OPERATIONALIZATION OF LEARNING SCENARIOS ON EXISTENT LEARNING MANAGEMENT SYSTEMS
The Moodle Case-study

Aymen Abedmouleh, Pierre Laforcade, Lahcen Oubahssi and Christophe Choquet
LIUM, IUT de Laval, 52 Rue des drs Calmette et Guérin, F-53020 Laval Cedex 9, France

Keywords: Learning Management System, Moodle, Learning Scenario, Visual Instructional Design Language, Operationalization, Domain Specific Modeling, Model Driven Engineering.

Abstract: Despite the increasing number of Technology Enhanced Learning platforms (eg. MOODLE) and their wide spreading, the operationalization of learning scenarios is still a problem for teachers. We aim to facilitate their implementation on existent platforms. We propose an approach based on the formalization of the implicit instructional design domain language embedded by these Learning Management Systems. It consists to identify and formalize this specific language in order to use it as a mean of communication with external design tools without losing the semantic of the designed scenarios. The originality of our approach relies in performing the scenarios operationalization by the development of a communication API based on the formalized language of the platform. Our proposal is also based on the application of theories and practices from the Domain Specific Modeling domain in order to formalize the domain language, to specify some graphical languages on top of it, and to help in the development of dedicated graphical editors. This paper details the implementation of our proposal (API and first editor) on the MOODLE platform.

1 INTRODUCTION

Most of academic organizations provide teachers with some Learning Management Systems (LMS) for improving or completing their face-to-face courses by some additional activities from simple resources access to scheduled communication or online assessments. However, the management and the appropriation of theses platforms by practitioners are complex tasks (Al-Ajlan and Zedan, 2007). Some LMSs, such as MOODLE, do not hide this complexity and struggle to provide assistance and appropriate means despite their numerous and active communities. Also, each platform embeds both a specific instructional design paradigm and a specific pedagogy. However, practitioners are not familiar with this implicit instructional design domain (Martinez-Ortiz et al., 2009). They are not able to implement scripts required by these platforms (Mekpiroona et al., 2008) or to adjust the many parameters of the form-based interfaces. The teachers-designers could be disappointed by the semantic distance between Educational Modeling Languages (EML) and the LMS because they cannot implement their scenarios specified by these EMLs. Several technical approaches provide design tools which do not instrument the operationalization. As a result, the practitioners are still looking for design and implementation approaches more appropriated to their practices and expertise.

We discuss in this paper the issue of the learning scenarios design for educational platforms. We present a new approach that facilitates the implementation of learning scenarios on these platforms. This approach consists (1) in identifying and expliciting the instructional design languages of platforms and (2) exploiting them by providing practitioners with some new tools based on these external languages in order to facilitate the design of learning scenarios.

Next section presents the operationalization activity and an overview of the current approaches for automating this activity.

2 EXISTING APPROACHES

The operationalization of learning scenarios consists in implementing teachers-intended scenarios on one TEL (*Technology Enhanced Learning*) environment. It resumes some manipulations as creation of activities, selection of participants, allocation of

Abedmouleh A., Laforcade P., Oubahssi L. and Choquet C..
OPERATIONALIZATION OF LEARNING SCENARIOS ON EXISTENT LEARNING MANAGEMENT SYSTEMS - The Moodle Case-study.
DOI: 10.5220/0003486001430148
In *Proceedings of the 6th International Conference on Software and Database Technologies* (ICSOFT-2011), pages 143-148
ISBN: 978-989-8425-77-5
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

roles and selection of required services and content for scenarios (e.g. pedagogical resources). However, the operationalization of learning scenarios is not only an engineer activity. It aims to translate teacher's intention and relative pedagogical semantics on a TEL system. Two approaches can be followed: manual or automatic ones.

Most of learning scenarios are manually implemented. It consists firstly in choosing the participants, then attributing roles foreseen by the scenario to the proper participants, and finally selecting the services and contents required by the scenario. This approach requires the intervention of a pedagogical engineer or an expert to make the necessary manipulations in the target platform.

