A Meta-Model to Acquire Relevant Knowledge for Interactive Learning Environments Personalization

Marie Lefevre, Alain Mille, Stéphanie Jean-Daubias and Nathalie Guin
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205
Villeurbanne, France
{Marie.Lefevre, Alain.Mille, Stephanie.Jean-Daubias, Nathalie.Guin}@liris.univ-lyon1.fr

***Abstract*— Outsourced personalization of Interactive Learning Environments (ILE) requires knowledge on these ILEs: their teaching description (type of activities, parameters for exercises generation, etc.) and their technical description (files location, content of configuration files, etc.). To make possible to support this personalization, we propose a meta-model to acquire, in interaction with an expert, relevant knowledge necessary to personalize an ILE. We combine this meta-model with two processes: the first one uses the meta-model in order to create a specific model to an ILE, and the second uses this model to personalize the ILE.**

Keywords- Meta-model of knowledge; models of knowledg; Interactive Learning Environments (ILE); personalization; externalized configuration

I. INTRODUCTION

Personalization of teaching and learning activities is widely considered as a topical issue in the field of research in educational technologies. Personalization of activities, i.e., their adaptation to the individuality of each learner, is an efficient mean of improving teaching, in particular in the classroom context. However, due to the diversity of learners, the variety of situations and subjects under study, personalization is often a complex and time-consuming task. For that reason, and because of a lack of adequate tools, teachers do not efficiently personalize pedagogical activities. Therefore, developing software to assist teachers in this personalization task can turn out to be very helpful.

Personalization is a multi-faceted research question. It can concern paper and pencil activities, Interactive Learning Environments (ILEs), interactions between teachers (or interactive environments) and students, as well as specific activities depending on the skills of students in one subject or another, etc. However, in any case, the personalization process has to rely on specific information about the context, the activities, and, of course, about learners. Usually, information about learners is gathered into "learners' profiles". A learner's profile contains elements characterizing knowledge, skills, perceptions, and/or behaviour of a given learner. According to [1], information contained in profiles is either collected or deduced from pedagogical activities which can be computerized or not.

In this paper, we propose a process and the associated meta-model to support personalization of ILEs. The meta-model relies on ILEs invariants. Hence is allows us to personalize various ILEs through a unified approach despite the wide variety of pedagogical situations these ILE cope with. We restrict our study to ILE assisting a situation of individual learning. First, we describe our context of research and we introduce the issue studied here. In section 3, we detail how the study of existing educational software has enabled us to detect common invariants that are of importance when personalizing ILEs. Next, we investigate if the existing standards, defined to describe an ILE, can describe the invariants identified. Then we describe our meta-model and the processes exploiting this meta-model in order to assist the personalization of an ILE. We restrict the formal description of the meta-model to the part concerning the pedagogical description of an ILE. The second part concerning the technical description of the ILE is not formalized in this paper. The conclusion is an opportunity to discuss the validation of the meta-model, its coverage and its limitations.

In this paper, the term ILE is used in a restricted sense. It is used to mention educational software only. These software can be freewares, sharewares, commercial softwares or research products. They implement a situation in which the learner is supposed to learn a new knowledge.

II. SCOPE OF THE STUDY: CONTEXT AND RESEARCH ISSUE

Personalization of human learning mediated by a computer environment (an ILE) requires a good knowledge of the possible settings and information on the mediated learning situation: information gathering knowledge on students, working habits, teaching goals of teachers, etc. This personalization is even more complicated to implement when one wants to cope with the great variety of educational software and of teaching situations.

The heterogeneity of ILE concern the proposed environment as well as the content of the environment [2, 3]. Indeed, educational software may take the form of an intelligent tutor, a microworld, a simulator, a hypertext document, etc. To each of these forms is associated a mode of use (free or guided curriculum), content (sequence of predefined activities, panel of objects can be manipulated, etc.), but also a variety of educational goals (acquisition of a method, acquisition of a set of knowledge, acquisition of practice, etc.).

The heterogeneity of educational situations depends on the various actors participating to a learning activity. Thus

ILEs can support individual learning situation, collective learning situation or collaborative learning situation. The teacher may have a role of the designer of an activity or tutor of an activity.

