A meta-modeling approach for capturing recurrent uses of Moodle tools into pedagogical activities

Esteban Loiseau, Nour El Mawas, Pierre Laforcade

A meta-modeling approach for capturing recurrent uses of Moodle tools into pedagogical activities

Esteban Loiseau, Nour El Mawas, and Pierre Laforcade

LIUM, Computer Science Laboratory,
University of Le Mans, France
{esteban.loiseau,nour.el_mawas,pierre.laforcade}@univ-lemans.fr

Abstract. Teacher's expertise on using Learning Management Systems (LMS) is tightly coupled to how they design their online courses. The GraphiT project aims to help teachers in specifying of pedagogically sound learning scenarios that can be technically executable for automatically setting-up the related LMS course. We intend to provide teachers with LMS-specific instructional design languages and editors. To achieve this goal, we have to raise the LMS semantics in order to enrich the pedagogical expressiveness of the produced models. We propose a specific LMS-centered approach for abstracting the low-level parameterizations and turning these semantics into higher-level pedagogical building blocks. We present and illustrate our propositions focused on Moodle. In this paper, we focus on the first abstraction level: identifying pedagogical activities according to recurrent uses of Moodle activities.

Keywords: Instructional Design, Learning Management System, Visual Instructional Design Language, Modeling and Meta-modeling, Weaving Models.

1 Introduction

Learning Management Systems (LMS) are presently widely spread in academic institutions and are used not only for distant courses but also during or in complement to face-to-face learning sessions [1]. We conducted a study with 203 teachers which put forward their heavy form-oriented human-interfaces and their tools/services-oriented course design, leading to reduce their uses. In order to set up complex learning activities, teachers have to develop high-level skills on how to use the existing LMS. Such skills can be acquired through specific teacher education programs, often focusing on the features and technical aspects of the platform, but few courses deal with how to design pedagogically sound learning situations for this very specific LMS. Because of the multiple educational theories and approaches [2], as well as the lack of tools and processes dedicated to existing LMSs, teachers develop *ad hoc* and individual learning design techniques.

In such context, it is relevant to help teachers in focusing on pedagogical aspects and their instructional design set-up for the specific LMS they have

at their disposal. Whereas improving their knowledge and abilities to use the platforms' features, a focus on the instructional design possibilities and how they can rely on the LMS features should encourage individual and collective understanding about the pedagogical uses of the targeted LMS.

For this purpose, we propose an LMS-centered design approach in opposition to the usual platform-independent approaches [3][4]. The GraphiT project (Graphical Visual Instructional Design Languages for Teachers) tackles this issue. Its main objective is to investigate Model Driven Engineering (MDE) techniques for supporting the specification of LMS-centered graphical instructional design languages and the development of dedicated editors. This paper deals with one central challenge: raising the pedagogical expressiveness of LMSs learning design semantics by using MDE techniques.

To this end, we detail in section 2 our research context, including the presentation of the GraphiT project, as well as a position of our current results. Section 3 is dedicated to a survey and series of interviews we conducted with designers in order to collect needs and requirements for the Moodle LMS. A global presentation of our abstraction proposition is done in section 4: a 4-level abstract syntax, formalized as a metamodel, for the future learning design language. Section 5 focuses on the first-level and includes the proposition of a specific method to identify the pedagogical activities and their bindings to LMS tools. We also use the specific weaving language we developed to formally capture these bindings. Section 6 illustrates our propositions by a concrete learning scenario.

2 Research Context

2.1 Instructional design and LMS compatibility

LMS development is usually based upon an educative theory rationale, or some specific pedagogical approach. For example, Moodle claims a socio-constructivist pedagogy philosophy [5]. Widespread LMS generally follow such an orientation because of the various production and communication tools provided. LMSs are the activity-centered evolution of former learning-object-centered TEL-systems. Indeed, current LMSs provide designers with some numerous functionalities that can be used to realise various learning activities and are not restricted to providing resource access to students.

Nevertheless, activity-centered standards like the *de facto* IMS-LD [6] fail to integrate existing LMSs. Experiments on extending Moodle to import IMS-LD learning scenarios proved that adapting existing LMSs requires some complex and heavy re-engineering (in particular integrating a dedicated runtime-engine) in order to overcome the limitations of the platform features and semantics [7]. Educational Modeling Languages [8] fail to provide a support for operationalizing EML-conformed learning scenarios into existing LMSs. For now widely spread LMSs like Moodle still do not propose an IMS-LD compliance.

Moodle proposes its own format for importing questions into quizzes. Our idea is to generalise it to the whole instructional design aspects. Similarly to the

SCORM [10] compliance about Learning Objects, the rationale of the GraphiT project is based on the idea that LMSs should make explicit their learning design format in order to facilitate the import/export of compliant learning scenarios, and, in addition, to foster the development of LMS-dedicated instructional design editors.

2.2 Some state-of-the-art learning design tools and languages

The project main goal is to study the possibilities and limitations about the pedagogical expressiveness of operationalizable languages. According to the classification of Educational Modeling Languages proposed by [11], our objectives map to **formal** languages, i.e. with closed set of concepts and rules for composing the designs, with an **implementation** level of elaboration, i.e. the highest level of detail achieving maximum precision, with a **visual** notation system.

The *Glue! architecture*, including the *Glue!PS* editor [3], and the CADMOS editor [4] are two recent research works sharing our learning design criteria about pedagogical sound and executable learning design editors. They both propose an LMS-independent solution offering an LMS deployment feature towards the most widespread and used platform: Moodle [12]. They both achieve the deployment by generating a Moodle course backup with all the information and mapping their own data model concepts to Moodle data model concepts; this backup is then imported and deployed within a Moodle course using the Moodle restoration process. Such approaches result in semantics adaptations and semantics losses during their internal mappings because of the gap between the instructional design language, the specific learning design capabilities as well as features of the targeted LMS.

Other research [13] shows that model transformations techniques from the MDE theories and tools can be useful to translate a designer-centered and LMS-independent learning scenario to a one that is LMS specific. Nevertheless, they also highlighted the complex transformation model to specify, the LMS meta-model to capture, the semantics losses during translation, and the requirement of an LMS-dedicated tool for embedding the scenarios into the LMS.