In contrast, the other approach aims to implement automatically learning scenarios on TEL environments. The intervention of LMS experts or engineers is not necessary. Nevertheless, this approach requires an infrastructure for interacting with the LMS and taking in charge the creation and configuration of the working spaces, as well as the activity performance, starting from a formalized description of the task. To this end, such approaches require a 'language' for specifying the learning scenario, and a binding technique (or a formal language mixing both) for allowing the machine-readability of scenarios.

We are focused on this last approach. Relevant works fall into four categories: standard oriented approach (as IMS-LD (De Vries et al., 2006) or CopperCore (Berggren et al., 2005), practitioners oriented approaches (as COLLAGE (Hernández et al., 2006) or LDL (Martel et al. 2007)), approaches proposed for specific platforms (as LAMS (Delziel et al., 2006)) and the hybrid approaches based on Model Driven Engineering techniques (as Bricoles (Caron et al., 2005)). The analysis of these approaches leads us to observe that:

- The different proposed solutions do not fit with the teachers-designers needs we mentioned, excepting the LAMS one which partially satisfies them with its user-friendly interface. Nevertheless, LAMS editor integration into existent LMSs does not take advantage of the potential internal semantics embedded in the platform: it requires to add to LMSs a new runtime engine with its dedicated course format.
- The COLLAGE proposition is interesting because the collaborative design patterns proposed to practitioners have been specified and developed on top of the IMS-LD standard: semantics about concepts/relations transformations have been taken into account when building the patterns; these patterns are this fully-compatible with IMS-LD. In the other hand, operationalizing COLLAGE models relies on operationalizing IMS-LD ones. Unfortunately, most of existing platforms are still not compatible with this standard (Berggren et al., 2005)(Burgos et al., 2007).

Research works dealing with exportation or transcription of learning scenarios have highlighted the semantic learning design gap that appears when considering learning scenarios concepts and platforms features (Caron et al., 2005). Such scenarios transcriptions lead to lose some informations when binding the source scenario on an LMS. This conceptual gap is inherent to the transformation process when both languages have been elaborated with no reciprocal relations.

Teachers naturally decline any tools or approaches that are not able to facilitate the course design on their platform. It seems that current research propositions have not yet reached a level of maturity for allowing teachers-designers to implement theirs scenarios.

3 OUR APPROACH

Current propositions rely on a same underlying idea about evolving existent Learning Management Systems by large add-ons (editors or runtime engines) and new semantics in order to integrate learning design standards or to improve the design. We do not aim to add new semantics to LMSs.

In contrast, we assume that each LMS is not pedagogically neutral and that it embeds an implicit language for describing the process of designing a learning activity. Thus, our proposition is based on the idea that this language can be identified and explicitly formalized in a computer-readable format. We propose in addition to use this formalization as a binding format for various external tools which will focus on different designing facets as well as interoperability purposes between various LMSs expliciting their instructional design semantics.

Nevertheless, LMSs have to be able to import/export learning scenarios in conformance with their own language: current platforms have notwithstanding to evolve in order to offer this new functionality. From an LMS viewpoint, our proposition is to add a similar 'import/export' functionality like the SCORM (Gonzalez and Anido, 2010) one but with the LMS language itself (as it is already proposed for specific sub-domains like the quiz one for MOODLE). We propose so a kind of self-compliance format specific to each LMS. This will warrant e-learning tools developers that they could exploit this explicit language (that will have to be accessible through an XML schema for example) for communicating with the LMS.

Operationalizing a learning scenario from this LMS-centered viewpoint will consist then in the importation of a learning scenario formalized in conformance to this explicit LMS language. It will not require the addition of a new runtime-engine like the SCORM/LAMS or other approaches.

