SPECIFICATION OF OBSERVATION NEEDS IN AN INSTRUCTIONAL DESIGN CONTEXT
A Model-Driven Engineering Approach

Pierre Laforcade, Boubekeur Zendagui and Vincent Barré
Maine University - LIUM
IUT de Laval, 52 rue des Docteurs Calmette et Guérin, 53020 Laval Cedex 9, France

Keywords: Instructional Design, Learning Scenarios, Observation, Model Driven Engineering, Meta-Modelling.

Abstract: Our works take place in the research field of distant learning situation observations. We want to help instructional designers to improve the learning scenarios they design within a re-engineering context. We think that the observation of the learners' behavior can be improved by taking into account the observation needs from the designers. We originally think these observation needs can be related to and guided by the information specified into the learning scenarios. This article presents a *Model-Driven Engineering* approach for the specification of these observation needs. A specific metamodel has been elaborated to support our conceptual proposition and process. A dedicated example illustrates the use of this metamodel to specify observation needs according to a given learning scenario and Educational Modeling Language. First elements of tooling are also presented.

1 INTRODUCTION

In the research field of distant learning situation observation, most of the research deals with the analysis of the collected data during the learning session. Our works take place within a re-engineering context of these learning situations.

We want to help instructional designers to improve the learning scenarios they design. To achieve that, we assume and believe that the observation of the learners' behavior could be improved by taking into account the observation needs from the designers *a priori* of the learning session.

We also originally think that these observation needs can be related to and guided by information from the learning scenarios specifying the various facets of the learning flow. We aim at supporting and guiding designers when defining these observation needs. To this end we have already proposed a conceptual model for the observation needs, whose originality relies on the use of 'signs' and 'behavior categories' techniques (Zendagui et al., 08) as possible features for detailing observation needs on the base of learning scenario information. This paper focuses on the *Model-Driven Engineering* context and approach we follow to propose both a theoretical formalization of the links between observation needs and learning scenarios, and a practical tool for helping designers to define these needs. We follow such an approach because our previous work about *Domain-Specific Modeling* (DSM) & *Educational Modeling Languages* (EML) (Laforcade et al., 08) focuses on the use of meta-modeling techniques and Eclipse tools to support our proposition as a first formalization step.

The next section presents our research context about instructional design and observation needs. In the following section we briefly present our conceptual proposition. The MDE approach for the support of our conceptual proposition is then detailed and discussed in section 4. Then, section 5 is dedicated to our first results illustrating the use of specific MDE tools for supporting our metamodel and providing a tree-based graphical editor. Finally we conclude by discussing our research in progress and the benefits designers can expect to obtain.

2 INSTRUCTIONAL DESIGN CONTEXT

2.1 Re-engineering of Learning Scenarios

Within the instructional design context, the re-engineering of learning scenarios forms a cycle. In the first step, designers use an *Educational Modeling Language* (EML) (Koper et al., 05)(Kinshuk et al., 06) to define a learning scenario. This predictive model allows designers to explicit their pedagogical objectives regarding learning situations in terms of activities/tasks, roles, resources/services, objectives/prerequisites, etc.

In the next step, the concrete learning situation is in progress, and leads students/tutors (and other associated actors) to use the designer's pedagogical scenario. In this step, data regarding effective use of the learning situation is collected either by the *Learning Management System* (LMS) or by other means. Many experiments have shown that the effective running of a pedagogical situation does not necessarily follow the predictive scenario, leading to the need to observe the real behavior of the actors.

To be used in a learning scenario re-engineering process, this data must be analyzed and abstracted from the system format that produced it ;this facilitates its interpretation by the analysts and the instructional designers. So, the last step of this cycle consists of analyzing data collected during or after the effective running of the learning situation. This data is then converted (filtered, structured, combined, etc.) leading to enriched data, having a pedagogical meaning for the designer.

The results of this step then allow designers to compare the predictive scenario with the observed situation. Thus, they can be brought to modify their predictive scenario as a first step of a new iteration.

2.2 REDiM Project

Our research works take place within the REDiM project (Choquet, 07) whose main objective is to provide teachers with dedicated techniques and tools supportingt the re-engineering of their learning scenarios. Within this project, the UTL language has been proposed for the XML specification of observation datum / observations means and observation needs (Choquet et al., 06). However there is still a lack of learning-scenario-centered tools and practices to help designers in specifying their observation needs.