2.3 Model Driven Engineering within the Graphit project

The project methodology consists in exploring how *Model Driven Engineering* and more particularly *Domain Specific Modeling* [9] techniques and tools can be relevant and useful to help in developing learning design editors that (1) are focusing on learning design for a specific existing LMS, (2) are enough expressive for abstracting the LMS's implicit learning design, and (3) are machine-readable, or *executable*, to be fully traduce into first LMS implementation settings.

Briefly, MDE is a software development methodology which focuses on creating and exploiting formal domain models and meta-models, rather than on producing code [23]. MDE is also a large research field about specifying / executing / transforming / composing (meta-)models. It comes with many specific tools to support all these activities. DSM can be considered as a specific MDE process. It

involves the systematic use of domain-specific languages. These languages tend to support formal higher-level abstractions in contrast to semi-formal general-purpose modeling languages like UML.

The approach and architecture we propose is different from other existing approaches. We propose an LMS-dependent architecture that only focuses on one existing LMS in order to provide an instructional design language that will be specified and tooled. Our idea is to conduct the platform abstraction in accordance with the formalisation of future learning scenarios. We do not aim at extending the LMS semantics with new add-ons/plugins, enriching it with more pedagogical-oriented features. Our objective is to support learning scenarios specification in conformance with the LMS semantics (its abilities as well as its limitations). Furthermore, we do not aim at only providing a notation layer on top of the LMS metamodel. The results of past experiments [14] show that the best solution (expressiveness/LMS compliance ratio) relies on extending the LMS metamodel. However, it requires a strong metamodeling expertise to reduce the developing cost while restoring the LMS compliance. This solution also highlights the importance to drive the expressiveness (and semantics) extension of the initial metamodel with the binding capacity. This paper focuses on our further results and propositions about this issue.

By extending the LMS metamodel we also extend the abstract syntax of the instructional design language and thus lose the LMS-compliance format. We plan to restore it by DSM (Domain-Specific Modeling) techniques (weaving and transformation models). We aim at guaranteeing that learning scenarios could be fully operationalized into the LMS without semantics losses. Our approach can take advantage of this LMS-dependance but it has also the disadvantage to be restricted to one LMS and one of its versions. The LMS instructional design semantics has first to be identified and formalized as a domain metamodel. This metamodel drives the elaboration of an XSD (XML Schema Definition) schema that will be used as a format reference for the API to develop. This API will be used through an import facility available to teachers-designers in their LMS courses. Its function is to parse the XML-based scenario and fill-up the LMS database. According to DSM techniques and tools (like the EMF/GMF ones for example [15]), the visual instructional design language will be composed of an abstract syntax (based on an extension of the LMS metamodel) from which the graphical, tooling and mapping models will be derived. The editor will also be developed using the code-generation feature of DSM tools. The produced scenarios have to be compliant with the initial LMS meta-model to be deployed by the API. We propose then to run two kinds of model transformations. The first one will consist of various, fine-grained transformations that will be during design-time: it will show some LMS mappings to teacher-designers in order to help and guide them in the design process. The second transformation, unique and large, will be used as an export feature (after design-time).

The main challenge of this project is to create enough abstraction from the LMS instructional design semantics to provide teachers with some pedagogically-sound higher design building blocks. The LMS expressiveness and limitations

have to be overcome in order to offer teachers some instructional design mechanisms closer to their practices and needs about specifying and sequencing learning activities.

Although the GraphiT project deals with different LMSs for guaranteeing the reproducibility of its results, we focus primarly on the Moodle platform which is the most popular open-source LMS.

3 Collection and analysis of requirements and needs

We evaluated several theoretical sources [16] as well as made practical exchanges with pedagogical engineerers in order to sketch our proposition orientations. Following, we decided to conduct a larger survey with complementary interviews to verify our initial assumptions, to collect feeback about our project orientations and positions, and to identify more precisely end-users' practices, needs and learning design tools requirements about the Moodle LMS.

3.1 Overview of the survey

We conducted an online survey that was diffused through international French-speaking higher education institutions during a 4-week period. This survey addressed teachers and pedagogical engineers using existing LMSs. The survey was composed of 21 mandatory questions, most of them accepting multiple answers. Some questions were conditioned to the selection of previous specific answers. For example, the first 8 questions (relative to the global design of courses) are LMS-independent, whereas the other ones were only available to people using Moodle (the LMS we wanted to focus on). We received and analysed 208 results. We only sketch here the most noticeable and relevant points in relation to the focus of this paper.

74% of those polled use an LMS in addition to their face-to-face courses (32% only for this purpose), 52% for distant courses and 37% during the face-to-face sessions. Main uses of the LMS concern the document transmission (91%), collection of works (52%), support for collaborative activities (47%), evaluations (47%), and new pedagogical practices (58%). On average, half of those polled considered having explored the LMS alone. Those who did not consider themselves as novices (56%) stated that they had improved their LMS knowledge on their own at 73%.

Although half of Moodle users consider that the global user-interface of a course is easily understandable, only 33% consider that the form-oriented parameterization screens are understandable. From a learning design perspective, they sketch all (38%) or part (37%) of the learning scenario before setting-up the equivalent course upon Moodle. 43% of this sub-population have met some difficulties during this manual step and have felt constrained in adapting their initial scenarios and intentions (12% failed to adapt the scenario). A majority of Moodle designers use the basic functionalities like the move left/right (64%), the hide/show (84%) parameters. Half of the group graded students' productions and

use Moodle's groups and groupings when required. 62% used the restrict access settings but only 34% the activity completion. 15 of 22 Moodle standard activities/functionalities (note that institutions can add/remove these blocks) are not well known by an average of 50% (sometimes more) of answerers, whereas the 7 others are regularly used. The *Forum* is largely preferred to the *Chat* feature to foster communication. For the setting-up of exercices, *Assignment* (47%) and *Quiz* (37%) are preferred to *Hot Potatoes* (15%) or *Lesson* (19%). The *Wiki* is the most preferred collaborative tool (23%) among others (*Journal* 8%, *Workshop* 8%).

3.2 Most relevant points from interviews analysis

From most relevant answerers that agreed to be contacted we conducted 20 one-to-one interviews, mostly by distant devices. Interviewees were selected depending on their instructional design expertise about the Moodle platform.

