Journal of Universal Computer Science, vol. 16, no. 19 (2010), 2862-2881
submitted: 1/3/10, accepted: 29/9/10, appeared: 1/10/10

Model-driven Transformation and Validation of Adaptive Educational Hypermedia using CAVIAr

Mark Melia
(School of Computing, Dublin City University, Dublin, Ireland
mark.melia@gmail.com)

Claus Pahl
(Center for Next Generation Localisation, School of Computing, Dublin City University, Dublin, Ireland
cpahl@computing.dcu.ie)

Abstract: Authoring of Adaptive Educational Hypermedia is a complex activity requiring the combination of a range of design and validation techniques. We demonstrate how Adaptive Educational Hypermedia can be transformed into CAVIAr courseware validation models allowing for its validation. The model-based representation and analysis of different concerns and model-based mappings and transformations are key contributors to this integrated solution. We illustrate the benefits of Model Driven Engineering methodologies that allow for interoperability between CAVIAr and a well known Adaptive Educational Hypermedia framework. By allowing for the validation of Adaptive Educational Hypermedia, the course creator limits the risk of pedagogical problems in migrating to Adaptive Educational Hypermedia from static courseware.
Key Words: Model-driven engineering, Courseware construction, Courseware authoring, Courseware Validation, Adaptive e-learning.
Category: L.2.0, L.3.0, D.2.2, H.1.0

1 Introduction

The authoring of Adaptive Educational Hypermedia (AEH) is a major task for any course creator to undertake. Although advances in this area have been made with the emergence of dedicated AEH authoring tools [Cristea et al., 2007, Dagger, 2006], there is still no way to check AEH for specific pedagogical problems. Courseware validation is a design activity that automatically ensures the presence of certain structural and pedagogical characteristics in constructed courseware such as correct topic sequencing or introducing broader concepts before narrower ones. Courseware validation allows the course creator to minimise the pedagogical problems which the learners must deal with when using immature courseware. Using courseware validation in AEH allows the course creator to test the AEH for specific pedagogical problems that may not be possible otherwise due to AEH's adaptive nature. This reduces the risk of migrating from static courseware to AEH to deliver a course.

Our contribution is a set of model-driven engineering (MDE) techniques to support the authoring of AEH beyond current state of the art tools such as MOT [Cristea et al., 2007] and the ACCT [Dagger, 2006] bulding up on AEH environments such as Interbook [Eklund and Brusilovsky, 1999], the ELM-ART [Weber and Brusilovsky, 2001] or AHA! [DeBra and Calvi, 1998]. MDE is a software development technology framework based on principles of model-based abstraction and automation of central activities that can be utilised to achieve integration and interoperability between different techniques in the AEH authoring context, specifically the integration of different model-driven learning content specification and validation approaches. In order to demonstrate this, we investigate how one AEH specification, the LAOS model as the model foundation of MOT as one of the most recent dedicated authoring environments, can be transformed into the Courseware Authoring Validation Information Architecture (CAVIAr), a set of models designed for the validation of courseware. We introduce mappings from LAOS to CAVIAr and show how the corresponding transformation results can be enhanced to perform model-driven validation. The benefit of model-based interoperability is the significant quality achievement based on reusing established notations and tools in an integrated setting.

The paper is organised as follows. We firstly outline the respective technologies, LAOS first and then CAVIAr [Melia and Pahl, 2007a]. Section 4 steps through AEH validation using CAVIAr. In section 5 we outline how the LAOS model is converted into CAVIAr models using MDE methodologies, allowing for its validation. Section 6 outlines how a domain model can be generated from a domain definition defined using SKOS. Interoperability with external specifications, such as SKOS, and standards is also done using model transformation technology. In section 7, we describe how CAVIAr is used to define a Learning Context Model using the generated Domain Model. Section 8 describes the definition of the CAVIAr validation model. Section 10 looks at related work. We conclude in section 11 outlining our contribution.

2 AEH modelling using MOT and LAOS

Adaptive Hypermedia (AH) looks at adapting hypermedia content to a user model, for example eliminating hyperlinks that are not relevant to a particular user [Brusilovsky, 1996]. Adaptive Educational Hypermedia (AEH) uses AH technologies in an educational context, for example, using a learner's prior knowledge to define an educationally-oriented hypermedia environment to present to the learner. In general, AEH systems operate at a low level of granularity. Among a number of adaptation aspects (learning style, device modality etc.), most commonly AEH systems focus on adapting to a learner's knowledge at the lesson navigation level. This is done by providing recommendations for a pedagogically sound learning path through the educational hypermedia. Examples of such AEH systems are Brusilovsky et al.'s Interbook [Eklund and Brusilovsky, 1999] and ELM-ART [Weber and Brusilovsky, 2001] and DeBra and Calvi's AHA! system [DeBra and Calvi, 1998]. The AHA! system also works at the level of the content unit, adapting the text presented to the learner depending on the learner model.

AEH research concentrates on delivery and the effect AEH-based personalisation has on learning. AEH courses are generally once-off implementations developed by an AEH researcher. One of the main criticisms of AEH is that its authoring is a time consuming and complex activity [Brusilovsky et al., 1998]. Here we look at one AEH authoring system, the My Online Teacher (MOT) system, that attempts to alleviate these problems in AEH authoring. The "My Online Teacher" (MOT) system [Cristea et al., 2007] allows course creators to create adaptive courses using the LAOS conceptual architecture for adaptive hypermedia [Cristea and de Mooij, 2003]. LAOS consists of five layered models, where higher layers are defined in terms of the lower ones:

- **domain model** - the lowest layer organises and structures resources of the learning environment, as well as intrinsic characteristics [Cristea et al., 2007].
- **goal and constraints model** - filters, regroups and restructures the domain model, with respect to an instructional goal used to express educational goals [Cristea et al., 2007]. This is done by specifying the instructional weights of domain model concepts and by ordering the domain concepts.
- **user model** - specifies the user knowledge, interests and learning styles.
- **adaptation model** - defines adaptive rules in terms of the lower layers using LAG, a 3-tier adaptive rule specification [Cristea and Verschoor, 2004].
- **presentation model** - defines delivery environments variables, allowing the AEH to adapt to the delivery environment being used by the learner.