Our current research focus on this lack. To our minds, the preparation of the learning situation observation is an activity that relies on the one dealing with the design of learning scenarios (Zendagui et al., 08). We aim at helping designers in defining what is important to observe during the elaboration phase of the learning scenario. This process must deal with many potential difficulties for the expressiveness, relevance and usefulness of observation needs:

- they are linked to the learning scenario expressiveness and, by extension, to the expressiveness of the underlying EML;
- they depend on the designers' ability to specify what they want to observe and which information they need;
- they have to be specified with such details and formalized in such a machine-readable format that it will be able to automatically handle them to, for example, guide and help the track analysis.

We think that the MDE approach can formalize and help us in tackling these issues.

To assist designers in their observation needs specification task, we have studied the observation activity and its preparation within both classic face-to-face and distance learning situations (De Ketele, 87)(Wragg, 99)(Dessus, 07). The next section presents our conceptual proposition.

3 THE CONCEPTUAL PROPOSITION

3.1 The Process

To assist designers in their observation needs specification task, we propose a two-step approach (see figure 1).

3.1.1 Observable Identification

Within this process, the effective EML used by designers is not *a priori* known. It can be improved or enriched to better express designers needs (improvement of the EML expressiveness) during its use. We propose to identify the elements (concepts, relations, attributes) of the EML that can potentially be observed, tagging them as "observable". We define an observable as *any EML element whose change could be meaningful to observe*; i.e. any element whose instantiation (for a concept), value (for an attribute), or acquaintances (for a r elation)

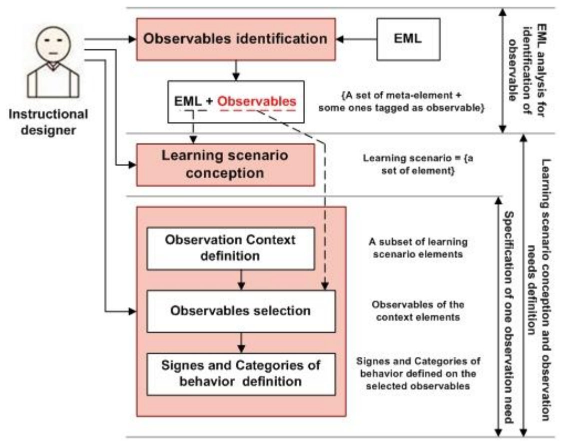
Instructional designer
Observables identification
EML
EML + Observables
{A set of meta-element + some ones tagged as observable}
EML analysis for identification of observable
Learning scenario conception
Learning scenario = {a set of element}
Observation Context definition
A subset of learning scenario elements
Observables selection
Observables of the context elements
Signes and Categories of behavior definition
Signes and Categories of behavior defined on the selected observables
Specification of one observation need
Learning scenario conception and observation needs definition

Figure 1: Teacher/designer-centred process for specifying observation needs.

can be useful to observe. For example, one might want to observe how many times an activity will be carried out (the "*Activity*" concept can thus be a possible observable), or to observe the different values of a "*duration*" property on a specific activity, or to concretely observe the order of an activity sequence (thus, identifying as observable the "*precedence*" relation defined on two activities).

This categorization of potential observables has to involve the instructional designers, or other experts of the pedagogical domain, in order to be sufficiently relevant. This identification of observables is, *a minima*, human-directed but it will be possible to eventually guide this activity thanks to a semi-automatic process based on a suggestion of observables according to a syntax-and-semantics analysis of the EML (Barré et al., 05).

3.1.2 Observation Needs Specification

The second step is the concrete specification activity of observation needs. Designers must precisely specify the observation needs they consider as relevant. Those observation needs will be intimately specified in relationship with a specific learning scenario. This activity can be done after the scenario specification or concurrently.

For each definition of an observation need, designers will follow three sub-activities: definition of the context, selection of observables, and definition of signs or categories of behavior.

The context definition aims at "contextualizing" the observation need. In addition to the observation objectives, designers have to define a subset of learning scenario elements concerned by the observation objectives. Thanks to this delimitation, only the observables concerned by elements within the context are selected and proposed to designers for the next sub-activity. The designers then have the responsibility to decide to use, or not, the proposed observables to clarify which data must be collected concerning the future learning situation. After this selection, designers can then define signs or categories of behavior (detailed in the 3.2 subsection) on these observables (Wragg E.C 99).

