



A TOOL TO GENERATE A COLLABORATIVE CONTENT COMPATIBLE WITH IMS-LD

Fauzi El Moudden¹, Souhaib Aammou², Mohamed Khaldi³

¹ Abdelmalek Essaâdi University, Faculty of Sciences, LIROSA, Tetouan, MOROCCO

² Abdelmalek Essaâdi University, Faculty of Sciences, LIROSA, Tetouan, MOROCCO

³ Abdelmalek Essaâdi University, Faculty of Sciences, LIROSA, Tetouan, MOROCCO

Abstract: In this research we try to adapt the meta-model IMS-LD with a model supporting collaborative learning. This adaptation will go through three stages, first the development of a collaborative model, secondly, the study of correspondence between the developed model and meta-model IMS-LD and their transformation to IMS-LD meta-model that reduced the MDA approach in a transformation that is based on the rules implemented in the ATL language.

Keywords: IMS-LD, MDA, ATL, collaborative learning.

I. Introduction

Distance learning is promoted through educational platforms: integrated systems offer a wide range of activities in the learning process. Teachers use the platforms to monitor or evaluate the work of students. They use content management systems (LCMS) to create courses, tests, etc. However, the platforms do not offer personalized services and therefore do not take into account the aspects of personalization such as the level of knowledge, interest, motivation and goals of learners. They access the same resource sets in the same way.

In fact, we present an easy way for teachers to create and administer the educational content online collaboratively. This tool allows the generation and editing structures of websites through database rather than pedagogical models, with a variety of choices that ensures better adaptation to the teaching of the course and learning style. Otherwise, the social constructivism approach is centered on the learner activity to support synchronous and asynchronous collaboration. Therefore, it is necessary to find a method to model all types of activities. In order to model the activities we have based ourselves upon the IMS-LD specification focusing on collaborative learning.

II. Theoretical approach

A. Collaborative Learning Online

A.1. Definition

According to [1], collaborative learning is any learning activity carried out by a group of learners with a common purpose, as every source of information, motivation, interaction, and support ... each with inputs from other, the synergy of the group and the help of a trainer facilitating individual and collective learning.

A.2. Importance of collaborative learning online.

Collaborative learning was experienced at the onset of online education in the late 1980s under the name, of computer conferencing, e-mail first, then by forums.

As online learning, collaborative learning allows learner's to benefit from great flexibility of time and place (stimulant autonomy and reflection) and an excellent asynchronous interaction (source of motivation for mutual, critical thinking, synthesis ...). That is why [2] reported in 1989 that "the collective nature of computer conferencing may be the single most critical and fundamental underlying theory development and the design and implementation of educational activities line. »

In this context, collaborative learning is the most important online educational contribution. And irrefutable logic [3], offering an online education without making those who follow the benefits of its "most fundamental" is absurd and devalues the remarkable educational tool that provides telemetric learners. This does not mean that online education should be limited to collaborative learning online! But it is important that any online program includes a minimum of collaborative learning and exploits the extent of appropriateness a suitable program and its student's way of learning.

Since the emergence of web, e-learning aroused great enthusiasm and developed rapidly in the form of educational materials posted on the web, often without human interaction, and sometimes email interaction between each learner and tutor, previous forums for interaction of each student with a tutor and peers. But the

online collaborative learning activities in small groups have been neglected by many directors online training; they are probably too busy to multiply online training to concentrate on the design and facilitation of small group activities, while they are the most beneficial innovation of online learning.

B. Instructional Management Systems (IMS-LD)

IMS-LD was published in 2003 by the IMS / GLC. (Instructional Management Systems Global Learning Consortium: Consortium for global learning management systems with training, the original name when IMS was started in 1997 Instructional Management Systems project). [4] Reminds us of its origins: the source (EML) of the proposed language was assessed by the European Committee for Standardization (CEN) in a comparative study of different SRMS [5], as best suited to satisfy the criteria definition of an EML. EML (Educational Modelling Language (EML)) is defined by CEN / ISS as "an information aggregation and semantic model describing the content and processes involved in a unit of learning from an educational perspective and in order to ensure the reusability and interoperability." In this context, the North American IMS consortium undertook a study and provided a specification of such a language, giving birth in February 2003, the Learning Design specification V1.0 (IMS-LD). She adds that proposal, largely inspired EML developed by [6] (OUNL) provides a conceptual framework for modeling a Learning Unit and claims to offer a good compromise between on the one hand to the generic implement a variety of instructional approaches and secondly, the power of expression that allows an accurate description of each learning unit.

