OPENING TEL SYSTEMS FOR TEACHERS
A Domain-specific Modeling & Model-driven Engineering Approach

El Amine Ouraiba, Christophe Choquet and Philippe Cottier
Maine University, IUT de Laval, 52 rue des Docteurs Calmette et Guérin, 53020 Laval Cedex 9, France

Keywords: Open Technology Enhanced Learning Systems, Instructional Design, Open Pedagogical scenario, Learning Session Adaptation, Domain-Specific Modeling, Model-Driven Engineering.

Abstract: Despite their quality, few TEL systems are actually adopted in educational institutions. These educational technologies have not always the necessary flexibility for use in real educational contexts that often requiring the rapid adaptations to new and often unexpected events (Cottier et *al*., 2008). Indeed, TEL environements should be designed as "open" in which the teacher himself is able to lead the adaptation and reengineering of learning system at an abstract level. In our work, we consider that opening of pedagogical scenario allows for the opening of TEL system. This article focuses on an approach based on the Domain-Specific Modeling and Model-driven Engineering for supporting practitioner teachers in their activities through the instructional design process. In order to verify our proposal we took Hop3x as experimentation field. Our objective is to open this TEL system for its users by providing them a user-friendly editor which allows the design and adaptation of learning sessions at a high-level of abstraction. We illustrate the development process of Hop3x's Domain-Specific Language and specific editor.

1 INTRODUCTION

Technology-enhanced learning (TEL) systems are dedicated to making their users learn. A TEL system is a complex environment that mobilizes human agents (learner, teacher) and artificial ones in interactions conceived in order to improve the quality of the human learning. The design of a TEL system is a significant effort for a learning institution. It is a complex process that is expensive in time and also in technical and human means. In this process the designer is involved to perform several choices about pedagogy, technology and interaction modalities. It must that the designed TEL system could be adapted to the evolution of usage's context, and be configured according to the evolution of users needs and activities. According to (Rogalski, 2003) the teacher, in his/her activity, should manage an "*open dynamic environment*". *Dynamic* because the learning process evolves even without teacher's intervention, this is called a spontaneous evolution. *Open* because the teacher cannot predict the spontaneous evolution of learners and the effect of his/her possible interventions (Roditi, 2003).

(Henri et *al.*, 2007) affirm that the learning environment is a *work in progress* which is improved session after session. This particularity requires, according to (Cottier and El-Kechaï, 2009), to think the design as a continuous and situated process. However, the engineering's classical methods are characterized by their rigidity and linearity. In one hand, after that the developed system is implanted, the only evolution that can have been is its maintenance. In other hand, for building this system, the design process is performed in steps that segment the time, the actors, and the works to realize (Bourguin and Derycke, 2005). This brings about a discontinuity between the design process and the usage process (separation between designers and users), and difficulties to fill the gap between the scientific disciplines mobilized, in particular between computer science and human and social sciences (Bowker, Star and Turner, 1997), (Bourguin and Derycke, 2005).

TEL systems should be designed as "open" (e.g. systems where the teacher is able to lead the adaptation and reengineering of learning system by himslef). The question asked by engineering of TEL systems is to "develop an adaptable and reconfigurable TEL system by its users according to

Amine Ouraiba E., Choquet C. and Cottier P..
OPENING TEL SYSTEMS FOR TEACHERS - A Domain-specific Modeling & Model-driven Engineering Approach.
DOI: 10.5220/0003345803120317
In *Proceedings of the 3rd International Conference on Computer Supported Education* (CSEDU-2011), pages 312-317
ISBN: 978-989-8425-49-2

usage's context evolution" rather than to "develop a TEL system according to a given specification". In the open TEL systems engineering, the decisions and practices of instructional designer are fundamental (Cottier and El-Kechaï, 2009).

We consider that the pedagogical scenario is a relevant and strong model for TEL systems engineering and that the opening of learning scenario allows the opening of TEL system. The scenario is formalized with the help of an EML (*Educational Modeling Langage*) (Koper, 2001), defined by a specific meta-model which is itself linked by conformity relations with the scenario. Within this framework, two approaches exist in the research field for the scenario based TEL systems engineering (Choquet, 2007) : (1) *interpretative approach*, where an existing EML (such as (IMS LD, 2003)) is proposed to designer for specifying scenarios; (2) *constructive approach*, where the designer, generally helped by modeling specialists, build the meta-model which describes their domain-specific EML and use it for specifying scenarios.

