A meta-model based approach for identifying and formalizing LMS instructional design languages

Nour El Mawas, Lahcen Oubahssi, Pierre Laforcade
Laboratoire d'Informatique de l'Université du Maine
LUNAM Université, Université du Maine
Le Mans, France
{nour.el_mawas, lahcen.oubahssi, pierre.laforcade}@univ-lemans.fr

***Abstract*— The use of existent LMSs presents many difficulties related to the design and operationalization of learning scenarios. Teachers have to encompass the LMS technical features and services in order to understand the underlying way of designing. Generic instructional design editors fail in bridging the gap between how they design a learning scenario and how the learning session can be set up within the target LMS. If LMSs could be able to explicit their intrinsic and implicit learning design model, it can be exploited as a proprietary format to build tools and facilities dedicated to this LMS. The research presented in this paper aims to present our method in terms of necessary analysis and steps for the identification and the formalization of Moodle instructional design language. The method takes into account three different viewpoints: a viewpoint centered on the LMS macro-HMIs (Human-Machine Interfaces), a functional viewpoint and a micro viewpoint. We validate the proposed method by applying this formalization on two versions of Moodle.**

Keywords - Learning Management System; E-learning; Instructional Design; Operationalization; Technology Enhanced Learning; Process; Moodle

I. INTRODUCTION

Our research work focuses on the field of Technology Enhanced Learning (TEL) engineering and re-engineering. TEL is a scientific domain where different disciplines such Computer Science,, education, psychology, philosophy, communication or sociology intersect [1]. We are particularly interested in applying and adapting Computer Science solutions for providing practitioners with some customized instructional design solutions.

Instructional Design (ID) is the systematic development of instructional specifications using learning and instructional theories to ensure the quality of instruction. It is the entire process of analysis about learning needs and goals as well as the development of a delivery system to support those needs. It includes development of instructional materials and activities and delivering and evaluation of all instruction and learner activities [2].

TEL is a large domain for research and practice, including e-learning, mobile learning, and Learning Management System (LMS). An LMS is the framework that handles all aspects of the learning process. An LMS is also the concrete infrastructure that delivers and manages instructional content, identifies and assesses individual and organizational learning or training goals, tracks the progress towards meeting those goals, and collects and presents data for supervising the learning process of organization as a whole [3]. LMSs support the use of standards for describing the learning objects, packaging them into larger content and learning units (such as lessons and courses), and applying various instructional design strategies and techniques [4]. Nowadays, LMSs are not restricted to distant learning only. Teachers use them for blended learning which combines traditional face-to-face learning with computer supported learning.

The research work presented in this paper is a continuity of other former works in our lab [5] [6] by proposing a new implementation approach of learning situations and pedagogical scenarios. It takes place into the context of the GraphiT project (Graphical Visual Instructional Design Languages for Teachers). Its main goal is to study the possibilities and limits about the pedagogical expressiveness of operationalizable languages to specify future leaning scenarios that could be fully deployed and automatically set up upon an existing LMS. Such instructional design languages aim at promoting and improving the uses of current LMSs by providing practitioners with some LMS-specific designing language and authoring-tool. Despite many existing standards [7] [8], approaches [9], languages [10], architectures [11], and tools [10] [12] to facilitate the instructional design, they are often not compatible with existing LMSs, or require a costly reengineering of the LMS (new web service API, new runtime engines, etc.). Moreover, they do not simplify the operationalization of the produced models. Some translations, leading to information or semantics losses, are still required to operationalize them into a targeted LMS.

In recent years, researchers have begun to formalize LMSs instructional languages in order to specify models in conformance with the infrastructure design languages of LMSs [13] [6]. These works propose a meta-model to

formalize LMS instructional language but to our knowledge, there is no proposition that focuses on identifying an explicit process or method to formalize it.

In this paper, we are focusing on the identification and formalization of LMSs implicit instructional design language. Indeed, the expected result will be the base for the development of binding solutions and will simplify the instructional design on platforms. These solutions must insure that future scenarios formalized in conformance to the language to identify will be operationalized without semantics losses into the LMS internal structures. This process is dedicated to LMSs active communities and more specifically to designers with a competence in IT and the service of information technology and communication for education (pedagogical engineers) who meets difficulties in appropriating the instructional design language of LMSs.

The paper is organized as follows. Section II highlights our motivation to extract the pedagogical LMS language. Section III details our approach. Section IV is dedicated to the application of our method on Moodle 2.4. Section V presents the comparison of two different versions of Moodle according to their meta-models. Section VI concludes our paper and presents our perspectives.

II. MOTIVATION

Many universities have adopted web-based LMSs as the TEL system. They use them to offer teachers a range of pedagogical and administrative tools for supporting teaching and learning activities [14]. However, many teachers have difficulty using LMSs to create learning designs that are truly engaging to their students. They are not familiar with the implicit learning design domains of LMSs [7]. Most of open source LMSs are very difficult to apply in real schools, because teachers are not familiar to using an LMS which needs to take an effort to appropriate it [8].