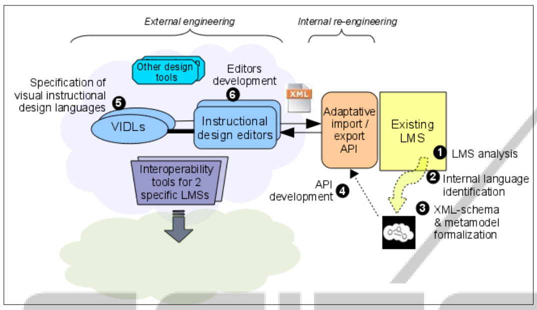
External engineering
Internal re-engineering
Other design tools
Editors development
6
Specification of visual instructional design languages
5
VIDLs
Instructional design editors
XML
Adaptative import / export API
Existing LMS
1 LMS analysis
2 Internal language identification
3 XML-schema & metamodel formalization
API development 4
Interoperability tools for 2 specific LMSs

Figure 1: Our external approach for improving learning designs on existent Learning Management Systems.

We also propose an original TEL-centered *Model-Driven Engineering* and *Domain-Specific-Modeling* (DSM) (Kelly et al., 2008) approach both to identify/formalize the LMS language and to use it as a basis for the elaboration of LMS-centered *Visual Instructional Design Languages* and their dedicated graphical authoring-tools. From a metamodeling viewpoint, every LMS language can be considered as composed of an abstract syntax (formalized as a metamodel and additional well-formed rules), a concrete syntax (the machine-readable textual notation that will be used for the binding of learning scenarios), and some semantics for both syntaxes.

The explicitation of LMSs languages allows the specification of VIDLs/EMLs on top of them. This approach will propose also a new opportunity to design and operationalize learning scenarios. A first step for this approach is to provide practitioners with some external learning design editors based on the LMSs languages. It is also important to provide practitioners with some learning design editors dedicated to the VIDLs built on top of the LMS language. Many VIDLs can be proposed for a same LMS language according to the LMS semantics they will include (whole or part). These LMS- centered VIDLs have to be composed of the same abstract syntax than the LMS language (same domain meta-model), but have to propose a visual notation (e.g. a concrete syntax) in order to facilitate thinking and communication for practitioners (human-interpre-table formalism). In contrast, the dedicated editors of these VIDLs have to manage the persistence of produced learning scenarios in the machine-readable format of the considered LMS (binding).

Our proposition is focusing on a DSM approach that aims to offer a practical solution to produce scenarios according to the semantics of the LMS language. DSM tools will manage the binding to the LMS machine-readable format. We propose to use these DSM tooling in order to elaborate some LMS-centered VIDLs and dedicated user-friendly editors based on the meta-model of the identified LMS language. We experimented such DSM tools, the ones from (Eclipse Project, 2011). They are able to specify all these artifacts (domain meta-models, graphical and textual notations, generation of dedicated editors, etc.). These tools have been experimented within several projects of different scopes and following practitioners centered viewpoints as well as TEL-centered ones (Laforcade et al., 2008; Laforcade, 2010).

Our LMS-centered solution guarantees that future produced scenarios will be implemented on the concerned LMS taking account the probability that this solution may restrict the pedagogical expressiveness of learning scenarios. But we assume that the explicitation of the internal LMS language will create the opportunity to build more practitioners-directed but LMS-centered on top of the initial LMS language. This external approach can also exceed the technological constraints and limits of existing TEL systems and offer more user-friendly computer artefacts to work with. Figure 1 illustrates our proposal detailed within this section.

4 THE MOODLE CASE- STUDY

We have chosen to apply our proposal on the MOODLE LMS. It is a distance learning platform based on a socio-constructivist pedagogy (Cocea, 2006). Our choice is motivated by: its open source code and its modular and extensible architecture, its large community of users and developers, and its use in our university.

4.1 Internal Language Identification

First of all, we had to identify concepts, attributes and relationships composing the abstract syntax of the MOODLE language. The relevant instructional design facet relies on the teacher's course space functionalities. We have performed an analysis work by combining three viewpoints: user interfaces analysis (what the designer sees), MOODLE database analysis (persistent objects and data associated to courses), and functional and technical analysis of the platform specific mechanisms linking human machine interfaces (HMI) and database (e.g. backup techniques, etc.).

