Specification of Visual Instructional Design Languages Dedicated to Learning Management Systems

Aymen Abedmouleh, Pierre Laforcade and Lahcen Oubahssi
LIUM, Avenue Olivier Messiaen, 72085, Le Mans Cedex 9, France

Keywords: Visual Instructional Design Language, Domain Specific Modeling, Learning Management System.

Abstract: Despite of the growing development of learning technologies into education, designing learning scenarios and exploiting them for setting up a learning situation is still a complex task. Visual Instructional Design Languages (VIDL) and their dedicated graphical editors have been identified as important conceptual tools for achieving more creative design solutions within a design process. In this article we propose the application of Domain-Specific Modeling tools for specifying and developing VIDLs and editors dedicated to specific Learning Management Systems. An experimentation concerning the Moodle LMS is discussed.

1 INTRODUCTION

The *Technology Enhanced Learning* (TEL) research domain has provided many solutions to support distant instructional design: *Educational Modeling Languages* (EML) facilitating the specification of learning situations as formal learning scenarios for delivering and exchanges purposes (Koper and Manderveld, 2004), *Visual Instructional Design Languages* (VIDL) (Botturi and Stubbs, 2007) focusing on the support of imagination, thinking, communication for practitioners communities, *Learning Management Systems* (LMS) providing an operational TEL environment for delivering online learning situations.

Despite of these tools and facilities, the learning design is still a complex task. The development of LMS systems has not decreased the complexity of design and learning processes in these systems (Martinez-Ortiz et al., 2009). Standards de facto like IMS-LD (Koper, 2006) have not succeeded in being integrated to the existent LMSs widely spread (Burgos et al. 2007). The VIDLs do not really allow to exploit the scenarios they ease to specify for automatizing the delivery and setting up of LMSs (Laforcade, 2010). Nowadays, teacher-designers within academic contexts are still designing their learning situations by directly using the parameters' forms-oriented screens of LMSs they have at their disposal.

In our research works we are interested in helping such practitioners to design distant or mixed (in relation to their face-to-face sessions) learning situations. We aim at providing them with specific VIDLs related both to the LMS they usually use and to their practices and needs. Such VIDLs have to meet the VIDLs added-values (visual notation improving the instructional design reflexion), the EMLs ones (formalization and binding), and those from LMSs (configurations and delivering).

Our proposition is based on the idea that every LMS embeds an implicit instructional design language. Our approach originally proposes a two steps approach: (1) identifying and formalizing this LMS language, and (2) exploiting this language for the specification of VIDLs and graphical editors. This LMS-centered approach is strongly relying on a *Model Driven Engineering* (MDE) and *Domain Specific Modeling* context (DSM) framework. This article mainly focuses on the second step. We then assume that the LMS language has been identified and formalized. Nevertheless, readers can also consider our proposition within a wide scope dealing with the use of a DSM approach for specifying visual languages on top of an existent XSD file.

2 BACKGROUND

2.1 Instructional Design

Instructional Design is the entire process of analysis of learning needs and goals and the development of a delivery system to meet those needs (Berger and

Abedmouleh A., Laforcade P. and Oubahssi L..
Specification of Visual Instructional Design Languages Dedicated to Learning Management Systems.
DOI: 10.5220/0004079501990204
In *Proceedings of the 7th International Conference on Software Paradigm Trends* (ICSOFT-2012), pages 199-204
ISBN: 978-989-8565-19-8

Kam, 1996).The specification of a learning scenario can be realized by means of an *Educational Modeling Language* (EML), which provides a framework of elements used for its formal specification (Koper and Manderveld, 2004). EMLs can be considered as authoring languages focusing on exchange, binding and delivery objectives.

Other EMLs propose a visual notation and focuses on specific instructional design theories or methodologies. These *Visual Instructional Design Languages* (VIDL) offer a support encouraging reflexion, communication, etc. (Botturi and Stubbs, 2007). However they do not systematically provide a binding support for formalizing models. Some of them provide then some partial translations towards some more abstract EMLs like SCORM or the *de facto* standard IMS-LD.

Learning scenarios are then deployed or executed in a *Learning Management System* (LMS). Nevertheless LMSs failed in providing EMLs support for automatic deployment. Past attempts proposed to deeply modify LMSs by adding them an execution engine dedicated to the considered EML (Berggren et al. 2005). Some other research works tried to bridge EMLs and LMSs by the means of web services (Dodero et al., 2010) or models transformation (Abdallah et al., 2008). Nevertheless they met some semantics losses inherent to the translation mapping. Finally, the most spread way to deliver learning scenarios consists in using them as a formal guide to directly set up "by hand" the equivalent course within the LMS.