If the selected observables are not useful, designers can go back to the observable selection phase, as well as the context definition phase, in order to add new observables or scenario elements. They can also modify the pedagogical expressiveness of the EML, as well as modifying the observable identification at the EML level.

3.2 The BOSIC Conceptual Model

We define an observation need as composed of four parts (see figure 2): observation objectives, observation context, elements to observe or observables, and signs and behavior categories defined on one or more observables.

The observation **objectives** are useful to define the "why" of observation needs. This information allows designers to explicit what they want to do when they know the results of their observation needs from the concrete observation of learning situation runtime. This information can also be useful to facilitate the reusing of observation needs for other learning situations or other learning scenarios sharing the same objectives.

The observation **context** allows the definition of conditions under which an observation need is defined and used. Contexts allow the delimitation the learning activity to observe. It consists of selecting one or more pedagogical scenario elements. Contexts are important and must be well defined since they allow the identification of the potential observables.

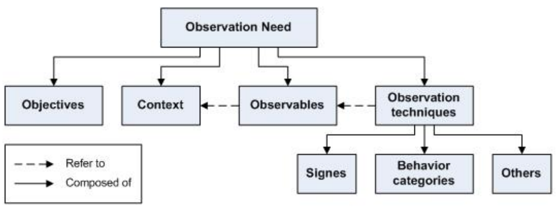

Figure 2: The BOSIC conceptual model.

The **observables** are pedagogical scenario elements for which designers want to get information after the learning situation execution. Concretely, these observables are defined at a scenario level butconform to those defined at the

EML level: they are contextualized and correspond to a kind of '*instantiation*' of the EML-level observables with respect to their context. Their specification is done by selecting observables among those than can be automatically proposed according to the context delimitation and the observables identified at the EML level. For example, "duration of activity X" is an automatically proposed observable since "activity X" is part of the observation context and since the "duration" attribute of the "activity" concept is identified as an observable at the EML level.

The **additional data** is defined on one or more selected observables at a scenario level to characterize a first representation of the observation means, in a domain-oriented language, identified by the experts. The **signs** correspond to particular events for which designers want to know apparition frequency, occurrence number or, *a minima,* to know if they occur or not, during a future observation. One or more signs can be defined on one or more scenario-level observables. For example, the "duration of at least 30 minutes" sign can be defined on the observable ("duration of activity X"). The **behavior categories** correspond to some grouping of events that will be analyzed as a block. Like the signs, one or more categories can be defined in relation to one or more observables at the scenario level. For example, the "handling events" category can be defined to indicate that we want to know all the undergone manipulations observed on the "document Y"; on the condition that the "document Y" is a selected observable, and by definition is also part of the delimited context, and that the underlying metaconcept ("Deliverable" for this case) was identified as an observable at the EML level.

The next section is about the formalization of our conceptual propositions following the MDE theories and practices.

4 THE MDE APPROACH

The formalization of the BOSIC conceptual model is necessary for two goals: firstly, to concretely support our proposition and propose a machine-readable notation for the observation needs (that will be useful, by extension, to manipulate/transform the specified needs to facilitate their reuse, etc.), and secondly, to help the building of software tools to support the observation needs specification activity by designers (user-friendly editors with helpful and guiding facilities, etc.).

4.1 MDE Projection

The *Model Driven Engineering* (MDE) is a software development methodology which focuses on creating models that describe the elements of a system (Schmidt, 06). A modeling paradigm for MDE is considered effective if its models make sense from the point of view of the user and can serve as a basis for implementing systems (productive models). The MDE principles (abstraction, modeling, meta-modeling, separation of concerns, etc.) have been applied within various educational disciplinary fields: adaptable learning materials generation, Computer Supported Cooperative Work, etc.; we have studied the application of its theories and practices for learning-scenario-centered instructional design processes in (Laforcade et al., 07)(Laforcade et al., 08).

From a MDE point-of-view, and if we do not take into account notation or concrete syntax aspects, the learning scenario is a model conformed to the metamodel specifying the terminology, or abstract syntax, of the EML used to define the scenario (see figure 3). Similarly, a set of observation needs, defined once for a given learning scenario, corresponds to a model. Because we want these observation needs defined in relation to information from the learning scenario, the models have to be linked together too. The model specifying some observation needs also has to be conformed to a specific metamodel for the definition of observation needs: the BOSIC metamodel (detailed in the next sub-section). This metamodel has to refer to the information from the EML metamodel.