Our work falls under this last approach. By the use of Domain-Specific Modeling (DSM) and Model-driven Engineering (MDE) paradigm we want to surpass the difficulties a teacher can encounter when using generic EMLs and existing editors (Ouraiba et *al.*, 2010). This, by defining the Domain-Specific Language (DSL) of teacher and developing accordingly a dedicated editor. The next section of this article presents the DSM/MDE paradigm how we instantiates it to support teacher both at design and run time. Thus, we took Hop3x as experimentation field of our approach. In section 3, we detail the work realized on the Hop3x TEL system for opening it where we illustrate our first results of the development process of Hop3x's Domain-Specific Language (DSL) and specific editor thanks to EMF tooling. We conclude by discussing the benefits of our approach and our future works.

2 DSM / MDE FOR SUPPORTING TEACHER

Model-Driven Engineering (MDE) is basically a software development approach where the code is produced by some transformations and combinations of models of the required artifact. It is an enhancement of the Model-Driven Architecture (MDA) approach, initially proposed by Object Management Group (MDA) in 2001 (the MDA document Guide (OMG, 2006) provides an overview and definitions of the used concepts) to provide a solution to the problem of software technologies continual emergence that forces companies to adapt their software systems every time a new "hot" technology appears (adaptability problem).

The first principle is to use modeling and models to develop software systems. The second principle is the separation of the enterprise functionalities of an information system from the implementation of those functionalities on specific technological platforms (EJB, CORBA, and so on). The abstracted specification of the system becomes the main asset in software development: many implementations using concrete technologies may be derived from the specification. It is model-driven while "it provides a means for using models to direct the course of understanding, design, construction, deployment, operation, maintenance and modification" (OMG, 2006). The software development process is based on automatic or semi-automatic transformations between models, from abstracted, domain centered and generally informal models (Computation Independent Model – CIM) to specific and platform dependent ones (Platform Specific Model – PSM).

Based on this initial proposal, the Domain Specific Modeling approach (DSM) was defined (1) to reduce the complexity of the transformations and the semantic losses they generate, and (2) to enhance the level of abstraction of the software specification (Kelly and Tolvanen, 2008). The principle here is to develop a Domain Specific Language (DSL), tailored for specifying software which instruments a specific activity in a specific context. This DSL has to be formal but its meta-model reflects the domain of the users: the modeling vocabulary used is the domain one. Then, code generators could be developed for directly transform models expressed with a DSL into a specific technological platform framework.

Particularly, we propose to adopt a DSM/MDE approach for allowing the teacher to assume his/her designer role, and the learning session actors to adapt the open pedagogical scenario dynamically. We consider that a pedagogical scenario, for being really designed and manipulated by a teacher, has to be considered as a domain specific model, expressed with a DSEML (Domain-Specific Educational Modeling Language) situated in his/her teaching context and rooted in his/her practices. In such a paradigm, MDE techniques have to support the transformation of the scenario from domain specific representation to operationalized one, both at the design phase to support the operationalization and at runtime to support the dynamic adaptation (Ouraiba

et *al.*, 2010).

The following figure 1 illustrates how we instantiates DSM/MDE paradigm to support teacher both at design and run time.

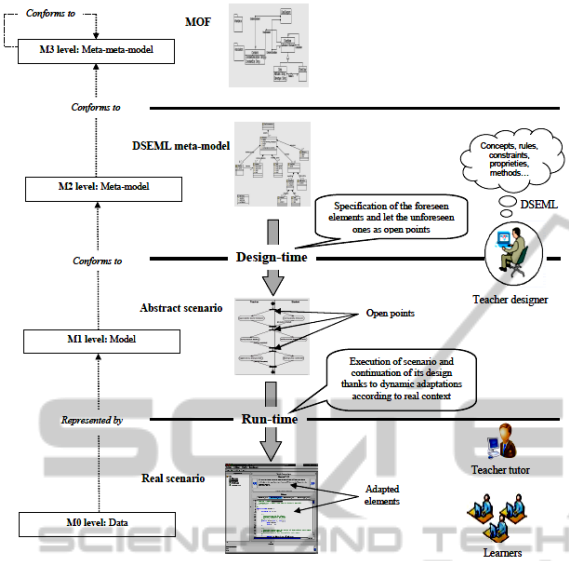

Figure 1: OMG layers view of the open pedagogical scenarios engineering.

