Pedagogical Validation of Courseware

Mark Melia and Claus Pahl

School of Computing, Dublin City University, Dublin 9, Ireland
mmelia@computing.dcu.ie

Abstract. A lack of pedagogy in courseware can lead to learner rejection. It is therefore vital that pedagogy is a central concern of courseware construction. Courseware validation allows the course creator to specify the pedagogical rules and principles that courseware must conform to. In this paper we investigate the information needed to automate courseware validation and propose an information architecture to be used as a basis for validation. We then demonstrate an approach to courseware validation in the context of the information architecture presented.

1 Introduction

To produce quality courseware, course creators aim to apply specific pedagogical principles to courseware they create. This can be difficult, especially when there are seemingly more pressing issues for courseware delivery, such as standards compliance and deadlines. Unfortunately, the neglection of pedagogy can lead to a course which confuses, demotivates and/or isolates the learner, ultimately leading to the rejection of the course [8].

Due to the importance of pedagogy in courseware, we must therefore ensure that a course creator's pedagogical principles are always adhered to in the courseware he or she produces. To do this, we propose automated post-construction courseware validation. The literature notes the importance of post-construction course validation or course auditing as an essential part of a holistic course construction methodology [7, 6]. Automated validation of courseware, with regard to some specified pedagogy, safeguards courseware from the possible implications of pedagogical neglect, mentioned above. Automated validation is now possible due to course packaging specifications formally separating learning design from content and the annotation of learning content with metadata.

In this paper we identify the information that is required for courseware validation and how it can be explicitly represented using an information architecture. After this, we use this information architecture to outline a courseware validation strategy and its implementation in a databases course. The paper concludes with a discussion on related and future work.

2 Layered Architecture for Courseware Validation

To define a layered information architecture for courseware validation it is necessary to firstly examine the information available for courseware validation. In

figure 1 we outline this implicit information. This information must then be explicitly represented in an information architecture, which can then be used to validate courseware.

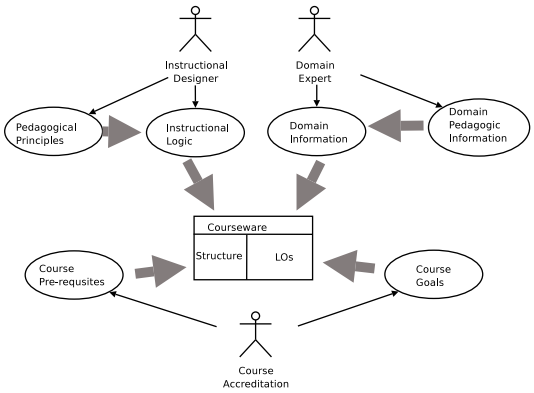

Fig. 1. Course Construction Elements

The "courseware aspects" define the scope, the content, and the design principles of courseware, and are based on the information presented in figure 1. During courseware construction any one of these aspects can unknowingly be compromised by the course creator. By making aspects explicitly available at the post-construction/pre-delivery stage of the course life-cycle, courseware can validated against its courseware aspects.

The Courseware Authoring Validation Architecture (CAVA), allows courseware aspects to be explicitly represented in a layered architecture. This architecture is an extension to the LAOS architecture [3] used to author Adaptive Educational Hypermedia (AEH). Each layer is developed in the context of the lower layers, for example the goal and constraint model is based on a domain model. We use a layered information architecture to keep the domain model free of pedagogical information, and also due to the implicit layering found in the course aspects in figure 1, where domain pedagogic information is layered on domain information and pedagogical principles is used to formulate instructional logic.

Each courseware construction aspect is captured at some layer of the CAVA. The domain model captures the domain to be taught to the learner, and is pedagogy neutral. The Goal and constraint layer allows the course creator to specify the goal of the courseware and domain pedagogic information, such as pre-requisite constraints between concepts in the domain model. The learner model captures the learner expected/pre-requisite knowledge for the courseware.

The courseware itself is represented as a Directed Cyclical Graph (DCG), where each node is a LO and each edge is a potential learner path. Each LO node

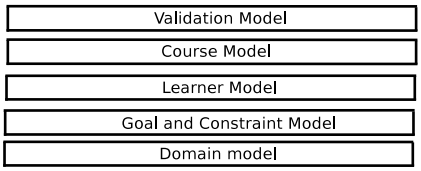

Fig. 2. Courseware Validation Information Architecture

in the courseware is associated with at least one concept in the domain model. This annotation allows for the the formation of concept groupings, grouping LOs according to the concepts they teach. We demonstrate this in figure 3. Concept groupings allows us to discriminate courseware validation between inter-conceptual pedagogy and intra-conceptual pedagogy.