Figure 3 is a four-layer OMG (OMG, 06) representation of these MDE artefacts.

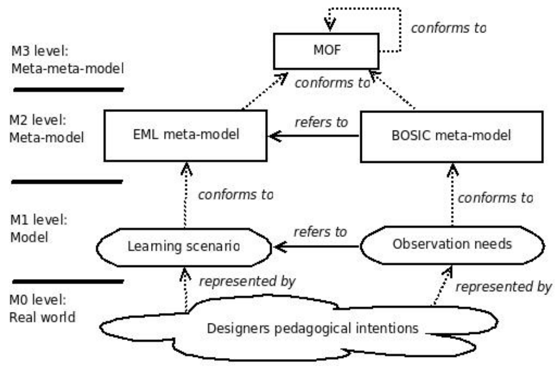

Figure 3: OMG layers view of the observation needs.

4.2 MDE Techniques Used

According to the conceptual process we outlined in the previous section, and to the MDE practices, we need concrete techniques to support both our conceptual process and BOSIC model.

The BOSIC metamodel as well as the EMLs used to define learning scenarios have to be specified using the meta-modeling technique: construction of a collection of "concepts" (things, terms, etc.) within a certain domain (Wikipedia, 08). A metamodel is a precise definition of the constructs and rules needed for creating semantic models. We illustrate an example of a metamodel for an EML in section 5, whereas the BOSIC metamodel is detailed in section 4.3.

We also need a technique to add information about the elements that designers want to tag as 'observable' to the EML metamodel . Because these potential elements can be concepts, attributes, as well as relations between concepts and because every metamodel conforms to the unique meta-meta-model MOF (OMG, 06), we need to use a MOF concept to be able to attach the 'observable' information to the class, attributes, association, etc. This meta-construction is called *annotation* (for the MOF 1.4), or *comment* (for the MOF 2.0). Section 5 shows how we use the equivalent *EAnnotation* mechanism from the *Eclipse Modeling Framework* (EMF) tooling.

Another issue from our conceptual process and model is how can the 'context' and 'observable' parts of an 'observation needs model' can refer to elements from a specific 'learning scenario'. Following the fact that the specification of an EML metamodel can use various MOF building blocks (class, attributes, relations...), it is not possible to specify a meta-relation in the BOSIC metamodel to 'anything-specified' in the EML metamodel. Also, because EMLs can differ, it is more relevant to concretely separate them. From our MDE expertise we choose to add in the BOSIC metamodel a specific concept which plays the role of a 'proxy' for the learning scenarios elements (see the next subsection).

4.3 The EMF Tooling

To concretely formalize and support the development of our proposition and dedicated editors, we chose to use a unified set of modeling frameworks, tooling, and standard implementations from the *Eclipse Modeling Projects* (Eclipse EMP, 08): EMF, GMF and ATL. In this article we only focus on the *Eclipse Modeling Framework* (EMF) because it provides the very first layer of support we need.

Indeed, the EMF is a modeling framework and code generation facility for building tools and other applications based on a structured metamodel (Steinberg et al., 2008). From a metamodel specification, EMF provides tools and runtime support to produce a set of Java classes for the metamodel, along with a set of adapter classes that enable viewing and command-based editing of the model, and a basic editor.

We illustrate the use of this very basic editor in section 5. Also, we have planned to use the *Graphical Modeling Framework* (GMF) in a second time to add a graphical layer on top of EMF, and, incidentally, to develop a graphical editor dedicated to the specification of observation needs.

Finally, the *ATLAS Transformation Language* (ATL) is the model-to-model transformation framework we will use to transform observation needs conformed to our BOSIC proposition into other machine-readable formats for the specification for observation needs (like the XML-based one proposed by UTL).

4.4 The BOSIC Metamodel

This sub-section details the BOSIC metamodel we have specified using the EMF tooling (metamodels are called ECORE models where ECORE is the MOF-like meta-meta-model in EMF). Figure 4 illustrates this metamodel in the class-diagram-oriented view proposed by the Ecore graphical internal editor of EMF.

The *ObservationNeeds* concept plays the role of the root for the specification of several observation needs in a same model, according to a same learning scenario too. In addition, the *ObservationNeed* concept is the root node for all the information in regard to one observation need. It is composed of several *Objectives* and of three other concepts representing the three layers of an observation need: *ContextLayer*, *ObservablesLayer* and *DataLayer*.