3 DSM/MDE FOR OPENING HOP3X TEL ENVIRONMENT FOR TEACHERS

3.1 Hop3x TEL System

Hop3x is a practical works TEL environment developed for learning and teaching object-oriented programming languages, like: C, Ruby, and Java. We will focus here on Java programming. On one hand, Hop3x is structured around a specific Java editor/compiler where the learner has to solve programming exercises. In the other hand, it allows a tutor to intervene remotely and at runtime by providing help and recommendation to learners thanks to feedback system which provides high level information about learners' productions and tasks by the way of indicators calculated from tracks with DCL4UTL language (Pham Thi Ngoc et *al.*, 2009).

Hop3x is operating-systems independent and its architecture is composed of three applications:

- **Hop3x-Learner Interface**, which allows learners to write, edit, compile and run code and program. It also allows them to ask for help from the teacher via a synchronous communication tool.
- **Hop3x-Server**, which collects interaction tracks of learners and save them as Hop3x events. It allows real-time calculation of indicators.
- **Hop3x-Tutor Interface**, which is a monitoring tool for tutors. It allows them to manage a group of learners in a situation of distance but synchronous practical work.

After their authentication, learners can use their working interface. Before trying to solve exercises, learners must read the what-to-do and how-to-do instructions. The set of questions is presented as a sequence that is not static, in a way where learners can browse the list of questions and choose the order that they want to follow. Answering questions requires writing a Java-code. Generally, before obtaining a correct and executable Java-code, learners perform a series of updates on it (writing, editing and deletion), sometimes compilations and runs. If a learner finds a difficulty in his/her activity he/she can request help from tutor via the audio communication functionality.

The tutor, after authentication, can use the relevant interface which is composed of a set of functionalities developed in order to help him in sessions control and learners monitoring during their activities. These functionalities allow to: know which learners are connected and which are not; observe at real-time the Java-code written by each learner (in fact, a mirror of the learner's interface); replay with a selected speed the different stages a learner's session; have a look on each compilations and executions results for a given learner; make tutoring interventions via audio or textual (proactive messages) modalities; responds orally or textually to a learner's call (reactive message); see the history of the tutoring interventions, read the content of questions and instructions; consult specific and transversal indicators calculated at real time.

Each tutoring intervention is characterized by its manner, its modality, its category and its strategy. A proactive intervention could be motivated by the Java-code written quality and/or thank to indicators characterizing the learners activities. A reactive intervention could be fed also by indicators. These indicators are calculated using the language DCL4UTL (Pham Thi Ngoc et *al.*, 2009). The dimensions of tutoring interventions are classified into four categories (Després, 2001): didactical support about the taught content; methodological support about learning organization; technical support about resources provided to perform activities; and motivation about the psychological and emotional state. Thus, the intervention strategy

corresponds to the action that can be performed by tutor such as: making a course reminding, modifying a question, recommending the consultation of an additional (external) resource, encouraging learner to document his/her Java-code, explaining in order to help learner to have some reflection, etc.

3.2 Hop3x Learning Sessions

The sessions we have chosen to highlight here are parts of a course entitled "Object-oriented programming and Java". The learners involved are undergraduate learners. These learners are novices in Java programming but they had, during the preceding term, an introductory course entitled "Introduction to Object-oriented Programming". They attended lectures and tutorials about the basics notions and concepts of object-oriented programming such as class, object, instance, message passing, inheritance, encapsulation, overriding, overwriting and polymorphism. Then they use the Hop3x TEL system in order to implement these concepts in situation.