MOT is an AEH authoring environment. Material created using MOT can be delivered using a delivery environment, such as AHA! [Cristea et al., 2003a] or WHURLE [Cristea et al., 2003b], which must be interoperable in terms of the representation languages. Cristea et. al. makes the distinction between static and dynamic elements of the LAOS interchange format [Cristea et al., 2007]. Static LAOS elements are exported from MOT (i.e. MOT implements parts of the LAOS model) through a common language, or lingua franca, known as the Common Adaptation Framework (CAF), which captures the domain model and the goal and constraint model. Dynamic elements, which describe the adaptive nature of the AEH and are captured using LAG. MOT exports to CAF by converting the domain model and the goal and constraint model, which is stored are the MOT database, to the CAF XML specification, this can then be imported by the AEH delivery environment. LAG captures the adaptation rules for AEH. The top level of the 3-tier LAG model is adaptation strategies, which are built on adaptation languages, which, in turn are built on direct adaption rules. In the LAOS context LAG direct adaptation rules are defined in terms of the lower layer models. The LAG direct adaptation rules are IF-THEN or condition-action style rules, defined in a context-free BNF (Backus-Naur Form) style meta-syntax notation[1]. We replicate the LAG definition in BNF in the following listing:

```
PROG          ::= STATEMENT
STATEMENT     ::= IFSTAT | WHILESTAT | FORSTAT | BREAKSTAT |
                  GENSTAT | SPECSTAT |
                  (STATEMENT)* STATEMENT | ACTION
IFSTAT        ::= if CONDITION then (STATEMENT)
WHILESTAT     ::= while CONDITION do (STATEMENT) [TARGETLABEL]
FORSTAT       ::= for RANGE do (STATEMENT) [TARGETLABEL]
BREAKSTAT     ::= break SOURCELABEL
GENSTAT       ::= generalize((CONDITION)*)
SPECSTAT      ::= specialize((CONDITION)*)
ACTION        ::= ATTRIBUTE OP VALUE
CONDITION     ::= enough((PREREQ)+, VALUE) | PREREQ
RANGE         ::= integer
PREREQ        ::= ATTRIBUTE COMPARE VALUE
LABEL         ::= text
TARGETLABEL   ::= text
SOURCELABEL   ::= text_label_a
ATTRIBUTE     ::= GENCONCEPT | SPECCONCEPT
GENCONCEPT    ::= CM_type.concept.attr | CM_type.concept.attr_z
SPECCONCEPT   ::= CM_x.concept_y.attr_z
OP            ::= = | += | -= | .=
COMPARE       ::= == | < | > | in
VALUE         ::= text
```

MOT provides an intuitive interface for designing AEH but does not provide a method for validating the AEH defined against a set of pedagogical and non-pedagogical requirements. The literature notes the importance of post-construction/pre-delivery course validation or "course auditing" as an essential part of a construction methodology [Samples, 2002, Rosmalen et al., 2006]. This is particularly important when defining AEH due to the additional complexity in its adaptive design which makes it near impossible to validate manually.

3 Validation-centric Courseware Modelling using CAVIAr

Model Driven Engineering (MDE) methodologies are traditionally used in the development of software, but have been applied to the development of courseware. In [Melia et al., 2006] a UML Activity Diagram is used to describe a courseware sequencing strategy, this is then transformed, using a model transformation language, into IMS Simple Sequencing [IMS, 2003]. We use the Eclipse Modelling

---

[1] http://wwwis.win.tue.nl/ acristea/MOT/help/LAGgrammar.doc

Framework [Steinburg et al., 2008] to represent the MDE-inspired CAVIAr models as it supports metamodels using ECore and model transformations through the Atlas Transformation Language framework [Jouault and Kurtev, 2005]. We use CAVIAr in courseware authoring to automatically validate courseware for structural and pedagogical concerns:

– inter-conceptual courseware sequencing - pedagogical concerns regarding the sequencing of concepts in courseware [Melia and Pahl, 2007b]

– intra-conceptual courseware sequencing - pedagogical concerns teaching one concept [Ullrich, 2005]

– appropriateness of type of learning material at particular points

– elements of the instructional design in use in the courseware

The aim is to validate the overall courseware consistency. The CAVIAr model allows to identify instructional problems in the courseware prior to delivery. This allows formative evaluation of courseware. AEH validation would allow the course creator to ensure that specific pedagogical and non-pedagogical requirements are satisfied before delivery. To do this CAVIAr must be interoperable with the AEH definition that the courseware wishes to validate, one such AEH definition is the LAOS definition. Validation using CAVIAr is achieved by modelling the courseware construction concerns. The CAVIAr uses modelling structures very similar to that of LAOS, using four modelling layers [Melia and Pahl, 2009]:

– **Domain Model** - a pedagogically neutral conceptual graph. The CAVIAr domain model is used to represent the structure of knowledge that is to be covered in the courseware and beyond. It does this by representing the knowledge as concepts and conceptual relationships.

– **Learning Context Model** - Defines conceptual sequencing constraints and the learner stereotypes, each learner stereotype is defined as having assumed initial knowledge and a course goal in terms of domain model concepts.

– **Courseware Model** - Defines courseware structure and behaviour. The Courseware Model is defined using courseware topics. Topics contain learning resources to be used by the learner. Courseware behaviour is defined using conditions that can be placed on topics that define what learners can access that topic and through topic sequencing constraints.

– **Learning Resource Model** - The Learning Resource Model represents courseware Learning Objects (LOs) and its metadata. Metadata used to describe LOs in CAVIAr is based on the IEEE LOM [IEEE LTSC, 2002].

- **Validation Model -** Is a constraints model which defines valid courseware. The validation model is defined using the Object Constraints Language (OCL). OCL defines validity in the Courseware Model and Learning Resource Model (using Domain Model and Learning Context Model). This allows ensuring that conceptual pre-requisite relationships defined in the Learning Context Model are adhered to in the Courseware Model.