2.2 Domain Specific Modeling

The *Domain-Specific Modeling* (DSM) (Kelly and Tolvanen, 2008) is a software engineering methodology for designing and developing systems. The DSM approach is an application of principles and techniques from the *Model Driven Engineering* domain. DSM involves the systematic use of graphical *languages* to represent the various facets of a system. They are specific to a domain and can be defined as the set of concepts and their relations within a specialized problem field. They offer primitives whose semantics are familiar to all practitioners in that domain.

Thanks to some past experimentations and studies about using DSM for specifying VIDLs we concluded in (Laforcade, 2010) that such DSM approach can help the emergence of communities of interests or practices sharing the same domain-vocabulary and formalisms. DSM tools can be used to support the specification of VIDLs and the development of specific editors.

3 LMS-CENTERED VIDLS WITH A DSM APPROACH

3.1 Context

Issues from existing approaches lead us to propose an original approach focusing on existent LMSs. We assume that LMSs are widely spread into academics institutions and that it is relevant to focus on helping teacher-designers in using them whereas propose yet another design solution that do not deal with binding or automatic deployment facilities compliant with these LMSs.

Our approach then follows two objectives: to facilitate the design of learning scenarios in accordance with the LMS abilities (hiding low level and technical details required by the form-oriented LMSs screens), and to propose a solution exploiting these scenarios as productive models for pre-configuring the corresponding courses within the targeted LMS. We then consider our instructional design proposition as LMS-centered.

In order to overcome the limits and constraints inherent to the technological and technical choices related to the development of the LMS, we propose to focus on an LMS external solution. However, a communication bridge is necessary between these external LMS-centered design tools and the LMS.

3.2 Formalization of the LMS Instructional Design Language

Our approach is based on the idea that every LMS is not pedagogically neutral but embeds an implicit instructional design language relying on specific paradigms and educative theories followed by the LMS providers.

We propose to identify and formalize them in order to exploit them as new specific formats for import/export exchanges between LMSs and external instructional design tools (similarly to the more specific formats some ones sometimes provide like the self-Moodle-format for importing quizzes).

In our mind, such self-labels can be considered as equivalent to the standard ones (SCORM, IMS-LD) because of their focus on instructional design but they have to exclude the managing of resources in order to be deployed as a self-contained XML file. From an MDE viewpoint, this LMS language could be considered as an abstract syntax (the instructional design entities, properties and relations), its related semantics, formalized with an XML-oriented concrete syntax. Our proposal requires an LMS modification: a specific import add-on has to be developed and added to the LMS. Nevertheless such extensions are generally allowed by most spread LMSs.

3.3 LMS-Centered Vidls following a DSM Approach

The explicit formalization of LMSs‘way of designing’ will allow tools providers to propose different design tools communicating with LMSs. Some ones could focus on delivering or implementation issues for LMS-compliant scenarios by means of transformations or direct binding facilities. Other tools could focus on LMSs interoperability by translating some source LMS-centered scenarios to a specific targeted LMS language.

In our research work our interest is to help teacher-designers that use to directly design their learning situations from the LMS interfaces. We then aim at developing specific external LMS-centered design tools helping them to focus on design aspects at a sufficient level of abstraction from a considered LMS (e.g. hiding some low-levels configurations required by LMSs). On one hand future-authoring tools could deal with some instructional design aspects in a first external design-time but, on the other hand, some low-level aspects will still require to be set up in a second design-time on the LMS.

We concretely propose to specify LMS-centered VIDLs and to develop external dedicated authoring-tools. According to the DSM approach we follow, such VIDLs specification can rely on the LMS abstract syntax previously captured by the XML schema. This concrete format acts as a base for the specification of VIDLs metamodels and as a binding target for the serialization of future produced models (machine-interpretable models). The automatic delivering of VIDLs-compliant models will be achieved by the means of both binding facilities (from authoring editors) and importation services (to add to LMSs). The DSM tooling will assist the specification of the VIDL domain model from the XML schema and they will guide the definition of graphical concrete syntaxes and mapping models linking abstract and concrete syntaxes as well as capturing other semantics. DSM tools also take charge of the editor code generation from domain, graphical and mapping models.