Due to the complexity of LMS functionalities, users are expected to have some pre-existing knowledge of these functionalities. Despite online forums, it is still difficult for a teacher to design his courses on platforms. LMSs are in continuous evolutions and discussions regarding different versions of a platform are interwoven. In addition, many forums, if not all, have input from developers, programmers, and software architects. That is why forums are difficult environments for non-expert LMSs users to make sense of.

In addition, there is no support (neither human nor software products) able to help teachers in clarifying, defining and then specifying their learning situations before setting them up within the LMS. They have to appropriate the various screens and form-based interfaces to abstract some low-level details to think about their global design courses.

Teachers need solutions to narrow the gap between their educational intention and the pedagogical features proposed by the LMS at their disposal. They ask for appropriate tools helping them in understand the underlying “way of thinking and designing” of this LMS.

In our work, we aim at supporting practitioners to overcome these LMSs’ obstacles in order to help them in focusing on the design of learning situations.

Our contribution consists of extracting, identifying, and formalizing the LMS implicit instructional design language. We also on purpose propose a meta-model formalism to capture it. The meta-model is obtained by the abstraction of pedagogical features and services provided by the considered LMS. This meta-model acts, according to the language theory, as an abstract syntax. It will then be used as a basis for the development of external editors [15].

III. OUR APPROACH

We propose a method to identify and formalize the instructional language of LMSs. Our approach takes into account a macro-HMI analysis, a functional analysis and a micro-analysis. In this section, we sketch an overview of our approach then we explain in details each step of the method.

A. *Overview of our approach*

In our work, we focus on pedagogical tasks and functionalities of a specific LMS. Our hypothesis is that LMSs are not pedagogically neutral and they embed an implicit language based on the LMS specific paradigm to specify the design of a learning activity. Our work aims at define the necessary analysis and steps for the identification and formalization of an LMS instructional design language.

Our proposed process takes into account the presence of common elements between pedagogical activities/resources on LMS. The final meta-model includes elements that are relevant for instructional design such as activity completion conditions, as well as outcomes and grade conditions.

Our method is specified according to three different viewpoints: a viewpoint centred on macro-HMI, a functional viewpoint and a micro viewpoint.

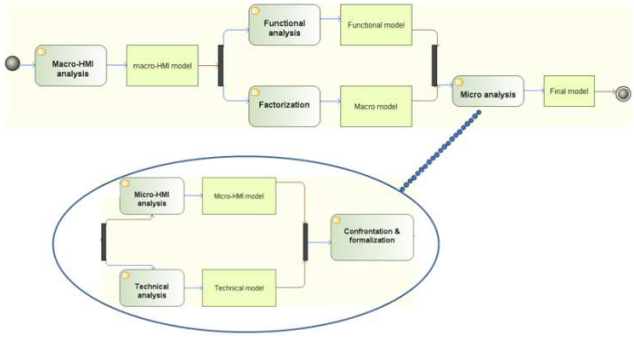

Figure 1. Analysis process of the instructional design language.

The first viewpoint consists of HMIs analysis according to two strategies: (1) the analysis of existing situations on the platform and (2) the analysis of interfaces related to the specification of new situations. After the macro-HMI analysis, we factorized the macro-HMI model in order to obtain the simplified macro model. The second viewpoint focuses on the identification of LMS existing functions. The third viewpoint concerns the micro analysis of the LMS instructional design language.

Figure 1 shows the proposed process. It is composed of the macro-HMI analysis, the factorization of HMI-macro model, the functional analysis and the micro analysis. The micro analysis is based on the micro-HMI analysis and technical analysis. The final model results from a confrontation of micro-HMI and technical models.

In the next sections, we present in details different steps of the process.

B. Macro-HMI Analysis

The macro-HMI analysis consists of identifying platform interfaces related to the Instructional Design (ID).

LMSs are usually composed of many interfaces, developed for different purposes and users' categories. In our work, we have ignored interfaces related to administration and management purposes; we are only interested in interfaces related to instructional design usages. The instructional design language is identified using two methods: the analysis of interfaces titles and the analysis of the navigation paths.

The first analysis step is to choose the main interface. Then, the analysis must determine whether or not the interface provides a pedagogical aspect. Interfaces related to ID are taken into account. The main interface concept is identified and presented on the macro-HMI model. Relations between model concepts are finally identified and defined.

Interfaces identification is an iterative process. When a new interface is identified, the analyst studies existing links inside this interface in order to access to new interfaces. Only Interfaces related to ID are analyzed and added to the macro-HMI model.

The macro-HMI model is presented by the meta-model format. We have chosen the meta-model format because it allows presenting clearly platform elements, their attributes, relations between them and their cardinalities.