They agree that Moodle is useful for simple pedagogical objectives but is time-consuming for elaborating more complex learning situations. Settings screens are considered too complex and difficult to handle. These screens mix pedagogical and technical parameters, requiring to test and observe the pedagogical implications of all combinations. Some interviewees stated that they encourage to use default parameters and then, hinder the setting-up of more complex activities.

A majority of interviewees accept the idea of both an external learning design editor dedicated to Moodle and an *import* block available through the Moodle internal design space to automatically set-up the course (the external feature allowing offline designs and the graphical notation helping to visualize the scenario at design-time). They approve the approach, emphasising its relevance if templates or concrete cases about pedagogical uses of Moodle tools can be handled within the editor. They highlight the need for a language/editor covering large pedagogical uses but without being too generic. Some of them consider important to continue using the editor for adapting the scenario after the import step although they agree that a round-trip use of both editor and Moodle can be an obstacle.

One issue highlighted is that practitioners did not expressed common design practises, as we expected them to, mainly because of the heterogeneity of their Moodle expertises and pedagogical backgrounds. Nevertheless they have in common to think about Moodle tools according to their basic pedagogical uses. Indeed, they all point the heavy parameterizations of tools and resources and the need for having an abstract view of what are the pedagogical uses in order to help and guide them in selecting and configuring the right implementation activities.

3.3 Requirements for a learning design language and editor

From all these practitioners feedback we listed some specific requirements for our Moodle language/editor to develop. First, they mentioned the need for the graphical authoring-tool to allow designers to select pedagogical blocks on top of

the LMS semantics as well as with Moodle building blocks to compose with. In their mind, the editor will not have to strictly follow a top-down process from abstracted specification elements to implementation one expressed in terms of Moodle; abstractions from Moodle and its own concepts should be mixed up together according to practitioners' expertise about instructional design (**specification and implementation concepts mix**). Secondly, they are interesting in the idea that mappings from pedagogical design blocks to Moodle concepts can be showed to practitioners (**default mapping**) and adapted if required (**mapping adaptation**). This design approach could help practitioners in the appropriation of the pedagogical constructs and guide them in designing more abstract learning scenario while mastering the translations into LMS elements.

Another design point highlighted (**declarative non-visible information**) is about the possibility to design and declare within the learning scenario some information that do not required to be mapped into LMS concepts or just mentioned as non-visible labels (for students/tutors) for the teacher him-self: information about the face-to-face sessions mixed up with the LMS-centered ones, about pedagogical strategies or pedagogical objectives, about activities to realize on the LMS at a specific runtime moment according to concrete data (enrolled students, dates, etc.). Finally, another design need was to help teachers in sequencing the course in more advanced structures (choices, sequences) with elements showed one-by-one according to their progress (**advanced activity structures**). Indeed, these can be done manually but it requires to parameterize many low-levels and technical-oriented properties (achievements, restricted access conditions...) that they would appreciate not to have to set up by themselves.

4 Abstractions based on the LMS metamodel

We plan to study how the LMS semantics about instructional design could be abstracted from two perspectives: a theoritical one, generalizable to different LMSs, and a practical one about the special case-study of the Moodle LMS. We chose to follow a bottom-up approach by focusing at first on the Moodle LMS. According to practitioners' needs, the abstraction could consist in raising the recurrent LMS uses supporting some learners and/or teachers activities. From an activity theory perspective [17][18], such activities should involve LMS's bindings of subject, objects, tools/artefacts, community, division of labor and rules. Because our survey and interviews highlight a special need to ease the parameterization of Moodle tools and resources for setting-up activities, we decided to focus at first on raising these tools and resource semantics, and to study later the other aspects.

The following sections present these abstractions in relation with their formalizations for the Moodle LMS. The metamodel from Figure 1 can be considered as part of the general abstract syntax of the instructional design language to be developed. This part focuses on the abstraction of Moodle tools and resources.

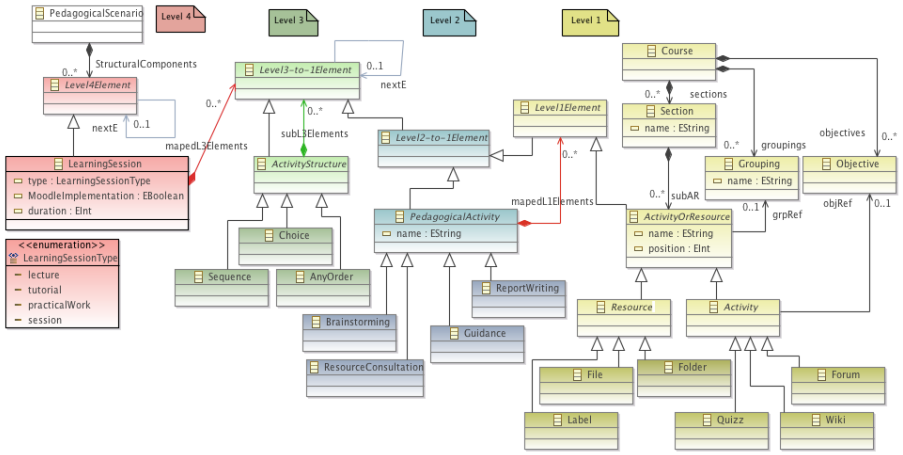

Fig. 1. The abstract syntax of an instructional design language on top of the Moodle metamodel.

4.1 Tool-or-resource-based pedagogical activities

We define an LMS-abstract pedagogical activity as an *encapsulation of parameters a teacher has to set-up when using a tool (or resource) for a specific pedagogical use*. From a single tool, for example a forum, a teacher can design several pedagogical use, depending on its configuration: to provide news to students, to set up group work, to propose a peer reviewing activity, etc.

Because several LMS functionalities can be used for the same pedagogical purpose, we have to find the discriminatory criteria that can guide to identify the right tool and default mapping (as well as the relations to objectives, resources, groupings, etc. that are involved in the right setting-up of the pedagogical activities).

To be used appropriately, this first abstract block requires a name, a description, and specific properties (the former discriminants), set at design-time by practitioners, that will drive the default mapping. For example an exchange activity, involving student communication, could either rely on a forum or a chat, depending on a synchronous property. The mappings will not be limited to the parameterization of a tool. It will also impact some other elements in relation with the tool/resource: grades, objectives, groupings, restriction access and achievements rules, etc.

