Seamless production of interoperable e-Learning units: stakes and pitfalls

Nadia Spang Bovey and Nicolas Dunand

University of Lausanne, Bâtiment Provence, 1015 Lausanne, Switzerland
Nadia.SpangBovey@unil.ch, Nicolas.Dunand@unil.ch

Abstract. The modeling of a 28-week course in Information Theory using IMS Learning Design Level B specification proved its efficiency for describing complex learning scenarios. This article briefly summarizes the method used to create the real-life unit of learning. The experiment showed that, although various types of editing software and rendering engines are available, the resulting production process relies too much on computer specialists to be adopted as a strategy at the institutional level, and that the lack of integration exhibited by both the software and the engines in terms of Virtual Learning Environments prevents large-scale deployment.

Keywords: learning scenario, interoperability, IMS Learning Design

Introduction

The Swiss Virtual Campus is a national initiative for the promotion of eLearning in Higher Education. It provides funding to multi-partner teaching projects and technical infrastructure such as a secure authentication system for all Swiss university students and access to a commercial Virtual Learning Environment (VLE). Therefore, the question of the reusability and interoperability of the learning material was not, at first, perceived as a major issue. However, the scope and complexity of the courses made it impossible to rely on a unique product, and brought to light the necessity of elaborating strategies for later redeployment of both content and learning scenarios.

As far as content is concerned, IMS Content Packaging or SCORM compliant VLEs provide basic export/import tools that allow a relatively easy redeployment. Learning scenarios, which convey the teachers’ personal views and expertise, are, on the contrary, stuck in the tool where they were initially developed. Pedagogically speaking, learning scenarios are indispensably beneficial, but they cannot conceivably be manually rewritten in case of a VLE change. Therefore, the publication of the IMS Learning Design (hereafter, “IMS-LD”) specification seemed to be a good omen.

The experiences and points of view presented here are those of a center that provides pedagogical support to the teaching staff. Its main duties are non-technical. Designing learning activities and scenarios is a core practice, with or without the support of technology, and the involvement of coding specialists to ensure their sustainability cannot be regarded as a viable possibility. We definitively needed to establish a design and production process that integrates the issue of reusability and interoperability of learning scenarios from the very start and – unlike the industrial approach adopted by large institutes for distance education [1] – which relies on generic computer skills. We decided to experiment on the potentials and shortcomings of the methodology proposed by IMS. Even though we were aware that no compliant VLE was available, we hoped – and still hold the view – that the next generation of authoring and teaching tools would provide us with a suitable long-term solution.

The work presented here uses the IMS Learning Design version 1.0 technical specification to create a rich learning unit. The latter can account for all aspects of the learning
scenario, allowing the rendering of the finest interactions between all actors, thus providing the largest possible didactic liberty to the teachers and developers [2] [3]. The following text aims at detailing each step that is needed to translate a real-life academic course into an interoperable learning unit; it also will provide a walkthrough that will point out the difficulties arising from the process, showing why the seamless production of such learning units is not yet at hand.

**Approach**

The approach used to design the learning unit presented here was a three-step process involving three different people. To fulfill our expectations, this process should, however, involve two, or even one single step, performed by the teacher him or herself. Firstly, discussions took place between the teacher and a learning engineer in order to create a model of the course. When a common agreement had been reached, an UML-like activity-diagram of the course scenario was made, taking into consideration all roles involved in the learning/teaching process, mapping activities related to them, and showing links between activities. In a subsequent step, the concepts of the UML diagram were translated into a learning unit (an IMS-LD compliant file) using various software applications and some manual coding. The next parts of this section describe these steps in more detail, and signal the places where simplifying the process would be a decisive advantage.

**Course Description**

The course being modeled is an online course in Information Theory targeted at:
- students of geography, linguistics, and statistics;
- Bachelor- and Masters-level computer scientists, more familiar with mathematics, and focusing on general knowledge and applications of the theory.

The course is composed of ten modules, each consisting of different Information Theory topics (core modules: general definitions and major theorems, and specialized modules: applications to geography, linguistics, statistics, and informatics). Each module includes three levels of difficulty and mathematical abstraction: general knowledge (level 1), main results with only simple proofs (level 2), and all results with complete proofs (level 3). One given implementation of the course is then built by combining different modules at different levels. For instance, the course focusing on linguists includes all common modules at level 1, and the linguistics module at level 1; the bachelor level course for computer scientists includes all common modules and all informatics modules at level 2. Similarly, a course for mathematicians would include all common modules and the statistics module at level 3.

During the first semester, students are required to study the fundamentals of Information Theory on their own, and must, in addition, explore its possible applications in their specialized fields of study. They are, therefore, provided with:
- a limited number of introductory or synthesis recorded videoconferences;
- the text of the relevant modules;
- concept maps which help the visualization of the organization and content, and revisions of important definitions;
- sets of control questions to verify their levels of understanding;
- various animated and interactive examples in terms of demonstrations;
- problems and exercises, with solutions;
- the help of one or more online tutor and of the professor, via forums.