While for instance, the mapping between the domain models of LAOS and CAVIAr are straightforward, tt is important to identify how the CAVIAr facilitates the representation of adaptive courseware as for instance captured in the LAOS user and adaptation model, allowing for the mapping from AEH to CAVIAr. The courseware model defines the courseware structure and the LOs in the courseware. The courseware model is defined using a metamodel; for an excerpt see figure 1.

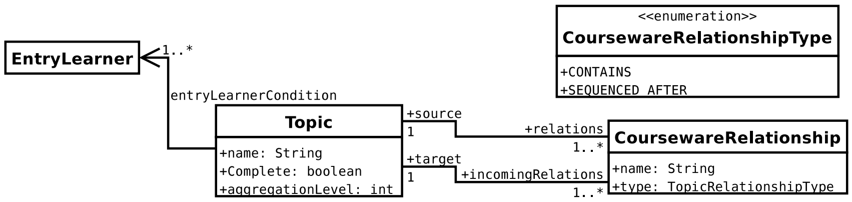

Figure 1: CAVIAr courseware metamodel excerpt

Adaptivity is achieved in a CAVIAr courseware model in two ways - specifying a "SEQUENCED_AFTER" relationship between two topics and specifying an "entryLearner" requirement for a topic. The "SEQUENCED_AFTER" relationship specifies explicit sequencing constraints between topics. The entryLearner requirement allows to place a gate condition on a topic, i.e. the topic is only delivered to learners which satisfy the entryLearner requirement. This is the format to capture input adaptation models.

4 Model-driven Validation of AEH using CAVIAr

In this section, we provide an overview of the CAVIAr principles of model definition, transformation and AEH-oriented courseware validation.

4.1 Model Transformation

Model transformations allow for one type of model to be transformed into another type of model. A model transformation is defined on a model's metamodel, which defines the model syntax and semantics. Metamodels are in turn defined by metametamodels. Figure 2 outlines the basics of model transformation. The transformation is defined between two metamodels and the transformation is invoked on the actual model. AEH and courseware syntax needs to be defined in a metamodel, allowing transformations to be defined between the different specifications allowing for interoperability. This interoperability can be used to transform an AEH model into a CAVIAr courseware model. Once in the CAVIAr format, the AEH definition can be checked for pedagogical faults.

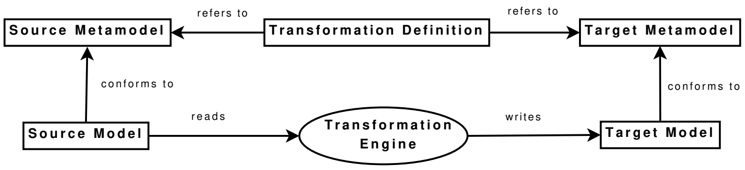

Figure 2: Basic concept of model transformation

4.2 CAVIAr Courseware Model Definition using AEH

In order to validate an AEH definition using CAVIAr a metamodel defining how the AEH is represented in its native form must be defined. This allows AEH courseware to be represented in the metamodelling technical space, in which CAVIAr is defined. Once the AEH courseware definition is represented in the metamodelling technical space model transformations can be defined to convert an AEH courseware definition to the CAVIAr courseware representation for courseware. Once defined the AEH metamodel definition and its transformation definition to the CAVIAr courseware model can be reused. We outline this process in details using the LAOS AEH definition language in section 5.

4.3 Other CAVIAr Model Definitions using AEH

After generating a CAVIAr courseware model from an AEH definition, the other CAVIAr models need to be defined (domain, learning context and validation model) to allow for its validation.

The domain model can be defined as a pedagogically neutral representation of the domain being taught or can be generated using some existing domain model definition [Melia et al., 2005, Holohan et al., 2005, Jovanović et al., 2005, Aroya et al., 2002, Yang et al., 2005, Holohan and Pahl, 2009]. The course creator can also reuse a domain representation defined using the Simple Knowledge

Organization System (SKOS) [Miles and Brickley, 2005]. SKOS is an ontology representation for knowledge in the form of concepts and concept relationships. We outline how the SKOS representation of the domain model is used to generate the CAVIAr domain model in section 6.

Based on the domain model, the CAVIAr Learning Context Model can be defined. Using the Learning Context Model, the course creator can define what the AEH courseware should and should not teach and how and who it should teach it. We outline how the Learning Context Model is created in section 7.

The validation model specifies constraints that must be adhered to in the AEH. For example, all concepts covered in the AEH must be introduced with a motivating example, see the following sample invariant specifying all topics must have a LO of type example:

context Topic
  inv has_example: self.resources→select(oclIsTypeOf(LO))
    →exists(metadata.educational.learningResourceType = EXAMPLE)

To do this, a constraint on the CAVIAr metamodel is defined, which must be adhered to in the CAVIAr models that represent the AEH. This constraint can be further refined by checking the sequencing, ensuring the example is sequenced first in the topic. The Validation Model is addressed in section 8.

The validation is then run using the CAVIAr validation engine, validating the generated models against constraints specified in the validation model. If any of the validation constraints are breached, the course creator is notified and he or she can then rectify them in the AEH.

In the following sections, we describe the technical contributions of this paper. Each section addresses how each of the CAVIAr models are created for the purpose of validating AEH.

5 LAOS AEH Definition to CAVIAr Courseware Model Mapping

In order to validate AEH defined by MOT using CAVIAr, the LAOS model is used to generate CAVIAr models. We define metamodels for LAOS, one looking at MOT's static elements in CAF and the other for its adaptive rules, defined in LAG. We also define transformations from the LAOS metamodel to the CAVIAr metamodel by identifying the relations between the metamodels. In this section, we firstly outline how the elements of the LAOS definition, CAF and LAG, can be represented in the metamodelling technical space. We then outline transformation relations from both CAF and LAG to CAVIAr. Note that the transformations specified here are sample mappings, all model mappings can be customised by the course creator.