This specification allows us to represent and encode learning structures for learners both alone and in groups, compiled by roles, such as "learners" and "Team". [7] We can model a lesson plan in IMS-LD, defining roles, learning activities, services and many other elements and building learning units. The syllabus is modeled and built with resources assembled in a compressed Zip file then started by an executable ("player"). It coordinates the teachers, students and activities as long as the respective learning process progresses. A user takes a "role" to play and execute the activities related to in order to achieve a satisfactory learning unit. In all, the unit structure, roles and activities build the learning scenario to be executed in a system compatible with IMS LD.

IMS-LD does not impose a particular pedagogical model but can be used with a large number of scenarios and pedagogical models, demonstrating its flexibility. That is why IMS-LD is often called a pedagogical meta-model. Previous initiatives in e-learning pretend pedagogically neutral, IMS-LS is not intended to pedagogical neutrality but seeks to raise awareness of e-learning on the need for a flexible approach.

IMS-LD has been developed for e-learning and virtual classes but during a face-to-face can be made and incorporated into a structure created with this specification, as an activity or learning support activities. If the ultimate goal of creating rich learning units, with support to achieve the learning objectives by providing the best possible experience, face-to-face and other learning resources are allowed such as videoconferencing, collaborative table or any action research field.

IMS-LD uses the theatrical metaphor, which implies the existence of roles, resources and himself learning scenario: a room is divided into one or more acts and conduct by several actors who can take on different roles at different times. Each role must perform a number of activities to complete the learning process. In addition, all roles must be synchronized at the end of each act before processing the next act. Figure 1 shows a conceptual model of these three levels.

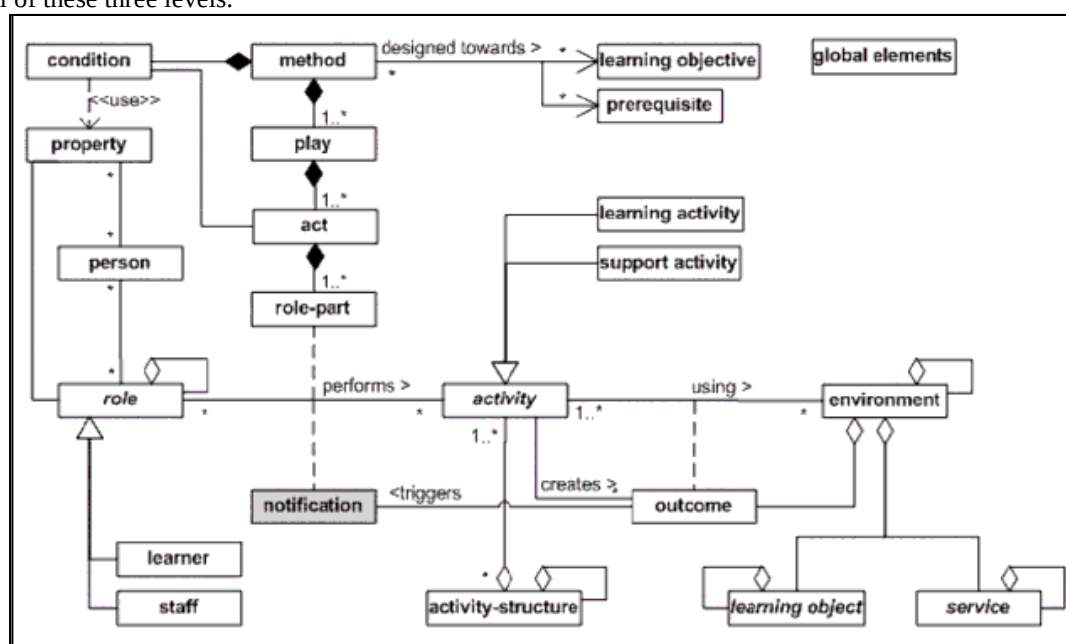


Fig. 1 The conceptual model of IMS LD [8]