The *ContextLayer* concept is the node element under which are specified the *PSElements* (scenario elements). As previously explained, this *EClass* is a kind of 'proxy' that refers to an element from the learning scenario: it can be an instance of an *EClass* from the EML metamodel, a property (the *EAttribute* and value pair) for a specific instance of an *EClass*, as well as a link (instance of a *ERelation*) between two instances of *EClasses*. For our very first prototype, these *PSElements* will have to be specified as new inputs even if they already exist in

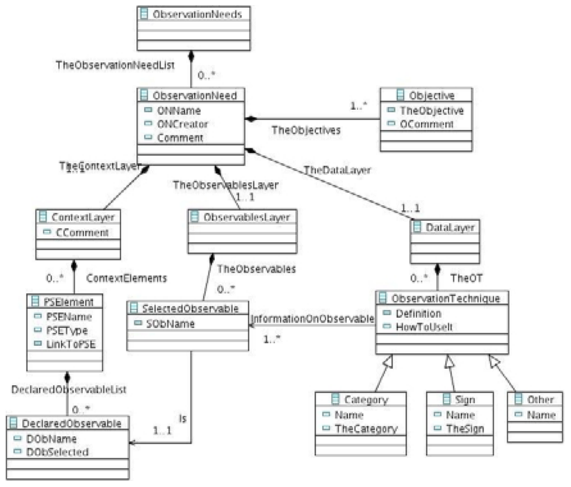

Figure 4: the BOSIC metamodel.

the learning scenario but future versions will provide designers with guiding facilities.

Similarly, the *ObservablesLayer* gathers the *SelectedObservable* which represents, according to the specification of the designers, a subset of the *DeclaredObservables*, the observables at a scenario level that can be deduced from the *PSElements* previously specified, and from the EML metamodel elements tagged as observable. We also plan to develop a specific algorithm and code routines dealing with the *DeclaredObservables* to automatically instantiate them with the deducible information.

Finally, the *ObservationTechniques* concept allows the definition of signs and categories of behavior (inheritance relation). It represents the third layer of an observation need. This information isdefined using one or more *SelectedObservable* via the *informationOnObservable* relation.

5 ILLUSTRATION

To illustrate our propositions as well as the first prototype we developed, we now present and discuss a concrete example of specification of observation needs according to a learning scenario and its underlying EML.

5.1 The EML Metamodel

Among the various case studies we have experimented on with EMF and GMF, we outline the following one for this article. Some practitioners have expressed these pedagogical expressiveness and notation needs: a UML UseCase-like diagram that shows the performing relations between roles and learning activities at a high-level of abstraction, and precedence/following relations between learning activities. We have therefore provided them with a specific graphical EML (or VIDL for *Visual Instructional Design Language*)(Botturi et al., 07) and a dedicated visual editor using EMF/GMF. A metamodel for the « *Learning Design Use Case* » (LDUC) view has been defined. It is illustrated in figure 5.

According to our conceptual process, designers have identified these potential observables from the terminology crystallized by this metamodel: some observables are EMF *EClass* (eg. *HighLevelActivity*, *Actor)*, some are EMF *EAttributes* (eg. *Duration*, *score*), others are EMF *EReferences* (eg. *nextActivities*). All these elements have been tagged as observables using the EMF *EAnnotation*

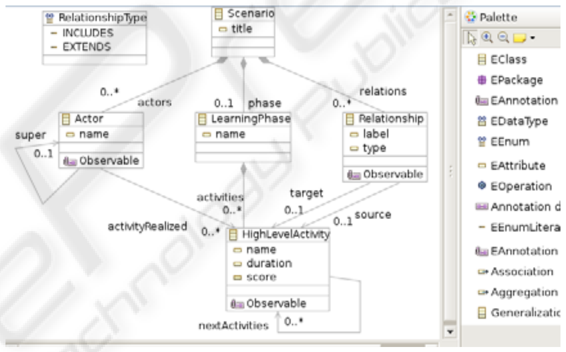

Figure 5: example of EML metamodel - the LDUC metamodel.

mechanism (only the *EAnnotations* on *EClass* are shown in figure 5).