5.1 Defining a Metamodel for the CAF Format

The CAF Document Type Definition (DTD) defines two key elements - “domainmodel” and “lesson”. The “domainmodel” is used to define the LAOS domain model in terms of concepts. The “lesson” defines domain model concepts in terms of a LAOS goal and constraint model where the “contents” attribute refers to a domain model concept using a weight and label as defined in LAOS [Cristea and de Mooij, 2003]. The CAF DTD is replicated from [Cristea et al., 2007]:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT CAF (domainmodel?, goalmodel?)>
<!ELEMENT domainmodel (concept+)>
<!ELEMENT concept (name, attribute*, concept*)>
<!ELEMENT attribute (name, contents)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT contents (#PCDATA)>
<!ATTLIST contents weight CDATA "" label CDATA "">
<!ELEMENT goalmodel (lesson)>
<!ELEMENT lesson (contents*, lesson*)>
```

To create a metamodel from the CAF definition, we use the Eclipse modeling framework. In order to automatically generate a metamodel, the CAF XML DTD was converted to an XML schema, as DTD has inferior semantic expressivity. For a generated XML schema to provide a correct metamodel for CAF, the following alterations were required: an explicit link between Link and Attribute (as only an implicit link exists in the DTD based on syntax), a “value” attribute added to CAF elements which contain text, and specified ordered relationships. The generated CAF metamodel is illustrated in figure 3.

5.2 Defining a Metamodel for the LAG Language

We use LAG adaptivity rules and transform them into CAVIAr courseware model restrictions. For this, the LAG language must be defined in the modelling technical space. We have defined a limited metamodel for the LAG abstract syntax in figure 4. This metamodel represents LAG in the modelling space by parsing a LAG rule and creating a LAG model. The LAG model is then transformed and integrated into the CAVIAr model created using the CAF format in section 5.3.

5.3 Transforming LAOS CAF to CAVIAr Courseware Model

Once the CAF ECore metamodel is defined, the transformation between the CAF format and CAVIAr courseware metamodel can be defined using a model transformation language [Jouault and Kurtev, 2005, OMG, 2005].

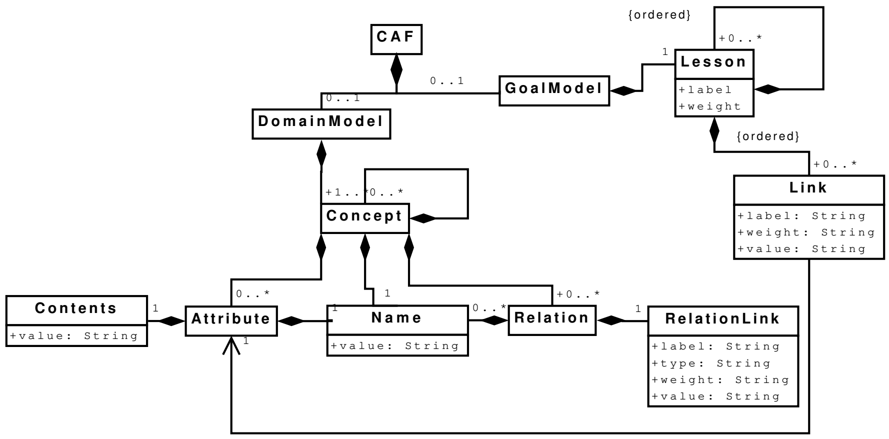

Figure 3: CAF Metamodel defined using ECore

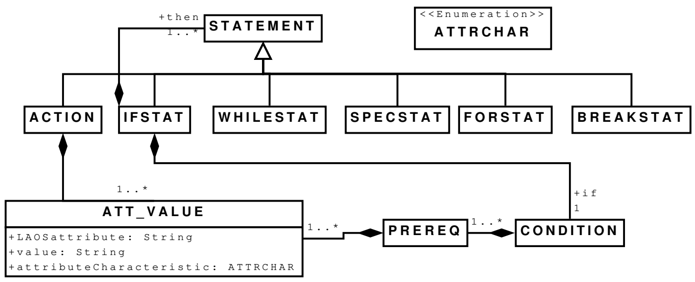

Figure 4: LAG defined as ECore metamodel

5.3.1 Generating Courseware Structure

A courseware model is not explicitly defined in the CAF model, but can be derived using the CAF domain model. In LAOS, the domain model contains the educational content to be delivered. Therefore, each of the concepts in the domain model are also topics in the courseware model. In our transformation, we create a new topic for each of the LO types. Wwhen defining adaptive behaviour in LAOS, it is possible to sequence not only concepts, but also concept attributes. In CAVIAr, only the sequencing of topics is possible, therefore all courseware elements which can be sequenced are divided up into courseware topics.

In defining the transformation from the CAF model to the CAVIAr courseware model, we specify a 1:1 relation between the concepts in CAF and the CAVIAr courseware topics. Concepts contained in other concepts in CAF are transformed to subtopics in the CAVIAr courseware model.

5.3.2 Learning Object and Learning Object Metadata Generation

In CAVIAr, learning material is typically a learning objects (LO), annotated with metadata. This metadata can be used to determine the suitability of a LO at some point in the courseware. In AEH, the domain model defines what is in the AEH lesson. The domain model not only defines a conceptual structure of the AEH course, but also defines the learning content. In LAOS, learning content is defined in concept attributes. To generate LOs from LAOS, we transform each conceptual attribute to a LO. The LO metadata is automatically derived for each LO generated, using the attribute type (e.g. title, conclusion). The concept is used to annotate the LO with the domain model concept that the LO covers. This links the LO with the CAVIAr domain model, see section 6.

5.4 Transforming LAOS LAG to CAVIAr Courseware Model

LAG rules define adaptivity in LAOS. The adaptive behaviour is represented in the CAVIAr courseware model through restrictions on topic sequencing and on learner profiles that can access a topic. This adaptivity is defined using modelling constructs, such as defining a sequencing relationship between topics. Transformation rules can be defined from the LAG metamodel to the CAVIAr metamodel. Below is an adaptive rule commonly used in LAOS to define AEH. This rule specifies that when a particular part of the domain model is accessed, a different part of the AEH should be made available. We also describe the transformation which converts LAG rules to CAVIAr. The rule states that if a domain model's concept title is accessed, then the text for that concept is shown. This type of LAG rule is made up of two different parts, an IF condition and an action. The condition and action are composed by checking (condition) and then setting (action) a characteristic of a domain model concept's attribute in LAOS. The condition checks the attribute "title" for domain model concepts has been accessed - "access" being the characteristic. In turn, the action sets the LAOS "text" attribute to be shown - "show" being the characteristic being set. This rule is parsed and creates an instance of the the LAG metamodel - a LAG model - as illustrated in figure 5. When a LAG model has been constructed for the sequencing rule