C. Factorization

Factorization is the process of finding common attributes shared between two or more pedagogical elements (classes) in the macro-HMI model and moving them into an existing or a new abstract parent element. The non-common attributes will not change place. The difference between an abstract class and a concrete class is that a concrete class can be instantiated. The role of an abstract class is that of possessing concrete subclasses. This is important for the factorization of the attributes and common methods realized by the sub-classes. Visually, an abstract class is represented implicitly with a cursive formatting (in italics) of the name of the class (cf. figure 2, Activity/Resource class).

Many works shows the relevance of classes and associations factorization in modelling languages. The factorization we propose is based on the fact that a model is a simplified view of a system. Therefore a model element can factorize the system collection of elements [12].

This step aims at find common elements in pedagogical activities/resources and common relations between them. Factorization is applied on the Macro-HMI model. The macro model, resulting for the factorization, is clearer and more simplified than the Macro-HMI model.

D. Functional analysis

In the software engineering field, a software life-cycle model includes a functional analysis in the requirements and specification phases. Functional requirements are associated with specific functions, tasks or behaviours the system must support. Functional specifications describe what the system must do as well as requested properties of inputs and outputs.

In our context, the functional analysis aims at identify the functionalities dedicated to the course instructional design. The HMIs of the Macro-HMI model are analyzed from both functional and pedagogical perspectives. Administrative perspectives (like display functions, etc.) are rejected from the functional model. The functionalities are implicitly embedded in interfaces via HMI widgets (buttons, links, etc.) facilitating the interactions between users and system. Each widget has to be tested in order to determine its pedagogical features. Then, the analyst has to give a function name for each pedagogical widget (as answer a quiz, etc.). The functional analysis is an interactive process, every time we identify a new function, we must verify its pedagogical use. Only functions with pedagogical use are presented on the model. Sub-functions are also added to the functional model.

E. Micro analysis

The micro analysis is based on the macro and the functional models. It takes into account two different viewpoints: micro-HMI and technical viewpoints. We propose a confrontation of micro-HMI and technical models to formalize the final model.

1) Micro-IHM analysis

The micro-HMI analysis consists of analyzing the concerned interfaces at a finer scale. It aims at identify all elements relevant to the instructional design, including their features (attributes, types, etc.). To conduct this analysis, we propose many steps. After choosing an element of the macro model, the analysis concerns the interfaces for realizing/defining a dedicated use case of the functional model. The concerned interface is break down into many areas. Each component of each area (titles of blocks, menus, forms, etc.) has to be analyzed in order to determine its pedagogical features. The analysis concerns also many pedagogical elements which are described by the use of various forms, widgets and software components (buttons, links, etc.). Two main categories of the forms elements/attributes can be identified: required elements and optional elements. The required ones have to be identified because they form the main elements of the LMS instructional design language. The non-setting of these elements prevents the ordinary working of system. These characteristics have to be identified: it presents an important feature about the instructional design language of learning platforms.

2) Technical analysis

The second step of the process concerns the technical analysis. Several technical analyses are possible: databases, source code, courses backup/restore, etc. During this step, the main source of information for identifying the instructional design language is the LMS database. The other technical analyses will be used during the confrontation step.

This analysis consists of specifying a reduced Conceptual Data Model from the one available by LMS providers if it exists. In our approach, the database analysis has to be restricted to the tables/columns in relation to instructional design data. The main obstacle is to identify these data. Information from the micro-HMI analysis could be useful to achieve this goal.

This technical analysis consists of (1) looking over all database tables in order to sketch a first draft of the model, (2) focusing on tables embedding elements in relation to instructional design concepts. These tables can be identified through the semantic analysis of their titles or their record fields. Some tables could be identified through their dependencies with others or through the foreign keys. The analysis consists then in specifying the database schema on the basis of the databases reverse engineering rules. The Conceptual Data Model can be finally specified from this schema. This model is relevant to represent the technical-model viewpoint because it hides ill-structured databases, misconceptions or redundancies.

3) Confrontation and formalization

The last process step concerns the confrontation of both micro-HMI and technical models, and the formalization of the final model. The micro-HMI and technical models are compared in order to (1) refine the micro-HMI model, (2) detect and correct the difference between models, (3) ensure that the final model can be easily bind to a computer-readable format for the existing LMS.

The confrontation conducts verifications on the definition of the instructional design elements on both models. Some differences or ambiguities (like the definition of similar elements, the non-existence of some attributes, divergences about the types of attributes, etc.) are so identified. They require a deeper and finer analysis of both HMI and technical analysis. At this step, other technical-centred analysis (source code, backup packages, etc.) can be useful. For example the source code analysis consists of directly reviewing the LMS code.

It primarily concerns the code of the HMI definition and the queries for inserting / selecting data. This analysis can reveal many details that developers have chosen to encode for effectiveness or portability reasons. The aim of this process step is to formalize the instructional design language.