During this first semester, the communication tools are mainly used to collect and answer students’ questions. Each computer scientist and engineer is then required to take an exam, while human sciences students begin a second semester, in which these tools are used to assist them in their personal work, allowing them to share knowledge and to monitor attendance.

The second semester is devoted to personal work based either on an individual or a group project, or on a study program leading to a traditional oral or written exam. Students use the VLE to deposit successive versions of their work, which can be examined by the whole group, and to inform the teacher of their pro-
gress. Individualized exam “contracts” are elaborated and negotiated online, using a learning journal as an asynchronous communication tool.

**Pedagogical Modeling**

According to the IMS methodology, pedagogical modeling should begin with the creation of a textual and visual representation of the course. [4]. Since no UML editor can generate automatically an IMS-LD compliant XML file, the modeling of the 28 weeks of teaching and learning activities had to be divided in two separate operations. The activity diagram was designed as a visual representation of the roles, activities, decision points, and timeline of the course.

Although necessary to obtain a full picture of the sequence of activities, this step is only a preliminary modeling of the teaching and learning activities. In an ideal world, this would not be an isolated operation, and would provide the modeler with a skeleton of the actual course in an IMS-LD compliant format, to which the learning environment, as well as the variables and conditions, could easily be added. MOTplus [5], although not an UML editor, heads in this direction and allows the generation of IMS-LD level A-compliant XML files. However, the modeling of a complex sequence of activities with this tool remains a task that is too disconnected from the daily practice of the average teacher, if only because the modeler needs to master the rules and constraints used for pedagogical modeling in the software, and a fair knowledge of the specification itself.

**Learning Unit Design and Conception**

Once the UML diagram of the pedagogical process had been established, the learning unit itself had to be created. This was achieved mainly by using the Reload Learning Design Editor software, which provides a comprehensive and intuitive graphical UI, allowing the creation of a learning unit complying with the Learning Design specification [6]. Some additional coding (i.e. the writing of XHTML files providing two-way interaction between the end-user and the learning scenario), however, needed to be done from scratch.

The learning unit that had been produced consists of a ZIP package containing an XML manifest file and all files needed to render the course properly: instructions, content, etc. The learning unit produced adopts the level B of the IMS-LD specification; this means that, for scenario flexibility purposes, the use of variables and of conditional events is possible.

The Reload Learning Design Editor allows the user to build a course scenario based on the IMS-LD concepts. These concepts include an approach based on roles and activities; each participant, being related to a role (i.e. teacher, tutor, student, etc.), performs a particular activity that is based on his or her role and personal preferences. Such activities can then be associated with various environments, which can provide facilities such as communication tools (e-mail, discussion forums, and so on), tracking and indexing functions, or simply the means to supply the user with additional content.

Building a course scenario based on these concepts (roles, activities, variables, conditional events) from an UML workflow diagram first requires identifying roles and activities: this is the easy part. The next step demands some reformulation, as a description of the various relations between the different roles and activities is needed. This reformulation consists mainly of converting human semantics into the limited number of concepts available from the IMS-LD specification.

The IMS-LD specification uses roles and activities to define role-parts, which are the building blocks of the learning scenario: each role-part associates one role to one activity. Several of these role-parts can then be grouped into acts: an act is a set of role-parts that takes place (that is, begins and ends) at the same time for all actors of the scenario, thus providing synchronization abilities. In the unit developed here, one act groups all activities of one academic semester (which, of course, takes place at the same time for everyone), while some other synchronization features are achieved using variables and conditions. It is indeed possible to use conditions on the values of variables to make visible or invisible
elements of the learning scenario, such as tools, content, or activities.

Firstly, all activities are built; each one is then mapped to one role, thus defining role-parts. The use of conditions allows making various elements visible or invisible depending on any constraint. Furthermore, the use of variables allows the precise tracking of students by the monitoring of their variables, which for instance may keep track of how (or if) some activity has been performed or of their learning preferences. To allow the setting and visualization of variables by users playing the learning scenario, XHTML files have to be written from scratch.

At this stage, the most arduous task is that of the learning designer first having to write down the complete learning scenario, intertwined with all of the variables and effects of the various conditions (i.e. availability of the various activities). Writing the IMS Learning Design itself becomes possible only once this work has been accomplished. It is, thus, very difficult to make even small modifications afterwards, such as adding or removing an activity. Besides, one thing was found to be missing from the IMS-LD specification: the ability to form groups of learners. Grouping learners in order to facilitate active learning, (e.g. problem-based or project-based) is, indeed a common practice, and the impossibility to describe such interactions in an easy way with the specification is, in one sense, a shortcoming which should be tackled.

![Figure 1](image.png)

Fig. 1. Different activities and environments are presented to actors of different roles. On the left side are shown the activities and corresponding environments which are presented, at some point of the scenario, to a tutor, allowing him or her to see, among other things, the various students’ choices made so far. On the right side of the figure are shown the activities available at the same time to a student.