```
IF (DM.Concept.title.access == 'true') THEN
    (DM.Concept.text.show == 'true')
```

the rule can be transformed into the CAVIAr courseware model based on a transformation mapping from the LAG metamodel to the CAVIAr metamodel. This transformation states when the DM.Concept.title attribute is accessed, the show the DM.Concept.text attribute. It maps this rule to a CAVIAr courseware model where each attribute in the LAG condition and action is a courseware topic. The topic mapped to the title attribute is the source of a "SEQUENCED_AFTER" CoursewareRelationship where the target is the topic mapped to the text attribute. "SEQUENCED_AFTER" is a construct that allows expressing topic sequencing constraints – for instance, a more specialised domain concept needs to be presented after a more generic one, giving rise to sequencing constraints for topics. We have formalised this in figure 5.

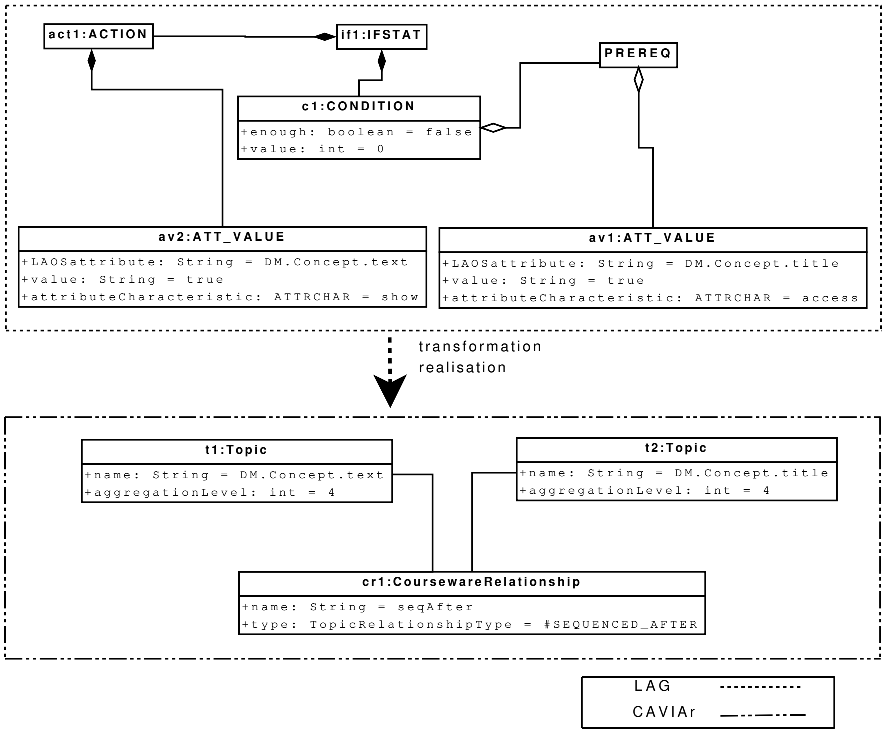

Figure 5: Transformation of LAG model to CAVIAr courseware model

6 CAVIAr Domain Model Mapping

Often, where courseware validation takes place, the course creator will use an existing domain model (section 4.3). A domain model is firstly identified and then transformed into a CAVIAr Domain Model. Any external domain model can be used in CAVIAr as long as the following conditions are met: the domain model has a formal metamodel or abstract syntax and the domain model's metamodel can be mapped and transformed to the CAVIAr Domain Model metamodel. For the domain model to be used in CAVIAr it is transformed from its native representation into CAVIAr. The following assumptions are made about the domain models used in CAVIAr:

- The domain model has a taxonomy of concepts, the scope of concepts becomes more specialised or generalised by moving up or down the taxonomy.
- The domain model is self-contained, in that the domain model does not reference any external resources.
- There is no pedagogical information in the domain model.
- There are no circular dependencies or relationships within the domain model.
- There are no orphan concepts in the domain model.

To illustrate interoperability at a conceptual level, we define mappings to the SKOS standard [Miles and Brickley, 2005]. SKOS operates in the ontological technical space, whereas CAVIAr operates in the metamodelling technical space. Mappings between these two spaces can be seen in the Ontology Definition Metamodel (ODM), where OWL classes are defined as MOF classes [Gašević et al., 2006]. ODM allows for SKOS to be represented in the metamodelling space. The primary component of a SKOS ontology is the *skos:concept* – an instance of *owl:class*. Concepts are related by two types of semantic relationships; associative relationships (*skos:related*) and taxonomic (*skos:narrower* and *skos:broader*). The latter are used to define when a *concept* has a narrower or broader scope than another.

SKOS can be mapped to the CAVIAr Domain Model abstract syntax:

- CAVIAr *DomainModel* is mapped to RDF graph which hosts SKOS *ontology*
- CAVIAr *Concept* is mapped to the *skos:concept*.
- CAVIAr *NARROWER ConceptRelationship* is mapped to the *skos:narrower* relationship, while the inverse of *NARROWER* from target to source can be mapped to SKOS *skos:broader* relationship
- CAVIAr *ConceptRelationship* of type *RELATED* is mapped to the SKOS relationship *skos:related*
- The set of names for a given concept, defined as synonyms and as the concept name attribute, can be mapped to the *ThesaurusTerm* in SKOS. Those terms that are related to the *skos:concept* through the *skos:altLabel* are a *Synonym* in the CAVIAr Domain Model, while the *skos:prefLabel* is used to determine the CAVIAr concept's *name* attribute.

Based on these mappings, a transformation can be defined from the metamodel defining SKOS to the CAVIAr domain metamodel. This transformation converts a SKOS domain model into a CAVIAr domain model. Again, the course creator has the option of adapying the Domain Model. For the domain model to be most effective for validation purposes, it should be integrated with LOs used in the courseware – done by using the ontology used to classify the courseware LOs [Melia and Pahl, 2009]. We assume here that the domain model used in CAVIAr is a superset of the domain model used in the LAOS AEH definition. LOs derived from the AEH definition are associated with concepts in the newly generated Domain Model.

7 CAVIAr Learning Context Model Definition

The CAVIAr Learning Context Model is defined in terms of the CAVIAr Domain Model. The Learning Context Model can be used to define pre-requisite constraints and can also define learner stereotypes for the course. Each learner stereotype is defined in terms of its goal concepts and its presumed knowledge upon entering the courseware. Using learner stereotypes, different learner groups may be distinguished. Learner stereotypes could be used to distinguish ability levels in a group of learners or to distinguish academic/professional background.

The course creator defines the Learning Context Model using domain knowledge elements in terms of a knowledge level and knowledge type. In the context of model mappings between for instance LAOS/MOT and CAVIAr, the learning context model captures input learning model knowledge as validation input and it allows to add further/missing elements.

7.1 Defining the Learning Goals for Learner Stereotypes

The goal concepts are derived from the course specification. This is done by identifying the concepts that capture the learning goals for each learner stereotype and defining the knowledge type and level required for that concept. The course creator must take the information supplied in the course descriptor and formulate CAVIAr knowledge elements. This is done as follows:

1. Identify the key knowledge concepts in the course descriptor (examples are domain concepts).
2. Identify type of knowledge required (examples are conceptual knowledge, skills or both).
3. Identify the level of knowledge required for each knowledge concept and type (examples are comprehension, application or analysis.
4. Map elements to knowledge elements in the CAVIAr Learning Context Model.

