# *Improving the design of courses thanks to graphical and external dedicated languages: a Moodle experimentation*

*Pierre Laforcade, Aymen Abedmouleh*

*LUNAM University, LIUM laboratory (France), firstname.lastname@univ-lemans.fr*

## Abstract

Designing learning scenarios and exploiting them for setting up a learning situation on Moodle is still a complex task. Visual Instructional Design Languages and their dedicated graphical editors have been identified as important conceptual tools for achieving more creative design solutions within a design process. However, binding and operationalizing such learning models is not tackled for now. In this article we propose the specification and development of visual languages and editors 1/ dedicated to Learning Management Systems like Moodle, and 2/ taking into account specific instructional expressiveness from communities of practices. This original approach requires the identification and formalization of the LMSs implicit and internal instructional design language. It also requires the development and addition to the LMS of a dedicated API in order to expose this proprietary format for communicating inside and outside LMSs. Then, we propose to use this LMS semantics as a base for the specification of external instructional languages. We follow a theoretical, methodological and technological research framework based on the Domain-Specific Modelling domain. Such software engineering domain provides some relevant tooling and techniques for assisting and guiding the elaboration of domain-specific languages. All our experimentations and development prototypes have been focused on the Moodle platform. These first propositions and results are presented. Also, a dedicated funded research project has just started in order to deeper address the highlighted issues and objectives.

## Keywords

Instructional design, operationalization, Domain-Specific Modelling, Visual Instructional Design Language

# Introduction

Nowadays, *Learning Management Systems* (LMS) such as Moodle are in widespread use and are not restricted to intensive and distant formations. Most of academic organizations provide teachers with some similar platforms. Teachers can then use them for improving or completing their face-to-face courses by some additional activities from simple resources access to scheduled communication or online assessments. During the last decade, many languages and tools have been designed in order to provide support to the instructional design actors: *Educational Modelling Languages* (EML) (Koper and Manderveld 2004) for the specification of learning scenarios ensuring interoperability and reusing purposes, dedicated authoring-tools and execution engines / 'players', various techniques and frameworks for operationalizing learning scenarios or other educational models, Visual Instructional Design Languages (VIDL) (Botturi and Stubbs 2007) providing diagrammatic notations facilitating communication and thinking for practitioners, etc.

Despite all this potential support, the operationalization of learning scenarios on existent LMS like Moodle has not reached a mature level for providing practitioners with some appropriated solutions. Most of teacher-designers directly use and handle platforms for setting up courses from the pedagogical situations they mentally designed or, at best, which they documented following specific paper-centric guidelines. They have to understand the LMS underlying "way of thinking and designing". They have to appropriate the various form-based inter-faces and screens in order to identify and separate pedagogical features from technical / low-levels parameters.

We aim at helping practitioners to improve their capacity of abstraction and understanding of the LMS they have at their disposal in order to more focus on the instructional design aspects of the distant or mixed learning situations they have to set up. We then assume that every LMS embeds an implicit instructional design language. Our original objective is to provide practitioners with dedicated instructional tools based on this LMS



ISBN

language. Concretely, we aim at specifying and developing specific *Visual Instructional Design Languages*, and dedicated external authoring-tools, related both to a target-LMS and to their practices and needs. Such external approach has to meet the VIDLs' added values (visual notation improving instructional design reflexion), the EMLs ones (formalization and binding), and those from LMSs (compliant-semantics and communication). We propose the following steps for a given LMS: (1) identification and formalisation of its language, (2) its exploitation for the development of a communication bridge between the LMS and its external environment, and (3) specification of VIDLs and graphical editors based on top of this language. As a methodological and practical framework we propose to apply theories and practices from the *Model-Driven Engineering* (MDE) and *Domain-Specific Modelling* (DSM) domains.