III. Model-driven engineering

A. Modeling Driven Architecture (MDA)

In November 2000, the OMG in the field of software engineering consortium of over 1,000 companies, has initiated the process MDA [OMG MDA], the concepts-oriented models rather than object-oriented. The Model Driven Architecture MDA [OMG MDA] offers the power of abstraction, refinement and different views of the models. This standard has to add a new way to design applications by separating business logic from business, any technical platform to increase the reuse of previously developed code, reducing development time and facilitating the integration of new technology [9]. It gives the opportunity to develop independent models platforms and implementation [10] environment. MDA is used to separate two extreme views of the same system [11]:

- its functional specifications on the one hand;
- its physical implementation on the other hand;

Including several aspects of the life of the software, namely its tests, its quality requirements, the definition of successive iterations, etc. The MDA architecture consists of four layers. In the center, there is a UML (Unified Modeling Language) standard MOF (Meta-Object Facility) and CWM (Common Warehouse Meta-model). The second layer contains the XMI (XML Metadata Interchange) standard for dialogue between the middleware (Java, CORBA, .NET, and Web services). The next layer refers to the services to manage events, transaction security, and directories. The last layer offers specific frameworks in scope (Telecommunications, medicine, electronic commerce, finance, etc.) A designer to create his own application can use UML as it can use other languages. So according to this architecture independent technical context, MDA proposes to structure the front needs to engage in a transformation of this functional modeling technical modeling while testing each product model [12]. This model of application is to be created independently of the target implementation (hardware or software). This allows greater reuse of patterns. MDA is considered an approach with the ambition to offer the widest possible view of the life cycle of the software, not content with only its production. Moreover, this is intended overview described in a unified syntax. One of the assumptions underlying the MDA is that the operationalization of an abstract model is not a trivial problem. One of the benefits of MDA is to solve this problem [13]. MDA proposes to design an application through software chain is divided into four phases with the aim of flexible implementation, integration, maintenance and test:

- The development of a computer model without concern (CIM: Computer Independent Model).
- The manual transformation into a model in a particular technological context (PIM: Platform Independent Model);
- The automatic transformation into a pattern associated with the target implementation of the platform (PSM: Platform Specific Model) model to be refined;
- Its implementation in the target platform.

B. Collaborative meta-model

In our research, we propose a meta-model for a system designed to achieve the needs of educational projects that require online collaboration, and the needs of teachers in terms of generation of collaborative educational content. Therefore, we establish the following diagram as a first proposal of a meta-model for collaborative learning:

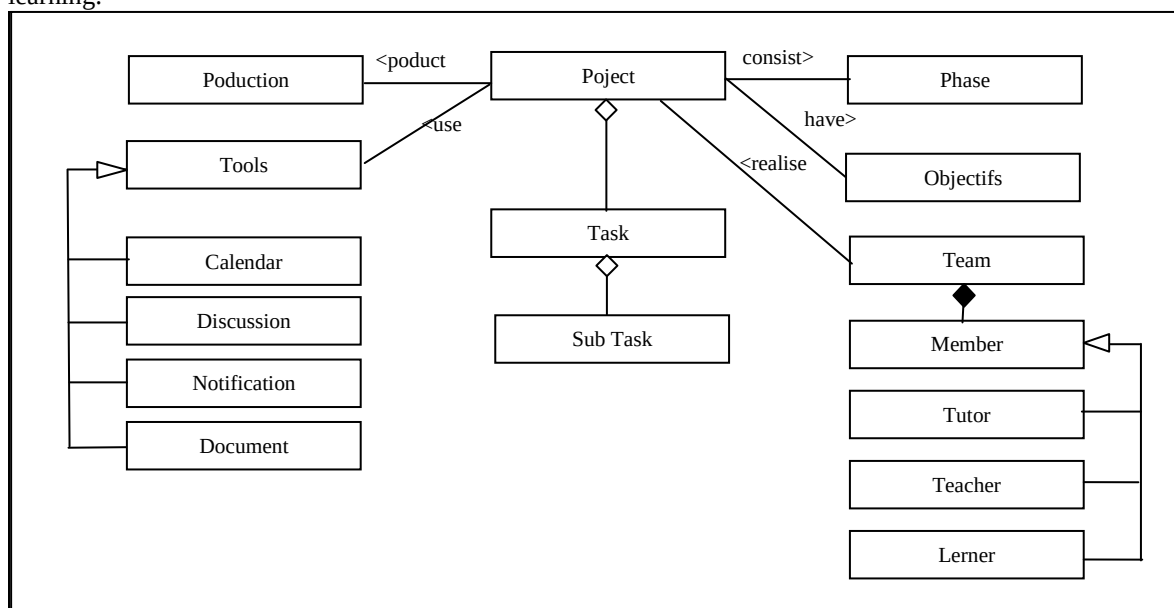


Fig. 2 Proposal of conceptual collaborative meta-model

We propose subsequently a collaborative model from our meta-model proposed in (Fig. 2) in which we defined each class properties and the relationships between them:

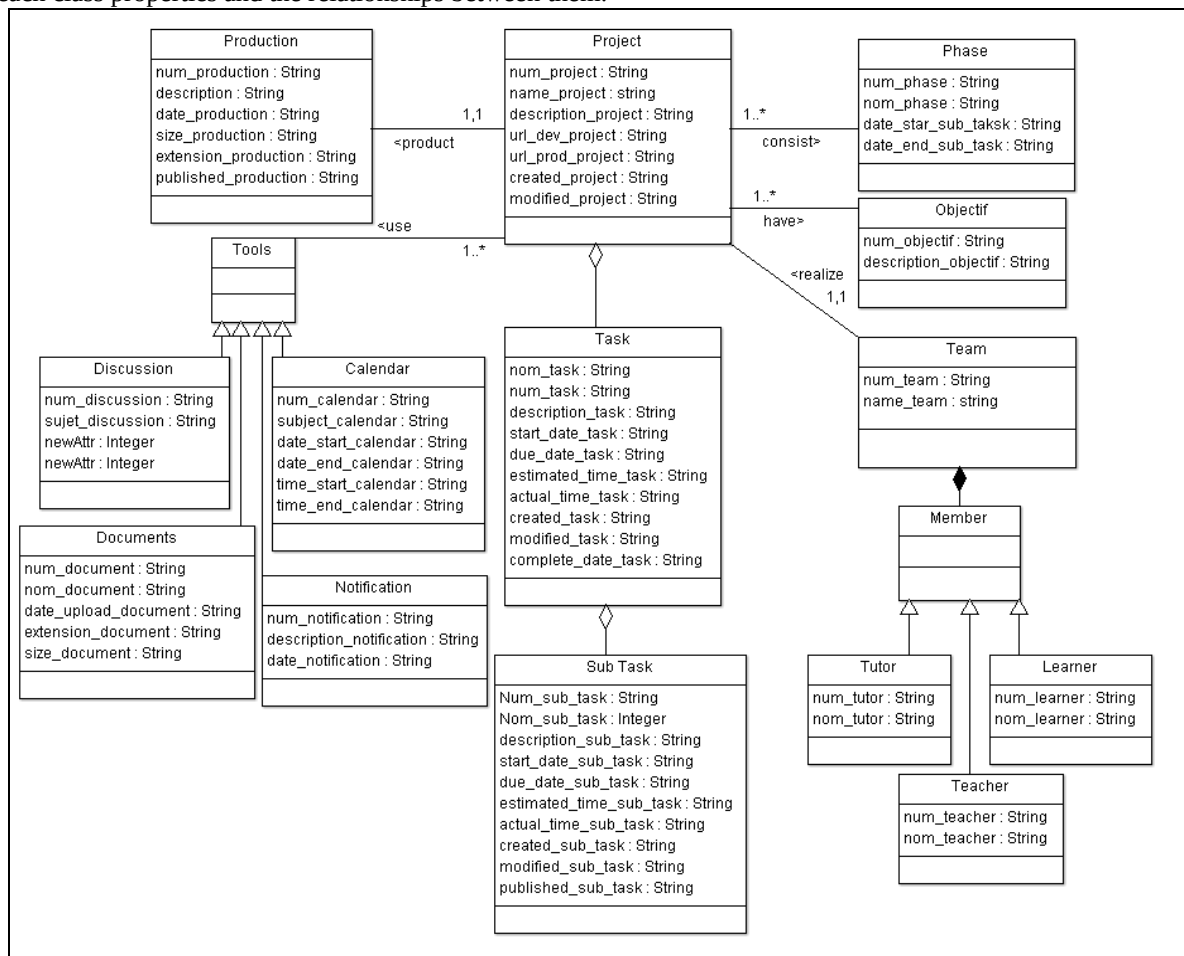


Fig. 3 Poposal of collaborative model

C. Correspondence between the terminology of IMS-LD and that of the collaborative model

The majority of classes designed in our collaborative model correspond perfectly with the IMS-LD model, which makes possible their transformations to it. The transformation of models is a technique aims to put links between models in order to avoid unnecessary reproductions. In the next section, we will discuss how to perform transformations between models, starting with the study of model driven engineering and ending with the rules of transformations that will be used to make our collaborative model compatible with meta-model IMS-LD. In the following table we have tried to collect all the collaborative model classes and their equivalent at the IMS-LD:

Table 1 Correspondence between the terminology of IMS-LD and that of the collaborative model

Collaborative meta-model	IMS-LD
Project	Activity
Task	Role
Subtask	Activity structure
Team, Members	Person
Teacher	Staff
Learner	Learner
Production	Outcome
Notification	Notification
Objective	Learning Objective
Services	Services

IV. Transformation rules

A. Atlas Language Transformation (ATL)

In their operational ATL, Canals et al use. [14] State that to deal with the transformation of models; it is difficult and cumbersome to use object languages since we spend so much effort to the development of transformation

definitions of Framework for the set work. The use of XSLT as a language if it is more direct and adapted by rest against difficult to maintain [14]. We follow their choice by focusing on the implementation of approaches centered on the MDA (Model Driven Architecture), MDE (Model Driven Engineering) and QVT (Queries View Transformation) tools. Query / View / Transformation (QVT) [18] is a standard defined by the OMG. This is a standardized language to express model transformations. QVT is not advanced sufficiently now in its definition for Queries and View aspects. Against transformation by the appearance expressed by the MDA approach has resulted in various experiments (eg Triskell, ATL ...) in both academic and commercial level. To determine the transformation, it is necessary to have tools of transformations. These are based on languages transformations must respect the QVT standard [18] proposed by the OMG [15]. There is an offer of free tools (ATL, MTF, MTL, QVTP, etc.) and commercial (eg MIA). We chose ATL (Atlas Transformation Language) from the provision of free tools, to the extent that only ATL has a spirit consistent with OMG / MDA / MOF / QVT [14].

B. ATL Description

Atlas Transformation Language (ATL) has been designed to perform transformations within the MDA Framework proposed by the OMG [16], [17]. The ATL language is mainly based on the fact that the models are first-class entities. Indeed, the transformations are considered models of Transformation. Since transformations are considered themselves as models, we can apply their transformations. This possibility of ATL is considered an important point. Indeed, it provides the means to achieve higher order transformations (HOT Higher- Order Processing) [17]. A higher-order transformation is a transformation including source and target models that are themselves transformations. As ATL is among the languages model transformation respecting the QVT [18] standard proposed by OMG [15], we describe its structure in relation to this standard (QVT).

The study of the abstract syntax of the ATL language is to study two features provided by this language more than rules changes. The first feature, navigation, allows to study the possibility of navigation between meta-models sources and targets. The second feature, Operations, used to describe the ability to define operations on model elements. Finally, the study of the transformation rules is used to describe these types of rules, how they are called and the type of results they return.

- Navigation [16], [15] this feature is offered by ATL language (Object Constraint Language). Navigation is allowed only if the model elements are fully initialized. The elements of the target model cannot be definitively initialized at the end of the execution of the transformation. Therefore, the navigation in ATL can only be made between elements of the model (or meta-model) source and model (or metamodel) target.
- Operations: [16] this feature ATL is also provided by the OCL (Object Constraint Language). In OCL, operations can be defined on the elements of the model. ATL takes this opportunity to OCL to allow defining operations on elements of the source model and the transformation model [15].
- The transformation rules: there are several types of transformation rules based on how they are called and what kind of results they return (Fig. 1)