5.2 The Learning Scenario

From the previous EML, designers have proposed the following scenario (extract on figure 6) using the graphic editor we developed thanks to the EMF/GMF frameworks. Briefly, this scenario focuses on the specific phase ("*OS introduction*") of a learning scenario they want to play. The first learning activity *"updating"* is composed of a sequence (specified thanks to the *"next"* and *"include"* relations) of sub-activities "*QCM*", "*Answers consultation*" and "*exchange*" (using a forum).

One must know that the graphic representation of the learning scenario does not reflect all the information specified with the LDUC editor: some have no graphic representations and can only be seen in the *properties* view of the editor (eg. the *duration* property for any activity).

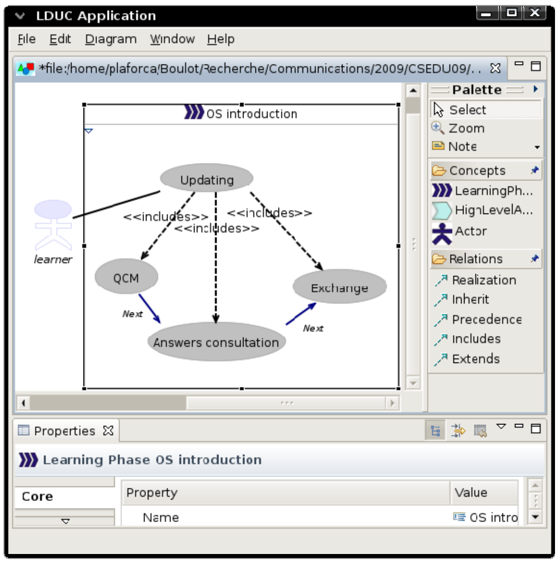

Figure 6: An example of a learning scenario.

5.3 The Observation Needs Model

Knowing both this learning scenario and the specificities of the distant platform (or LMS) that will be used to play this scenario, the instructional design team (where the instructional designers are the experts for formalizing of pedagogical intentions) want to know if the learners really take the time to consult the answers and resources available after the QCM activity, before using the forum to get explanations from their peers. For them, this question is meaningful because the LMS leaves learners independent for the order in which they perform learning activities and because the LMS does not limit the time activity to strictly follow what the designers specify using the *duration* property.

This observation need aims at gathering information about the use of the "Answers consultation" activity. The most meaningful information for this activity are that about learners who fail the "*QCM*" activity with a low score (less than 50%). It is the concrete **objective** of the teachers.

Concerning the **context** of this observation need, designers must select these elements from the learning scenario: the activity "*answers consultation*", the "*QCM*" activity and the link "*next*" between the two activities. This context filters the potential observables (*Declared-Observables*) that can be proposed to the designers: "*next*", "*answers_consultation*", "*answers_consultation.duration*", "*answers consultation.score*", "*QCM*", "*QCM.duration*" and "*QCM.Score*".

From this list of potential observables, designers only selected the following **observables:** "*answers_consultation.duration*", "*QCM.Score*" and the *next*" link.

Designers then specified three observation techniques: two **signs** and one **category of behavior**. The first sign aims to give information about the number of learners who consult the resources/answers directly at the end of the *QCM* activity. The second sign focuses on the number of learners that score less than 50% for the *QCM* activity and that spend less than one minute on the "*answers consultation*" activity. A large number here will indicate that learners did not make an adequate effort to understand their mistakes. Finally, the category of behavior designers have defined aims at collecting all the durations (for each learner) for the "*answers consultation*" activity. This expected set of results will give designers with the time spent by each learner in the "*Answers consultation*" activity.

5.4 The First Prototype

The first prototype has been developed thanks to the EMF tooling. This framework generated a first version of the editor directly from our BOSIC metamodel. This editor provides designers with a tree-view of the models where each node is an instance of an *EClass* from the metamodel and child nodes are the instances of *EClass* linked by a relation of containment between the two *EClasses* in metamodels (see figure 7). Properties and links (kind of "instances" at a model-level of the *EAttributes* and *ERelation* defined at the metamodel level) appear in the property view according to the element selected in the tree-view.

In addition to the Java-code generated by EMF we have added some specific modifications to adapt the editor: personalized labels in the tree-view, interrogating routines to gather meta-information on any elements from the learning scenario, and generation routines to automatically instantiate *DeclaredObservable* according to the potential observables information that can be deduced. The screen-capture depicted in figure 7 shows how we used this editor to formally specify the observation needs we used as illustration.