In the version used here, Hop3x-server can only run learning sessions that are provided in the format of a XML file. Thus, to design a *Hop3x learning session,* designer (which is a teacher) writes manually a XML file in which he/she defines its structure (help by a template). He/she identifies firstly the basic elements that compose the mandatory layer of the session's structure (section 4), these elements are: *session's characteristics (*name, programming language, pedagogical objective, time of its start and end*)*, *actors* who can be involved *(learners*, *tutors* and *groups)*, the *instructions* to be respected by learners, and some elements that can be used such as the necessary *resources*. Secondly, he/she defines different *learning sequences (scenarios)* that he/she can anticipate. The set of these sequences defines the foreseen contextual layer in the session's structure. A learning scenario is composed of a set of *questions* characterized by an ID, a content, a set of indicators and three types of tasks: required (what the learner has to do), optional (what the learner has of doing to think) and prohibited (what the learner cannot do).

3.3 Development Process of Hop3x DSEML and Specific Editor

The Hop3x system is developed initially for executing linearly learning sessions, without deviation and dynamic adaptations at runtime of predictive learning scenarios. Our ultimate objective is to "open" more this TEL system which remains rather closed in its current version. We think that the opening of a TEL system can be realized by its reengineering in order to have a new version which can allows its users (especially teachers) to design and adapt dynamically learning sessions according to the current context. We want indeed to enable teachers to participate actively in the adaptation and reengineering activities.

We aim by our work to provide the necessary tools and conceptual means to teachers who use Hop3x to enable them to design and adapt learning sessions at a high-level of abstraction without needing to create XML files by hand or with the help of a generic tool (such as Reload Editor (Reload, 2004)), we adopt the DSM/MDE approach to help teachers to define their business language (DSL) for developing accordingly a specific editor more user-friendly. For this, we followed a pragmatic methodology that starts from an existing situation that is of legacy Hop3x TEL system, and then try to propose improvement solutions to make it more open for users who haven't much technical knowledge. First of all, through a first process that is presented in the rest of this article we want to develop knowledge about the relevance of the implementation of specific tools, which are based on the users business, without taking into account the dynamic adaptation aspects.

This first process of our methodology lasts 4 steps (see Figure 2). In the first step, we investigated the semantic of Hop3x: we collected and analyzed the use cases of the actual version of Hop3x for extracting domain specific concepts and rules (Sections 3.1 and 3.2). Then based on this, we specified the metamodel that describes the Hop3x's domain specific language. This metamodel formalizes the semantic of Hop3x field by specifying the meaning of each concept and how it can be used according to domain's rules and respecting constraints. In the third step, a Hop3x-specific editor was generated from the DSL metamodel. This editor makes available as specification tools the concepts and rules that are handled usually in the Hop3x practices. Finally, a teacher could use this editor for designing the practical works sessions at an abstract level.

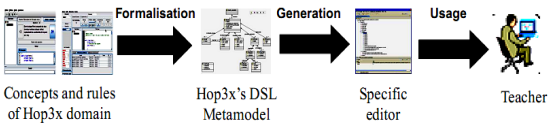

Figure 2: Development process of Hop3x DSL and specific editor.

3.4 Use of the EMF Tooling

The Eclipse Modeling Projects (EMP, 2008) provides a unified set of modelling frameworks, tooling, and standard implementations, such as EMF (Eclipse Modeling Framework), GMF (Graphical Modeling Framework) and ATL (ATLAS Transformation Language). In the following, we use the EMF because it facilitates code generation for building tools and other applications based on a structured metamodel (Steinberg et *al*., 2008). Our objective was to specify a metamodel which describes the Hop3x's DSL, and then generating from this metamodel the code of the editor thanks to tools provided in EMF. Indeed, this metamodel is an ECORE model where ECORE is the MOF-like meta-meta-model in EMF. Figure 3 illustrates this metamodel in the class-diagram-oriented view proposed by the ECORE graphical internal editor of EMF.

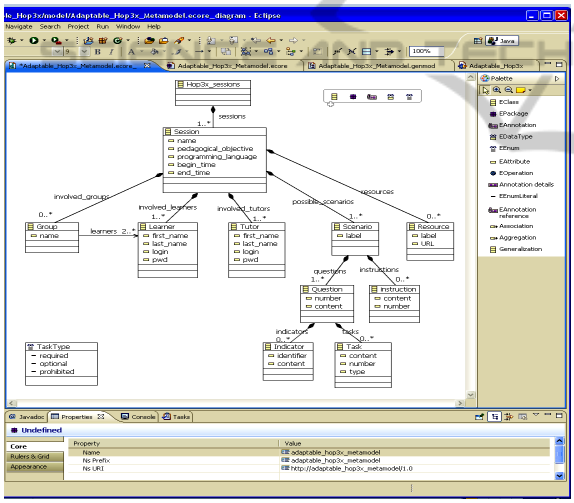

Figure 3: Hop3x's DSL metamodel.