By analysing the user interfaces, we have identified a set of challenges faced by teachers when using platforms. For example, changing informations in a Moodle course requires to fill a long list of *operationalization-oriented* parameters: full name, short name, identification number, course format, number of topics, etc. Some parameters are required and mandatory (full name and short name) for the creation of courses but others are optional or too technical (course registration parameters, file size, etc.). We have set up concepts used in space courses and also links or relations between them according to screens sequences. Briefly, a course is a set of editable sections that serve as structural components for defining and positioning activities and resources.

By analysing the MOODLE database, we have identified the system data organization. We have analysed about 266 tables of MOODLE 2.0. This has highlighted relationships between different data, in particular those used by many types of activities and resources available in a course. Crossed with HMI analysis, data analysis let us to identify properties, attributes, concepts (linked to some portions of screen-forms), and to verify and validate relationships between concepts. We have also checked off required data and optional ones (i.e. can be omitted without making database inconsistent or incoherent).

By analysing the platform functionalities, which has consisted to successively test all possible handling of the identified objects, attributes and relationships, we have characterized the MOODLE provided backup functionality for saving existing courses into an external format (here an archive containing many XML files and course resources, organized in folders). MOODLE also provides a restore functionality to import courses from external packages previously saved.

4.2 Language Explicitation

The explicitation of the specific MOODLE instructional design domain requires the choice of a concrete syntax for the representation of the language vocabulary and grammar.

Its widely use in interoperability standards and its use by the backup/restore functionality of MOODLE has oriented us to choose XML. Firstly, we have chosen to develop an instance of an XML document representing a complete course in order to help us to set representation choices (tags for concepts, sub-tags for attributes, etc.). Then, we have specified an XML schema that have been completed and refined in order to clarify all possible semantics (limited to the XML Schema expressiveness) that was not explicit in the previous XML instance: multiplicities of relationships, enumerated types, etc. These two steps have been primarily directed by the analysis of the package contents generated by the pedagogical course backup. We have proceeded by successive refinements: modifying/adding/deleting data in order to observe the generated course package files and to isolate the XML fragments to deduce the concrete syntax of our XML document. We have also taken into account the results of our previous analysis to specify relationships between different concepts. Finally, our XML instance which specify a full course, has a different structure and formalization in comparison to the numerous XML files generated by the backup/restore functionality.

The associated XML schema has been specified by using the more relevant Russian dolls design pattern. This schema formally represents the MOODLE domain language we propose as a communication format between external tools and this LMS. It may also be used to validate XML documents that will be imported or exported.

4.3 Import/Export Communication API

In order to import (and to export) learning scenarios into (from) the MOODLE platform, it is necessary both to develop and to integrate a new communication module (IN/OUT Course) in its design space. This new module provides an API between the external design tools and the LMS. It ensures the operationalization of learning scenarios. We have taken into account the possibility to successively perform import/export operations in order to adapt a course with external design tools. Therefore, we also took into account that external tools could only be compliant with all or part of the formalized language (to focus on specific design facets for example).

The communication module integrates two processes: an export process to generate an XML instance which specifies all course details and an import process able to operationalize scenarios specified in conformance with the platform language. In order to ensure the cooperation of these two processes in terms of first creation (first import into an empty course) and adaptations (by addition and deletion of concepts), we have initially tested it on the basis of two concepts (forum and label). Then, in a second step, we have extended the development throughout the whole language.

The first process has to deal with the course backup and its formalization in the specified platform language. Concretely, it relies on the backup functionality to generate the MOODLE course in an external package. Considering the XML format of the data package, we have developed an XSLT transformation using the XML schema we defined, in order to generate and to apply the necessary transformations rules on the different XML files. This process leads to generate an XML document instance conformed to our XML schema and handleable by the specific external design tools.

The second process leads to import scenarios in the platform. We have chosen to use the existing restore function of MOODLE to operationalize the scenarios. The general idea of the transformation algorithm we specified is to: (1) perform a course backup, (2) complete and modify this package with information specified within the XML scenario instance (with XSLT transformations), then (3) achieve the restore of this modified package. This process takes into account some constraints when the API is used to adapt an existing course.

Concretely, the XSLT transformations take into account four possible cases: (a) modification of some informations from an existing course concept; (b) creation of a new scenario concept not already present in the course (c) deletion of a course concept (i.e. concept appearing in the backup but not in the scenario to import with the additional information that the external tool was able to handle this element); (d) omission of a concept (i.e. concept only appearing in the backup files while knowing that the external tool was *not* able to deal with it). External tools that do not support all the LMS language have then to precise their "domain language perimeter of understanding", an imposed constraint for future external tools, by providing a subset of the XML schema we defined.

4.4 A First DSM-based Editor

From the XML schema dedicated to the MOODLE learning management system, we automatically generated an equivalent metamodel (figure 5) thanks to the EMF tooling. This metamodel can then be used as a basis for the development of Visual Instructional Design Languages, and their dedicated graphical editors, according to the DSM approach.

We have developed a first visual editor. It aims to graphically ease the specification of a course structure for MOODLE. We used the EMF DSM tool already used to formalize the MOODLE metamodel. EMF is a Java framework and code generation facility for building tools and other applications based on a structured model. Once you specify an EMF model (or metamodel), the EMF generator can create a corresponding set of Java implementation classes. In addition to simply increasing your productivity, building your application using EMF provides several other benefits like model change notification, persistence support including default XMI and schema-based XML serialization (the feature that interests us in order to offer a binding towards the platform XML schema), a framework for model validation, and a very efficient reflective API for manipulating EMF objects generically. Most important of all, given an EMF model definition, the EMF code generator can produce a fully functional editor tool with a tree-based interface that will allow to view instances of the model using several common viewers and to add, remove, cut, copy, and paste model objects, or modify the objects into property sheets.

Although we also plan to use GMF (Graphical Modeling Framework) as a complementary framework to EMF in order to generate user-friendly and graphical editors, we decided at first to focus on the EMF preliminary editor. By so, a full- generated prototype, as a plug-in for Eclipse, has then been generated by the EMF tooling in accordance with the MOODLE domain model. This editor does not require some computer skills but a graphical version with GMF will be more adapted for teachers/designers. From this EMF-based editor, visual scenarios are automatically serialized in an XML machine-readable format in compliance with the XML schema we propose (ie. the tree-based view is synchronized with the XML-formatted scenario; no other binding functions or services have to be performed).

We have experimented and tested this editor in relation to the previous communication API. We verified the efficiency of the four manipulations of MOODLE course scenarios (creation, deletion, omission, modification) and succeeded in operationalizing the successive models. Its is quite important to notice that appearing named resources have to be concretely added manually: only their labeled name is set. We planed others experiments with best-practices MOODLE courses as well as direct use with teachers-designers.

5 CONCLUSIONS

Increasing the spreading and the using of computer-based learning is a crucial issue of this decade. Nowadays, the supply of TEL environments as LMSs is effective and the technology is mature for their use. But most of teachers do not effectively use these new artifacts and claim for more suitable and easy-to-use tools. We do not think the solution to this problem only relies on teachers' training or on institutional initiatives and supports. Our proposal is (1) to simplify the authoring tasks and (2) to take more into account the teachers' requirements by the automation of the operationalization process of a course on a given platform. The idea that underpins our work is the explicitation of the domain specific language of a platform in order to emphasize the underlying pedagogical approach embedded in the platform while allowing the courses designers to focus, at the knowledge level, on pedagogical concepts rather than technical ones.

To reach this goal, we have decided to explore the potential of Domain Specific Modeling techniques. Our proposal is to externalize the design of a learning scenario by the mean of editors, reifying a Visual Instructional Design Language based on the instructional design language of the targeted platform, and to develop a bridge between this editor and the platform for ensuring the operationalization of the scenario. This paper details the architecture and the functionalities of this kind of bridge, an API for a platform based on its implicit language. We have tested this approach with MOODLE. This API, is able to both import and export learning scenarios.

For going farther this first result, we actually work on two directions. One is to define a visual instructional design for MOODLE and to develop its dedicated graphical editor. To this aim we are working on exploiting the Graphical Modeling Framework tool in order to specify a graphical notation and a mapping model with the existent domain model. We also plan to evaluate the usability of the editors by teachers-designers and to experiment them on concrete learning situations case studies. The second direction is to study at least one another LMS, to repeat the same approach (ie. identifying and formalizing the internal instructional design language, develoment of communication API, defining of VIDLs and externals editors) in order to evaluate the possibilities of interoperability between two different technical frameworks, helped by Model Driven Engineering techniques of models transformations.