6 CONCLUSIONS

This article has presented and discussed a specific *Model-Driven Engineering* approach for the support of the definition of observation needs as models in relation to the 'learning scenario' model. We have proposed a support that is both theoretical, by providing a conceptual model and a specific process, and practical, by specifying a dedicated metamodel and by generating the first prototype of a dedicated editor, according to the use of the *Eclipse Modeling Framework*.

For now, we are working on several improvements. Some are conceptual like the use of a neutral referential for any constructivist-oriented EML to ease the definition of the context for an observation need. Other improvements are related to our MDE approach and tooling: we want to improve the editor prototype by dealing with concrete syntaxe (notation) aspects for a graphic definition of observation needs. To this aim, we have already used Eclipse's GMF (*Graphical Modeling Framework*) to provide practitioners with *Visual Instructional Design Languages* (VIDL) and dedicated editors. We plan also to use GMF to add a graphic layer on top of the EMF-generated tree-view editor for the specification of observation needs. We think that this graphical layer will give us access to facilities and services for more user-friendly editors.

REFERENCES

Barré V., Choquet C., 2005. *Language Independent Rules for Suggesting and Formalizing Observed Uses in a Pedagogical Reengineering Context*. In: ICALT'05. July 5-8. Kaohsiung (Taiwan). pp. 550-554.

Botturi, L., Stubbs, T. (eds.), 2007. *Handbook of Visual Languages in Instructional Design: Theories and Practices*. Hershey, PA: Idea Group.

Chikofsky, E. J., Cross, II J. H., 1990. Reverse engineering and design recovery: A taxonomy. *IEEE Software*, 7(1).

Choquet, C., Iksal S., 2006. Usage Tracking Language: a meta language for modelling tracks in TEL systems. In *International Conference on Software and Data Technologies (ICSOFT)*. September 11-14th. Setubal (Portugal). Filipe J., Shishkov B., Helfert M. (ed.), ISBN 978-972-8865-69-6.

Choquet, C. 2007. Engineering and re-engineering of TEL systems, the REDiM approach. Habilitation à Diriger des Recherches. Le Maine University. France. *In french.*

De Ketele, J.M., 1987. *Méthodologie de l'observation.* Bruxelles, De Boeck.

Dessus, P., 2007. Systèmes d'observation de classes et prise en compte de la complexité des événements scolaires. In *Carrefours de l'Éducation*. Vol(23). pp. 103-118.

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

Favre, J.M., 2004. Towards a Basic Theory to Model Driven Engineering. *In WISME'04, the 3rd Worksho p in Software Model Engineering*. Lisbon, Portugal .

Kent S., 2002. Model Driven Engineering. In *IFM 2002.* LNCS 2335. pp. 286 – 298. Springer – Verlag.

Kinshuk, Sampson D.G., Patel A., Oppermann R., 2006. Special issue: Current Research in Learning Design. In *Journal of Educational Technology & Society*. (9)- 1.

Koper, R., & Tattersall, C., 2005. *Learning Design – a handbook on Modelling and Delivering Networked Education and Training*. Springer-Verlag. Berlin Heidelberg.

Laforcade, P., 2007. Visualization of Learning Scenarios with UML4LD. In: *Journal of Learning Design*. Vol. 2 (2). pp. 31-42.

Laforcade P., Zendagui B., Barré V., 2008. *Supporting the Specification of Educational Modeling Languages and Learning Scenarios with a Domain-Specific-Modeling Approach*. In: ICALT'08. July 01-05. Santander (Spain). pp. 819-821. IEEE Computer Society.

Metamodel, 2008. http://www.metamodel.com/. Retrieved from December 2008.

OMG, 2006. Meta Object Facility specification (MOF 2.0). Formal/2006-01-01. http://www.omg.org/mof/

Schmidt, D.C., 2006. Model-Driven Engineering. *IEEE Computer*. 39 (2).

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

Wikipedia, 2008. Meta-modeling. http://en.wikipedia.org/wiki/Metamodeling. Retrieved from December 2008.

Wragg E.C., 1999. *Introduction to classroom observation.* Second edition. Routeledge, May 14.

Zendagui B., Barré V., Laforcade P., 2008. *Support to the specification of observation needs*. In: ICALT'08. July 01-05. Santander (Spain). pp. 793-797. IEEE Computer Society. ISBN 978-0-7695-3167-0.