IV. MOODLE 2.4 CASE STUDY

In this section we present the application of the identification and formalization process on Moodle 2.4 [7] for many reasons: (1) Moodle is increasingly used in schools, universities and companies, (2) Moodle is also used in our university, and (3) Moodle has an active community that continuously develops APIs and tools. Note that the version 2.4 is the installed version in our university.

A. Application of the Macro-HMI Analysis on Moodle

The application of macro-HMI analysis on Moodle consists of identifying interfaces related to course design. We analyzed interfaces titles and navigation paths / URLs.

We studiously browse all the links in a specific interface. These links often point to new interfaces. Moodle is

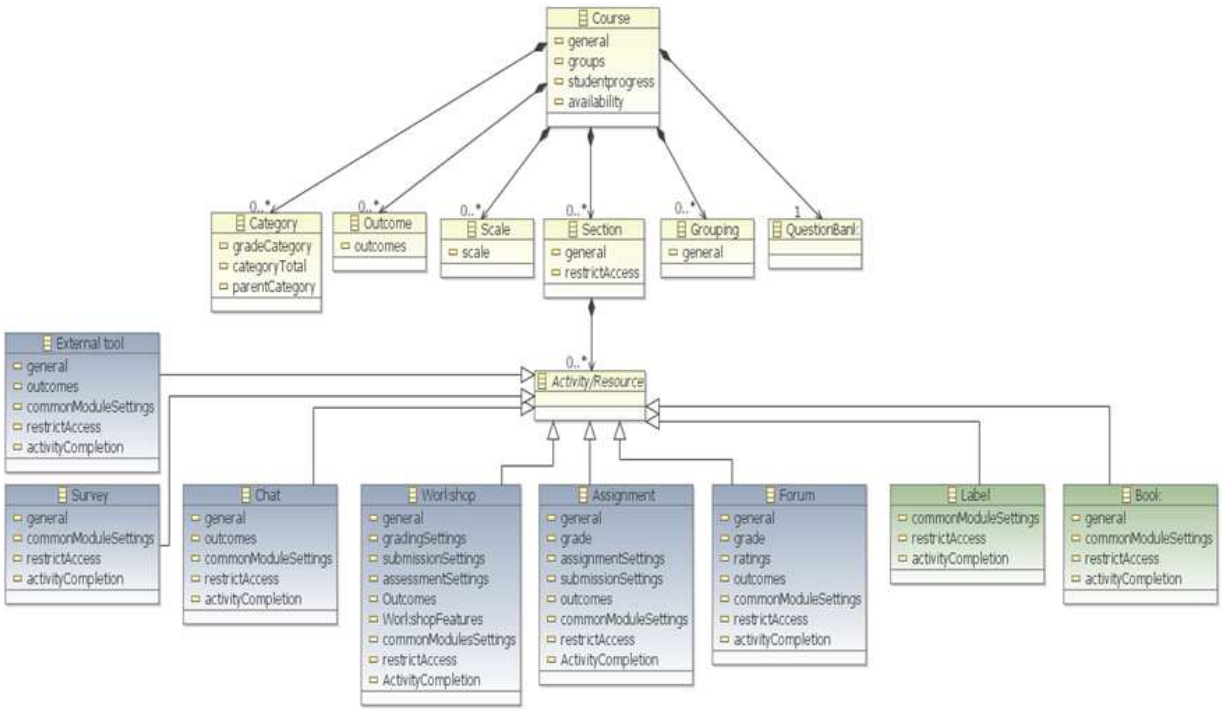

Figure 2. An extract of the Moodle macro-HMI model.

designed based on a top-down approach: the main interface is about specification and presentation of the course content, other interfaces (like add a forum, a label...) are accessible from the main interface.

The figure 2 shows the result of applying the macro-HMI analysis on Moodle. A course is composed of categorie(s), outcome(s), scale(s), section(s), group(s), grouping(s) and one question bank. Course sections are organized into resources and activities for students. 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). In figure 2, we present only one resource (Label), and 5 activities (Survey, Chat, Workshop, Quiz, and Forum) for clarity reasons.

In the page specification of each concept, attributes are divided into different parts. For example, for the Chat activity, its fields are divided into 4 parts named: general, common module settings, restrict access, and activity completion. These parts names are presented in the macro-HMI model. Note that there are only two types of relationships within this model: composition relationship and inheritance relationship.

B. Application of the factorization on Moodle

We then applied the factorization process. We noticed that all activities/resources had the common attributes: “commonModuleSettings”, “restrictAccess”, and “activityCompletion”. So we moved these attributes to the Activity/Resource class. All activities had the common attribute “general” according to the macro-HMI model, that’s why we created a class called “Activity” and we moved the attribute “general” into it. Some Moodle activities could have outcomes like Chat activity, Workshop, and Quiz. We added in the macro model a class named “ActivityWithOutcomes”. This class had “outcomes” as an attribute. We noticed that some activities with outcomes could be graded. Therefore, we added the class “GradedActivityWithOutcomes”.