The paper is structured as follows. Section 2 discusses the current internal/external instructional approaches about designing, binding and operationalizing when using an LMS. Section 3 is dedicated to the presentation of our LMS-centred approach for providing dedicated instructional design tools. Because Moodle is the main LMS addressed by our research works, this article will draw and discuss applications and results of our propositions in relation to it in Section 4. Before concluding, we present and discuss within Section 5 the funded GraphiT project that has just started. Indeed, this research project aims at prolonging our current results by shifting the problematic to the specification of VIDLs both LMS-centred and practices/needs-oriented (i.e. further from the LMSs semantics as concretely addressed for now).

## Background: instructional design and LMSs

Several approaches aim at facilitating the design of courses by focusing on the specification of learning scenarios (a formal description of a learning situation). Nevertheless they require some infrastructures and facilities for interacting with LMSs and for taking in charge the automatic creation and configuration of the corresponding courses.

The instructional design requires a 'domain language' or Educational Modelling Language (EML). Current EMLs propositions (mainly standards or de facto standards such as IMS-LD (De Vries et al. 2006), SCORM 2004 (Gonzalez-Barbone & Anido-Rifon 2010) (Mekpiroona et al. 2008), IMS-SS, etc.) allow the design of learning activities and unit of learning and propose a 'binding' technique (machine-readable scenario). Thus, they focus on "abilities" (interoperability, reusability, etc.). Differently, Visual Instructional Design Languages (VIDLs) are other specific instructional languages (e.g. MISA, CPM, E2ML, PoEML, etc.) (Botturi and Stubbs 2007) offering some visual notations from simple drawing with a few symbols to complex diagrams. These VIDLs focus on higher-level languages, i.e. with syntaxes and semantics closer to some instructional theories or to some specific communities of instructional designers' practices. VIDLs aim at supporting imagination, creative thinking, communication, etc. Because VIDLs are rather visual domain-specific languages focusing on human-interpretations, they do not systematically provide some binding techniques. Learning models are generally saved as proprietary files mixing learning scenarios data with graphical information. Some VIDLs or authoring-tools provide instructional designers with a standard binding by the means of a dedicated export service (e.g. MISA and CPM provide a model exportation towards IMS-LD). Nevertheless, resulting models are generally less expressive than the original scenarios because of the semantic gap between the source/target languages.

The COLLAGE approach (Hernández-Leo and al., 2006) proposes some collaborative design patterns to practitioners in order to specify scenarios. These patterns are specified and developed on top of the IMS-LD standard: their semantics and the LD-translation has been taken into account from the start. Thus, the operationalization of COLLAGE models then shifts to the problem of operationalizing IMS-LD models.

With the exception of all-in-one infrastructures (authoring-tool and execution environment specific to some TEL approaches / paradigms), none of the current design languages proposes a direct binding and operationalization to some existing LMSs. Some research works have dealt with partial transformations from practitioners-centred scenarios towards LMS-centred models (e.g. from PPC to Moodle (Abdallah et al. 2008), from IMS-LD to Moodle (Burgos et al. 2007). Target-models are based on the LMS semantics by the means of a subjective Moodle metamodel specified by researchers. Such transformations attempts showed a semantic gap leading to some information losses or lacks from the source models. However, they have also highlighted the relevance of applying techniques and tools from the Model-Driven Engineering domain (MDE). They also demonstrated that LMS semantics is difficult to identify and very subjective.



ISBN

Next issue of these works concerns the operationalizing of LMS-compliant models. Some attempts have been tried by means of a Web service-oriented approach (from LD to LMSs including Moodle in (Caron et al. 2005) and (Ortiz et al. 2007) but it requires the development of a dedicated service-oriented API within the LMS. Such technical approaches require the development of specific LMSs' add-ons more complex than developing additional modules to LMSs allowing such extensions.

The LAMS solution was at first a specific TEL environment with an internal graphical editor (Dalziel 2003) whose user-friendliness has been considered as one of the most appropriate and is used as a reference for the specification of recent graphical VIDLs or graphical authoring-tool dedicated to EMLs (e.g. the IMS-LD compliant GLM graphical editor (Neumann and Oberheumer 2009)). LAMS integration to existent LMSs has been studied (Lengyel et al. 2007) and has led to some concrete results like the one for Moodle: the LAMS graphical editor has been integrated with the same notations and semantics but produced scenarios or courses are related to a specific LAMS runtime engine also added to the LMS. This approach does not rely on the LMS internal semantics. Moodle designers have to choose the LAMS format for a new course and cannot use the LAMS editor for a basic Moodle course.