A first version of the editor has been generated automatically from Hop3x's DSL metamodel thanks to the EMF tooling. This editor provides a tree-view of the models which are namely the Hop3x learning sessions (see Figure 4). By using this editor the teachers who want to use Hop3x can design the practical works sessions at an abstract level compared to the manual creation of XML files as it is the case currently.

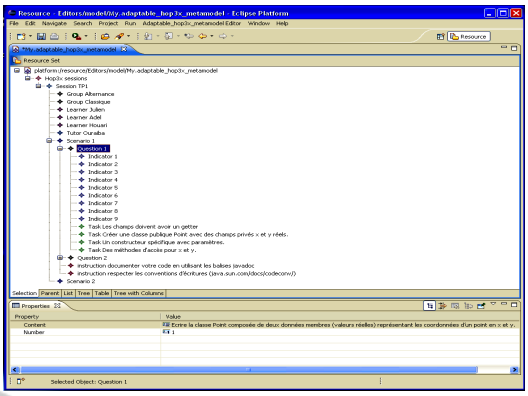

Figure 4: Example of a Hop3x learning session designed by the specific editor.

Finally, thanks to this editor, designer can generate learning sessions in the XML format required by Hop3x system. The following figure 5 shows an example of a Hop3x learning session generated as an XML file after its design by the specific editor.

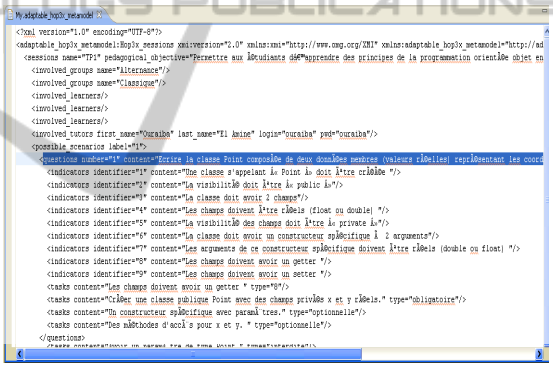

Figure 5: XML file of a Hop3x learning session designed by the specific editor.

Recently, we have conducted a testing experimentation of the specific editor with the students who preparing the "Professional License in the Design and Realization of Multimedia Services and Products". These 22 students had particularly lessons related to learning design. As work they had to describe a practical learning session of Hop3x using two editors separately, the first one is generic (Reload Editor (Reload, 2004) which implements IMS LD (IMS LD, 2003)) and the second one is the specific editor which we have developed based on Hop3x's DSEML (see Figure 4). Our goal was simply to verify which editor was intuitive enough to enable the autonomy of its user. Beyond this testing, we have noted the interests of "putting in the hands of users" a specific editor, freed from the conceptual and technical barriers of learning session's representation.

4 CONCLUSIONS

The pedagogical scenario can be considered among the strong models for investigating the engineering of open TEL systems. It is relevant to open TEL system through the opening of scenario. Our work falls under the constructive approch of instructional design of open scenarios where we have adopted the DSM/MDE paradigm. To verify our proposal we took Hop3x as experimentation field. The ultimate objective is to further open this TEL system for its users. For this, we follow a pragmatic methodology that spans on two processes. In the first one, presented in this article, we deal with the potential of specific tools of instructional design. While in the second one, we plan to investigate the adaptation possibilities by specific tools. Indeed, although our methodology spends more time and effort, a first benefit is that teachers have to develop a reflexive analysis on their practices and what they could do with the TEL system.