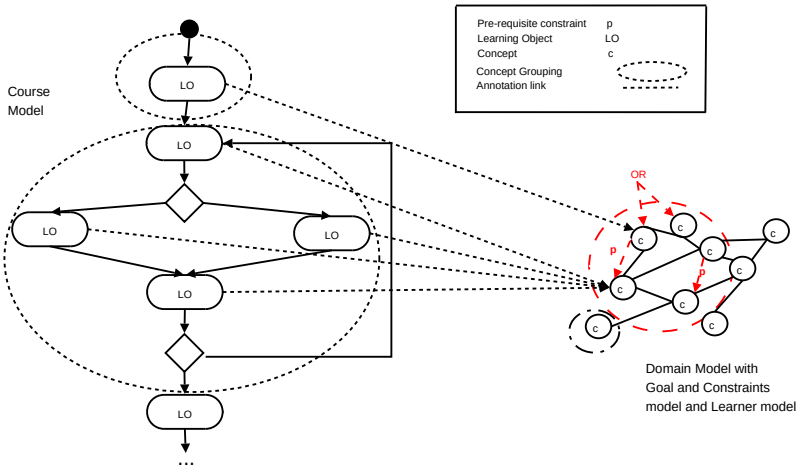

Fig. 3. Grouping LOs according to the concept they cover

The top layer, the validation model layer allows the course creator to express pedagogical rules that the course must adhere to.

3 Courseware Validation Strategy

The architecture we have presented splits pedagogical validation into two parts, validating pedagogical strategy and validating pedagogical rules. Pedagogical strategy is defined by the domain and constraint model. Pedagogical rules are expressed in the validation model.

3.1 Validating Course Pre-requisites

In this section we demonstrate our approach to one type of pedagogical strategy validation, pre-requisite constraint validation, as specified in the goal and constraint model. Pre-requsite validation aims to verify that the learner has any needed pre-requisite knowledge needed for a course element. In validating courseware pre-requisites, we classify the pre-requisite constraints into categories. These categories are "Pre-requisite verified", "Minor ordering error" - simple sequencing fix required, "Warning" - possible for the learner to miss some needed course material, "Error" - learner cannot view pre-requisite material due to the sequencing constraints or the pre-requisite material covered in the course

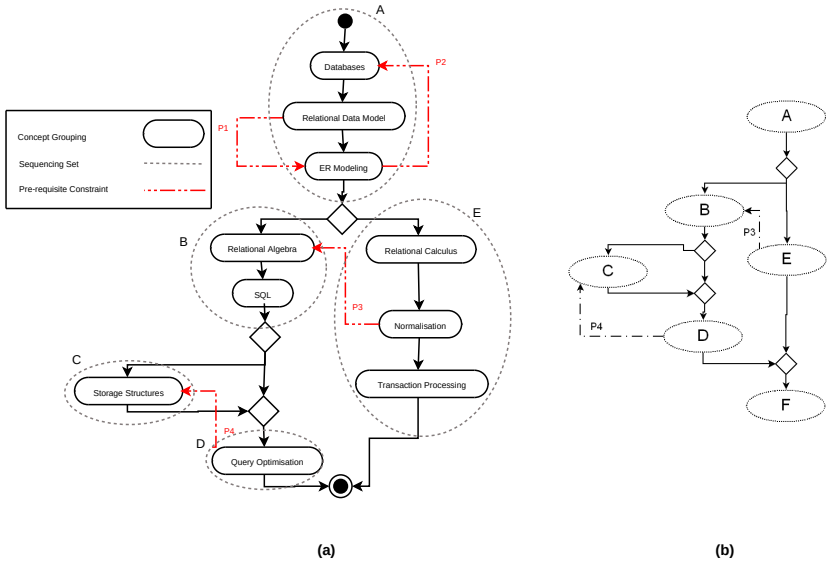

Fig. 4. (a) Database course model with pre-requisites (b) Course divided into sequencing sets

To demonstrate how pre-requisite constraint validation will work and to demonstrate its value we will use a case study course.

In figure 4(a), we have outlined a databases course. The databases course model has been divided into concept groups (black, solid ellipses), the name of each concept group indicates the name of the concept that the concept group refers to in the domain model. The dashed arrows, labeled Pχ, indicate pre-requisite constraints between concept groupings, which are derived from pre-requisite constraints between concepts in the goal and constraint model.

There are four pre-requisites specified for the database course in figure 4(a). The directional pre-requisite arrow points to the concept grouping which is the

pre-requisite, therefore according to P1 "ER modelling" must be understood before the "Relation Data Model".

Concept groupings in the course are further grouped into sequencing sets - sets of concept groups which are no linear (i.e. contain no choices in learner paths). We use letters to denote each sequencing set.