Similarly, the importation of SCORM content (1.3 being the most integrated format within existent LMSs while SCORM 2004 is rarer) requires choosing a specific SCORM format for a new course (in relation to a Scorm dedicated runtime-engine). A study has been realized about the integration of IMS-LD in Moodle (Berggren et al. 2005) (Burgos et al. 2007) but it concluded that such compliance would imply Moodle to evolve by extending its semantics and integrating additional and dedicated runtime-engines.

For now, because external learning design approaches fail to support the operationalization of their models, most of LMSs' practitioners within academic contexts are still designing their learning situations by directly setting-up the various parameters from the forms-oriented screens of the LMSs they have at their disposal. Existent LMSs are widely built upon a client-server architecture. Their underlying technological features, programming languages, frameworks, etc. limit and constraint the development of user-friendly internal editors.

# An LMS-centred external designing approach

Current proposals rely on a same underlying idea about evolving existent LMS by large add-ons (editors or runtime engines) in order to integrate external learning design standards or improving the internal design. Their common position is to evolve the internal learning design logic with some new semantics in order to bridge the gap between them. We do not aim to add new semantics to the domain-specific model embedded into the LMS.

We assume that LMSs are widely spread into academic institutions and that it is relevant to focus on helping teachers-designers in using them instead of proposing yet another design solution that do not deal with binding or automatic deployment facilities. 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 LMS screens), and to propose a solution exploiting these scenarios as productive models for pre-configuring the corresponding courses within the target-LMS. We then consider our instructional design proposition as LMS-centred. In order to overcome the limits and constraints inherent to the technological and technical choices related to the development of LMSs, we propose to focus on an LMS external solution although it requires then a communication bridge.

### Formalization of the LMS Instructional Design Language and communication bridge

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 sometimes provided 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. Our proposal also requires an LMS modification: a specific import/export add-on has to be developed and added to the LMS. Nevertheless such code extensions are generally allowed by most spread LMSs.

### Towards LMS-centred Visual Instructional Design Languages



ISBN

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-centred scenarios to a specific targeted LMS language.

In our research work our interest is to help teachers-designers that use to directly design their learning situations from the LMS interfaces. We then aim at developing specific external LMS-centred 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-centred 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).

### About the Domain-Specific Modeling framework

*Domain-Specific Modelling* (DSM) (Kelly and Tolvanen, 2008) is a software engineering methodology for designing and developing systems applying principles and techniques from the *Model-Driven Engineering* domain. It involves the systematic use of graphical DSM 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.

The DSM tools propose meta-modelling techniques capable of expressing domain-specific vocabularies (abstract syntaxes) and propose facilities to construct various notations (concrete syntaxes). Generally they also provide model persistence facilities. These editing frameworks and tools are supporting those techniques and many more customizations with minimal programming effort. As a result, these tools can generate powerful and usable dedicated editors for DSM languages. Because of some past experimentations and studies about using DSM for specifying VIDLs we concluded in (Laforcade, 2010) that such a DSM approach can help the emergence of communities of interests or practices sharing the same domain-vocabulary and formalisms. Indeed, DSM tools can be used to support the specification of VIDLs, with their specific editors.

From an MDE viewpoint, an LMS language could be considered as composed of an abstract syntax (the instructional design entities/properties/relations formalized in a metamodel), one or several concrete syntaxes (visual or textual notations), and semantics (the semantic domain and the mappings from abstract syntaxes to concrete ones). We propose an XML-oriented format in order to act as a concrete format to propose to practitioners (similarly to the Moodle XML format for specifying quizzes).
Such XML concrete syntax will also allow fixing the format for the VIDLs binding.
The main issue for specifying VIDLs on top of the LMS language relies on how their abstract syntaxes, concrete syntaxes (graphical notations) and semantics will be specified in relation to an LMS.