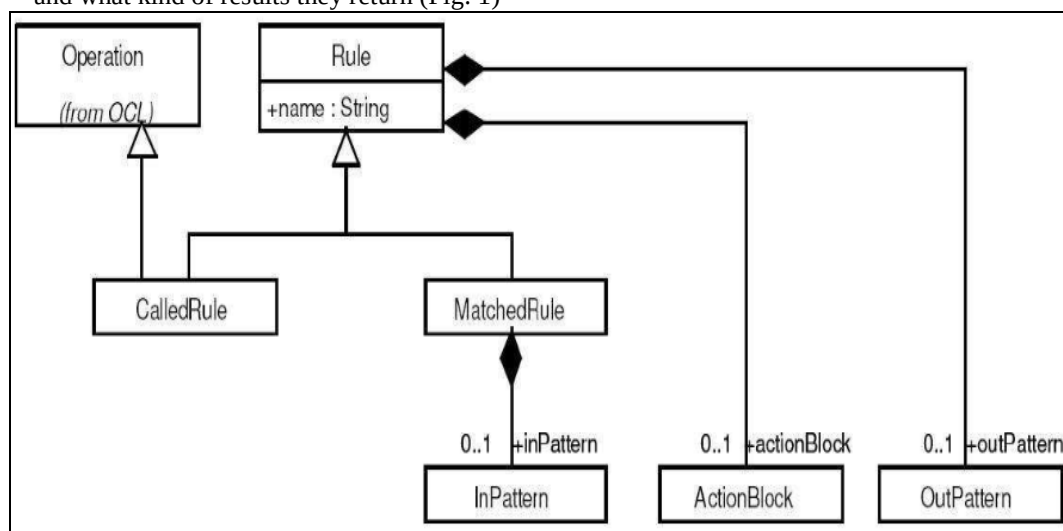


Fig. 4 ATL Transformation rules

CalledRule [16] rule explicitly called by its name and by setting its parameters.

MatchedRule [16] rule executed when a guy (InPattern) scheme is recognized in the source model.

The result of a rule may be a set of predefined models (OutPattern) or a block of mandatory (ActionBlock). If the rule is MatchedRule type and if its result is a set of elements of the target model (OutPattern), it was named declarative. If it is CalledRule type whose result is a block of statements, it is then called procedure. Combinations of rules (declarative and imperative) are called hybrid rules. [15]

C. Rules of transformations from the metamodel Collaborative Project to the meta-model IMS-LD

In this section, we define, explain and argue the transformation rules developed from the collaborative *méta_modèle* to that of IMS-LD metamodel. A rule defines the "mapping" between classes of meta-models and the rules for handling attributes and relationships of classes.

D. Rules between the collaborative meta-model and IMS-LD Meta-model activity structure side.

In the collaborative meta-model, there are three levels of conceptualization to meet: Project, Task (task) and subtask (Subtask). These three concepts are linked by relations of composition (a project consists of tasks that are composed of sub-tasks). While in the meta-model IMS-LD, we found that the concept and structure activity that describe these three concepts.

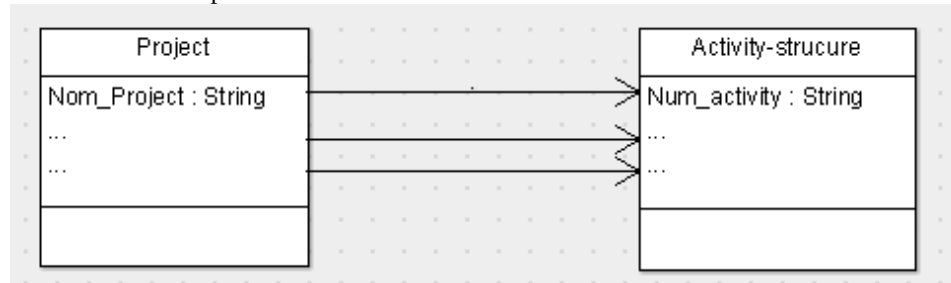


Fig. 5 Rule Project to Activity-structure

Regarding the first rule (rule *Projet2Structure*) (Fig. 5), we need to create a rule type (*MatchedRule*) that will have as source class (*Draft*) and a target class (*Structure of activity*). This rule is the "mapping" between the Project of class collaborative meta-model structure and activity meta-model IMS-LD and handling attributes of these classes.