4.2 Activity structures

According to [19], successful implementation of an online course that is facilitated by an LMS needs careful planning, including structural strategies in the design of the course. In order to ease and assist the practitioners when assembling and setting-up combinations of activities or resources we propose then usual structural elements (selection, sequence, conditional activities, etc.). These blocks will

be composed of activities or other activity structures. Every instructional design language feature some of them. In the case of Moodle they will be concretely translated as complex combinations of *labels* (stating the structure name, kind and use for users), shifted content (*move left/right* feature) according to the activity structure components in the learning scenario. After various translations and mappings until reaching the LMS low-level elements, all its content will be parameterized (*restrict access*, *visibility*, *achievement...*) with appropriate properties in order to set up the desired behavior.

4.3 Instructional Design Language Abstract syntax

The global architecture we propose for the abstract syntax of the Moodle-centered instructional design language is composed of four levels. Figure 1 illustrates our proposition with a graphical representation of the ecore domain model (the EMF metamodel format).

Level 1 fits the Moodle metamodel. Readers have to consider Figure 1 as a part of the whole metamodel. Only important structural relations and concepts are depicted because of our current interest. *Level 1 elements* (restricted to the Moodle *activities* – name for tools given by Moodle — and resources) can be directly used by teachers-designers and parameterized for building a learning session. From the Moodle metamodel point of view these elements require a global *Course* and a *Section* container to be attached to. In the extended metamodel they will be specified at first as child of *level 4 elements*. The model transformation, at post-design-time, will deal with restoring a model in full-compliance with the Moodle metamodel: creation of the global *Course* instance, *Section* instances, attachment of all the corresponding Moodle elements according to the orders and positions deductible from the source scenario.

Level 2 includes our pedagogical activities. They are composed of *Level 1 elements*, i.e. Moodle activities and resources. Level 3 captures the activity structures. The activity structures are composed of *Level 3 elements* specified during the design-time. Finally, the fourth level is the contextual level focusing on the global structure of the learning session in relation to the different face-to-face, complementary, distant sessions or other teacher-defined customized sessions.

Such *Level 4 elements* rely on the Moodle *section* concept. Indeed, Moodle only proposes sections into the space of the course for aggregating the tools and resources. However, designers have at their disposal an *indentation* feature (*position* property in the Moodle metamodel) to shift activities and resources in order to visually indicate their collective relationships. This *position* property will be used by the dynamical mappings, in order to position the corresponding elements in accordance to the source element position in the global learning scenario.

The relations with a red composition indicate that the content will not be displayed in the future concrete syntax (notation) as nested elements but will appear in another sub-diagram where the parent container will be the root canvas. Differently, the green composition indicates that content will be showed as nested elements of the parent container in the same diagram. Finally, the *nextE*

reflexive relation allows, by inheritance, to provide a previous/next information to sequence the various elements within their dynamic pedagogical context (the ordering concerns the child elements sharing a same *Level Element* parent).

The leaf meta-classes from figure 1 (dark elements) sketch some examples of future elements. They are on purpose not showing their attributes (for ease of reading). However each of them owns specific properties in accordance with the different in-progress formal specifications we are studying about the Moodle instructional design semantics, pedagogical activities, and activity structures.

Overview of our instructional design tool The proposed authoring-tool will directly propose to practitioners the *level-4 elements* in the tool palette. Indeed, these elements are required to map to Moodle sections in order to sequentially structure the course skeleton. Sessions that do not rely on Moodle features can also be described if designers need an overall view of a global module/course larger than the ones involving the use of an LMS. Other *level-4 elements* will then open an empty sub-diagram when double-clicked. It can then be used to arrange *levels 3-to-1 elements* from the new palette. Indeed, practitioners can then choose the method (top-bottom, bottom-up), the description level (specification versus implementation) and the elements to select, combine and adapt. Pedagogical activities can be opened up as another sub-diagram containing the default mapping to *levels 1 elements*. Every mapping can be adapted and modified by deleting/adding new elements (according to those accepted under the parent element) or modifying the elements properties. This layer-oriented notation and functionalities fit the practitioners' need depicted in section 3.3.

5 Main abstraction: pedagogical activities

5.1 Identifying pedagogical activities

In order to identify the most appropriate tool for a specific pedagogical activity, we followed these three steps: (1) analysis for each Moodle tools of its recurrent uses (bottom-up method), (2) identification of tools offering common uses (top-down method), and (3) specification of discriminating criteria to drive the selection of a suitable tool. Moodle 2.4 offers 7 resources (*Book*, *Page*, *Label*, *IMS content package*, *File*, *Folder*, and *URL*) and 13 activities (*Forum*, *Database*, *Glossary*, *Assignment*, *Lesson*, *Quiz*, *Workshop*, *SCORM package*, *External tool*, *Choice*, *Survey*, *Wiki*, and *Feedback*). We have study their recurrent uses. We notice that some activities/resources can be diverted to serve for different uses. For example, everyone knows that the *Forum* is used for discussion reasons but it can also be used to allow students to introduce themselves in a course or to consult a Frequently Asked Questions (FAQ) or to share documents between learners. After looking at all Moodle's activities/resources uses, we have identified those supporting the same uses. Three tools can be used to consult a FAQ: the Forum, the Wiki, and the Glossary.

We have then specified discriminating criteria to help a teacher in deciding which tool he must use if he has many choices. We chose $m \times n$ matrix A format to present these discriminating criteria (A has m rows and n columns, first row and column are headers and not part of the matrix data) according to seven rules :

R1 The pedagogical activity name is only from a teacher perspective if no students are concerned (= with parameter *hide* on). For example, for a survey, we choose the expression "answer a survey" (students viewpoint) instead of "create a survey" (teachers viewpoint). Note that A_{11} presents this pedagogical activity.
R2 Tools participating to the realization of the activity are the elements $A_{12}...A_{1n}$.
R3 Discriminating criteria are the elements $A_{21}...A_{m1}$.
R4 Discriminating criteria are expressed as much as possible as a pedagogical question designers have to answer by *Yes* or *No*.
R5 Cells intersecting a discriminating criterion and a tool must embed all answers that can implied to choose this tool (*Yes/No* are both possible if this criterion is not directly discriminant for this tool, i.e. the tool can support both pedagogical cases).
R6 A valid discriminating criterion must cause at least one different answers for one tool.
R7 The matrix is terminated if there is no similar combination of answers for two tools.