## Application to Moodle: first results

We chose to mainly focus our research works on Moodle. Indeed, Moodle has an open source code and has a modular and extensible architecture allowing the addition of new modules. It also has a large community of users and developers. Also, teacher-designers and pedagogical engineers involved in our experimentations are members of Le Maine University that provides them with a Moodle-based LMS (called UMTice).

### Identification and formalization of the implicit instructional design language

The identification and formalization of the implicit Moodle instructional design language have already been performed and discussed in (Abedmouleh et al. 2012a) 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 objective is to propose a detailed method in order to help LMS' communities in realizing this step and avoiding the proposition of subjective metamodels.



ISBN

At first, we have conducted many studies and experimentations from a teacher-designer viewpoint on several platforms and LMS systems (Moodle, Ganesha, etc.). Many uses (as the creation of courses, the specification of quizzes, the addition of learning resources, etc.) were realized in order to appropriate the LMSs. Although every LMS has its specific paradigm and instructional design language, differences were not critical and we were able to propose a common analysis process according to two strategies: the analysis of an existent course and the analysis of the creation of new courses.

Three main parts compose the analysis process: the HMI-centred analysis (Human-Machine Interface), the technical-centred analysis and the confrontation & formalization. The first one is centred on the HMI analysis. Performed with a top-down approach, it is conducted by three sub-sequential analyses (macro-HMI analysis, functional analysis and micro-HMI analysis). Each analysis has its specific features and provides its own models and formalisms. The composition sub-activity follows the three other ones. It consists in specifying the main model for the HMI-centred analysis. The second step concerns the technical-centred analysis. Several analysis methods could be adopted (data-bases, source code, course backup, etc.). During this step, the focus is put on the data persistence and handling. It results a specific conceptual model. Finally, the last process step concerns the confrontation between the HMI-centred and technical-centred models. The underlying objective is to specify the abstract syntax (e.g. the meta-model) of the instructional design language for the considered LMS. Although we aim at providing a very detailed process dealing with all particular cases in order to avoid a subjective analysis, the final confrontation/formalization step requires some specific skills about meta-modelling to prevent different resulting meta-models. Also the specification of an equivalent XML-schema from the meta-model offers several possibilities due to the XSD expressiveness.

### Development of an import/export module

The final XML schema we finally fixed from the formalization of the Moodle metamodel has also been used to develop a dedicated Moodle module allowing importation of course contents (Abedmouleh and al., 2011). 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 information) will be preserved and merged according to the changes realized outside the LMS.

The realized Moodle API exploited the backup/restore functionalities. Thanks to several hidden XSLT transformations we generate and adapt files compatible with the existent backup/restore PHP functions. Concretely speaking, the API we developed takes into account four possible cases: (a) modification of some information from an existing course concept; (b) creation of a new scenario concept not already presents in course (c) deletion of a course concept (i.e. concept appearing in the course but not in the scenario to import); (d) omission of a concept (i.e. same that deletion but with the additional information that the external tool do not deal with this element).

### Exploitation of the explicit language

For a very 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 (our concrete design rationale). We then discussed and fixed some very basic requirements conjointly with several teachers-designers using Moodle from our university: graphically design sections by spatially arranging them without a definitive ordering; allowing the drawing of connecting arrows between sections to represent their future ordering on the LMS; offering into the palette the basics activities and resources facilities proposed by the LMS; allowing 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. These practitioners' requirements are offering a weak pedagogical added value in comparison to directly using Moodle but, in contrast, they allow us to test and experiment our approach.

To this end we used the open-source unified set of modelling frameworks, tooling, and standard implementations from the Eclipse Modelling Projects (Eclipse 2012): EMF and GMF. From the XML schema dedicated to Moodle, we automatically generated an equivalent metamodel thanks to the EMF tooling. EMF is a Java framework and code generation facility for building tools and other applications based on a structured model. According to the GMF guidance, we then specified the notation model in conformance with the practitioners needs. It specifies some inter-related drawing primitives (line, rectangle, compartment, etc.). The next step concerns the specification of the tooling model: what users will have at their disposal in the palette, menus, toolbars, etc. Finally, the mapping model is specified. It aims to link the three previous models. From all the previous models (domain, graphical, tooling, mapping), GMF provides a generator model to give access to



ISBN

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 uses the one generated by EMF from the domain model that we have customized to solve some binding issues about the ordering of sections (discussed in (Abedmouleh et al. 2012b)): the VIDL domain model had been extended to separate the *creation* order of sections (when created into the drawing space) from their implicit order (deduced from the arrow relations path).