E. Rule between the Collaborative meta-model task and the IMS-LD activity model side

Regarding the second rule (rule *Task2Activity*) (Fig. 6), we also chose a type rule (*MatchedRule*) of source class (*Task*) and a target class (*Activity*). This rule is the "mapping" between the Task of class collaborative meta-model and meta-model Activity IMS-LD and handling attributes of these classes.

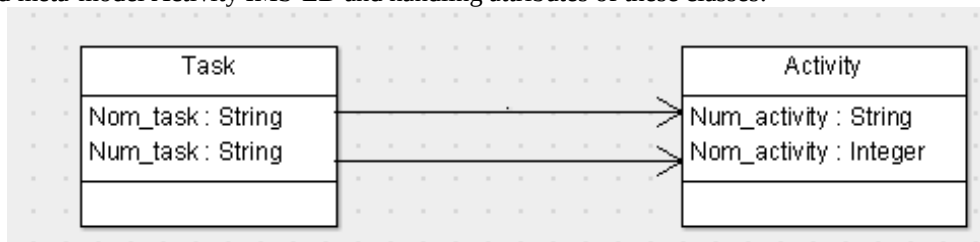


Fig. 6 Rule Task2Activity

F. Rule between the Collaborative meta-model Subtask and the IMS-LD activity model side

For the third rule (rule *Subtask2Activity*) (Fig. 7), we also chose a type rule (*MatchedRule*) of a source class (*Subtask*) and a target class (*Activity*). This rule is the "mapping" between the Task of class collaborative meta-model and meta-model Activity IMS-LD.

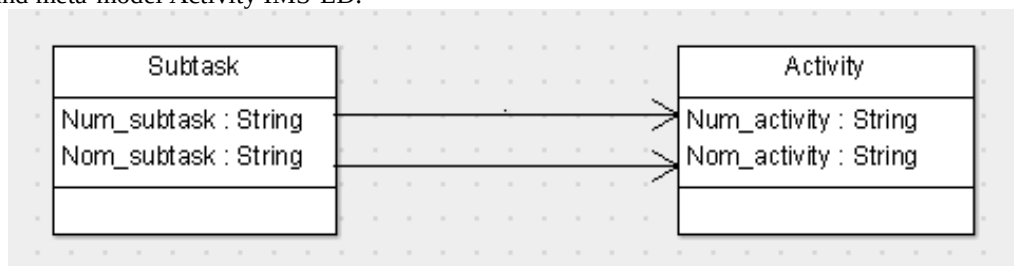


Fig. 7 Rule Subtask2Activity

Concerning the handling of the relationship between the classes Task and Subtask of collaborative meta-model, we are unable to have a close relationship in the meta-model IMS-LD because it does not exist in the meta-target model (IMS-LD) of equivalent relationship. In IMS-LD, we do not have the opportunity to build an Activity which consists of several Activities. For this, there is at this level a semantic loss (certain specified information in a scenario disappear) (Fig. 8). And it is the responsibility of the user to verify the name Subtask project to see what Subtask correspond to which task. We will see in the following rule how we concatenate the title of Subtask with that of its Task and Subtask this number with that of the identifier of the task to address this problem.

```

Rule Subtask2Activity {
  From
    to
      e: collaboratif !subtask

      c: IMS-LD!Activity(
        full_name<-e.x+'-'+e.subtask_nom
        num_activity<-e.w+'-'+e.subtask_num
      )
}

```

Fig. 8 Concatenation ATL code

Concatenation is not ensured within the rule directly. To do this, we specify that we use a "Helper". [15] We define two "helper" (Fig. 9) in the context of the concept "Subtask". The first is used to return the value of the attribute "name_subtask" concept "Subtask" by assigning the "Helper". The second is used to return the value of the attribute "Number_Subtask" concept "Subtask" by assigning the "Helper".

```

helper context collaboratif!subtask def: x:string=
  self.subtask.nom_task;

helper context collaboratif!subtask def: w:string=
  self.subtask.num_task;

```

Fig. 9 Code ATL helper

V. Conclusion

In this chapter we took the road to the modeling of a meta-model of collaborative online education compatible with IMS-LD, we started in part by the expression of the need for this modeling suite we developed a section regarding the IMS LD specification in the theoretical framework of this chapter.