An unachieved matrix indicates to experts that they have to add one more discriminating criteria and verify again the rule R7. Table 1 shows an example

Table 1. Example of identification matrix.

PA	T1	T2	T3	T4
C1	Yes/No	No	Yes/No	Yes
C2	Yes/No	No	Yes/No	Yes
C3	No	No	No	Yes
C4	Yes/No	No	No	No
C5	No	No	Yes/No	No
C6	Yes	No	No	No
C7	Yes	No	Yes	No

of identification matrix for the pedagogical activity (PA) "Answer a poll". Four Tools can support this activity: *Quiz* (T1), *Choice* (T2), *Feedback* (T3), and *Survey* (T4). Experts have found 7 discriminating criteria. Each criterion is presented in the form of a question:

- (C1) More than one question?
- (C2) Only multiple choice questions?
- (C3) Pre-populated with questions?
- (C4) Time limit?

- (C5) Anonymous?
- (C6) Graded?
- (C7) Feedback after submission?

In table 1, we have three different answers that can imply these four tools: Yes, No, and Yes/No. For example with a survey (T4), we can have more than one question (A_{25} = Yes), only multiple choice questions are allowed (A_{35} = Yes), it is a pre-populated survey with questions (A_{45} = Yes), it can not have a time limit/countdown timer for students' navigation (A_{55} = No), it is always nominative (A_{65} = No), it can not be graded (A_{75} = No), and students can not have a feedback after their submissions (A_{85} = No). Note that a designer can reply to C1, C2, C3, C4 , C5, C6, and C7 in any order. Some combinations cannot lead to a specific tool choice for two reasons: (1) a non-valid combination, or (2) a non-response to all questions. In the first case, the experts will be notified to adapt their pedagogical choices while in the second case they will be asked to precise more choices.

Such identification matrix has to be completed by additional information in order to precise the general (whatever the answers that guide the binding like a tool name) or contextualized (depending on some specific answers like a tool format) parameters for the related LMS activity or resource.

5.2 Formalization through model weaving

According to our Model Driven Engineering research framework, we can use model transformations to achieve the mappings specified by experts. The transformations will be run at design-time, to add mapped elements to the model and populate the sub-diagrams. Such transformations are complex (proportionally to the mapping complexity) and numerous, thus costly to write.

We on purpose propose to use the model weaving technique we studied in [20] to capture the mapping semantics in dedicated weaving models and automatically generate models transformations. From a practical viewpoint, thanks to the matrice and additional information from an LMS expert using our method and formalisms depicted in section 5.1, an engineer will formalize the mappings in a weaving model, using a tree based editor. He can then run a generic *High Order Transformation* (HOT) that will generate the concrete "mapping transformations". These final transformations can then be integrated within the graphical editor to be automatically run at design-time when teachers-designers will specify the pedagogical activity properties.

The weaving models can be expressed using a weaving language, based on a generic weaving metamodel we designed. This weaving metamodel defines the "syntax" of the mapping/weaving model. Each mapping (or binding) has one *source* element and one or several *targets* (chosen from the extended instructional design metamodel). Targets can have conditions on whether they have to be instantiated or not, attributes can be set to specific values (also with conditions). Figure 2 is a screenshot of the weaving editor. It is used to formally specify as a weaving model the corresponding binding specified by LMS's experts. Concretely, the left part of the figure should concern level-2 elements of

our instructional design metamodel for Moodle (1) whereas the right part should only concern level-1 elements, i.e. Moodle elements. Because our instructional design metamodel includes the Moodle one, source and target metamodels involved in the weaving are the same. The central part of the figure is the concrete tree-based editor for specifying bindings.

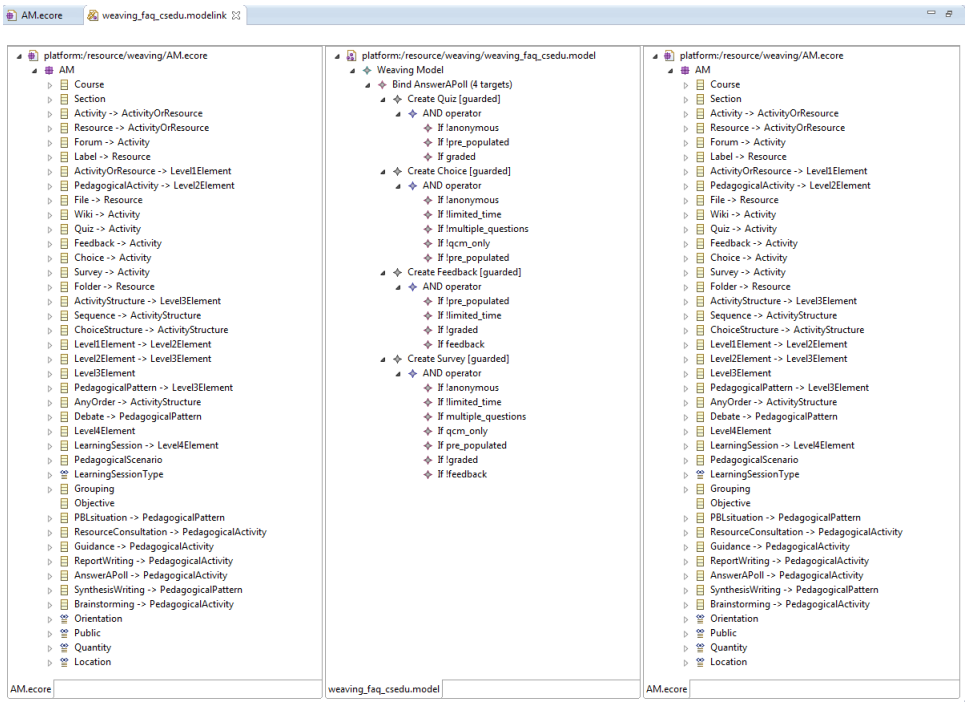

Fig. 2. Screen caption of our weaving tool for formalizing the bindings.