To cope with this double heterogeneity (ILE and educational situations), the teacher must be able to configure ILE in order to satisfy his own pedagogical goals and to build different types of profiles enabling him to manage personalized educational situations for learners.

In this research, we restrict our study to the case of the teacher which prepares a learning situation by personalizing an ILE and by preparing the appropriate profiles to manage this situation; and to the case of an individual learning situation, even if it is likely that we could have a similar approach to collective learning situation.

This personalization uses profiles of learners [1]: profiles designed by teachers regardless of a particular ILE and profiles designed for concerned ILEs, regardless of a particular teacher. Personalization is done in accordance with the educational goals chosen by the teacher to the prepared learning situation.

We rely on the EPROFILEA environment (PERLEA project [1, 4]) as support to the process of personalization. EPROFILEA consists of a set of modules: module for description of profiles structures, modules for integration of date profile, module for visualization, etc. Our meta-model aims at supporting efficiently the personalization process of any ILE. Indeed, it guides the design of models dedicated to the configuration of ILEs. In the same way, it supports the design of models of profiles associated to specific pedagogical goals. As a consequence, this meta-model helps to prepare an individual learning situation with a given ILE. This meta-model is implemented in a new module of EPROFILEA: the Adapte module [5].

Defining such a meta-model requires to identify the invariants which exist in the models of individual learning activities and in the models of ILE's settings. For this purpose, we have analysed 30 ILES and we have studied how learning activities were described, by focusing in particular on standardized descriptions.

III. A SURVEY OF EXISTING SYSTEMS AND STANDARDS

In this section, we address the issue of the description of an ILE for personalization and then how the metadata allow describing an ILE.

A. *Describing an ILE for Personalization*

In order to allow a personalization of an ILE, it is necessary to have information (knowledge, skills, etc.) on the learner for which the personalized ILE will be designed, but also on the situation in which this ILE will be used (place, time, pedagogical goal of the teacher). The information on the learner may be contained in a profile of learner, but information on the educational situation should be described by the teacher himself. From this information, it is possible to change the ILE to adapt it on the one hand to the learner and the other hand to educational goals of the teacher.

In order to determine the main types of personalization in an ILE and the way it can be implemented, we have conducted a systematic study of 30 ILEs. These ILEs are of different types: intelligent tutors as Andes [6] or AMBRE-add [7], microworlds as Aplusix [8], simulators as Teleos [9] or web applications as ActiveMath [10]. Some of them come from research as SQL-Tutor [11], or from practices of teachers as LiliMath [12], or from free development as Mathenpoche [13] or from commercial softwares as Exomatiks [14]. Moreover, they are intended to various public from nursery school to high school and for professional training. They cover various learning areas (mathematics, orthopaedic surgery, German, etc.).

We have determined that in these ILE, personalization may include: the proposed *activities*, the *sequence* of activities (number of activities, order), available *functionalities*, the *feedback* offered to learners and the customization of ILE's *interfaces*. These targets of personalization are confirmed from a theoretical point of view in the literature studying the techniques used to adapt systems [15].

The Table 1 provides an overview of personalization possibilities, choosing typical examples among studied ILEs. The columns in this table reflect the five targets of personalization that we have identified.

TABLE I. OVERVIEW OF POSSIBLE PERSONALIZATIONS IN VARIOUS ILES

ILE	Acti-vities	Sequence of activities	Functio-nalities	Feed-back	Inter-face
AMBRE-add [7]	X	X	X		X
Aplusix [8]	X		X		
Téléos [9]	X				
Mathenpoche [13]	X	X			
LiliMath [12]	X			X	X
Exomatiks [14]	X				X
ActiveMath [10]	X				
Andes [6]	X				

After having identified five targets of personalization, we focused on how implementing this personalization. To do this, we studied how ILEs could be "configured" when used by the learner.

An ILE is usually set up by a configuration file or a configuration interface. Activities can then be generated automatically or selected from a predefined list, possibly in compliance with the constraints expressed by the settings.

Fig 1. summarizes these different cases giving, each time, examples of ILEs operating in this manner.