REFERENCES

Al-Ajlan, A., Zedan, H., 2007. E-learning (Moodle) based on service oriented architecture. *In EADTU'07*. Lisbon, Portugal, 8-9 November, Lisbon-Portugal, vol. 1, 2007, pp. 62-70.

Berggren, A., Burgos, D., Fontana, J.M., Hinkelman, D., Hung, V., Hursh, A., and Tielemans, G. 2005. In Practical and Pedagogical Issues for Teacher Adoption of IMS Learning Design Standards in Moodle LMS. In *Teacher Adoption of IMS Learning Design Standards in Moodle LMS*. Journal of Interactive Media in Education, 2005.

Burgos , D., Tattersall , C., Dougiamas , M., Vogten , H., Koper, R., 2007. A First Step Mapping IMS Learning Design and Moodle . *Journal of Universal Computer Science*, vol. 13, no. 7 (2007), 924-931 .

Caron, P.A., Derycke, A., and X. Le Pallec, 2005. Bricolage and Model Driven Approach to design distant course. In *E-learn 2005*, pp. 2856-2863.

Cocea M., 2006. Extendibility of Educational Systems to Include a Learner-Adaptive Motivational Module. In *EATA NETTIES'06*, Timisoara, Romania, pp. 195-198.

Delziel, J.L., 2006. Lessons from LAMS for IMS Learning Lessons from LAMS for IMS Learning Design. In *ICALT'06*, Sydney,5-7 July, pp. 1101-1102.

Eclipse, 2011. Eclipse Modeling Projects, http://www.eclipse.org/modeling/, retrieved from May 2011.

De Vries, F., Tattersall, C., and Koper, R., 2006. Future developments of IMS Learning Design tooling. In *Educational Technology & Society*, 9 (1), pp. 9-12.

Gonzalez, V. B., and Anido, L. R., 2010. From SCORM to Common Cartridge: A step forward. In *Computers & Education* 54 Issue 1, January, pp. 88–102.

Hernández, L. D., Villasclaras-Fernández, E.D., Jorrín-Abellán, I.M., Asensio-Pérez, J.I., Dimitriadis, Y., Ruiz-Requies, I., and Rubia-Avi, B. 2006. Collage, a Collaborative Learning Design Editor Based on Patterns Special Issue on Learning Design. *Educational Technology & Society*. 9(1), pp. 58-71.

Kelly, S., Tolvanen, J.P., 2008. *Domain-Specific Modeling*. ISBN: 978-0-470-03666-2. Paperback. 427 pages. Wiley-IEEE Computer Society Press.

Laforcade, P., Zendagui, B., Barré, V. 2008. A Domain-Specific-Modeling Approach to Support Scenarios-Based Instructional Design. In: ECTEL'08, Sept. 16--19, Maastricht Netherlands.

Laforcade, P., 2010. A Domain-Specific Modeling approach for supporting the specification of Visual Instructional Design Languages and the building of dedicated editors. In: *Journal of Visual Languages & Computing*, vol. 21, issue 6, pp. 347-358.

Martel, C., Vignollet, L., Ferraris, C., David, J.-P., and Lejeune, A. 2007. Modeling collaborative learning activities on e-learning platforms. In *ICALT'07*, pp. 707-709.

Martinez-Ortiz, I., Sierra, J.L., and Fernández-Manjón, B., 2009. Enhancing IMS LD Units of Learning Comprehension. In *the 4th International Conference on Internet and Web Applications and Services*. May 24-28, Venice, Italy.

Mekpiroona, O., Tammarattananonta, P., Buasrounga, N., Apitiwongmanita, N., Pravalpruka, B., and Supnithia, T. 2008. SCORM in Open Source LMS: A case study of Learnsquare. In *ICCE'08*. Taipei, Taiwan, pp. 166-170.