The algorithm firstly locates any pre-requisite constraint where the pre-requisite's source and target are in the same sequencing set. In this case P2 and P1 are identified as such constraints (source and target are in set A). The order of the concept groupings are checked where the target of the pre-requisite is encountered before its source. P2 is satisfied, but P1 is not valid as the source of the pre-requisite is encountered before the target. The P1 constraint violation is classified as causing a "Minor ordering error", while P2 is classified as "Pre-requisite verified".

Pre-requisite constraints can be represented at the sequencing set level, where pre-requisites between concept groupings in different sequencing sets are represented as pre-requisites between sequencing sets. Figure 4(b) depicts the sets derived from the database course outlined in figure 4(a). We also show two pre-requisites from the database course which are concerned with concept groupings from two different sequencing sets, P4 and P3. Pre-requisites, which have not been classified at this stage either fall into the "Warning" category (when there is a possibility through learner choice that pre-requisite concepts will be by-passed) or "Error" category (where it is not possible for the learner to encounter the pre-requisite before the sequencing set which requires it).

When the algorithm checks P4, the target sequencing set is first located (set C). The course ordering constraints are then traversed (every path from set C) in order to find the source sequencing set of the pre-requisite, sequencing set D. In this case set D is found, this means that the pre-requisite constraint could be respected, although there is a possibility that the learner may violate this pre-requisite, depending on the learner's individual path through the courseware. P4 is classified in the "Warning" category as it is possible for the constraint to be satisfied but is subject to learner path choice. If the source of the pre-requisite is not found, which is the case for P3 we can conclude that the pre-requisite will never be seen by the learner and is classified as causing a sequencing "Error", and is categorised accordingly.

3.2 Pedagogical Rule Validation

Pedagogical rules allow the course creator to express desirable or undesirable small-grained pedagogical traits, such as the suitability of LOs at a particular point in a courseware design [4]. Pedagogical rules are expressed in the validation model of the CAVA.

The implementation of the validation rules in the validation model can be captured using a logics-based rule language. In our investigation we have implemented a validation model using the JESS rule language [5]. When validation rules are expressed in JESS, a JESS rule engine can be used to validate a given course against the validation rules.

4 Discussion

From our investigation we have found the literature does not address the diversity of information available for courseware validation. The CoCoA tool developed at Carnegie Technology Education maps a course to a concept map so to reason about learning material in the context of the concept map [2]. Baldoni et. al. have investigated using of logics for courseware representation and then reason about possible problems [1]. In our research we look to further this research by addressing the diversity of information available for courseware validation. In this paper we introduce an information architecture which allows for the explicit representation of the information needs of courseware validation. In order to validate courseware, we must identify the various information elements necessary for courseware construction, and bring these elements together under the context of an information architecture for courseware validation.

In the context of the courseware validation information architecture we were able to develop an approach to courseware validation, which addresses the validation of pedagogical strategy and pedagogical rules. Pedagogical strategy looks at issues such as conceptual sequencing, while pedagogical rules allows the course creator to specify rules which must hold for a course to be deemed valid.

References

1. M. Baldoni, C. Baroglio, V. Patti, and L. Torasso. Reasoning about learning object metadata for adapting SCORM courseware. In *Proceeding of the International Workshop on Engineering the Adaptive Web: Methods and Technologies for personalization adn adaptation in the Semantic Web (EAW204)*. Springer-Verlag LNCS Series, Aug 2004.
2. P. Brusilovsky and J. Vassileva. Course sequencing techniques for large-scale web-based education. *International Journal Continuing Engineering Education adn Lifelong Learning*, 13(1/2):75–94, 2003.
3. A. I. Cristea and A. de Mooij. LAOS: Layered WWW AHS Authoring Model and their corresponding Algebraic Operators. In *Proceedings of The Twelfth International World Wide Web Conference (WWW03), Alternate Track on Education*. ACM, May 20th - 24th 2003.
4. D. Dagger. *Personalised eLearning Development Environments*. PhD thesis, University of Dublin, 2006.
5. E. Friedman-Hill. *JESS in Action: Java Rule-Based Systems*. Manning Publications, Greenwich, Connecticut, 2003.
6. D. Persico. Courseware validation: a case study. *Journal of Computer Assisted Learning*, 12:232–244, 1996.
7. P. V. Rosmalen, H. Vogten, R. V. Es, H. Passier, P. Poelmans, and R. Koper. Authoring a full life cycle model in standards-based, adaptive e-learning. *Journal of Educational Technology and Society*, 9(1):72–83, 2006.
8. J. W. Samples. The pedagogy of technology - our next frontier? *Connexions*, 14(2):4–5, 2002.