Among “GradedActivityWithOutcomes” class, some activities had the common attributes “grade”. The “ActivityWithGradedSection” class is created and contained the “grade” attribute. Some activities from the “ActivityWithGradeSection” had the common attributes “ratings”. The class “ActivityWithRatingsSection” is added to the macro model with the attribute “ratings”. All coming steps are carried out on the basis of this analysis.

C. Application of the factorization on Moodle

We then applied the factorization process. We noticed that all activities/resources had the common attributes: “commonModuleSettings”, “restrictAccess”, and “activityCompletion”. So we moved these attributes to the Activity/Resource class. All activities had the common attribute “general” according to the macro-HMI model, that’s why we created a class called “Activity” and we moved the attribute “general” into it. Some Moodle activities could have outcomes like Chat activity, Workshop, and Quiz. We added in the macro model a class named “ActivityWithOutcomes”. This class had “outcomes” as an attribute. We noticed that some activities with outcomes could be graded. Therefore, we added the class “GradedActivityWithOutcomes”.

Among “GradedActivityWithOutcomes” class, some activities had the common attributes “grade”. The “ActivityWithGradedSection” class is created and contained the “grade” attribute. Some activities from the “ActivityWithGradeSection” had the common attributes “ratings”. The class “ActivityWithRatingsSection” is added to the macro model with the attribute “ratings”. All coming steps are carried out on the basis of this analysis.

D. Application of the functional analysis on Moodle

Based on the macro-HMI model, we proceeded to the functional analysis on Moodle. We divided each interface to several areas. Then, for each area, we studied the graphical interface components to identify functionalities related to instructional design. For example, from the main interface of a Moodle course, a teacher can show/hide/move a section. He can modify the course description, and manage different groups. He can also add an activity/resource in a specific section. If the teacher adds a forum, he will be pointed to a new page about forum specification. He can add files, add/modify/delete/separate a discussion and also reply to a discussion.

We have grounded the formalism of the functional model on the SADT (Structured Analysis and Design Technique) Model [13]. SADT is a multi language supporting the communication between users and designers. It is based on simple concepts in an easy graphical and textual formalism. This language is conformed to our functional analysis approach: top-down, hierarchical, modular and structured. This analysis is very important in our process; it can verify existence and relation between macro-HMI elements.

E. Application of the micro analysis on Moodle

The micro analysis consists of several sub-analysis: the micro-HMI analysis, the technical analysis, and the confrontation and formalization process.

1) Micro-IHM analysis

The application of IHM-micro analysis is about characteristics identification of instructional design elements. It is based on the macro and functional models.

For example, the “Course” class has “general” as attribute. In this phase, we study in details fields with pedagogical use related to this attribute. “Fullname” and “shortname” are these fields, so we replace “general” attribute in the macro-HMI model by “fullname” and “shortname” attributes in the micro-HMI model.

The figure 3 shows an extract of Moodle micro-HMI model (without taking into account corrections in red). Reference relationships appear in this model. For example

the abstract class "ActivityWithOutcomes" refers to "Outcome" class: a teacher can define outcomes to a course then he can associate a specific outcome to Moodle activities except for Choice, Survey, Wiki and Feedback activities.