Figure 3 is an example of weaving model related to the binding example from the table 1. It specifies the corresponding translations for all possible combinations by using AND/OR/! operators. Such weaving model is realized by following the matrice formalism, tool by tool. Informations about the tools parameterizations are deduced from the additional informations given by experts.

We used languages and tools from the Epsilon [21] project to build a software framework fulfilling our model weaving requirements. This project is compliant with the Ecore formalization of metamodels we already used to formalize the various metamodels we illustrated. This Ecore format is from the Eclipse Modeling Framework [15]. Weaving models are edited through ModeLink, a three pane editor displaying the source and target metamodels in side panels (which are the same in our use case). The final "mapping" transformations are expressed using Epsilon Object Language (EOL), and are generated through a Model-to-text

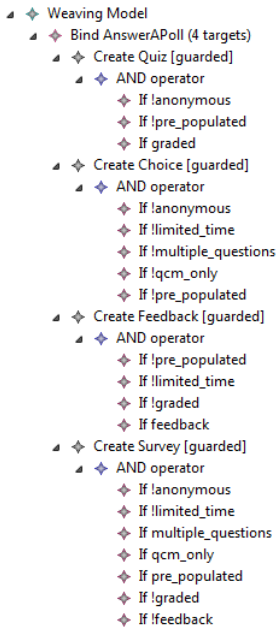

Fig. 3. Example of resulting weaving model (tree-based representation of a concrete XML file).

transformation using EGL language. This last transformation replaces the HOT traditionally used in model weaving environments.

6 Example of a learning scenario

We on purpose propose to illustrate our proposal by formalizing a very simple but representative learning scenario for the Moodle LMS. We propose at first a brief textual description, then the equivalent specification as a model conformed to the dedicated metamodel we proposed in section 4 (Figure 4 is a screenshot of the EMF-tree-based model editor).

6.1 Description and formalization

The learning scenario is composed of two learning sessions. The first one is a *lecture* session for which the teacher only want to provide learners with a *Resource consultation* corresponding to his face-to-face course material. This pedagogical activity has the *quantity* property set to "one" and the location set to "local". These properties will lead the dynamic mapping process to add the *File* Moodle element to the scenario.

The second learning session is a *practical work* that the teacher wants to realize in face-to-face within a computerized classroom. He would like to use

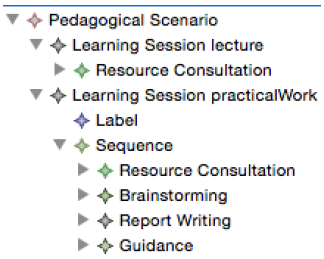

Fig. 4. Example of simplified learning scenario composed of elements from the 4 levels.

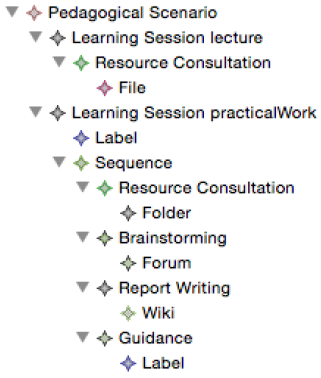

Fig. 5. Same example of simplified learning scenario with generated contextualized tools and resources bindings.

the Moodle platform for supporting a *sequence* activity structure embedding 4 sub-components. The first one is another *Resource consultation*. This time, the properties set to "many" (quantity) and "local" (location) by the teacher will lead the transformation process to add a *Folder* tool. The second sub-element is a *Brainstorming* pedagogical activity. Its *orientation* property set to "discussion" leads to propose a Forum tool. Similarly the third one is another pedagogical activity *Report writing* leading to a *Wiki* tool because of the *collaborative* property set to "true". Finally the fourth sub-component is a *Guidance* activity that aims at reminding the teacher to evaluate the synthesis in the wiki. Thanks to a *public* property set to "tutor" it leads the mapping process to set the corresponding *Label* to be invisible (*visible*="false") to students (it will only be displayed to the teacher).

The teacher can change at any time the activities properties, leading to other mapping adaptations. He can also manually delete the mapping elements, rearrange their order, or add some other elements. Figure 4 shows a global overview of the learning scenario elements including all the automatic mappings according to the various properties and values (not depicted within the figure).

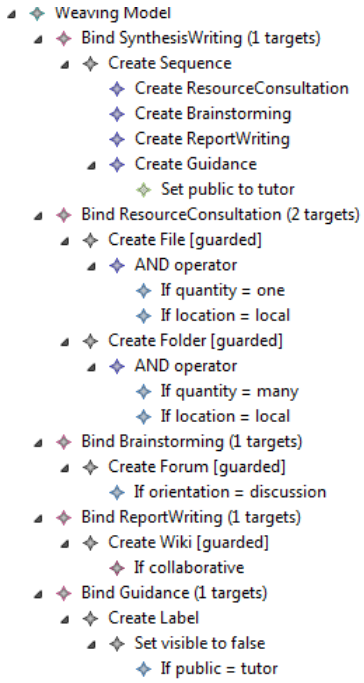

Fig. 6. Example of weaving model specifying mappings from Figure 4.

6.2 Prototype of the learning scenario editor

We are currently working on the development of a prototype adding a notation layer on top of the abstract syntax we propose for the Moodle-centered instructional design language. We choose for now the Sirius tooling [22] because it allows to quickly define custom multiview for workbenches with less technical knowledge compared to the well-known GMF tooling (from [15]). The notation, or concrete syntax for our instructional design language, is derived from the abstract syntax formalized as an Ecore metamodel. Sirius also facilitates the development of dedicated graphical tools by generating most of features (diagrams, trees, tables, etc.) from the sirius-specific model we build when using it. It reduces the cost and complexity of developing a graphical editors.