In the second part of this chapter we began by studying the MDE approach (Model Driven Engineering) and we adopted the MDA (Model Driven Architecture) defined by the OMG approach based on four stages of implementation:

- The development of a computer model without concern (CIM: Computer Independent Model).
- The manual transformation into a model in a particular technological context (PIM: Platform Independent Model);
- The automatic transformation into a pattern associated with the target implementation platform (PSM: Platform Specific Model) model to be refined;
- Its implementation in the target platform.
- This led us to study the transformation language ATL that in turn has allowed us to define the transformation rules to our proposed meta-model IMS-LD meta-model.

Concerning executed transformations in this chapter, we limited ourselves to a few examples by what we have given way to forward for other transformations to perform all other transformations from collaborative proposed meta-model to the meta-model IMS-LD.

References

- [1] Henri F., Lundgren-Cayrol K., Apprentissage collaboratif à distance - Pour comprendre et concevoir les *environnements d'apprentissages virtuels*, Sainte-Foy (Canada), Presses de l'Université du Québec, 2001
- [2] Harasim L., « Online education: a new domain », in Mason R. & al. (ed.), *Mindweave*, Oxford, Pergamon, 1989, p. 50-52, <http://www-icdl.open.ac.uk/literaturestore/mindweave/chap4.html>
- [3] Salmon G., *E-Moderating - The key to teaching and learning online*, Londres, Kogan Page, 2000.
- [4] Lejeune, A. *IMS Learning Design : Étude d'un langage de modélisation pédagogique*, Revue Distances et Savoirs, volume 2.
- [5] CEN/ISS WS/LT, *Learning Technologies Workshop "Survey of Educational Modelling Languages (EMLs)". Version 1*, septembre 2002.
- [6] Koper, R. (2001). *Modeling Units of Study from a Pedagogical Perspective*, the pedagogical meta-model behind EML. <http://eml.ou.nl/introduction/docs/pedmetamodel.pdf>.
- [7] Burgos, D., Berbegal, N., Griffiths, D. (2005a). *IMS Learning Design Level 0*. <http://moodle.learningnetworks.org/>.
- [8] *IMS Learning Design* www.epi.asso.fr/revue/articles/a0512c.htm
- [9] P. Boulet, J.L. Dekeyser, C. Dumoulin, and P. Marquet. *Mda for soc embeddeb systems design*, intensive signal processing experiment. In *SIVOESMDA workshop at UML 2003*, San Francisco, October 2003.
- [10] D. Thi-Lan-anh, G. Olivier, and S. Houari. *Gestion de modèles : définitions, besoins et revue de littérature*. In *Premières Journées sur l'Ingénierie Dirigée par les Modèles*, pages 1–15, Paris, France, 30 Juin- 1 Juillet 2005.
- [11] A. Clave. *D'uml à mda en passant par les méta modèles*. Technical report, La Lettre d'ADELI n 56, 2004.

- [12] A. Clave. D'uml à mda en passant par les méta modèles. Technical report, La Lettre d'ADELI n 56, 2004.
- [13] P.A. Caron, F. Hoogstoel, X. Le Pallec, and B. Warin. Construire des dispositifs sur la plateforme moodle - application de l'ingénierie bricoles. In MoodleMoot-2007, Castres, France, 14 - 15 Juin 2007.
- [14] A. Canals, C. Le-Camus, M. Feau, G. Jolly, V. Bonnafous, and P. Bazavan. Une utilisation opérationnelle d'atl : l'intégration de la transformation de modèles dans le projet topcased. In Génie Logiciel (73), pages 21–26, 2005.
- [15] J. Bézivin, G. Dupé, F. Jouault, G. Pitette, and J. Eddine Rougui. First experiments with the atl model transformation language : Transforming xslt into xquery. In OOPSLA 2003 Workshop, Anaheim, California, 2003.
- [16] ATLAS group LINA and INRIA Nantes. Atlas transformation language atl user manual - version 0.7. Technical report, INRIA University of Nantes, February 2006.
- [17] B. Combemale and S. Rougemaille. ATL - Atlas Transformation Language. Master 2 Recherche SLCP, module rtm edition, 2005.
- [18] OMG/RFP/QVT MOF 2.0 Query/Views/Transformations RFP. OMG, October 2002.