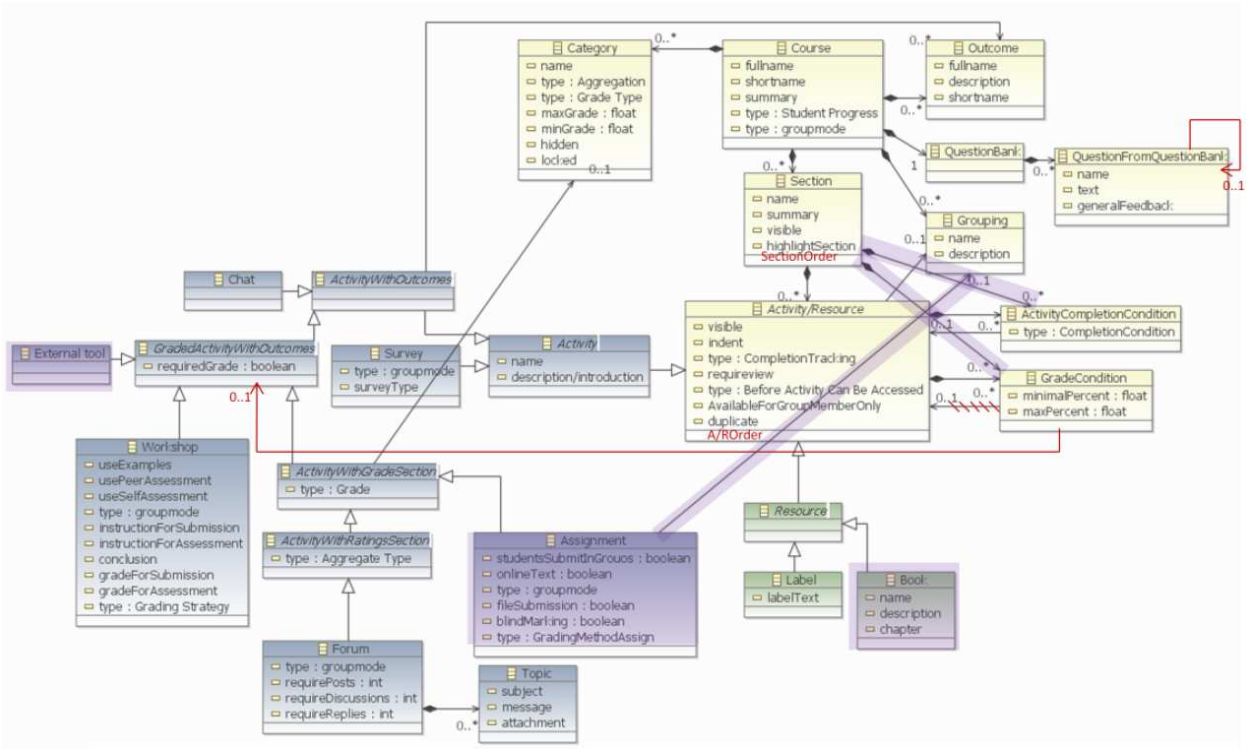

Figure 3. An extract of Moodle micro-HMI model (without corrections in red), an extract of Moodle 2.4 final model (with corrections in red).

2) Technical analysis

The technical analysis consists of analyzing the Moodle database. Our goal is to identify the Moodle instruction design language from a technical viewpoint to approve the relevant of specific data for this language.

This analysis consists of specifying the reduced Conceptual Data Model for Moodle in relation with the instruction design. We have reviewed all Moodle database tables. Titles semantic analysis of tables and fields allows to (1) gather the tables related to the ID, and (2) ignores those related to technical specifications (users' management, learners' tracking…). Then we studied dependences and relations between database tables. The generated Conceptual Data Model is based on reverse engineering rules. Foreign keys enable the specification of required multiplicities.

3) Confrontation and formalization

The micro-HMI analysis and the technical analysis have specified two Moodle instructional design models according to two different viewpoints. In this step, we are interested in the confrontation of these models to formalize Moodle instructional design language. This step is very important in our process. We think that the use of only one analysis method presents many negative points. For example, the micro-HMI model depends directly on the Moodle analyst competence. This means the possibility lack of pedagogical attributes. Similarly, the technical analysis is not an easy task. Many data structures are not explicitly reported when creating the database.

From the 2 models comparison, we noticed that every element/ attribute existing in the micro-HMI model is certainly presented in the technical model. But some elements exist in the technical model without being present in the micro-HMI model. That is why we refer to the PHP source code analysis of Moodle to verify the presence of these elements.

Figure 3 (including corrections in red) shows an extract of Moodle final model. Corrections in red present the confrontation result of the two models. For example thanks to the technical analysis, we found that every section has an order. This attribute has not been detected by the micro-HMI model. The code source analysis confirms the presence of this attribute. The attribute "SectionOrder" is presented in the final model.

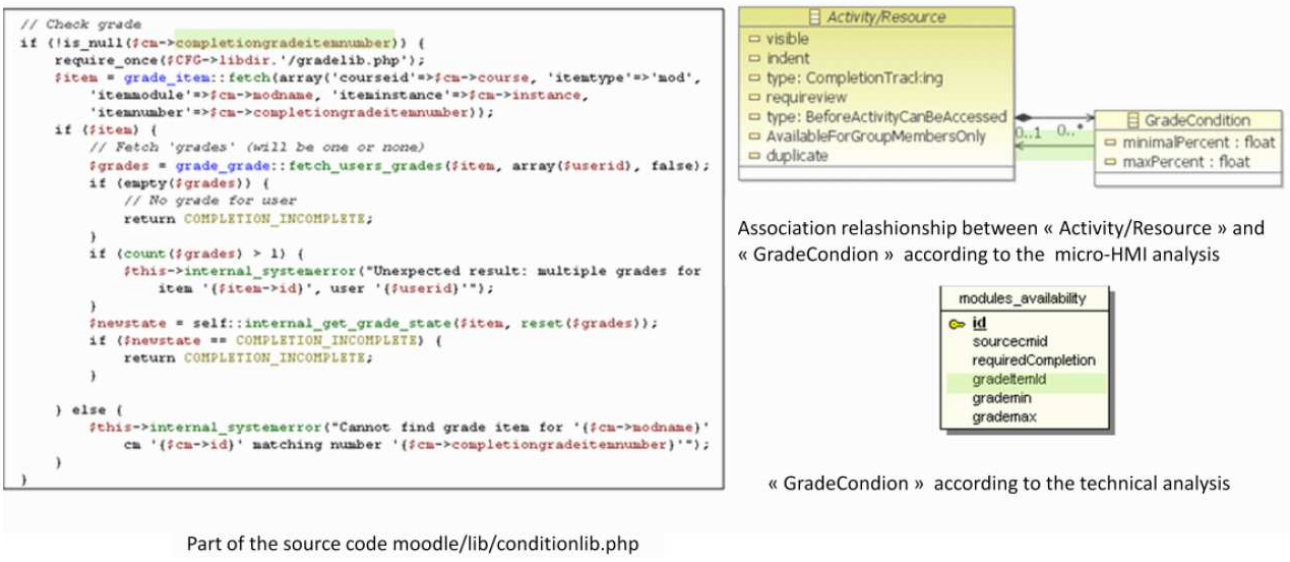

Figure 4. An example about relationship verification between the "GradeCondition" class and the "Activity/Resource" class.