We succedeed in integrating the mappings transformations within this prototype. For now, when a user double-clicks on a session within the first level diagram (depicted in Figure 7), it opens a new diagram where he can mix elements from levels 1 to 3 according to his Moodle expertise (Figure 8). Pedagogical properties of level-2 elements can be set at this stage. When these level-2 elements are double-clicked, a transformation process is launched for checking all transformation rules automatically generated from the weaving models we produced. The execution of an eligible rule modifies at run-time the current scenario by adding the corresponding binding towards Moodle elements (level-1). Figure 9 shows resulting mappings. The result is part of the pedagogical scenario: it can

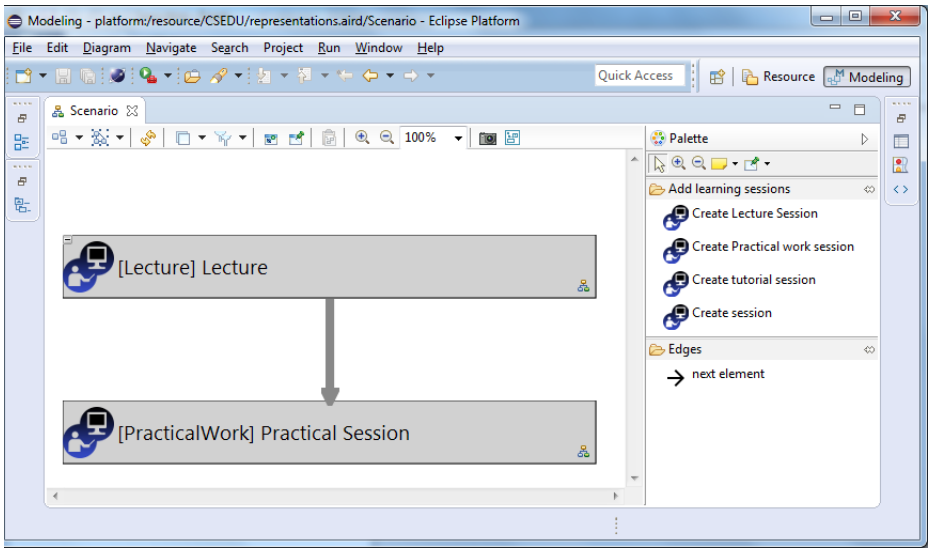

Fig. 7. Screenshot of a session diagram (level 4) in our current prototype.

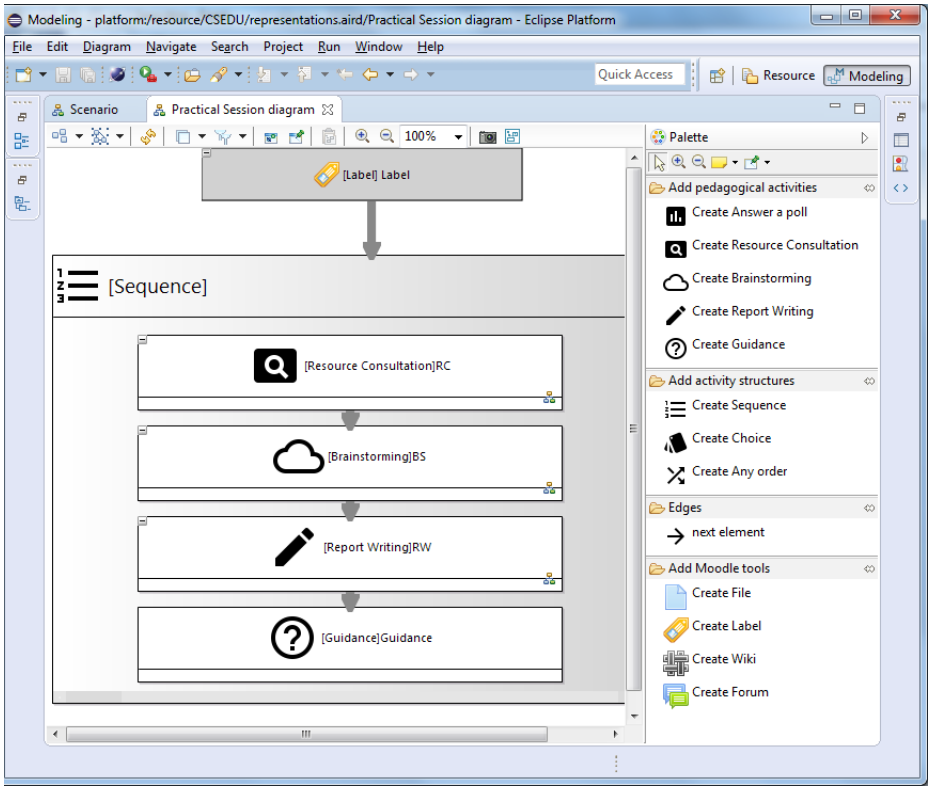

Fig. 8. Screenshot of an activities diagram (levels 3, 2 and 1) in our current prototype.

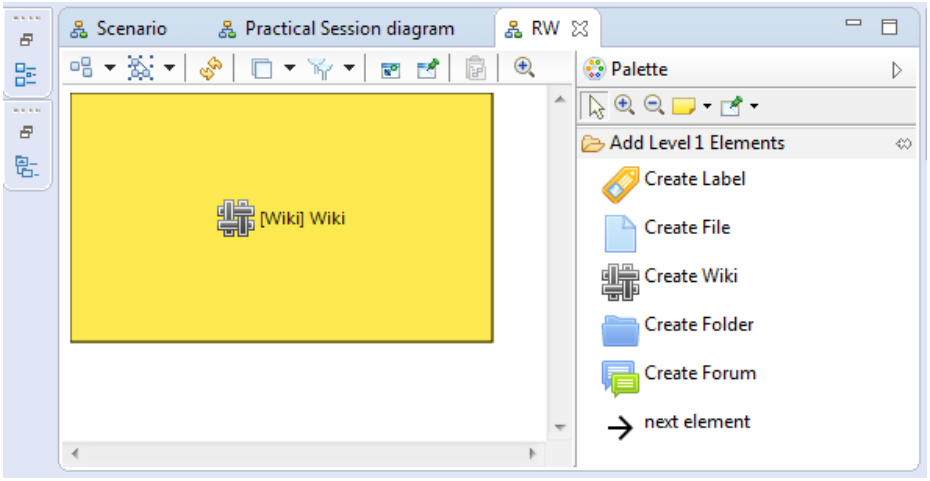

Fig. 9. Screenshot of a moodle diagram (level 1) in our current prototype.

be modified by adding/editing/removing new level-1 elements. Mappings can also be updated if a user changes the pedagogical properties of a level-2 element.