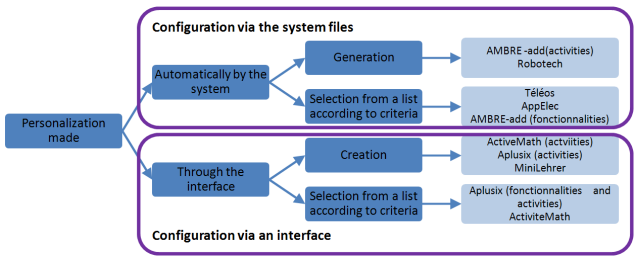

Figure 1. Categorization of ways to customize an ILE.

Finally, to know how to act from outside the ILE on its parameters, we studied the different configuration files as well as the interfaces that have an impact on system setting. We give below some examples to determine how to discover the invariants in the personalization process. Indeed, the normalization process needs to match the setting information as accessible to the user and the information to adapt the ILE to implement the desired personalization.

First example, the AMBRE-add ITS [12] has an exercises generator allowing to act on the creation of an **activity**. The **configuration file** (see an extract in Fig. 2) of the generator contains, among others, the value of each parameter necessary for the creation of exercises. This configuration file is made up of specific codes for adapting the AMBRE-add ITS.

Taking the first code of the configuration file "['c_comparaison_dif','de plus']", it corresponds actually, for the first field, to the value of a parameter "classe de problème", **scale_list** type, whose possible values (explicitly provided to the user in the graphical environment) are "réunion, changement, *comparaison*", concatenated to the value of the associated parameter "place de l'inconnue", **scale_list** type, whose possible values are "résultat, opérande, opérateur, *différence*, un des deux – min, un des deux– max" and for the second field, to the value of a parameter "variation", also **scale_list** type, and whose possible values are "*augmentation*, diminution". Shown here that under the apparent specificity of the configuration file, there is a setting, relatively quite simple, to choose the values in predefined lists. We have put in *italic* the values of the lists corresponding to the code used in the configuration file example.

In the same way, taking the fifth parameter of configuration file "[0,0,1,0]", it corresponds to the value of parameters "utiliser une seule sorte d'objets", "utiliser plusieurs sortes d'objets", "affecter les objets" and "affecter les personages", all **scale_boolean** type. The reader will easily understand that true is represented by 1 and false by 0.

To conclude this example with another type of parameter, we choose the sixth parameter of the configuration file [5,15], it corresponds to the value of the parameter "intervalles pour les valeurs", **scale_interval** type, and possible values in [LowerBound = 0..∞, Upperbound = 0..∞].

Second example, in the Aplusix microworld [8], the learner can obtain a **sequence of activities** through a test card which is represented by a table with double entries. This test card is available from the **interface** of the ILE. The rows of table corresponding to the types of calculation can be: "numeric calculation (A), development and resolution (B), factoring (C), solving equation (D), resolution inequality (E), solving system (F)" and the columns correspond to the level of difficulty between 1 and 9. Some cases of the table are inactive, such as A6, A7, A8 and A9. The learner then chooses one case of the table and the ILE generates exercises corresponding to the type of calculation and at the level requested. The card test can be described with the parameter "type of calculation", **scale_list** type, whose possible values are "A, B, C, D, E, F" and with the parameter "level", **scale_numerical** type, of values between 1 and 9. These two parameters are constrained by a set of rules like "If {Value (type of calculation) = A} Then {ValueDomain(level) = 1..5}".

Third example, in the Andes Physics Tutor [6], the learner may request to do an **activity** by selecting, from menus of the **interface**, the subject among those of one discipline then the number of the activity.These menus can be described by three parameters. The first parameter is the "discipline", **scale_list** type, whose values are "mechanics, electricity and magnetism". The second parameter is the "subject**", scale_list** type, whose an overview of the values is "vectors, translational kinematics, free body diagrams, static, etc.". The third parameter is the "number of exercise", **scale_list** type, whose an overview of values is "vect1a, ..., relvec3a, ..., mirror1, mirror2, ...". These three parameters are constrained by a set of rules like "If {Value (subject) = vectors} Then {ValueDomain(number of exercise) = "vect1a, ..., relvec3a"}".