7.2 Defining the Presumed Learner Stereotype Knowledge

The presumed learner stereotype knowledge allows the course creator to define the knowledge (in terms of Domain Model concepts) he or she expects a particular learner stereotype to have upon starting. The course creator must again define knowledge type and level for the presumed knowledge concepts. This involves assessing the knowledge a typical learner in a particular stereotype will have for a specific Domain Model concept. Presumed knowledge is added to the CAVIAr model through a relationship between the learner stereotype's *PresumedKnowledge* class and a *KnowledgeElement* that references the Domain Model concept. This tags the concept as presumed knowledge for a specific learner stereotype.

8 CAVIAr Validation Model Definition

The CAVIAr Validation Model is a constraints model that defines explicit constraints on the courseware and resources model in terms of domain and learning context. We have divided the model definition process for AEH into three parts: identify an instruction design theory, derive instructional constraints, formulate instructional constraints in OCL. We examine the following: determine what instructional design the course creator used, derive informal instructional constraints from the instructional design of choice and form CAVIAr-based constraints from these, and defining instructional constraints in OCL.

Determine the Instructional Design. Instructional design theory defines how knowledge should be taught to a learner given a learning scenario. The course creator defines the instructional design to ensure courseware is as effective and efficient as possible in facilitating the learner from an initial knowledge state to their learning goal. A course creator may apply their own instructional design or may apply one of the many instructional design theories found in the literature [Reigeluth, 1999, Gagné et al., 2005]. If the course creator uses an instructional design from the literature he or she may be able to reuse an existing validation model.

Deriving Instructional Constraints from an Instructional Design. In validating courseware, the course creator looks to confirm that an instructional design theory has been applied correctly in courseware. In order to validate the courseware against an instructional design theory, the course creator must break down the theory into instructional constraints.

An instructional design theory can be defined as a set of instructional principles. Reigeluth demonstrates this by summarising an instructional design theory as a set of instructional principles. For an instructional design theory to be used correctly, the course creator must adhere to these principles. In seeing instructional principles as requirements, the course creator defines instructional constraints for the principles, which must be true for the correct application of the instructional design theory. Once the instructional constraints have been defined, the course creator specifies each in terms of the CAVIAr metamodel.

Formulating Instructional Constraints in OCL. At this stage the instructional design has been broken up into instructional constraints that are defined in terms of CAVIAr. These constraints must then be used to formulate a CAVIAr Validation Model using OCL. The instructional designer takes each of the constraints and defines them in terms of the CAVIAr metamodels and expresses them formally in OCL. This OCL is then used to validate the AEH represented using CAVIAr.

9 Evaluation and Discussion

We have carried out a number of case-study evaluations, where instructional design models represented in different formats have been defined in terms of a CAVIAr validation model. Tests have demonstrated that CAVIAr is expressive enough to capture the essence of instructional design theories as well as core personalization strategies. By expressing well-established instructional design theories as CAVIAr constraints [Holohan et al., 2005] and using concrete courseware systems such as a database learning environment as experimental environments [Murray et al., 2003, Kenny and Pahl, 2005], we have shown our approach to be capable of checking courseware for real-world problems. The most common problem is that of domain concept mismatches in the sequencing of courseware content.

It is worth noting that here are different roles with different expected knowledge/skills involved here. The context of this paper is courseware content componentisation. Thus, the content creator is here not the same as the composer and validator, i.e. different skills levels are involved.

Adaptation and validation as activities both relate to the operationalization of instructional principles. Our approach is a post-construction, pre-delivery technique. We have brought a dynamic monitoring of for instance adaptivity principles forward to a pre-delivery stage. The question that would arise here is whether (adaptivity-related) validation as described here can be integrated into the adaptivity design stage. The key contribution of our validation technique is the formalisation of constraints and their formal analysis using a constraints language. Consequently, in a fully integrated AEH development environment, a constraints-compliant AEH construction would be possible, if due consideration is given to the expected skills as discussed above.

10 Related Work

We look at research on validation of courseware in general – we are not aware of validation research that addresses exclusively AEH needs [Brusilovsky, 1996].

