware authoring and thus needs more automated and intelligent support. Crucial issue related to CBC is the development and maintenance of subject domain and course ontologies by the instructor (author). In this paper we suggest to provide authoring support for CBC by building an authoring reference model on the basis of formalizing basic domain related authoring tasks. This comes as an elaboration of our approach to concept map knowledge classification and indexing in a Web-based learning/training environment aimed at supporting students in retrieving information necessary to perform course related tasks.

The key idea is to formalize the knowledge about the authoring process for CBC with respect to both content development and content delivery, and use it as a basis to construct an intelligent authoring tool. In an attempt to decompose the authoring process we define a set of primitive atomic tasks underlying higher-level authoring

activities related to the three models in the concept-based courseware architecture: subject domain ontology, course structure, and library metadata. This is the first step in our longer-term plan to build ontology of courseware authoring functional concepts (generic authoring tasks), in the same line as introduced by [6]. The intention is to use it as a basis for defining intelligent behavior in a set of support tools for both phases of courseware authoring process.

#### 5. References

- [1] L. Aroyo, *Task-oriented approach to information handling support within Web-based education*. PhD Thesis, University of Twente, PrintPartners Publ., Enschede, 2001.
- [2] L. Aroyo and D. Dicheva, "AIMS: Learning and Teaching Support for WWW-based Education", // *Int. Journal for Continuing Engineering Education and Lifelong Learning*, 11(1/2), 2001, pp. 152-164.
- [3] L. Aroyo, D. Dicheva and A. Cristea, "Ontological Support for Web Courseware Authoring". *Int. Conf. on Intelligent Tutoring Systems (ITS'02)*, Biarritz, France, 2002.
- [4] T. Berners-Lee, "Semantic Web road map". IN, W3C, <http://www.w3.org/DesignIssues/Semantic.html>, 1998.

[5] M. Gruninger & J. Lee, "Ontology: Applications and Design". *Comm.s of the ACM*, 45(2), 2002, pp.39-41.

[6] Y. Kitamura, T. Sano and R. Mizoguchi, "Functional understanding based on an ontology of functional concepts", *Proc. of PRICAI*, 2000, pp 723-733.

[7] J. Sowa, "Building, Sharing, and Merging Ontologies",  
<http://www.jfsowa.com/ontology/ontoshar.htm>, 2001.