On one hand this LMS-centered approach allows to overcome the translation losses inherent to the semantics gap between the VIDL and the targeted LMS. On the other hand, such approach necessary limits the VIDL expressiveness and usages (design close to the LMS semantics in opposition to conceptual design close to the practitioners needs). Nevertheless, first practitioners’ feedbacks (from surveys and interviews conducted within the teachers-designers community from our academic institution using a Moodle-based LMS) argue in favor of our original position. They do not have well-formalized instructional design background and practices to support. They ask for very first design tools allowing an abstraction of the LMS low-level details.

4 LMS EXPERIMENTATION

4.1 The Moodle LMS

We chose to apply our proposal on the Moodle LMS. It provides a learning environment to create courses, define activities, manage and grade students and so forth. Moodle include many types of activities (as lessons, assessments, forums, databases, quizzes, etc.). Moodle has an open source code and has a modular and extensible architecture allowing the addition of new modules. It has also a large community of users and developers. The design of courses on Moodle is based on the setting up of many interfaces based on long forms mixing pedagogical elements with technical ones.

The identification and formalization of the implicit Moodle instructional design language have already been performed and discussed into (Abedmouleh et al., 2012) by combining three viewpoints: users interfaces analysis (what designer see), functional analysis (what the LMS can do), and database and other technical sources analysis (how the LMS realizes and persists the design components). The final XML schema we finally fixed has also been used to develop a dedicated Moodle module allowing importation of course contents. This module appears as a block in the course space for a teacher-designer. So it requires the context of an empty created course to be used. The importation/exportation process allows a kind of round-trip design process ensuring that configurations directly made using the LMS designing facilities (including low-level data) will be preserved and merged according to the changes realized outside the LMS.

4.2 Practitioners' Requirements

For a first experimentation we decided to focus on objectives and practitioners' needs allowing the specification/development of a prototypal VIDL and editor in order to verify our DSM approach and tooling. These are the requirements:

- to design graphically sections by spatially arranging them without a definitive ordering;
- to allow the drawing of connecting arrows between sections to represent their future ordering within the course;
- to propose in the palette the basics activities and resources facilities provided by the LMS;
- to allow the addition of these activities and resources into the sections to define their use without having to specify all the usual data required for each of them.

Practitioners concretely would like a diagrammatic-oriented authoring tool, specific to their LMS, allowing them to focus on the global design of their courses.

4.3 DSM Tooling

Since several years we use the open-source unified set of modeling frameworks, tooling, and standard implementations from the Eclipse Modeling Projects (Eclipse, 2012): EMF and GMF. Our experience proves us that final editors, developed thanks to this tooling, tackle the need for graphical editors about learning scenarios (Laforcade, 2010).

Nevertheless, this Eclipse tooling requires some expertise about DSM and MDE principles (meta-modeling at first). In order to customize the generated editor or design more complex user-friendly editors (e.g. for modeling various views or perspectives for a same learning scenario), developers will have to acquire a higher level of expertise about the frameworks and the underlying Eclipse RCP principles.

4.4 Using EMF

First of all, the domain model defining the abstract syntax of the VIDL to build has been directly specified from the LMS XML schema. Indeed, EMF provides such a facility. EMF also keeps a trace of the mapping in order to drive, when the model code will be automatically generated from the meta-model, the persistence of future models. This tackles our need for a binding facility.

The figure 1 illustrates an extract of the domain model (as a class-diagram representation). This model specifies that a *Course* is composed of one *Sections*, itself composed of ordered *Section* (the ordering is a propriety of the 'Ereference' between *Sections* and *Section*; it can be checked in the Eclipse 'property' view). *Section* can include many *Activities* (forum, workshop, chat quiz, etc.). All the Eclasses (Moodle, Course, and so on) are defined in the model in order to map to the corresponding ComplexTypes from the initial XML schema.

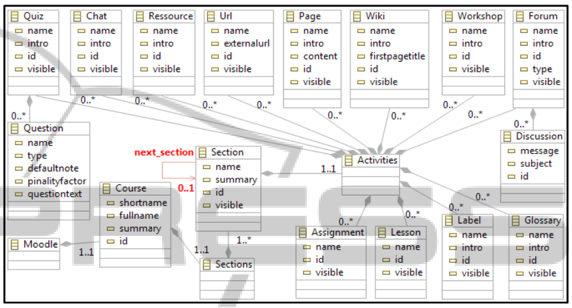

Figure 1: Extract of the domain model.