The confrontation phase allows also rectifying information on the micro-HMI model. Figure 4 shows an example about relationship verification between the "GradeCondition" class and the "Activity/Resource" class.

Based on the micro-HMI analysis, the "GradeCondition" class refers to the abstract class "Activity/Resource" while the same class refers to a graded activity in the technical model. The code source analysis of Moodle conditionlib.php file confirms that the grade condition refers to a graded activity. That is why the reference relationship is between the two classes "GradeCondition" and "GradedActivityWithOutcomes" in the final model. The final model resulting from the confrontation phase formalizes the Moodle instructional design language.

V. COMPARISON BETWEEN MOODLE 2.0 AND MOODLE 2.4 META-MODELS

In this section, we apply our identifying and formalizing approach on Moodle 2.0 then we compare the two meta-models (Moodle 2.0 and 2.4) in order to identify differences between these versions. Figure 5 (including corrections in red) shows an extract of the Moodle 2.0 final model.

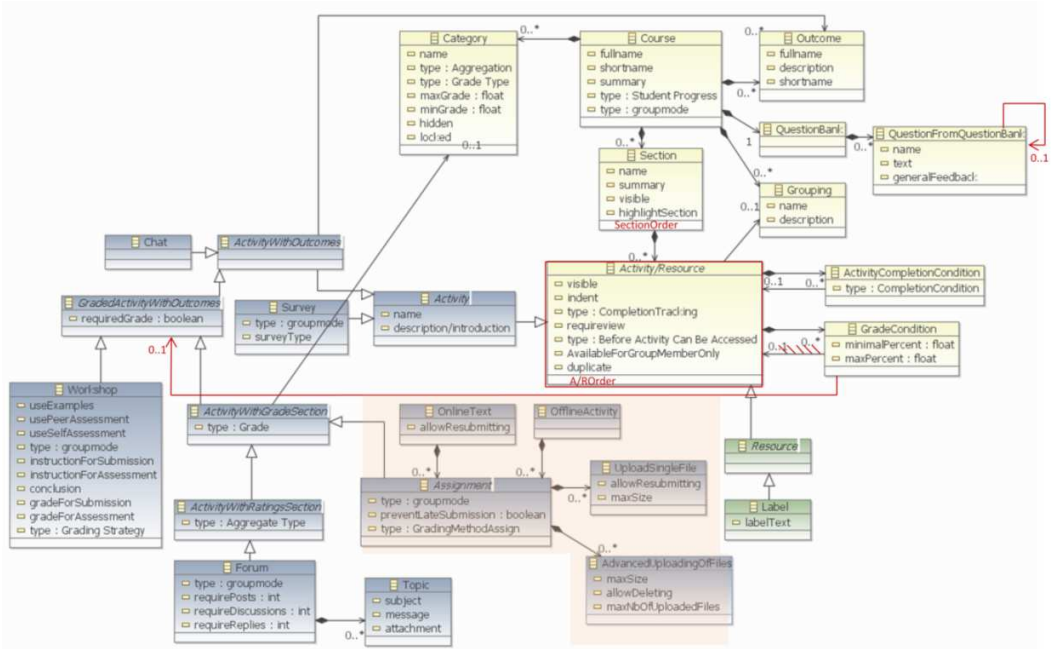

Figure 5. An extract of the Moodle micro-HMI model/ the Moodle 2.0 final model (without/with corrections in red).

TABLE I. DIFFENCE BETWEEN MOODLE 2.4 AND MOODLE 2.0 META-MODELS

	Moodle 2.4	Moodle 2.0	Comments
External Tool class	Yes	No	The external tool activity module enables students to interact with learning resources and activities on other web sites.
Book class	Yes	No	The book module enables a teacher to create a multi-page resource in a book-like format, with chapters and subchapters.
Relation between Section & ActivityCompletionCondition classes	Yes	No	This relation determines any activity completion conditions which must be met in order to access the section.
Relation between Section and GradeCondition classes	Yes	No	This relation determines any grade conditions which must be met in order to access the activity.
Assignment class	Yes	Yes	In Moodle 2.4 we have 1 class for assignment (Assignment) while in Moodle 2.0 we have 4 classes for assignment (Online text, Advanced uploading files, Offline activity and Upload single file).
blindMarking attribute for Assignment class	Yes	No	Blind marking hides the identity of students to markers.
gradingMethodAssignment attribute for Assignment class	Yes	No	This attribute defines the advanced grading method (Simple direct grading, Marking guide, Rubric) used for calculating grades in the assignment.
Relation between Assignment and Grouping classes	Yes	No	Students are able to collaborate on an assignment.