The final editor (left part of Figure 1) 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 1 shows a caption-screen of a learning scenario within the editor (left part) and the equivalent result after importing the scenario in Moodle (right part). The illustrated scenario is very basic: 4 sections have been drawn within the editor canvas and composed of various Moodle tools/services; these ones have been parameterized using the property view (only the parameters considered as relevant at this design-time); the sections ordering is explicitly defined by using the *next_relations* arrows from one section to another. Teachers-designers can complete their design directly on Moodle by focusing on the low-level details not dealt with by the external design tool.

We proposed the several practitioners involved to handle this editor (qualitative evaluation). They considered it with interests: visual notation more flexible to pick and position Moodle resources and tools than the equivalent form-based from Moodle, focus on the selection and ordering of activities rather than focus on their technical parameters. Nevertheless, such approach still requires them to load materials and set up low-level parameters in the Moodle's user interface. They also have to manage two different design environments (one external from Moodle) instead of one. In conclusion they ask for external design tools offering a more important pedagogical added value in order to counter balance the use of a new authoring-tool. We discussed also this point with some pedagogical engineers, from our University, that have to set some Moodle courses for teachers according to information they specified to them using a document-oriented pedagogical method. We identified different Moodle-abstracted pedagogical grains to consider for future: pedagogical activities ("online discussion", "web research", etc.) with different groups/groupings, pedagogical patterns (brainstorming, debate, etc.), courses types (complement to a face-to-face course, training course, etc.), and so on.

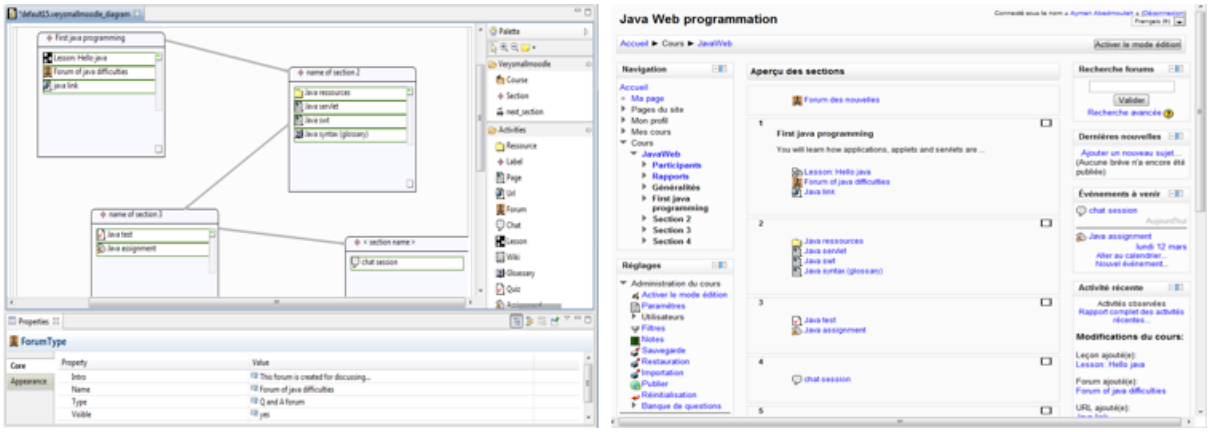


**Figure 1: A first DSM-generated authoring tool for Moodle (left part) and the resulting course set up after using the import/export module (right part)**

## The GraphIT Project

Current results have mainly focused on propositions very close to the Moodle semantics. The specific import/export module has also to be improved for taking into account the global Moodle metamodel we are completing thanks to the application of our analysis process. The very first VIDL we specified and its dedicated graphical authoring-tool has justified the DSM frameworks and the added value of some DSM tooling. Nevertheless the captured and tooled practitioners' needs are very close to the Moodle semantics and the LAMS notation. This proximity allowed us to start with a VIDL abstract syntax (metamodel) mapping the one from the



ISBN

explicit Moodle language but it also highlighted issues symptomatic of the expressiveness limits one can reach when both fixing the domain model to an existent one (for binding issues) and only trying to abstract the LMS semantics by specifying a specific concrete syntax (and its mapping to the abstract syntax).