**Running the Learning Unit**

In our institutional context, the primary purpose of modeling units of learning would be to ensure their portability from one VLE to the other. Although sharing learning objects and scenarios in repositories might be an additional incentive, our main worry is the issue of durability, and this is where the experiment turns out to be inconclusive. Getting the learning unit running necessitates an IMS Learning Design rendering engine, as, at this
moment, no course management system allows the importation of such a unit. The CopperCore engine [7] was chosen to test the produced learning unit because it was able to render most elements of the learning scenario. It, however, provides only a basic rendering layer; it was, indeed, not aimed at providing a virtual learning environment, but a low-level Learning Design engine (which could be incorporated into a real learning environment).

The first step to get the learning scenario running is, then, the instantiation step. That is, one needs to map actual users to the roles of the scenario, thus creating one scenario instance. Users have to be manually added to the different learning scenario roles, which, again, can be quite a hassle, as this has to be done within a command line interface. Once all needed users have been added, the CopperCore engine allows users to run the learning scenario. Fig. 1 shows the various activities proposed at the same moment to two different actors having different roles within the learning unit.

A need for integration

In our real-life settings, both the production and use of the learning unit would, however, have to be different to be accepted as a viable strategy. Going through all of the following steps writing a UML diagram of the learning scenario to identify the needed roles and activities

- writing down the whole IMS Learning Design in order to correctly use the variables and conditional events
- writing the IMS-LD compliant files with Reload Learning Design Editor (some of them, from scratch)
- rendering the learning scenario with the CopperCore engine

is certainly suitable for learning and testing the specification itself.\(^1\) Except for a small number of projects, the courses would have to be designed and produced by the professors or their teaching assistants. Therefore, the generation of IMS-LD files should be embedded into simple design tools, preferably within the VLE, in a way similar to that implemented in LAMS (Learning Activities Management System) [8]. Although not based on IMS-LD, LAMS illustrates a concept that might bridge the gap in a context where teachers are the main producers of technology-enhanced courses. While the framework, tools and sequencing of the course is provided by the VLE,\(^2\) the teacher models one learning activity after the other, dragging icons representing the tools that are needed to proceed with the activity on a design screen where instructions, resources and conditions can be added in a very natural way.\(^3\) The relevant product would be an IMS-LD compliant VLE with learning activities design functionalities. Those would provide visual and intuitive means to create sets of instructions linked to the relevant resources and tools, and be able to automate the generation of the XML files needed to redeploy the course in another compliant VLE. Specialized help could, thus, be restricted to a few highly sophisticated courses and the specification be adopted on a large scale.

Conclusion

At the University of Lausanne, the production of technology-enhanced courses is done mainly by the teachers themselves. Therefore, the issues of sustainability and interoperability of the learning scenarios, although fully appreciated by the eLearning support staff, must be kept behind the scenes. Regular teaching staff members would very easily be discouraged by additional technical constraints imposed upon their work. In such an institutional context, the modeling of the Online Course in Information Theory according to the IMS Learning Design specification served two major objectives. The first was to test the adequacy of the specification to describe real-life courses that were not designed on purpose, and the second was to identify the conditions needed for the adoption of the specification to

\(^1\) A method for a full evaluation benchmark of expressiveness and suitability of IMS-LD is proposed by Caeiro-Rodriguez et al [9]. The authors designed a comprehensive methodology based on pattern recognition.

\(^2\) Moodle, Blackboard, Sakai and WebCT in a near future.

\(^3\) For a technical discussion of LAMS and IMS-LD, see the article by Berggren et al [10].
ensure the portability of the online courses on a large scale.

The result of the experiment is promising, but it also clearly shows that the natural integration of the specification with actual practice is not yet at hand. Surely enough, IMS-LD proved adequate to successfully model the 28 weeks of learning activities and all the related tools and interactions. The complete process required a three-person team composed of a professor, a learning engineer, and a skilled computer staff member willing to dig into the specification, who produced the level B-compliant XML file using the Reload Learning Design Editor. The resulting learning unit can be run using a rendering engine such as CopperCore, with each role correctly performing the intended actions with the adequate tools.

However, although IMS-LD seems to provide a potential solution to a problem encountered by many Higher Education institutions, its practical use is hampered by a much too complex flow of production. The UML modeling and the translation of the activity diagram into the IMS-LD concepts of activities, activity-structures, and proprieties are out of reach of the typical staff in an educational context. Unless both the visual modeling of the learning activities and the generation of the compliant XML files can be integrated into the usual pedagogical design practice of the teachers, the large-scale use of the specification will remain an unviable option in our institution.

While assumedly a technical and commercial challenge, the missing integrating product can easily be described: an IMS-LD compliant Learning Management System that would provide the course framework and set of tools, equipped with a LAMS-like visual learning activities design tool which would allow the teacher to sequence simply activities and type in instructions and resources references. In addition, the VLE would be able to generate a proper IMS-LD file with all of the necessary resources and proprieties, ready for importation into any other compliant VLE. A natural and intuitive production process could then be implemented, ensuring that teachers’ work and creativity are not at risk of being lost.

References