Using the developed Hop3x-specific editor we are currently conducting interviews with Hop3x's users in order to promote the expression of the adaptation requirements of learning sessions and the openness needs of TEL system. The information gathered from these interviews will help us to define another version of metamodel which will be more optimized in order to: (1) guide us in the perfection of Hop3x's functionalities for transforming it into a TEL system more open, and (2) to develop a graphical editor dedicated to the design and adaptation of learning sessions at a high-level of abstraction. We have planned to use the GMF in a second time to add a graphical layer on top of EMF.

ACKNOWLEDGEMENTS

Authors acknowledge the designers and users of Hop3x TEL Environment for their help and information about their habitual practices.

REFERENCES

Bourguin, G., Derycke, A., 2005. *Systèmes Interactifs en Co-évolution Réflexions sur les apports de la Théorie de l'Activité au support des Pratiques Collectives Distribuées*. Revue d'Interaction Homme-Machine Vol 6 N°1.

Bowkers, G., Leigh Star, S., Turner, W., Gasser, L., 1997. *Social science, technical systems and cooperative work: beyond the great divide*. Lawrence Erlbaum Associates, "Computer, cognition and work" series, 470 p.

Choquet, C., 2007. *Ingénierie et réingénierie des EIAH-L'approche REDiM*. Habilitation à diriger les recherches en informatique, Université du Maine.

Cottier, P., Choquet, C., Tchounikine, P. 2008. *Repenser l'ingénierie des EIAH pour des enseignants concepteurs*. In*: Usages, usagers et compétences informationnelles au XXIème siècle,* Jérôme Dinet (ed.), p.159-193. Édité par Hermes Lavoisier, ISBN 978-2-7462-2193-2.

Cottier, P., EL-Kechaï, H., 2009. *L'usager concepteur en situation: Conception collective d'un livret électronique d'apprentissage (LÉA)*.In*: Ingénierie des systèmes d'information*, ISSN1633-1311. vol.14, no3. p162

Després, C., 2001. *Modélisation et conception d'un environnement de suivi pédagogique synchrone d'activités d'apprentissage à distance*, Phd Thesis, Université du Maine, 286p

Eclipse EMP, 2008. Eclipse Modeling Projects. http://www.eclipse.org/modeling/. Retrieved from Octobre 2008.

Henri, F., Compte, C., Charlier, B., 2007. La scénarisation pédagogique dans tous ses débats. *Revue internationale des technologies en pédagogie universitaire*, 4(2).

IMS Global Learning Consortium, "IMS Learning Design Specification, Version 1 - Final Specification", 2003, available at: http://www.imsglobal.org/learningdesign/

Kelly, S., Tolvanen, J., 2008. *Domain-specific modeling: enabling full code generation*. 448 pages. Wiley-IEEE Computer Society Press. ISBN: 978-0-470-03666-2.

Koper, R., 2001. *Modeling units of study from a pedagogical perspective: the pedagogical meta-model behind EML*, Open University of the Netherlands.

OMG, 2006. *MDA specification guide*. Version 1.0.1. Report—omg/03-06-01.

Ouraiba, E. A., Choquet, C., Cottier, P., Després, C., Jacoboni, P., 2010. Engineering of open learning scenarios: the case of Hop3x learning scenarios. *Proceedings of IEEE ICALT'10*, Tunisia, pp. 264-268.

Pham Thi Ngoc, D., Iksal, S., Choquet, C., Klinger, E., 2009. UTL-CL:A Declarative Calculation Language Proposal for a Learning Tracks Analysis Process. *Proceeding of the 9th IEEE International Conference on Advanced Learning Technologies (ICALT'09)*, Riga, Latvia, July, pp.681-685.

Roditi, É., 2003. Régularité et variabilité des pratiques ordinaires d'enseignement. Le cas de la multiplication des nombres décimaux en sixième, *Recherches en Didactique des Mathématique*. 23 (2).

Rogalski, J., 2003.Y a-t-il un pilote dans la classe ? Une analyse de l'activité de l'enseignant comme gestion d'un environnement dynamique ouvert*. Recherches en Didactique des Mathématiques*. 23(3)342-348.

Steinberg, D., Budinsky, F., Paternostro, M., 2008. *EMF: Eclipse Modeling Framework, Second Edition*. Publisher: Addison Wesley Professional.

The Reload Project, The University of Bolton, The University of Strathclyde and JISC, 2004. Available at: http://www.reload.ac.uk/editor.html