The use of logics for validating courseware has been investigated by the ALICE project at the University of Torino [Baldoni et al., 2004a]. The project looks at a range of course construction activities including course verification [Baldoni et al., 2004b] and construction [Baldoni et al., 2004a]. Courses are represented using action theory, where each course component is an action with pre-conditions and post-conditions. Traditional AI reasoning, such as temporal projection, is used to check that pre-conditions are respected. Curricula models are formalised using temporal constraints and are independent of the learning resources, operating at the knowledge level. Using linear-time temporal logic to represent temporal constraints allows for the verification of the curriculum. The motivation is to validate Italian student study plans. Each year students may alter their study plan. These alterations may have adverse effects to the students overall learning goal. Verifying compliance means that the curriculum respects the constraints at the knowledge level represented using the curricula model, constraints at the resource level represented using action theory and that the curriculum allows the learner to reach some goal state. In order for logics to be a viable method for curriculum construction and verification the logic coding would have to be hidden from the user. Baldoni et al. look primarily at the sequencing of modules. Our work concentrates on the sequencing of topics within a degree module, the granularity level is smaller. Our work also looks to not only validate the sequencing of topics in courseware, but other instructional concerns.

The Concept-based Courseware Analysis tool can be considered as an authoring support tool that uses two types of validation; typed items and advanced concept roles [Brusilovsky and Vassileva, 2003]. Typed items allow for validation of the positioning of particular teaching operations. Advanced concept roles defines a LO with regard to pre-requsite knowledge and knowledge outcome. The tool checks only sequential learning paths by simulating a learner's progression through the material and indicates any "content holes" where the learner encounters a LO without having the necessary pre-requsite knowledge. It was prototypical in nature and as such there is no consideration for TEL standards. Pedagogical problems are defined by the developer, there is no facility for the course creator to manipulate the validation rules.

11 Conclusion

AEH authoring, particularly in the context of composite courseware based on reusable learning objects is a challenging activity due to the complexity of the overall courseware and the variety of individual objects involved. In this paper we have described courseware validation as a method for course creators to minimise the risk involved in creating and deploying AEH. The CAVIAr has been introduced in the AEH context, as a way for course creators to test the AEH developed for specific pedagogical concerns.

New technologies auch as learning objects or AEH authoring are initially often illustrated through prototypes. Our focus has been on model-driven engineering (MDE) technologies used here not to integrate learning objects, but to integrate different authoring approaches for AEH, specifically to enable interoperability between LAOS and CAVIAr as examples of AEH design and validation activities, respectively. We have outlined the application of MDE technologies and methodologies, provided model mappings from LAOS to the CAVIAr, and detailed an implementation infrastructure with which the conversion from LAOS to CAVIAr can take place. An example of another interoperability benefit here is the domain model mapping between SKOS and CAVIAr that we used.

We have outlined how MDE offers a generic approach to AEH interoperability, where interoperability can be achieved when a metamodel is defined for the AEH technology in use and transformations between the metamodels are implemented. It has turned out that some modelling concerns, such as domain or courseware modelling, are common to different individual approaches (showing some convergence in the research area) and thus allow, given a common metamodel, integration through transformation. The methodology outlined is also highly customisable, all AEH to CAVIAr mappings can be changed to reflect the course creator's opinions on metamodel relationships.

Extensions of our approach can be considered as part of our future research. Sicilia observes that the rationale used to define an IMS Learning Design specification is not captured [Sicilia, 2006]. He suggests defining courseware and its rationale using a common point of context. Models as constraint structures can be used to guide the CAVIAr courseware design process or generate tentative courseware designs automatically.

Acknowledgement

The authors would like to thank the reviewers for their invaluable comments. This material is partly based upon works supported by Science Foundation Ireland (Grant 07/CE/I1142 - Center for Next Generation Localisation CNGL).

References

[Aroya et al., 2002] Aroya, L., Cristea, A. I., and Dicheva, D. A Layered Approach towards Domain Authoring. In *Proceedings of The International Conference on Artificial Intelligence (ICAI02)*, pages 615–621. CSREA. 2002.

[Baldoni et al., 2004a] Baldoni, M., Baroglio, C., and Patti, V. Web-Based Adaptive Tutoring: An Approach Based on Logic Agents and Reasoning about Actions. *Artificial Intelligence Review*, 22(1):3–39. 2004.

[Baldoni et al., 2004b] Baldoni, M., Baroglio, C., Patti, V., and Torasso, L. 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)*, pages 4–13. Springer-Verlag LNCS Series. 2004.

[Brusilovsky, 1996] Brusilovsky, P. Methods and techniques of adaptive hypermedia. *Methods and Techniques of Adaptive Hypermedia*, 6(2-3):87–129. 1996.

[Brusilovsky et al., 1998] Brusilovsky, P., Eklund, J., and Schwarz, E. Web-based education for all: a tool for development adaptive courseware. *Computer Networks and ISDN Systems*, 30(1–7):291–300. 1996.

[Brusilovsky and Vassileva, 2003] Brusilovsky, P. and Vassileva, J. Course sequencing techniques for large-scale web-based education. *International Journal Continuing Engineering Education and Lifelong Learning*, 13(1/2):75–94. 2003.

[Cristea et al., 2007] Cristea, A., Smits, D., and deBra, P. Towards a generic adaptive hypermedia platform: a conversion case study. *Journal of Digital Information*, 8(3).

[Cristea and de Mooij, 2003] Cristea, A. I. and de Mooij, A. 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. 2003.