Our LMS-centred approach with a DSM framework is original and promising but some further experimentation is required. We have also to shift our current courseware-design position to an instructional design one with more complex and LMS-abstracted practitioners' needs and practices (in relation to some pedagogical approach or educative theories to address). Solutions for specifying LMS-centred and practices-oriented VIDLs have to be experimented in order to increase the semantic gap between these instructional languages and Moodle. Developed authoring-tools usefulness and usability have to be experimented with practitioners.

These research aspects are considered by the GraphiT project that has just started. The project also proposes to focus on design patterns and patterns languages to formalize the identified practices and needs of Moodle teacher-designers communities. This should help in formalizing what future VIDLs metamodels would have to take into account by abstracting building blocks taken from LMS-closer VIDLs/metamodels (analogy with high-level programming languages for which the translation towards machine code is part of their specification: important information are preserved whereas human-oriented structures can be lost when mapped to low-level machine-instructions).

In order to help the specification of metamodels/VIDLs on top of other ones, we plan to study theories and techniques from our Model-Driven Engineering framework. Indeed metamodels weaving and composition appear to address the specification of semantic links between metamodels that will later be used for the generation of declarative rules for models transformation. This research topic is at the intersection of MDE and instructional design domains and has never been addressed or tackled so far. The following issues will also have to be tackled in order to reach our objectives:

- Limits of the meta-model (for formalizing the abstract syntaxes of VIDLs) and visual notations (for the concrete syntaxes of VIDLs) expressiveness specified on top of the expressiveness of the LMS internal semantics;
- Limits of the automatic translation of models from one VIDL to another one with no semantic losses about information to bind to the LMS semantics;
- Limits of relations possibilities and expressiveness between two VIDLs;
- Investment cost, programming effort required when following a DSM approach and tooling;
- User-friendliness, appropriation, usability and other validation aspects of propositions by the target-communities.

# Conclusion

This article has presented an original approach for improving the focus on the instructional design facets of learning situations to deliver on existent LMSs like Moodle. We propose an external approach for designing, binding and operationalizing learning situations. At first, we propose to identify and formalize the internal and implicit instructional design languages of LMSs, and then to expose them thanks to some specific API/modules available for teacher-designers within their course spaces. These explicit languages will be used as proprietary formats for communicating in and out LMSs. If they could propose such self-format, it would open the range of instructional tools focusing on various aspects (design, interoperability, etc.) and not constraint and limited by the LMSs architecture.

Our next proposition concerns the specification of Visual Instructional Design Languages both based on LMSs' semantics and directed towards some specific practitioners' needs and practices to capture. The *Domain-Specific Modelling* framework we chose provides a very challenging trend for supporting the specification of such visual languages as well as the generation of dedicated graphical editors thanks to the synchronized representation it proposes: human-interpretable visual models with machine-readable persistence.

Nevertheless, the research works conducted for now, mainly on the Moodle LMS, has also depicted some first semantics limits while reducing VIDLs metamodels to the LMS one. It tackles the binding and operationalizing objectives but also limits the VIDL expressiveness and usages: the learning situations designed are too close to the LMS semantics. We have just started a French ANR funded project in order to study the specification of more complex LMS-centred VIDLs. This research work will be realized in relation to a specific community of



ISBN

Moodle practitioners from our university: pedagogical engineers (experts) and teachers-designers (novice and regular practitioners). They will be frequently involved in the project realization.

## References


Abdallah, F., Toffolon, C., & Warin, B., (2008). Models transformation to implement a project-based collaborative learning (pbcl) scenario: Moodle case study. The 8th IEEE International Conference on Advanced Learning Technologies (ICALT'2008), Santander, Cantabria, Spain.