Last example, in the Exomatiks software [14], **functionalities** and **interface** of the software (language, graphic options, tools, users, etc.) are defined from a **configuration file** (see Fig. 3). Taking the code "langue=0", it corresponds to the value of parameter "language", **scale_list** type, whose possible values are "French, English, etc." and the value of the file indicates the index of the enumerated list.

```
appel_generation([
['c_comparaison_dif'
,
'de plus'],[jeu],
['bille','bille
jaune',
'bille jaune'],
[],[0,0,1,0],[5,15],
[],
[intervalle,1],[0,2,
0,1,0,0]],apercu).
```

Figure 2. Excerpt from configuration file for the generation of exercises in AMBRE-add ITS.

```
[generales]
langue=0
…
[Internet]
utiliser_ie=1
…
[outils]
executer=-1
auto_mutiplier=1
auto_reduction=0
activer_coloration=1
…
```

Figure 3. Excerpt from configuration file for interface and functionalities of the Exomatiks software.

Through these examples, we can see that it is possible to describe the parameters involved in setting of an ILE through a common formalism : a set of parameters constrained by a set of rules. These parameters are of predefined types (scale_list, scale_boolean, etc.) and their values are dependent on the ILE to configure.

At the conclusion of the study that we made on 30 ILEs, we can observe that the personalization of an ILE may cover five targets: the selection or the creation of activities, the organisation of these activities to form working sequences, the functionalities offered to the learner, the feedback offered by the ILE and the interface of the ILE. Parameters acting on these targets may be contained in files or be changed via a configuration interface in the ILE.

Each of these parameters can be described using a common formalism. Thus these parameters can be presented

in a uniform way for a teacher and thus allow easier personalization of heterogeneous ILE. The parameters influencing the ILE must be accompanied by pedagogical competences associated with them. Indeed, for a teacher can change the setting of an ILE, he needs to know what competence is associated with each setting and what influence will get change from one parameter to ILE.

In conclusion, we can say that in order to personalize an ILE in outsourced way, we need to know the parameters impacting on the setting of the software, the competences associated with changing these settings and a technical description on how, from the values of each parameter, to modify or to create the configuration files. We therefore examined the existing standards to see if they allow us to describe an ILE according to this granularity.

B. *Describing an ILE Using Metadata*

The question of ILE description has historically quickly asked to enable the sharing and reuse of resources within them. Several proposals for standardization have been made with different objectives. The Dublin Core includes a set of general metadata to describe all types of resources [16] [17] while others standards are more specific to areas or to professions. With regard to educational resources, we can cite LOM, which allows us to describe the educational tools [18], notably e-learning softwares, or SCORM which allows us to create interoperable and reusable structured teaching objects [19]. These standards are not suitable for the description of components constituting ILEs because they do not have enough metadata to describe the software appearance of these components, such as technical needs (hardware and software) to run it, their characteristics (services, properties, methods) and let-alone rules and dependencies required to assemble it by computer specialist developing ILEs.

In response to this observation, the software component description pattern LSCM proposes a set of metadata devoted to software components [20]. LSCM has two sections: a common section to all classes of components describing software engineering, and a specific section to the category of educational software components describing the pedagogical and didactic appearance of software component. Hence, LSCM allows us to describe components for reuse, but not for personalization.

Thus, the various standards proposed in current work do not offer solutions to describe systems for personalization. They do not permit a description of their educational content allowing personalization. They not do either a description of technical knowledge allowing stepping in the system to implement this outsourced personalization.

IV. AKEPI, A META-MODEL TO ACQUIRE KNOWLEDGE NECESSARY FOR OUTSOURCED PERSONALIZATION OF ILES

The outsourced personalization of an ILE requires obtaining some knowledge on the ILE. First, the properties characterizing the activities and the environment, allow for each ILE, to know the type of proposed activities and how it is possible to choose or generate activities by this ILE. These properties also permit to know if the ILE proposes customizable sequences of activities, customizable functionalities… To personalize the environment and activities offered to learners, it is necessary to have a description of all the parameters relating to activities, sequences of activities, functionalities and / or interface of the ILE. Then, teaching competences associated with these properties allow teachers to be assisted when making educational choices when setting the ILE. