Di Nitto, E., Mainetti, L., Monga, M., Sbattella, L. & Tedesco, R. (2006). Supporting Interoperability and Reusability of Learning Objects: The Virtual Campus Approach. *Educational Technology & Society*, 9 (2), 33-50.

Supporting Interoperability and Reusability of Learning Objects: The Virtual Campus Approach

Elisabetta Di Nitto
Politecnico di Milano - DEI, via Ponzio 34/5, I-20133 Milano, Italy
dinitto@elet.polimi.it

Luca Mainetti
Università degli Studi di Lecce - DII, via per Monteroni, I-73100, Lecce, Italy
luca.mainetti@unile.it

Mattia Monga
Università degli Studi di Milano - DICo, via Comelico 39/41, I-20135 Milano, Italy
mattia.monga@unimi.it

Licia Sbattella and Roberto Tedesco
Politecnico di Milano - DEI, via Ponzio 34/5, I-20133 Milano, Italy
sbattell@elet.polimi.it
tedesco@elet.polimi.it

Abstract

E-learning has the potential to offer significant advantages over traditional classroom learning. However, it requires a complete redefinition of the dynamics of interaction between the various actors of a classroom. Moreover, in this context, the authoring of instructional material requires much more time than in traditional learning. Therefore, special care has to be posed to the definition of proper authoring approaches where educators can reuse and easily assemble existing materials. In this scenario, a comprehensive learning platform addressing the various interrelated aspects of *authoring* and *fruition* of instructional material is needed. Such a platform should enable *reusability of materials*, so that it is possible to make efficient use of preexisting experiences, and *interoperability* with existing platforms so that it is possible to take advantage of their strengths. The SCORM standard offers, among other features, a rich data model that can be used to define and share Learning Objects through different e-learning platforms. We argue that, despite the fact that it is the most emerging and promising standard, SCORM does not address some key issues properly, such as specification of metadata and LO composition. In this paper we focus on these issues and propose some extensions to SCORM that aim to address the above issues.

Keywords

SCORM, Learning Object, Metadata, Aggregation, Sequencing.

Introduction

Pervasiveness of computing in modern societies enables computer-supported learning tools to be available both in the classroom, and at home. In the classroom, such learning tools facilitate specific – often collaborative – activities, while at home they support students in self-study, in taking part in virtual classes, etc.

As pointed out in (Dongsong et al. 2004), e-learning has the potential to deliver significant advantages with respect to traditional classroom learning. As a drawback, by exploiting e-learning, the natural dynamic of interaction between the various actors in a real classroom cannot be recreated completely and new dynamics have to be enabled and driven. Moreover, the preparation of instructional material requires much more time than in traditional learning, therefore, a special care has to be posed to the definition of proper authoring approaches where educators can reuse and easily assemble existing materials. In this scenario, a comprehensive learning platform addressing the various complex and interrelated aspects of *authoring* and *fruition* of instructional material is needed.

Such a platform should address the needs of three main classes of actors: Authors, Teachers, and Learners. Authors design and build courses, possibly modifying and composing LOs; Teachers enact and manage courses, exploiting available LOs; finally, Learners attend courses by consuming LOs, possibly with the supervision of Teachers. Moreover, the platform should enable *reusability of materials*, so that it is possible to make efficient use of preexisting experiences, and *interoperability* with existing platforms, so that it is possible to take advantage of the functionality they offer.

In general, there are at least two possible strategies that can be adopted and combined for interoperability between heterogeneous systems: one *interface-oriented* and the other *model-oriented*. The former refers to the idea of defining well-known interfaces that systems should expose. By exploiting such interfaces, systems can call each other services thus enabling exchange of data, execution of queries on remote data, etc. The Simple Query Interface (SQI) (Simon et al., 2004) is an example of such an interface for the e-learning domain. The model-oriented strategy refers to the idea of having standard, semantically-rich data models shared by various systems. Sharing the same data model enables the possibility of reusing the same data in different systems, dramatically increasing the interoperability among them.

Within the context of e-learning, the SCORM standard (ADL, 2004a) supports both strategies. Restricting our analysis to the model-oriented aspect (which is the focus of our paper), SCORM provides a rich data model that can be used to define and share *Learning Objects* (LOs). LOs are defined not only in terms of their content, but also in terms of the technological platform needed to exploit them and in terms of the way they need to be sequenced for the purpose of a particular course. While experimenting with this standard in our Virtual Campus project (Cesarini et al., 2004), we have realized that, despite the fact that it is the most emerging and promising standard, SCORM does not properly address some key issues, which concern the specification of metadata describing LOs, and the composition of LOs. As we discuss in this paper, such issues directly affect the possibility to effectively perform two main activities related to reuse of instructional materials both within the same e-learning system and, even more, across different systems: *reuse for authoring* and *reuse for teaching*. Reuse for authoring, during creation of instructional materials, requires a data model for LOs that is rich enough to support reuse of parts (independently of their level of granularity), creation of LOs that reassemble existing ones, modification of the workflow that defines the way LOs will be executed (*sequencing*, in the SCORM terminology). Reuse for teaching, at the time a course is given, requires mechanisms and metadata that simplify the publication of the LOs and their enactment.

In this paper, based on the experiences we gained within the Virtual Campus project, we propose some extensions to SCORM aiming at supporting reuse for authoring by empowering the mechanisms for LO composition and reuse for teaching through the definition of proper metadata for LOs.

The paper is structured as follows. In Section 2 we present the SCORM approach and highlight its advantages and disadvantages. In Section 3 we present the core contribution of this paper, that is, our extensions to SCORM. In section 4 we present the high level architecture of the Virtual Campus runtime platform. In Section 5 we evaluate our approach, comparing it against SCORM through an example. Finally, in Section 6 we present some related work, and in Section 7 we draw some conclusions.

The SCORM Approach: Pros and Cons

SCORM (Sharable Content Object Reference Model) has been defined by the Advanced Distributed Learning initiative established in 1997 by the US DoD. The aim was to develop an overall standardization strategy to modernize education and training, and to promote cooperation between government, academia and business. To address these goals, the initiative has defined high-level requirements for learning contents, such as content reusability, accessibility, durability, and interoperability, and has defined three main standards aiming at addressing the aforementioned requirements: The *Content Aggregation Model*, the *Run-Time Environment*, and the *Sequencing and Navigation*.

The *Content Aggregation Model* (ADL, 2004b) contains guidance for identifying and aggregating resources into structured learning contents. The model is based on the IMS Content Packaging Information Model (IMS, 2004). It is based on three components *Assets*, *Sharable Content Objects* (SCOs), and *Content Organizations* that are enacted by means of the Run-Time Environment (see next point). Assets are electronic representations of media, text, images, sound, web pages, assessment objects or other pieces of data that can be delivered to a Web client. Assets can be grouped to produce complex Assets. A SCO is a collection of one or more Assets and represents a single executable learning resource. Since SCOs represent the lowest level of granularity of a learning resource that communicates with the Run-Time Environment, they can be launched by the Learner, while Assets cannot. Finally, a Content Organization is a tree composed of so-called *Activity items*, which can be mapped on SCOs or Assets (see Figure 1, extracted from (ADL, 2004b)). All of these components can be tagged with metadata. As a metadata standard, SCORM defines a variation of the IEEE Learning Object Model (LOM) (IEEE LTSC, 2002).

LOM envelops instructional material in metadata that describe it. Examples of metadata are the Language of the instructional material, the Description associated to the instructional material, etc. LOM defines a Learning Object (LO) as “any entity -digital or non-digital- that may be used for learning, education or training”.

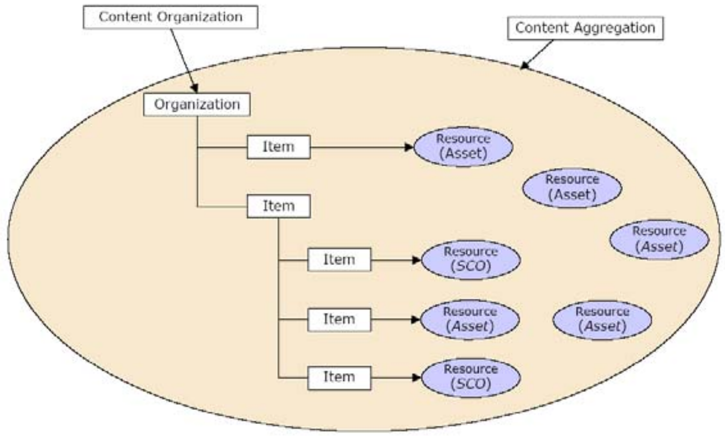

Figure 1 - SCORM Content Organization

The *Run-Time Environment* (ADL, 2004c) describes a content object launch mechanism, a communication mechanism between content objects and the SCORM engine on the server, as well as a data model for tracking Learners’ experience with content objects. It is derived from the run-time environment functionality defined in (AICC, 2004) CMI001 Guidelines for Interoperability.

The *Sequencing and Navigation* (ADL, 2004d) describes how SCORM-conformant contents may be sequenced through a set of Learner-initiated or system-initiated navigation events. It allows the Author to define a path through Activities. This path is used to guide the Learner in the way she/he takes the instructional material. Sequencing and Navigation is based on the IMS Simple Sequencing standard (IMS, 2005). It introduces new structures to aggregate Activities. An *Activity Tree* describes the branching and flow of Activities. Notice that the Activity Tree coexists with the Content Organization structure, as they represent two different "views" on the same objects. As in the Content Organization structure, Activities in the Activity Tree need to be mapped on the previously defined SCOs and Assets. Within the tree, a single parent Activity and its first-level children are considered as a new entity, called Cluster (see Figure 2, extracted from (ADL, 2004d)). Each Cluster has associated a Sequencing Definition Model (SDM) to define sequencing behaviors of its first-level children.

SDM permits to define sequencing behavior in several ways. In particular, Sequencing Control Modes define the instructional path in terms of workflow-like statements; Sequencing Rules represent a set of conditions that are evaluated in the context of the Activities for which they are defined; Rollup Rules define how to evaluate tracking information collected during the fruition; Objectives define how to evaluate Activities’ progress information, etc.

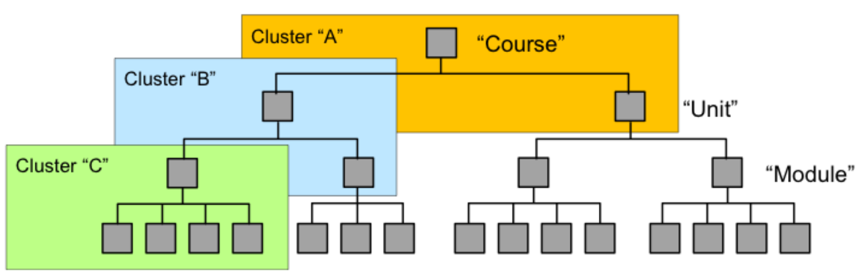

Figure 2 – Simple Sequencing Activity Tree and Clusters

SCORM can certainly be considered a step towards the establishment of a comprehensive framework for the definition and execution of LOs. If it will be actually accepted by the various vendors working in the area, interoperability and reuse of existing assets will be dramatically enhanced. However, while experimenting with it we have identified some weaknesses that are summarized in the following of this section.

The Content Aggregation Model defines components (i.e., SCOs and Assets) that seem to be slightly different implementations of the same concept (the Learning Object) under different reusability properties and limitations. As SCOs and Assets are both tagged with metadata and can be composed by means of the Content Organization tree, the only difference seems to be how they are handled at runtime. Only SCOs are able to fully exploit the functionalities provided by the Run-Time Environment in order, for example, to keep track of the interaction of Learners with the system.

The Content Organization tree permits to aggregate SCOs and Assets. However, it actually aggregates Activity items, which, in turn, need to be mapped onto previously defined SCOs and Assets. This choice maximizes the flexibility of the standard but leads to a very complicated and redundant definition, as Authors need to define Assets and SCOs, build the Content Organization tree based on Activities, and then provide a mapping scheme.

The Sequencing and Navigation specification further complicates the situation, as it adds new aggregation structures - Activity Trees and Clusters - which seem to partially replicate the Content Organization definition. The aim of allowing their coexistence is to separate simple aggregation (provided by the Content Aggregation Model) from sequencing (defined via Sequencing and Navigation). The drawback is that the overall specification becomes very complex and odd, since several concepts overlap and do not seem to be clearly defined.

Sequencing and Navigation provides several ways to define a learning path. This is interesting as it enables the definition of very complex learning paths. However, the interactions among these sequencing statements can be very hard to understand and lead to unexpected run-time behaviors. In fact, an Author can use the Sequencing Control Mode to define a path through a workflow-like specification. Then, she/he can add Sequencing Rules and Limit Conditions to assert when Learners are allowed to start a given Activity. These rules and conditions are completely independent of the workflow specification and can easily result in contradictory definitions. Authors are likely to commit such errors, as the complexity of the instructional material increases. Thus, they need a verification tool able to detect such inconsistencies. This however makes the authoring cycle more complex, and increases the difficulty in developing and using authoring tools.

Another aspect that needs to be analyzed concerns the purpose and potential of LOM. Shortcomings of LOM are well known in literature. In particular (Duval & Hodgins 2003) proposes an LOM research agenda to improve the standard in areas such as LO taxonomy, LO composition, automatic metadata generation, search, etc. In (Quemada & Simon, 2003) it is argued that the fundamental problem with LOM is the broad definition of the term "learning object" and the consequent inadequacy of the metadata set, when trying to apply it to a particular scenario. In our opinion, LOM has been mainly defined to support search and discovery of instructional material, but it could be extended to provide support to many other activities such as:

a) Automatic configuration of *required software*. Following the metadata specification, the platform could automatically configure pieces of software required by the LO. As an example, whenever a given LO exports a video content, a video streaming server could be automatically installed/configured.
b) Automatic configuration of *supporting software*. By supporting software we mean tools that are not mandatory for LO execution, but that could improve the way the LO is exploited. As an example, an LO requiring asynchronous communication among Learners and Tutors could be supported by a forum, one requiring synchronous communication could exploit a chat, while one requiring the cooperative production of some homework could take advantage of a shared, versioned repository. In summary, information about the kind of communication and cooperation required by an LO could trigger the configuration of proper software supporting these activities.
c) Tutoring. Metadata expressing instructional requirements could be useful to provide Learners with personalized automatic tutoring. In fact, they could support the selection of the most appropriate LO for a Learner, depending on his personal preferences and attitudes.
d) Evaluation. Metadata could be used by Teachers to analyze and evaluate the effectiveness of LOs.

These activities cannot take advantage of LOM metadata in their current form, because of the following weaknesses. First of all, the exact meaning of some metadata is not clearly specified (e.g., Semantic Density, Difficulty, etc.). Moreover, some important characteristics of LOs cannot be expressed. As an example, there is no way to say whether a given LO has been designed to support group study or individual study. Finally, the defined metadata are not fully machine-processable: Some of them are defined as free-text (e.g. Installation Remarks that could be relevant to the configuration of required software) while others rely on vocabularies that are not precise enough to enable a fully automatic processing.

The LOM extensions we propose in Section 3.1 try to address the points from A to D, while the LO composition model we present in Section 3.2 integrates the composition and the sequencing aspect thus reducing the effort of Authors and avoiding inconsistencies at runtime.

The Virtual Campus conceptual model as an extension to SCORM

The Virtual Campus project aims at offering a clean and simple conceptual model, called VC-LOM, that supports both LO definition and fruition in an interoperable and reusable way. Moreover, it offers a software platform supporting the entire life cycle of LO and their usage by Learners and Instructors. For the purpose of our approach, we define a Learning Object (LO) as the union of instructional materials and the corresponding metadata. In this section we focus on the conceptual model. In the following paragraphs, we present, in more detail, the extensions we propose to the LOM standard, and then discuss our composition approach that supports both aggregation and sequencing of LOs.

Metadata specification

Our extensions to the LOM mainly aim at supporting the activities of automatic configuration, tutoring, and evaluation by explicitly expressing some LO properties the LOM does not consider. Of course other extensions aiming at improving other aspects, such as discovery, could be included, but these are not our focus. The extensions we have defined are summarized as follows (see Table 1):

- A new set of metadata providing details on how an LO should be used. These concern the level of supervision a given LO requires (e.g., if a tutor should be available during fruition of the LO), whether the LO requires group (cooperative) study, whether some artifacts have to be created at the end of fruition, whether the LO requires communication facilities among Learners, and whether the communication or cooperation have to be synchronous or asynchronous. Such metadata can have various uses. In particular, they can help address issues presented in point B of Section 2. For instance, if the “Cooperation Attribute” and “Artifact Attribute” are set, the runtime platform could automatically configure a version-control server to facilitate the learners in working on shared documents. In addition, if the “Supervision Mode” is set to *tutored*, an upload facility could be configured in order to allow Learners to release the created documents, and Teachers to manage and evaluate them. These data can also provide information useful for automating the tutoring of Learners and the evaluation of LOs (points C and D in Section 2). For example, the fact that the "Cooperation Attribute" of an LO is set to true triggers the monitoring of the cooperative abilities of Learners taking that LO (for instance, the runtime environment could check the number of communications among Learners, the number of check-in/out operations they perform on shared documents, etc.). The collected data are added to the Learners’ profile and could be used by a tutoring system to push shy Learners toward interaction with the others, or to infer, through automatic evaluation, other data both about the performance of LOs and of Learners.
- Pre-conditions and Learning Objectives constraining the start and end of LO fruition. Both can predicate on Time and on data available from the Learner’s profile.
- The possibility of considering traditional situated didactical activities and digital e-learning material in a uniform way. Both of them in fact can be represented as LOs. A LO representing a situated activity will have the Access Modality attribute set to “Situated”, and Date and Place will provide information on the schedule and location where the activity will take place.

Table 1 - Examples of Virtual Campus additional metadata

Field name	Field description
Supervision Mode	The level of supervision on Learners' activities: “none” (no supervision); “tutored” (a tutor is available, during fruition Learners can explicitly request his/her supervision); “supervised” (the supervisor is always present during the instructional process); “driven” (Learners act in a passive way, by strictly following the Teacher's instructions.)
Cooperation Attribute	Whether Learners should take the LO in cooperation.
Communication Attribute	Whether Learners will be provided with communication facilities, while exploiting the LO.
Synchronism Attribute	In case of a cooperative or communicative LO, it specifies if the LO must be taken synchronously by all Learners or asynchronously.
Group Cardinality	The cardinality of the group involved into the fruition of the LO. Meaningful group

	cardinalities are “1” (self-study), “2” (pair study, both Learner-Learner and Learner-Teacher), “m” (group study). Note: 2 is the minimum group cardinality for cooperative LOs and the maximum one for non-cooperative LOs.
Artifact Attribute	Whether the LO requires Lerner(s) to produce an artifact.
Access Modality	“Situated” (in a specific physical location, e.g. a live lecture held in a classroom), “Digital”.
Place	The physical location name. If Modality is “Digital”, this field is ignored.
Date	The date (including starting and ending time) the LO is given. It is used for “Situated” LOs, but also for some kinds of “Digital” LOs that can be taken only at certain points in time.
Precondition on time	Constraints on time that have to hold before the LO is taken.
Precondition on user profiles	The skills and knowledge a Learner must have in order to exploit the LO. It can also predicate on administrative constraints that have to be fulfilled by the user before exploiting the LO (e.g., he/she must have paid the enrollment fee.)
Learning objective on time	The min/max amount of time required/allowed to complete the LO.
Learning objective on user profiles	Educational objectives of the LO in terms of skills a Learner can obtain by exploiting it. It can also express objectives that are not strictly educational (e.g. the fact that the Learner achieves some kind of degree by completing the LO).

In order to support automatic installation of software required by LOs (see point A in Section 2), we are working at a more precise specification of the LOM metadata 4.4 Requirements. In particular, we are currently investigating on the possibility to express requirements and capabilities of software, by using the CC/PP (W3C, 2004) (Composite Capabilities/Preference Profiles) standard. The CC/PP is a standard aiming at describing device capabilities and user preferences. It introduces the structure for representing delivery-context information by means of *profiles*. Profiles consists of attribute/value pairs, which characterize in detail a certain feature of the delivery context. A vocabulary defines a specific set of attributes and component classes, designed for bringing together and describing the characteristics of a certain category of devices. The CC/PP itself does not depend on specific vocabularies, it just provides a structure for constructing and representing them. Although CC/PP’s initial goal is to describe devices and user-preferences, we argue its flexibility allows for the implementation of an LO ad-hoc profile. This profile will be able to express LO requirements for software in a far more precise way than the one permitted by LOM metadata.

We also plan to investigate the IMS Learning Design specification (IMS, 2003), as it explicitly models the concept of “service facilities”. Service facilities are resources that cannot be given a URL at design time. They have to be instantiated by a local runtime service. Current specification allows the following services: *send-mail*, *conference*, *monitor*, and *index search.* This seems an interesting approach to support automatic configuration of supporting software (see point B in Section 2).

Other approaches for the design of LOM extensions can be found in literature. In (Rivera et al., 2004), the LOM Application Profiles is used to adapt semantics of LOM elements, and make LOM-conformant extension to represent usage conditions and learning activities, in the context of the Elena project. In (Brase et al., 2003), constraints on LOM’s fields are defined by means of inference rules. RDF is exploited to define metadata, while an inference language explicitly developed for RDF (TRIPLE) represents axioms.

LO composition

We overcome the limits imposed on LO reusability by the Content Aggregation Model by defining a unique model for aggregated and simple Learning Objects and by enabling a powerful recursive composition mechanism. Moreover, we coherently integrate it with proper sequencing mechanisms.

As shown in Figure 3, we define an Atomic LO (ALO) as the smallest unit of reuse for LOs that may or may not be associated to one or more multimedia contents. A Complex LO (CLO) is defined as an LO whose instructional material is an aggregation of Learning Objects. Being an LO, a Complex LO can be treated exactly as any other LO. Moreover, our composition mechanism defines the way some metadata of a CLO are automatically derived from the metadata of the component LOs (e.g., the Size of a CLO can be computed from the size of its components).

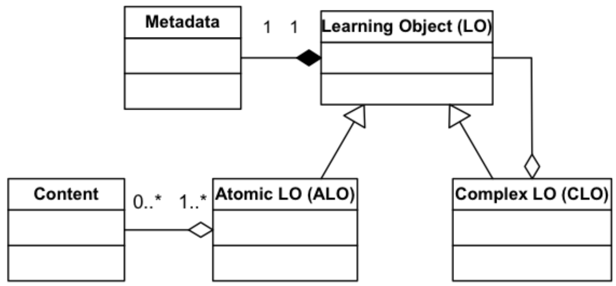

Figure 3 - The Virtual Campus LO model

Of course, LOs defined within a CLO can be sequenced in a way that depends on their content, on some external conditions such as the availability of Teachers and Tutors, on the purpose of the specific CLO in which they are contextualized, etc. As it can be noticed, the information that constrain such sequencing are not all known at the same time (e.g., A Teacher's appointment can be known few days before a seminar, while the need for taking Mathematics before Physics could be intrinsic in some specific curricula). Indeed, they do not have the same scope (again, the Teacher's appointment can enforce certain sequencing only on a specific edition of a course, while the need for having Mathematics before Physics should be always enforced). Based on these considerations, we distinguish between three levels of abstraction that offer different but coherently related facilities to define sequencing constraints LOs:

- At the *Reusable Level*, Authors define each CLO in terms of a graph where nodes univocally represent LOs (either Atomic or Complex) while edges represent relationships between LOs. For instance, in Figure 4, History of Mathematics is referred into Basic Concepts LO and, therefore, can be seen as an optional, in-dept instructional material.
- At the *Didactical Level*, each CLO is seen in terms of a workflow (see Figure 5) that can be automatically generated starting from the Reusable Level definition. Such a workflow shows all possible learning paths for a CLO and it can be customized by the Teacher for specific purposes. For instance, an available path can be disabled for some specific reasons (see Figure 6).
- At the *Fruition Level*, some details needed for fruition of LOs are introduced (see later).

More in detail, at the Reusable Level, rounded-corner rectangles inside a CLO represent particular CLOs called *Inner CLOs*. They provide a mechanism to aggregate LOs, but, differently from other CLOs, they do not have an identity and cannot be reused outside of the context of the CLO in which they are defined. They can however participate in any relationship connecting two generic LOs. LOs inside an Inner CLO can be mutually exclusive or not: The Inner CLO is labeled with two values indicating the minimum and the maximum number of LOs the Learner has to exploit. The syntax is *(n,m)* where *n* represents the minimum and *m* the maximum. Values as *(1,1)* indicates that LOs are mutually exclusive, while *(0,1)* states that Learners are allowed to skip the fruition of LOs. Finally, if the label is not indicated, all of the LOs have to be exploited. LOs labeled with T are called Test-LOs. They are used to model assessments Learners have to take.

Relationships indicate the presence of instructional constraints between two LOs in the context of a containing CLO (outside that CLO, the relationship is no longer valid). A generic relationship from *x* to *y* in the context of *z*, with *x,y* being LOs (either Atomic or Complex) and *z* a CLO (on Inner CLO), is represented by an arrow from *x* to *y* inside *z*, labeled with the relationship name. The relationships can be named *IsRequiredBy*, *References*, and *RequiresOnFailure*. Their meaning is summarized in Table 2, where, for the sake of brevity, we omit the indication of the CLO where the relationship takes place.

Table 2 - Relationships between Reusable-Level LOs

Relationship	Description
IsRequiredBy	*A IsRequiredBy B* indicates that LO *A* must be completed before starting LO *B*; i.e., the Learner has to possess *A*-related knowledge in order to achieve a correct understanding of *B*. However, the *IsRequiredBy* relationship does not mean that Learners must complete *A* immediately before *B*: Learners are allowed to make use of other LOs after *A* and before *B*'s fruition.
References	*A References B* indicates that *A* cites *B* as a source of more details on a topic related to *A* itself. Taking *B* at fruition time is not compulsory but, in case it is taken, it has to be entered after A and before taking any other LO not connected to A through a References relationship. Many *References* can depart from the same referencing LO. In this case,

	Learners can make use of one or more of the referenced LOs.
RequiresOnFailure	The *RequiresOnFailure* relationship always connects a Test-LO with some other LO. If the Test-LO is failed, then the LO at the other end of the *RequiresOnFailure* relationship has to be taken by the Learner. If no *RequiresOnFailure* is specified, Learners failing a Test-LO have to re-start the fruition of the whole CLO.

The example shown in Figure 4 defines two different CLOs. The first one, “Mathematics”, is composed of several LOs. “Basic concepts” and “Algebra” are both required by the Inner CLO enclosing “Calculus”, “Geometry”, and “Limits”, so they should be taken in the first place. Notice that the relative order between “Basic concepts” and “Algebra” is not constrained. All the LOs inside the Inner CLO have to be exploited, since no label is present. The “History of mathematics” is left as an optional activity and, in case it is taken, it must immediately follow “Basic concepts” that references to it. “Exam” is a Test-LO. In this example, since no *RequiresOnFailure* relationship is defined, if “Exam” is failed the whole CLO have to be repeated.

The second CLO, “Engineering first year”, is composed by reusing “Mathematics” as well as some other LOs. Learners have to complete “Mathematics” before entering “Physics”. “Chemistry A” and “Chemistry B” are alternative. This is defined by the label *(1,1)* that states the Learner must exploit only one of the two. Notice that Learners can take one of the “Chemistry” LOs either before “Mathematics”, or after “Physics”, or in between. It is interesting to note that “Mathematics”, being reused in this context, appears as a black-box. Its internal complexity is hidden thus allowing for an easy composition. Even more interesting is the fact that, in such a representation, *less arcs are drawn means that more freedom is left to Learners*. As an extreme example, a simple collection of LOs, with no arcs at all, permits the fruition of a course in which all possible paths are allowed.

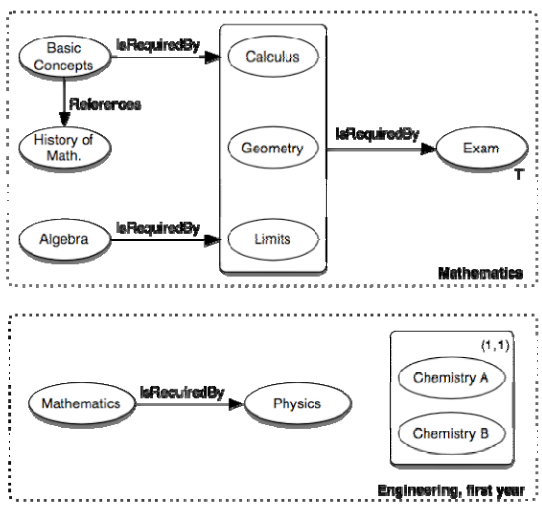

Figure 4 - CLO definitions

LOs can be (re)used either to define other LOs or to provide them to the Learners. In the second case, they need to be translated into their didactical level representation. For instance, Figure 5 shows the workflow representations of the two CLOs defined in Figure 4. In this representation, LOs are mapped into activities that represent the fruition of the corresponding LOs. The syntax is derived from the UML activity diagram. However, it has been adapted to our particular case. Simple arrows connecting activities represent a sequence. Vertical bars encode the *fork/join* semantics (i.e. parallel execution or, more properly in our case, absence of constraints on the relative fruition order), as well as the *multiple switch* semantics (i.e. execution of one or more of the involved activities). A label *(n,m)* permits to define the behavior of the vertical bar (the meaning of *(n,m)* is the same as in the Reusable Level language). The dashed, bidirectional arc denotes the fact that the corresponding path is not mandatory (i.e. there is an optional activity). Finally, the double-arrow arc indicates the path Learners must follow whenever they fail a Test-LO. If such an arc is not indicated, Learners have to take again the whole CLO (in the example, the arc is drawn in order to permit a complete explanation of the workflow syntax and it could be removed).

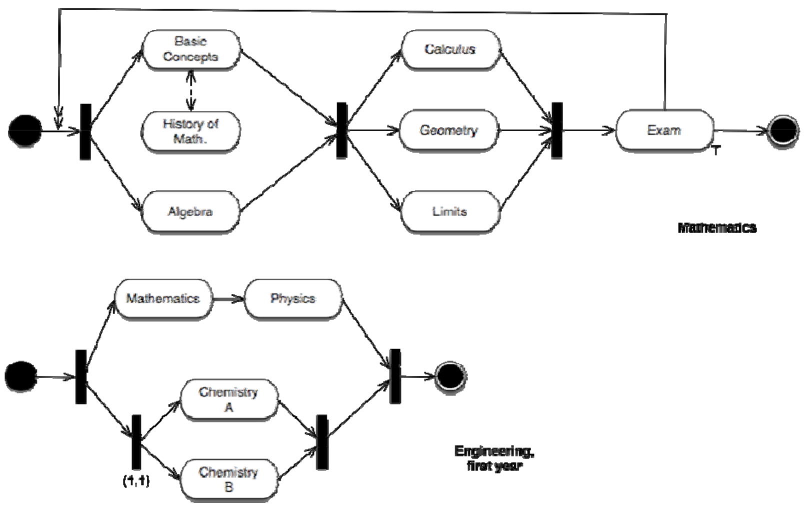

Figure 5 - Workflow description of CLOs

By using the editing tools provided by Virtual Campus (see Section 4), such a workflow representation is automatically derived from the Reusable Level model of an LO. If needed, the Teacher can customize this representation by performing the following operations:

- elimination of alternative paths by selecting a single path or a subset of the available ones;
- elimination/enforcement of optional activities;
- enforcement of the order of fruition, in case of parallel activities.

All these operations preserve the consistency between the resulting workflow and the corresponding high-level description since they further constrain the way LOs are used by Learners. Figure 6 shows a possible customization of the workflows depicted in Figure 5.

It is important to notice that if the Teacher does not need to customize the generated workflow, this step can be left completely hidden. Thus, in the simplest case, the teacher does not have to deal with the workflow representation.

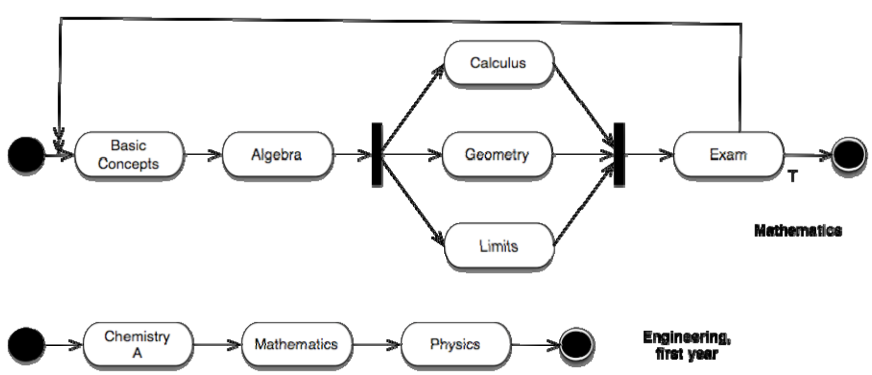

Figure 6 – Workflow customization

At the Didactical Level an LO is not ready for fruition yet. Some additional details need to be added concerning the course edition, the enrollment method, start and end dates, the course calendar, announcements, the Teacher's name, the list of already enrolled students, etc. Such information is defined at the Fruition Level of abstraction and cause the creation of a Course which is ready for fruition.

The Virtual Campus runtime platform

The Virtual Campus platform implements the VC-LOM and supports the design, deployment, fruition, and evaluation of learning materials. It is composed of two main subsystems: The Authoring Environment and the Fruition Environment (see Figure 7).

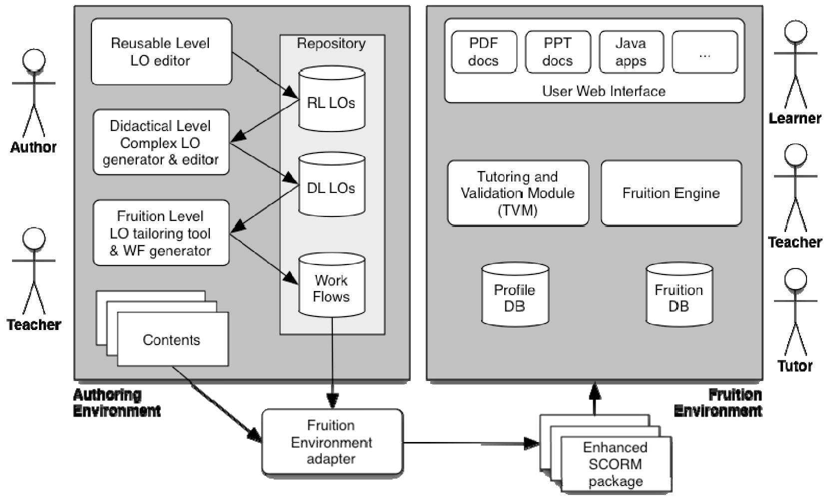

Figure 7 - Virtual Campus high-level architecture

The Virtual Campus authoring environment provides several editors to define Atomic LOs and Complex LOs at the Reusable, the Didactical, and the Fruition Levels. The editor used at the Reusable Level is shown in Figure 8. It is provided as an extended and personalized version of Microsoft Visio, to take full advantage of its drawing capabilities. The tool permits the definition of a new Atomic LO, specifying its content (if any) and its metadata. Complex LOs can be created, reusing and aggregating other LOs. In order to find LOs within the Virtual Campus repositories, a dialog box permits the specification of searching criteria, using metadata. It is also possible for authors to import SCORM LOs from external systems. Such LOs can be either standard SCORM compliant, or extended SCORM compliant (incorporating the Virtual Campus extensions).

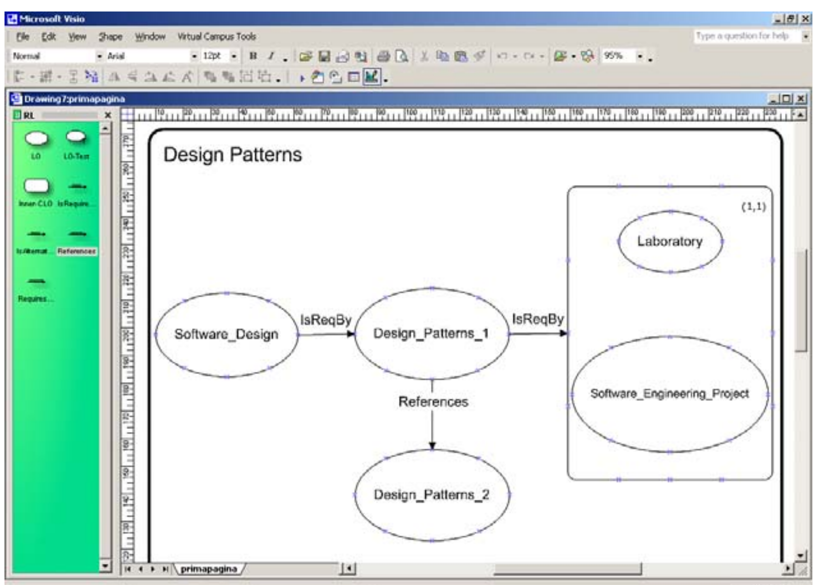

Figure 8 - The CLO editor (Reusable Level)

The application also features a Didactical Level Complex LO generator that automatically generates a first version of the workflow associated to a Complex LO and then supports Teachers in customizing it. Finally, a Fruition Level LO tailoring tool supports the insertion of all the necessary Fruition Level details (e.g. class start time).

A LO can be serialized in the SCORM format at any of the abstraction levels defined in the Virtual Campus conceptual model. In such a way, Authors and Teachers are free to choose the preferred trade-off between abstract, highly reusable LOs (needing the final tailoring phase by the Teacher), and less reusable, ready-to-use LOs (no effort by the Teacher is needed). Reusable Level, Didactical Level, and Fruition Level specifications can all be exported at the same time in the same SCORM packet. The serialization algorithm maps both the Reusable Level and the Didactical Level languages on a subset of the SCORM Simple Sequencing tags. Moreover, the VC-LOM metadata are stored into the package as extensions to the standard LOM. When SCORM packages, generated with our platform, are exported to a Virtual Campus system, the whole specification is extracted. Otherwise, standard SCORM systems would simply ignore our extensions.

The Fruition Environment is based on the SCORM Run-Time specifications; it allows Learners to attend courses, and Teachers to guide fruition (when required). The environment is Web-based. Learners and Teachers only need a Web browser (possibly augmented with plug-ins, such as Flash, the Java Virtual Machine, etc.) to exploit the contents of LOs.

Our SCORM-compatible Engine incorporates both the standard SCORM Run-Time engine, and a Sequencing Engine. The Sequencing Engine enacts the fruition workflow associated to an LO, guiding Learners and Teachers in the execution of the activities related to the usage of the LO. It is composed of several modules. Besides the Workflow Engine, the other components are: a Precondition Resolution module which takes into account constraints added to LO metadata (e.g. class date and time); a Learning Objectives Evaluation module which evaluates the Learning Objectives specified in LO metadata. Whenever a given Learner finishes an LO, the Learning Objectives Evaluation module is invoked in order to apply the specified rules. Then, the Workflow Engine is asked for the list of LOs the Learner is allowed to enter as a next step. The resulting list is then filtered by the Precondition Resolution module, leading to the final list that is returned to the SCORM Engine. Finally, a Web page showing the list is presented to the involved Learner who can communicate her/his decision by selecting one of the proposed alternatives. Once the confirm button is pressed, an event is generated and notified to the Sequencing Engine.

The Engine activates the tools that are required for the execution of LOs. These tools can be of the following types:

- Stand-alone applications (e.g., Microsoft PowerPoint or Flash), in this case off-line study is enabled by the platform.
- Server-based applications (e.g., tools supporting, collaboration among Learners).

As an example, WebTalk (Barbieri et al., 1999) is an application, developed at Politecnico of Milano, providing a 3D metaphor to browse through teaching material. The tool allows the user to maintain awareness of presence of other users who browse through the same material, thus enabling unstructured interaction and sharing of information among Learners. WebTalk exploits the Macromedia Shockwave Multiuser Server to permit information exchange among Learner, while a Web browser (enhanced with the Macromedia Shockwave plug-in) is required as a client. In this case fruition requires Learners to be on-line, connected to the Virtual Campus platform.

The Fruition Environment (Sbattella et al., 2004) is instrumented with a monitoring tool that collects data on actions performed by users. Profiles are then generated and exploited, on one side, to provide feedbacks to Teachers on the validity of specific LOs and applications, and on the behaviour of Learners. On the other side, they are exploited to instruct automated tutoring agents, enabling them to suggest and guide Learners throughout the fruition process.

Virtual Campus Vs. Scorm – Comparative Evaluation

We have tested the Virtual Campus platform by experiments. The experimentation aimed at assessing the possibility of arranging in complex structures heterogeneous LOs (lectures, studying activities, cooperative sessions of work, and exams), and at testing the whole platform covering all the phases of course development and fruition. A first experiment focused on evaluating the behavior of students while working on a group project, exploiting collaborative LOs. Then, we developed a complete on-line version of a course on "Web Design", taught at the Politecnico di Milano. Finally, the course on "Software Architectures", depicted below, has been given to some undergraduate students of a software engineering course at Politecnico di Milano.

The results are still under evaluation. In general, students enjoyed the user-friendly interface and the easy interaction with the system and their colleagues. Moreover, they appreciated the support to collaboration offered by the Virtual Campus platform. All of them took the LOs defined as optional in the course, probably because of the novelty of being able to autonomously make choices. We also found that the added value of contents based on virtual environments was not perceived as determinant for the learning objectives. We expect that virtual environments could be more deeply exploited and appreciated in presence of students with disabilities, for whom traditional media are difficult to use. From the teacher's perspective the system has been helpful to integrate existing materials and build Complex LOs in a few hours. Moreover, the monitoring facilities enabled him and the tutors to profile students' behavior and to evaluate the effectiveness of the proposed Complex and Atomic LOs.

As a complementary way to test Virtual Campus, we have evaluated the expressiveness of the model against the one of SCORM, referring to the example of the "Software Architectures" course. In the following we discuss this comparison.

The Virtual Campus model

The LO we have realized is composed of four main modules: "Introduction", "Software Design", "Design Patterns", and "Architectural Styles". The optional "Seminary" focuses on presenting advanced mechanisms for late-binding of software components. Finally, students are evaluated on both theoretical aspects (through a quiz) and practical abilities (by means of a laboratory session).

Figure 9 shows the Reusable Complex LO associated to the course. The module on "Design Patterns" is offered by two alternative Atomic LOs. One of them is based on slides while the other provides a video. Note that the fact that such LOs are alternative is established by the Author of the course (considering the goals of the course), and the scope of such relationship is limited to the Complex LO "Software Architectures" itself. The quiz is implemented by the Test LO. Learners failing the test are required to take again the course starting from the "Software Design" LO.

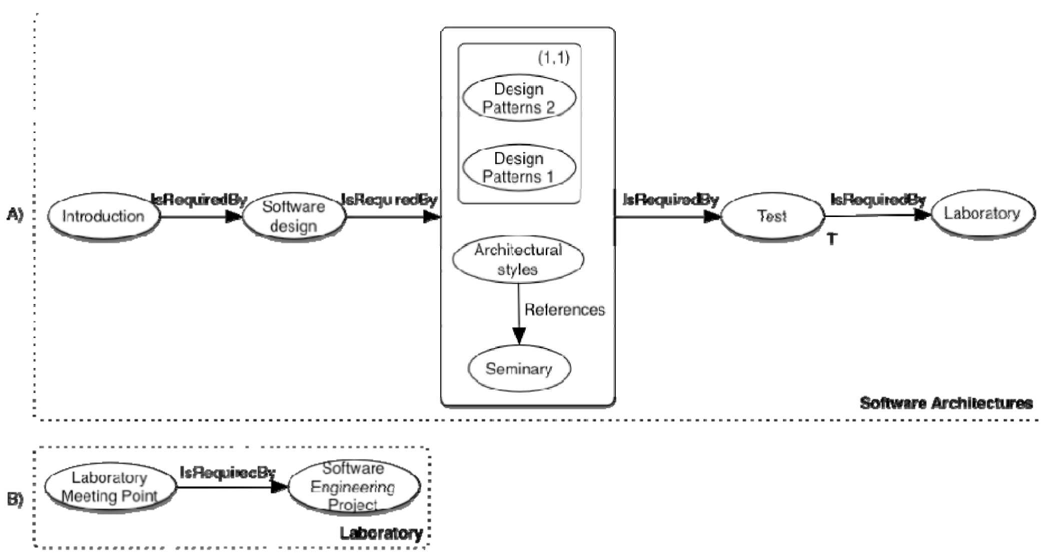

Figure 9 - The Reusable Complex LO of the course on Software Architectures

The laboratory session is a Complex LO whose components are "Meeting Point" and "Software Engineering Project". The former requires that Learners create groups, download the requirements of the simple event based application to be developed and exchange comments on them. The latter requires Learners implement the application. Since they have to work in teams, they are requested to coordinate themselves by using a configuration management application.

Notice the usage of a nested Inner CLO asserting that "Design Patterns 1" and "Design Patterns 2" are alternatives. It is enclosed in another Inner CLO that requires all its LOs to be exploited. In this specific case this means that one of the two "Design Patterns" LOs, and the "Architectural Styles" LO must be exploited. The optional LO "Seminary" can be exploited after "Architectural Styles".

All the LOs depicted in Figure 9 are decorated with metadata. For example, "Software Engineering Project" is a collaborative LO in which groups of Learners have to build a simple software application. Since it is tagged with "Supervision Mode = tutored", the Virtual Campus platform enables a Tutor to access all the "Software Engineering Project" instances exploited by the Learner groups. The tag "Artifact Attribute = true" tells the system to configure an up-load page to let the groups deliver their artifacts. The tags "Cooperation attribute = true" and "Group Cardinality = 2" allow for the creation of student pairs and for the installation of a versioning server to support the cooperative creation of the artifact. The tags "Communication Attribute = true" and "Synchronism = asynchronous" instruct the system to provide a forum. Finally, "Access Modality = Digital" specifies that the LO does not need a classroom.

Figure 10 shows the Didactical Level Complex LO that is automatically derived from the Reusable LO in Figure 9. It can be customized depending on the needs of the specific course and teacher. For instance, Figure 11 shows a customization where the video version of "Design Patterns" has been eliminated and a fruition order between "Design Patterns" and "Architectural Styles" has been forced. This customized version of the course was then translated into a Fruition level LO and made ready for fruition.

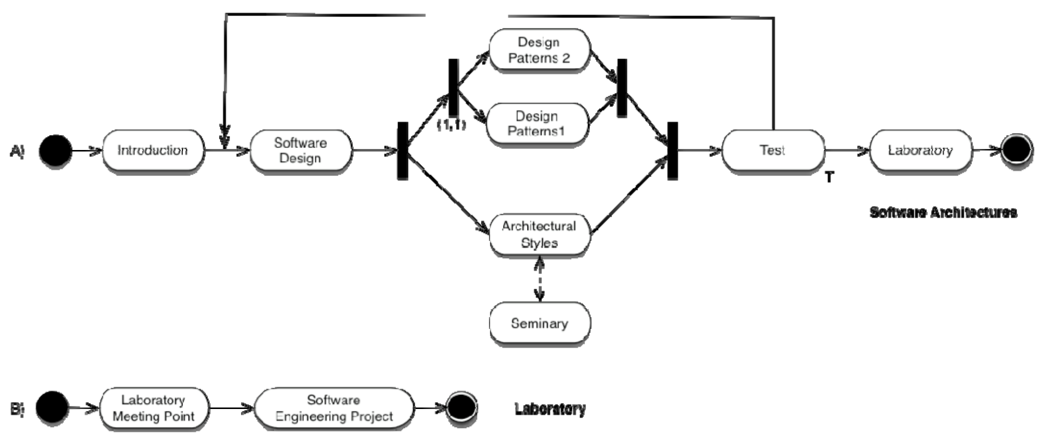

Figure 10 - The Didactical Level Complex LO of the course on Software Architectures

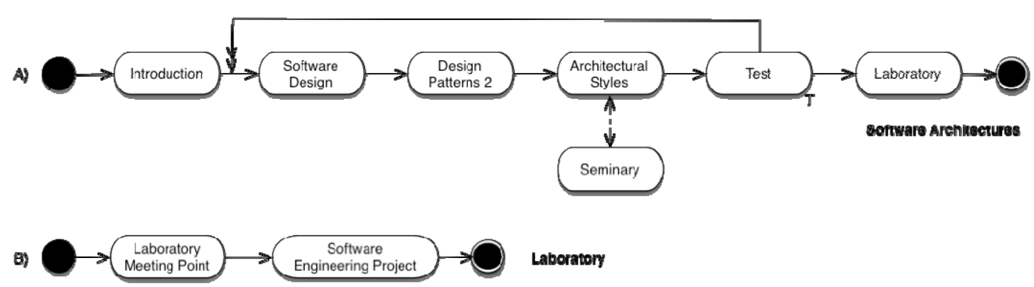

Figure 11 - The customized version of the course on Software Architectures

The SCORM model

The aforementioned example can be described using the SCORM Sequencing and Navigation. Figure 12 depicts the SCORM Activity Tree representing our "Software Architectures" LO. Boxes labeled as "Anonim-*n*" represent Activities we need to introduce, in order to reproduce the CLO aggregation structure shown in Figure 9. For example, "Anonym-4" represents the fact that "Architectural Styles" and "Seminary" can be seen as a single entity (see the semantics of the References relationship).

Figure 13 depicts the sequencing information needed to reproduce the behavior of the LO "Test". For the sake of brevity, the description of the whole example is not depicted. It is easy to see that such a specification tends to become very complex, as the number of Activities and/or the number of allowed instructional paths increases. By contrast, the Reusable Level specification tends to be less and less complicated as the number of allowed instructional paths increases (the less the number of arcs drawn, the more freedom is left to Learners).

Standard LOM metadata do not allow the specification of the aforementioned "configure up-load page" or "install versioning server" requirements, in a structured, machine-processable manner. Authors can add such requirements as textual notes, but machines cannot easily process such kind of unstructured specification.

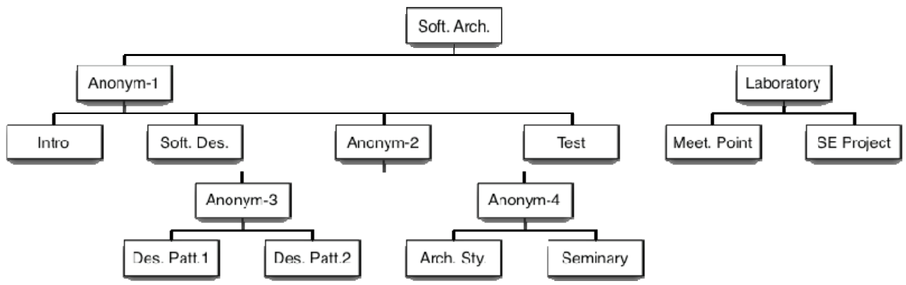

Figure 12 – Activity Tree

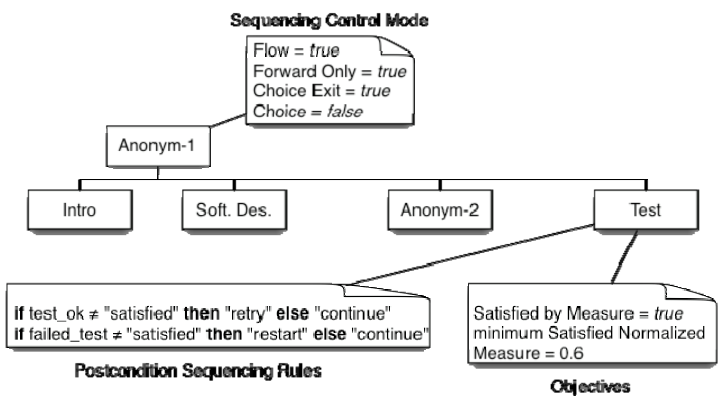

Figure 13 - Sequencing information for the LO "Test"

SCORM graphic representations

In (Jun-Ming Su et al., 2005) a workflow-like, graphic representation of sequencing information is proposed, so we think it is interesting for us to compare our approach against this. SCORM Clusters and the associated set of SDM rules define a so-called Sequencing Object (SO). The model defines six kinds of SOs, each one representing a workflow construct: Linear SO (a simple, linear sequence), Conditional Linear SO (adds conditions α_i on arcs, permitting to modify the strict linear behavior), Choice SO (selection among two or more objects), Conditional Choice SO (adds conditions α_i, permitting to describe more complex selection behaviors), Loop SO (uses two conditions; useful to describe tests), and Exit SO (adds conditions α_i that stop the instructional process).

Figure 14 depicts how the SO-based language models the "Software Architectures" example. Gray boxes represent SOs, while white boxes model elementary activities (the leafs in the Activity Tree). The special nodes "N_{in}" and "N_{out}" do not represent any actual instructional content. They just act as input and output gates for Choice and Conditional SOs. Notice that, in case of condition failure, the standard behavior of the Loop SO lets learners return only to the SO that immediately precedes the current one. Therefore, a Linear SO is needed in order to group all the previous LOs.

This specification resembles our Didactical Level language, while does not provide features equivalent to our Reusable Level language. Thus, the main lack we found is the absence of a composition scheme that permits to decompose Complex LOs into simpler objects, describing them as separate entities ("Software Architectures" and "Laboratory", in this example).

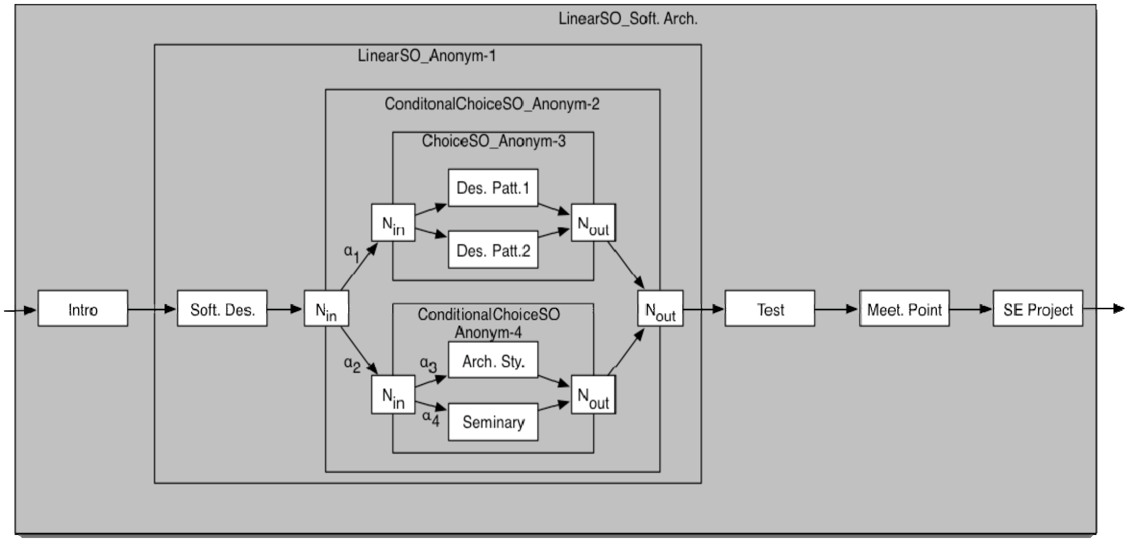

Figure 14 – SO-based description of Software Architectures

Related work

Besides SCORM, and the approaches directly related to it, in the literature there are various platforms that support authoring and/or fruition of LOs. In this section we focus on the approaches supporting authoring, and classify them into three categories: relationship-based, workflow-based, and rule-based.

Relationships, workflows, and rules

Relationship-based systems allow teachers to define a course structure by means of logic relationships among the course components. MediBook (Knolmayer, 2001) is an example of such systems. In MediBook the important domain concepts are formalized and related to each other by semantic relationships. In turn, LOs are associated with concepts and are connected through so-called rhetorical relationships (e.g. LO-A *deepens* LO-B, LO-C *is-part-of* LO-D). MediBook uses the LOM standard to define LO metadata and to store rhetorical relationships. Learners can navigate through both the rhetorical relationships structure and the semantic relationships structure. In the latter case, they discover LOs starting from the associated concepts.

As an alternative approach (Steinacker et al., 2002) uses a sort of "direct prerequisite" relationship to order LOs (e.g. LO-A *is a direct prerequisite for* LO-B). The matrix associated to the resulting graph shows the total number of direct and indirect prerequisites between two LOs. When Learners choose an LO to exploit, it is possible to calculate the list of required LOs. An integer-programming model is then built, taking into account further constraints (e.g. the time effort required by a given LO). By minimizing the model's target function, some LOs are removed from the list. A sequencing procedure determining the "best" schedule for the remaining LOs is then executed.

A similar approach, described in (Carchiolo et al., 2001), uses the same relationship, but adds weights in order to represent how hard is to access a given topic, coming from a previous one. To choose a path, Learners select it from the whole graph provided by the system. Each route is associated with a numeric index weighting the "effort to learn" the target topic.

Ontologies are proposed in (Jin-Tan David Yang et al., 2004) as a mean to create relationships among instructional material. The paper makes use of RDF/RDFS to define the ontology of a given instructional domain. Sharing the ontology among authors, it is possible to generate SCORM-compliant Learning Objects, all of which provide an outline using precisely defined terms.

Workflow-based systems allow teachers to define a course structure as a workflow. In (Padrón et al., 2004) workflow languages such as BPEL4WS are exploited for the composition of learning web services and their adaptation to the needs of a Learner or group of Learners. Once composed and packaged as Learning Objects, these composite processes can be executed, instantiated and adapted to the Learner's particular needs. These adaptations can be realized, either by predefined rules implemented into the process description and driven by the Learner behavior, or in a supervised manner.

Rule-based systems customize the fruition of LOs by mean of *rules*. Preconditions and postconditions rules are exploited in (Sicilia, M.A. et al., 2004). Preconditions encode prerequisites required for an instructional process to take place, while postconditions states the expected outcomes. Relying on such rules it is possible to calculate the list of LOs a given Learner is allowed to enter.

In (Quarati, 2003), weights are associated to LOs. Weights represent the cognitive contribution of objects within a given context. If an LO has subcomponents, its weight is defined as the maximum weight of its subcomponents. The knowledge already acquired by Learners, together with the weights of LOs, is used to identify the part of the content domain available to an individual user at a certain point in time.

Discussion

Relationship-based, workflow-based, and rule-based systems offer both advantages and disadvantages. Relationship-based systems focus mainly on the definition of complex association structures between LOs, while the other classes of systems provide run-time mechanisms to check that such relationships are properly satisfied and, in some case, enforced during the fruition of courses.

Workflow-based systems define a strict temporal order among activities. In some cases this forces teachers to impose unnecessary constraints. On the contrary, rule-based systems provide support to the definition of constraints on fruition paths, but do not offer mechanisms to let teachers define precise paths if this is needed.

Based on our practical experience in giving courses and preparing the related material we have noticed that educators working on LOs at different levels of abstraction may need to exploit features offered by all the three kinds of approaches. In particular, educators in charge of defining the structure of courses want to compose LOs just by aggregating them and imposing some instructional constraints. For instance, should they need to build a math course using LOs "Limits", "Derivatives", and "Geometry", they would probably impose the constraint "Limits" *is required by* "Derivatives", while they would not constrain any sequencing between "Geometry" and the other two LOs. To achieve this goal they would probably prefer a relationship-based system. Vice versa, teachers willing to adapt LOs to a specific course instance (e.g., "Mathematics" for Mechanical Engineering) wish to impose more restrictive constraints on the LOs in order to define the structure of a course. In particular, teachers wish to have complete control over the fruition paths, for instance, imposing that "Geometry" has to be taken before "Limits" when the organization of that specific course instance requires it, e.g., because of problems related to the calendar structure. Thus, they would prefer to exploit a workflow-based system. Finally, teachers willing to precisely define some preconditions to the LO fruition (e.g., administrative constraints that have to be fulfilled), would prefer a rule-based system.

For the above reasons, we have chosen to exploit all these approaches at the various levels of abstraction. At the Reusable Level, the composition of LOs is expressed according to a relationship-based approach, while at the Didactical Level, workflow-based and rule-based supports are provided. SCORM supports all the three approaches as well, however --as it should be clear from what we presented in the previous sections-- the lack of a clear definition of different levels of abstraction in which these approaches are exploited, and the presence of various overlapping concepts, makes it difficult to be used.

Conclusions

We see SCORM as a good opportunity to support interoperability among e-learning tools since it enables the definition of a data model that can be shared among them. However, we have noticed some weaknesses in such a data model. These weaknesses mainly concern the way learning resources can be structured and made available for reuse.

In our vision all the learning resources have to be thought as LOs, that is, entities described by proper metadata that can be recursively composed. Thanks to the recursive composition mechanisms, reuse both within a single platform and among platforms can be greatly enhanced: A LO at any level of composition can be reused and composed in another context. The definition of proper metadata can support not only browsing and reuse of LOs, but also their installation and execution, and also enable tutoring and evaluation.

The Virtual Campus project provides an implementation of the aforementioned concepts. Moreover, it tries to enhance the SCORM run-time environment, exploiting a workflow engine to guide Learners through the instructional paths.

We rely on model-oriented strategies to improve interoperability. However, we are aware that adopting standard interfaces is of course a crucial aspect. Therefore, as a future work, we plan to include the Simple Query Interface specification into Virtual Campus. Another aspect that merits further investigation is the definition of proper guidelines to support Authors and Teachers in the design of LOs. Clearly, the more their LOs correspond to fine granularity learning materials, the more such materials are reusable and applicable in various contexts. Indeed, the mechanisms to compose fine granularity LOs are essential in this case in order to avoid the difficulties of having a huge, non-organized collection of LOs.

Acknowledgements

The Virtual Campus project has been funded by Microsoft Research, Cambridge (UK). We thank Sam Guinea who participated to the implementation of the Virtual Campus authoring environment, and helped us reviewing the paper. We also thank the editor and the anonymous reviewers for their helpful feedbacks and suggestions.

References

ADL (2004a). *SCORM 2004 2nd edition - overview*, retrieved March 22, 2006 from http://www.adlnet.org/downloads/70.cfm.

ADL (2004b). *SCORM content aggregation model - version 1.3.1*, retrieved March 22, 2006 from http://www.adlnet.org/downloads/70.cfm.

ADL (2004c). *SCORM run-time environment - version 1.3*, retrieved March 22, 2006 from http://www.adlnet.org/downloads/70.cfm.

ADL (2004d). *SCORM sequencing and navigation - version 1.3.1*, retrieved March 22, 2006 from http://www.adlnet.org/downloads/70.cfm.

AICC (2004), *CMI001 - AICC/CMI Guidelines For Interoperability v4*, retrieved March 22, 2006 from http://www.aicc.org/pages/down-docs-index.htm.

Barbieri, T., & Paolini, P. (1999). WebTalk: a 3D Collaborative Environment to Access the Web. In *Proceedings of EUROGRAPHICS '99*, 111-113.

Brase, J., Painter, M. & Nejdl, W. (2003). Completing LOM - How Additional Axioms Increase the Utility of Learning Object Metadata. In *Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies (ICALT'03)*, Athens, Greece.

Carchiolo, V., Longheu, A. & Malgeri, M. (2001). Learning through ad-hoc formative paths. In *Proceedings of the International Conference on Advanced Learning Technologies (ICALT)*, Madison, Wisconsin, USA.

Cesarini, M., Guinea, S., Sbattella, L. & Tedesco, R. (2004). Innovative learning and teaching scenarios in Virtual Campus. In *Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications (ED-MEDIA)*, Lugano, Switzerland.

Dongsong, Z. L., Zhao, J., Zhou, L. & Nunamaker, J. F. (2004). Can e-learning replace classroom learning? *Communications of the ACM, 47* (5), 75-79.

Duval, E., & Hodgins, W. (2003). A LOM Research Agenda. In *Proceedings of the 12th World Wide Web Conference*. Budapest, Hungary.

IEEE LTSC (2002). *IEEE Draft Standard for Learning Object Metadata*, retrieved March 22, 2006 from http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf.

IMS (2004). *Content Packaging Information Model, version 1.1.4 Final Specification*, retrieved March 22, 2006 from http://www.imsglobal.org/content/packaging/#version1.1.4.

IMS (2003). *Learning Design Information Model, Version 1.0 Final Specification*, retrieved March 22, 2006 from http://www.imsglobal.org/learningdesign.

IMS (2005). *Simple Sequencing v1*, retrieved March 22, 2006 from http://www.imsglobal.org/simplesequencing/index.html.

Knolmayer, G. F. (2002). Decision support models for composing and navigating through e-learning objects. In *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS)*, Big Island, Hawaii.

Padrón, C. L., Torres, J., Dodero, J. M., Díaz, P., & Aedo, I. (2004). Learning Web Services Composition and Learner Communities Support for the Deployment of Complex Learning Processes. In *Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'04)*, Joensuu, Finland.

Quarati, A. (2003). Designing shareable and personalisable e-learning paths. In *Proceedings of the International Conference on Information Technology: Computer and Communications (ITCC)*, Las Vegas, Nevada, USA.

Quemada, J. & Simon, B. (2003). A Use-Case Based Model for Learning Resources in Educational Mediators. *Educational Technology & Society, 6* (4), 149-163.

Rivera, G. M., Simon, B., Quemada, J., & Salvachúa, J. (2004). Improving LOM-Based Interoperability of Learning Repositories. In *Proceedings of the Workshop on Ontologies, Semantics, and E-learning (WOSE)*, Larnaca, Cyprus.

Sbattella, L. & Tedesco, R. (2004). Profiling and tutoring users in virtual campus. In *Proceedings of the 5th International Conference on Information Technology Based Higher Education and Training (ITHET '04*), Istanbul, Turkey.

Sicilia, M.-A., García, E., Pagés, C., Sánchez-Alonso, S., & Rius, A. (2004). Specifying semantic conformance profiles in reusable learning object metadata. In *Proceedings of the* 5th *International Conference on Information Technology Based Higher Education and Training (ITHET 2004)*. Istanbul, Turkey.

Simon, B., Massart, D., & Duval, E. (2004). *Simple query interface specification*, retrieved February 15, 2006 from http://nm.wu-wien.ac.at/e-learning/interoperability/query.pdf.

Steinacker, A., Faatz, A., Seeberg, C., Rimac, I., Hrmann, S., El Saddik, A., & Steinmetz, R. (2001). Medibook: Combining semantic networks with metadata for learning resources to build a web based learning system. In *Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunication (ED-MEDIA)*, Tampere, Finland.

Su, J.-M., Tseng, S.-S., Weng, J.-F., Chen, K.-T., Liu, Y.-L., & Tsai, Y.-T. (2005). An Object based Authoring Tool for Creating SCORM Compliant Corse. In *Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05)*, Tamkang University, Taiwan.

W3C (2004). *Composite Capability/Preference Profiles (CC/PP): Structure and vocabularies 1.0*, retrieved March 22, 2006 from http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115.

Yang J.-T., Yu, P.-T., & Chen, W. (2004). An Ontology-Based Course Editor (OBCE) for SCORM-compliant Learning Objects. In *Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'04)*, Joensuu, Finland.