Abedmouleh, A. Oubahssi, L., Laforcade, P., & Choquet, C. (2012a). Expressing the implicit instructional design language embedded in an LMS: motivations and process, The 15th IASTED International Conference on Computers and Advanced Technology in Education (CATE 2012), June 25 – 27, Napoli, Italy.

Abedmouleh, A. Laforcade, P., & Oubahssi, L. (2012b). Specification of Visual Instructional Design Languages dedicated to Learning Management Systems. International Conference on Software and Data Technologies. Seville, Spain, 143–148.

Abedmouleh A, Laforcade P, Oubahssi L, Choquet C. (2011). Operationalization of learning scenarios on existent Learning Management Systems: the MOODLE case-study. International Conference on Software and Data Technologies, July 18-21 2011.

Berggren, A., Burgos, D., Fontana, J., Hinkelman, D., Hung, V., Hursh, A., & Tielemans, G., (2005). Practical and pedagogical issues for teacher adoption of IMSLD standards in Moodle LMS, Journal of Interactive Media in Education2. http://www-jime.open.ac.uk/2005/02/berggren-2005-02-paper.htmlS.

Botturi, L. & Stubbs, S.T. (2007). Handbook of Visual Languages for Instructional Design: Theories and Practices. Information Science Reference.

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, 924—931.

Caron, P.A., Derycke, A., & Le Pallec, X. (2005). The Bricoles project: support socially informed design of learning environment. The 12th International Conference on Artificial Intelligence in Education (AIED 2005), Amsterdam.

Dalziel, J. (2003). Implementing Learning Design: the Learning Activity Management System (LAMS). Proceedings of the 20th ASCILITE, 593–596.

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

Eclipse, (2012). The Eclipse Modeling Project. http://www.eclipse.org/modeling/. [viewed 10 March 2012].

Gonzalez-Barbone, V., & Anido-Rifon, L. (2010). From SCORM to Common Cartridge: A step forward. Computers & Education 54, 88–102.

Hernández-Leo, D., Villasclaras-Fernández, E.D., Asensio-Pérez, J.I., & Dimitriadis, Y. (2006). COLLAGE: collaborative learning design editor based on patterns, Educational Technology & Society, 9(1), 58–71.

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

Koper, R., & Manderveld, J., (2004). Educational modelling language: modelling reusable, interoperable, rich and personalised units of learning. British Journal of Educational Technology Vol 35 (5), 537–551.

Laforcade, P. (2010). A Domain-Specific Modeling approach for supporting the specification of Visual Instructional Design Languages and the building of dedicated editors. Journal of Visual Languages and Computing 2, 347–358.

Lengyel, P., Szilágyi, R., & Herdon, M. (2007) Moodle and LAMS integration. Summer university on IT in agriculture and rural development.Debrecen, Hungary.

Mekpiroona, O., Tammarattananonta, P., Buasrounga, N., Apitiwongmanita, N., Pravalpruka, B., & Supnithia, T. (2008). SCORM in Open Source LMS: A case study of LEARNSQUARE. 16th international conference on computer in Education (ICCE'08), 166–170. Taipei, Taiwan.

Neumann, S. & Oberheumer, P. (2009). User evaluation of a graphical modeling tool for IMS Learning Design. M. Spaniol, Q.Li,R. Klamma, R.W.H.Lau(Eds.), Proceedings of the Eighth ICWL, Lecture Notes in Computer Science, vol. 5686, 287–296.

Ortiz, I. M., Ger, P. M., Rodríguez, J. L. S. & Manjón, B.F., (2007). Supporting the authoring and operationalization of educational modelling languages. Journal of Universal Computer Science 13(7), 938-947.


## Acknowledgements


This work and submission are funded by the French GraphiT project [ANR-2011-SI02-011] (http://www-lium.univ-lemans.fr/~laforcad/graphit/).





ISBN

Proceedings of the 1st International Moodle Scientific Conference – MoodleSCo2012

ISBN