4.5 Using GMF

According to the GMF guidance, we had then specified the notation model in conformance with the practitioners needs. It specifies some inter-related drawing primitives (line, rectangle, compartment, etc.).

Next step concerns the specification of the tooling model: what users will have at their disposal in the palette, menus, toolbars, etc.

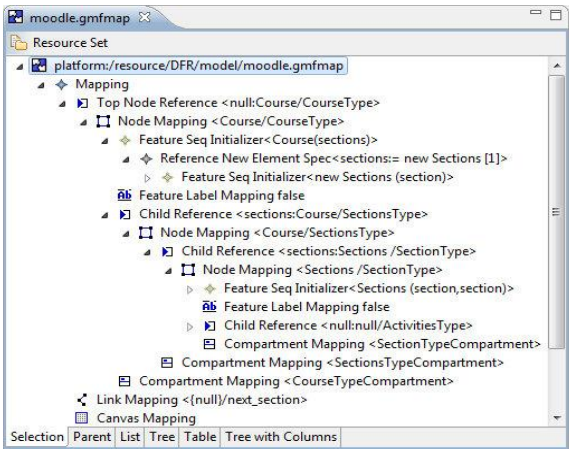

Figure 2: Extract of the GMF mapping model.

Finally, the mapping model is specified (Figure 2). It aims to link the three previous models. For instance, it specifies that the canvas (drawing space) maps to the ‘Moodle’Eclass from the domain model (itself in relation with the top tag “Moodle” defined within the XSD file), that the compartment within every ‘Section’/’Rectangle’ is related to the ‘Activities’ Eclass, and so on.

4.6 Resolving Some Meta-modeling and Binding Issues

We on purpose propose to relate a specific obstacle we met when dealing with the ordering of sections within a course. Their ordering is defined within the XSD file as the one of the “Section” tags from the future XML files. When getting the meta-model from the XSD file by the EMF importation service, this information has been translated as an Ereference between the Eclasses*Sections* and *Section* (both relating to the Sections and Section tags/complexTypes) with the property *ordered* fixed at ‘true’. Unfortunately it is not able to map an arrowed drawing line notation (from the graphical model) to this property. The GMF logic consists in mapping this information according to the creation sequence of visual *Section*(s) within the compartment of a *course*. This is concretely an issue because practitioners do not know in advance the concrete order of the sections they are designing. In order to use the authoring-tool for sketching the global design of the course, we have to consider separately the section instantiation order from their pedagogical one. To solve this issue we made the following modifications:

- addition of a self *nextSection* Ereference on the Section eClass with lower/upper bounds to “0..1” and a *transient* attribute to ‘true’ in order to inform the EMF persistence mechanism to not deal with it (in red color within figure 1).
- definition of the corresponding notation (arrowed line) into the graphical model, and of the corresponding tool for the palette;
- mapping specified into the related model for bridging together these new elements;
- addition of OCL constraints, to the mapping model, in order to disable self next relation on a section and to detect cycles.
- modification of the model code (generated at first by EMF) in order to redefine the behavior of the save/load methods: the save method have to re-order the *Section* instances from their *Sections* parent according to the *nextSection* relations specified; similarly, the load method have to set the *nextSection* according to the *Section* tags order parsed from the XML file.

4.7 The Final Authoring-Tool

From all the previous models (domain, graphical, tooling, mapping), GMF provides a generator model to give access to implementation customizations. Then this last model drives the GMF generative component to generate an editor code directly usable as a plugin for Eclipse (a *Rich Client Platform* standalone application can be further configured). This code use the one generated by EMF from the domain model that we have customized to solve the meta-modeling issues we met.

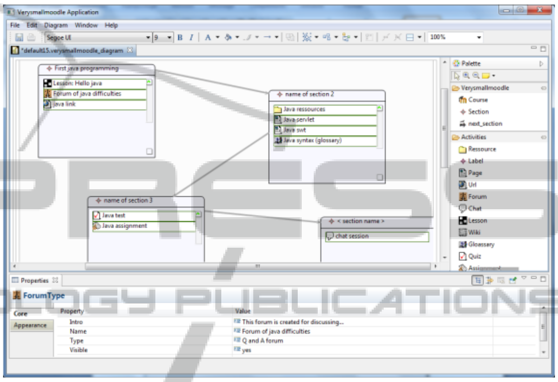

Figure 3: Screenshot of a scenario specified thanks to the generated authoring-tool.

The final editor (figure 3) can be used for two purposes: (1) to draw and then design learning scenarios as graphical models and (2) to visualize a learning scenario from another tool, which depends on the condition that this file/model is compliant to the schema used by the persistence facility. The models are both visualized by a diagram-oriented view and synchronously serialized as machine-interpretable XML file in conformance with the XML schema we started from.

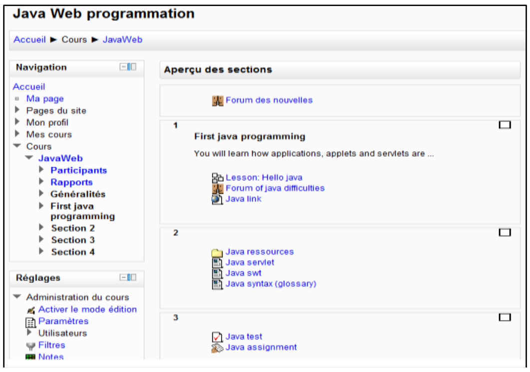

Figure 4: Screenshot of the Moodle course space after importing the previous model.

This graphical editor meets the practitioners’ requirements. Figure 3 shows a caption-screen of a learning scenario within the editor whereas Figure 4 shows the equivalent result after importing the learning model into Moodle. Teacher-designers can complete their design directly on Moodle by focusing on the low-level details not dealt with by the external design tool.

5 CONCLUSIONS

This article has presented and discussed a *Domain Specific Modeling* approach for the specification of *Visual Instructional Design Languages* centered on the *Learning Management System* semantics. The DSM theories and practices provide a very challenging trend for supporting the specification of VIDLs as well as the generation of dedicated visual editors. The main practical advantage and added value is to synchronize human-interpretable visual models with machine-readable persistence. The EMF/GMF-based editor delivers learning scenarios in a machine readable format compliant to a specific schema like the one we propose to identify and formalize from a specific analysis of the internal and implicit instructional design principles embedded within the LMS.

Nevertheless, the work conducted for now has also depicted the difficulty to adapt the resulting meta-model from the XSD file in order to capture the practitioners semantics and allowing a mapping in conformance to the notation targeted by end-users. We have just started a French ANR funded project in order to study the specification of more complex LMS-centered VIDLs that will capture practices and requirements closer to teachers-designers than the experimentation we discussed within this submission.

REFERENCES

Abdallah, F., Toffolon, C., and Warin, B., 2008. Models transformation to implement a Project-Based Collaborative Learning (PBCL) Scenario: Moodle case study. In *ICALT 2008*. Santander (Spain) pp. 639-643.

Abedmouleh, A., Oubahssi, L., Laforcade, P., and Choquet, C., 2012. Expressing the implicit instructional design language embedded in an LMS: motivations and process. In *the 15th IASTED International Conference on Computers and Advanced Technology in Education (CATE 2012)* Napoli, Italy. (To be published).

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

Berger, C., and Kam, R., 1996. Definitions of instructional design. In *http://www.umich.edu/ed626/define.html*. Retrieved fromMarch 2012 pp. 16.

Botturi, L., and Todd, S., (Eds.). 2007. *Handbook of Visual Languages for Instructional Design: Theories and Practices*. Information Science Reference.

Burgos, D., Tattersall, C., Dougiamas, M., Vogten, H., and Koper., R., 2007. A First Step, Mapping IMS Learning Design and Moodle. In *Journal of Universal Computer Science* 13 (7).pp. 924-931.

Dodero, J. M., Martìnez del Val, A., and Torres, J., 2010. An extensible approach to visually editing adaptive learning activities and designs based on services.In *Journal of Visual Languages and Computing* 21 (2010) pp.332-346.

Eclipse, 2012. The Eclipse Modeling Project. http://www.eclipse.org/modeling/. Retrieved from March 2012.

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

Koper, R., and Manderveld, J., 2004. Educational modelling language: modelling reusable, interoperable, rich and personalised units of learning. In *British Journal of Educational Technology* Vol 35 N°5.pp 537-551

Koper, R., 2006.Current Research in Learning Design. In *Educational Technology & Society*, 9 (1), pp.13-22.

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 and comuting*. pp. 347-358.

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*. Venice, Italy.