Table 1 presents the differences between the two meta-models process by being capable of identifying the new functionalities added by Moodle 2.4 developers in comparison to Moodle 2.0 (The whole Moodle's meta-models are available at the following link : http://www-lium.univ-lemans.fr/~laforcad/graphit/wp-content/uploads/2015/02/metamodels.pdf).

VI. CONCLUSION

In this paper, we present a meta-model-based approach and method for identifying and formalizing LMS languages. We apply our proposed method on the Moodle platform. Note that we are applying this method on three other LMSs: Ganesha, Dokeos and Sakai. The meta-model will be used as a basis for the development of the external editor by using a Model Driven Engineering tooling. It will guide and generate most of the final code for the editor. We have then been able to propose a graphical and external editor communicating with the system thanks to a dedicated API developed and integrated to the LMS. It will offer more user-friendly and soundly computer artefacts when development is freed from the technological choices related to the initial design of the TEL system considered. This will facilitate the use of LMS and allow to teachers and pedagogical engineers (service information technology and communication for education) of becoming more familiar with the specific design upon this LMS. Our approach promotes the use of all LMS activities and resources and expands LMS pedagogical concepts not by adding new concepts to users but by facilitating and clarifying the existing tools thanks to the external editor.

REFERENCES

[1] P.Tchounikine, A. Morch, and L. Bannon, "A computer science perspective on technology-enhanced learning", The Netherlands: Springer. (2009) 275-288.

[2] B. Gros, J. Elen, M. Kerres, J. Merrienboer, and M. Spector, "Instructional Design and the Authoring of Multimedia and Hypermedia Systems: Doeas a Marriage make Sense?", Educational Technology, 37 (1), (1997) 48-56.

[3] M. Szabo, and k. Flesher, "CMI Theory and Practice: Historical Roots of Learning Managment Systems", E-Learn 2002 World Conference, Montreal, Canada (2002) 929-936.

[4] J. Jovanovic, D. Gasevic, C. Brooks, V. Devedzic, M. Hatala, "LOCO-Analyst: a Tool for Raising Teachers' Awareness", Online Learning Environments. ECTEL, Crete, Greece (2007) 112-126.

[5] L. Oubahssi, P. Laforcade, and P. Cottier, "Re-engineering of the Apprenticeship Elec-tronic Booklet : Adaptation to new users requirements", IEEE-ICALT, Sousse (July 2010) 511-515.

[6] F. Abdallah, C. Toffolon, and B. Warin, "Models transformation to implement a Project- Based Collaborative Learning (PBCL) scenario: Moodle case study", ICALT, Spain (2008) 639-643.

[7] I. Martinez-Ortiz, J.L. Sierra, and B. Fernández-Manjón, "Enhancing IMS LD Units of Learning Comprehension", In the 4th ICIW, Venice, Italy (May 2009) 561-566.

[8] O. Mekpiroona, P. Tammarattananonta, N. Buasrounga, N. Apitiwongmanita, B. Pravalpruka, and T. Supnithia, "SCORM in Open Source LMS : A case study of LEARNSQUARE", In ICCE2008, Taipei, Taiwan, (2008) 166-170.

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

[10] R. Baggetun, E. Rusman, and C. Poggi, "Design Patterns for collaborative learning: from practice to theory and back", In EdMEdia, pp. 2493-2498, Chesapeake, VA: AACE 2004.

[11] C. Alario-Hoyos, M.L. Bote-Lorenzo, E. Gómez-Sánchez, J.I. Asensio-Pérez, G. Vega-Gorgojo, and A. Ruiz-Calleja, "GLUE!: An Architecture for the Integration of External Tools in Virtual Learning Environments", Computers & Education. 60(1), (2013) 122-137.

[12] A. Al-Ajlan, and H. Zedan, "E-learning (Moodle) based on service oriented architecture", EADTU's, 8-9 November, Lisbon-Portugal, vol. 1, (2007) pp. 62-70.

[13] P.-A. Caron, A. Derycke, and X. Le Pallec, "The Bricoles project: support socially informed design of learning environment", 12th International Conference on AIED, Amsterdam, 2005 p 759 - 761.

[14] H. Coates, R. James, and G. Baldwin, "A critical examination of the effects of learning management systems on university teaching and learning", Tertiary Education and Management, 11, (2005) 19-36.

[15] E. Loiseau, P. Laforcade, "Specification of learning management system-centred graphical instructional design languages - A DSM experimentation about the Moodle platform", ICSOFT'13, Reykjavik, Iceland (2013) 29-31.