7 Conclusions

7.1 Tools uses as a first abstraction

We proposed a specific LMS-centered approach for raising the pedagogical expressiveness of its implicit learning design semantics. We discussed how the LMS low-level parameterizations could be abstracted in order to build higher-level building blocks capturing some recurrent resources or tools uses into pedagogical activities. We also presented a specific method for helping and guiding LMS experts to describe how these activities should be binded to appropriated tools or resources. In addition we propose a specific model weaving approach for formalizing these mappings. These resulting weaving models will drive at run-time and in real-time the automatic translations when using the authoring-tool. Based on a Moodle application, we present and illustrate our approach by formalising the abstract syntax of a Moodle-dedicated instructional design language following a specific 4-levels architecture. Thanks to illustrative examples and an overview of our current prototyping editor, we concretely argued and verified our propositions.

7.2 Ongoing work and future perspectives

We are working on integrating groups and pedagogical objectives in both learning design language and prototype editor. The objetives could be linked to *Level 4-to-1 elements* and mapped to Moodle outcomes, contained by the root *Course*, labels, or description fields depending on its status (goals/competencies for teacher

or activities objectives for learners understanding). Also, roles or groups refer to the division of labour in the learning scenario. Mappings to the Moodle concepts of *Group* and *Grouping* are considered.

The 4-levels metamodel extension we propose breaks compatibility of the learning scenarios with the LMS format as it differ from the platform metamodel. We plan to restore it using a global model transformation (written with the *Atlas Transformation Language*), available as an export feature of the authoring tool.

We are planning an experimentation of the prototype with end-users (teachers), once it has reached a stable enough version. The main objective is the validation of the language, along with the design approach, more than the tool itself (ergonomy or user experience). By providing teachers with only general guidelines about the scenario to produce, we would like to evaluate how they are able to design a pedagogically sound course with the editor. We are interested in observing which abstraction levels and elements they will use, if they are following a top-down or bottom-up approach, if they will use the default-mapping feature and other specific features we propose.

References

1. Garrisson, D.R., Kanuka, H.: Blended learning: Uncovering its transformative potential in higher education. The Internet and Higher Education 7, 95–105 (2004)
2. Ormrod, J.E.: Human Learning. Pearson College Division, Upper Saddle River (2011)
3. Alario-Hoyos, C., Munoz-Cristobal, J.A., Prieto-Santos, L.P., Bote-Lorenzo, M.L., Asensio-Perez, J.I., Gomez-Sanchez, E., Vega-Gorgojo, G., Dimitriadis, Y.: GLUE!-PS: An approach to deploy non-trivial collaborative learning situations that require the integration of external tools in VLEs. In: 1st Moodle Research Conference, pp.77-85. Greece (2012)
4. Katsamani, M., Retalis, S., Boloudakis, M.: Designing a Moodle course with the CADMOS learning design tool. Educational Media International 49, 317–331 (2012)
5. Dougiamas, M., Taylor, P.: Moodle: Using Learning Communities to Create an Open Source Course Management System. In: The World Conference on Educational Multimedia, Hypermedia and Telecommunications, pp. 171–178. Waynesville, USA (2003)
6. IMS Learning Design specification, `http://www.imsglobal.org/learningdesign/index.html`
7. Burgos, D., Tattersall, C., Dougiamas M., Vogten, H., Koper, R.: A First Step Mapping IMS Learning Design and Moodle. Journal of universal computer science 13, 924–931 (2007)
8. Berggren, A., Burgos, D., Fontana, J.M., Hinkelman, D., Hung, V., Hursh, A., Tielemans, G.: Practical and Pedagogical Issues for Teacher Adoption of IMS Learning Design Standards in Moodle LMS. Journal of Interactive Media in Education, special issue: Advances in Learning Design. (2005)
9. Kelly, S., Tolvanen, J-P.: Domain Specific Modeling: Enabling Full Code Generation. Wiley (2008)
10. Advanced Distributed Learning: The SCORM specification, `http://www.adlnet.gov/scorm/`

11. Botturi, L., Derntl, M., Boot, E., Figl, K.: A Classification Framework for Educational Modeling Languages in Instructional Design. In: 6th IEEE International Conference on Advanced Learning Technologies, pp. 1216–1220. Kerkrade, The Netherlands (2006)
12. Moodle Official Website, `https://moodle.org`
13. Abdallah, F., Toffolon, C., Warin, B.: Models transformation to implement a Project-Based Collaborative Learning (PBCL) Scenario : Moodle case study. In: 8th IEEE International Conference on Advanced Learning Technologies, pp. 639–643. IEEE Computer Society, Washington DC, USA (2008)
14. Loiseau, E., Laforcade, P.: Specification of learning management system-centered graphical instructional design languages - A DSM experimentation about the Moodle platform. In: 8th International Joint Conference on Software Technologies, pp. 504–511. Scitepress, Reykjavik, Iceland (2013)
15. Eclipse Modeling Project Official Website, `http://www.eclipse.org/modeling/`
16. Conole, G., Dyke, M., Oliver, M., Seale, J.: Mapping pedagogy and tools for effective learning design. Computers & Education 4, 17–33 (2004)
17. Engestrom, Y.: Learning by Expanding: An Activity Theoretical Approach to Developmental Research. Orienta-Konsultit Oy, Helsinki, Finland (1987)
18. Benson, A., Lawler, C., Whitworth, A.: Rules, roles and tools: Activity theory and the comparative study of e-learning. British Journal of Educational Technology, Vol. 39, No. 3., 456–467. (2008)
19. Gedera, D. S. P, Williams, P. J.: Using Activity Theory to understand contradictions in an online university course facilitated by Moodle. International Journal of Information Technology and Computer Science 10, 32–40 (2013)
20. Loiseau, E., Laforcade, P., Iksal, S.: Model Weaving and Pedagogy Mapping Abstraction Levels in Instructional Design Languages. In: 9th International Joint Conference on Software Technologies. ScitePress, Vienna, Austria (2014)
21. Paige, R. F., Kolovos, D. S., Rose L. M. , Drivalos N. , Polack F. A. C.: The Design of a Conceptual Framework and Technical Infrastructure for Model Management Language Engineering. In: 14th IEEE International Conference on Engineering of Complex Computer Systems, pp. 162–171. IEEE Computer Society, Washington, DC, USA (2009)
22. Sirius Project, `http://eclipse.org/sirius/`
23. Schmidt, D.C.: Model-Driven Engineering. IEEE Computer 39 (2) (2006)