[Cristea et al., 2003a] Cristea, A. I., Smits, D., and de Bra, P. Writing MOT, Reading AHA! - converting between an authoring and a delivery system for adaptive educational hypermedia. In *Proceedings of The Third International Workshop on Authoring of Adaptive and Adaptable Educational Hypermedia at AIED05*. 2003.

[Cristea et al., 2003b] Cristea, A. I., Stewart, C., Brailsford, T., and Cristea, P. Evaluation of Interoperability of Adaptive Hypermedia Systems: testing the MOT to WHURLE conversion in a classroom setting. In *Proceedings of The Third International Workshop on Authoring of Adaptive and Adaptable Educational Hypermedia at AIED03*. 2003.

[Cristea and Verschoor, 2004] Cristea, A. I. and Verschoor, M. The LAG Grammar for Authoring the Adaptive Web . In *Proceedings of The International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 1*, pages 382–386. 2004.

[Dagger, 2006] Dagger, D. *Personalised eLearning Development Environments*. PhD thesis, University of Dublin. 2006.

[DeBra and Calvi, 1998] DeBra, P. and Calvi, L. AHA!: a Generic Adaptive Hypermedia System. In *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia*, http://wwwis.win.tue.nl/ah98/Proceedings.html. Eindhoven University of Technology. 1998.

[Eklund and Brusilovsky, 1999] Eklund, J. and Brusilovsky, P. InterBook: An Adaptive Tutoring System. *UniServe Science News*, 12:8–13. 1999.

[Gagné et al., 2005] Gagné, R., Wager, W., Golas, K., and Keller, J. *Principles of Instructional Design*. Wadsworth, California, USA, 5th edition. 2005.

[Gašević et al., 2006] Gašević, D., Djurić, D., and Devedžić, V. *Model-Driven Architecture and Ontology Development*, chapter The Ontology Definition Metamodel (ODM), pages 181–199. Springer-Verlag. 2006.

[Holohan et al., 2005] Holohan, E., McMullen, D., Melia, M., and Pahl, C. Adaptive Courseware Generation based on Semantic Web Technologies. In *Proceeding of the International Workshop on Applications of Semantic Web Technologies for E-Learning (SW-EL2005) at the Twelveth International Conference on Artificial Intelligence in Education (AIED2005)* . IOS Press. 2005.

[Holohan and Pahl, 2009] Pahl, C. and Holohan, C. Applications of Semantic Web Technology to Support Learning Content Development. *Interdisciplinary Journal of E-Learning and Learning Objects*. Vol. 5: 1-25. 2009.

[IEEE LTSC, 2002] IEEE LTSC. LTSC WG12:Learning Object Metadata. IEEE Learning Technology Standards Committee. 2002.

[IMS, 2003] IMS. IMS Simple Sequencing Specification. Technical Report 03/19, IMS, Global Learning Consortium. Available from http://www.imsglobal.org/simplesequencing/. 2003.

[Jouault and Kurtev, 2005] Jouault, F. and Kurtev, I. Transforming Models with ATL. In *Proceedings of the Model Transformations in Practice Workshop at MoDELS*

2005. 2005.
[Jovanović et al., 2005] Jovanović, J., Gašević, D., Verbert, K., and Erik, D. Ontology of learning object content structure. In *Proceeding of the 12th International Conference on Artifical Intelligence in Education*, pages 322–329. IOS Press. 2005.
[Kenny and Pahl, 2005] Kenny, C. and Pahl, C. Automated tutoring for a database skills training environment. In *ACM SIGCSE Symposium 2005*, pages 58–64. ACM. 2005.
[Melia et al., 2006] Melia, M., Barrett, R., and Pahl, C. A Model-Based Approach to SCORM Sequencing. In *Proceeding of the Sixth Annual Irish Educational Technology User's Conference (EdTech06) - Research Track*. ILTA. 2006.
[Melia et al., 2005] Melia, M., Holohan, E., McMullen, D., and Pahl, C. Ontology-based Adaptive Content Navigation. In *Proceeding of the First International Conference on Methods and Technologies for Learning (ICMTL2005)*, pages 435–440. WIT Press. 2005.
[Melia and Pahl, 2007a] Melia, M. and Pahl, C. An information architecture for validating courseware. In Massart, D. and Colin, J.-N., editors, *Proceeding of the First International Workshop on Learning Object Discovery and Exchange (LODE2007) at EC-TEL2007*, Crete, Greece. CEUR Workshop Proceedings. 2007
[Melia and Pahl, 2007b] Melia, M. and Pahl, C. Pedagogical validation of courseware. In Duval, E., Klamma, R., and Wolpers, M., editors, *Proceedings of the Second European Conference on Technology Enhanced Learning*, number 4753 in LNCS. Springer. 2007.
[Melia and Pahl, 2009] Melia, M. and Pahl, C. Constraint-based validation of adaptive e-learning courseware. *IEEE Transactions on Learning Technology*, 2(1):37–49. 2009.
[Miles and Brickley, 2005] Miles, A. and Brickley, D. SKOS Core Guide. Technical report. 2005.
[Murray et al., 2003] Murray, S., Ryan, J., and Pahl, C. A Tool-mediated Cognitive Apprenticeship Approach for a Computer Engineering Course. International Conference on Advanced Learning Technologies ICALT'2003. Athens, Greece. IEEE Press. Pages 2-6. 2003.
[OMG, 2005] OMG Meta Object Facility (MOF) 2.0 Query View Transformation (QVT). OMG Final Adopted Specification. 2005.
[Reigeluth, 1999] Reigeluth, C. M. *Instructional Design: Theories and Models*, volume 2. Lawrence Erlbaum Associates, Publishers. 1999.
[Rosmalen et al., 2006] Rosmalen, P. V., Vogten, H., Es, R. V., Passier, H., Poelmans, P., and Koper, R. Authoring a full life cycle model in standards-based, adaptive e-learning. *Journal of Educational Technology and Society*, 9(1):72–83. 2006.
[Samples, 2002] Samples, J. W. The pedagogy of technology - our next frontier? *Connexions*, 14(2):4–5. 2002.
[Sicilia, 2006] Sicilia, M.-A. Semantic learning designs: recording assumptions and guidelines. *British Journal of Educational Technology*, 37(3):331–350. 2006.
[Steinburg et al., 2008] Steinburg, D., Budinsky, F., Paternostro, M., and Merks, E. *Eclipse Modeling Framework*. Pearson Education, 2nd edition. 2008.
[Ullrich, 2005] Ullrich, C. Course generation based on HTN planning. In *Proceeding of the Thirteenth Annual Workshop of the SIG Adaptivity and User Modeling in Interactive Systems*, pages 74–79. 2005.
[Weber and Brusilovsky, 2001] Weber, G. and Brusilovsky, P. Elm-art: An adaptive versatile system for web based instruction. *International Journal of Artificial Intelligence in Education*, 12:351–384. 2001.
[Yang et al., 2005] Yang, J.-T. D., Chen, W.-C., Tsai, C.-Y., and Chao, M.-S. An Ontological Model for SCORM-Compliant Authoring Tools. *Journal of Information Science and Engineering*, 21(5):891–909. 2005.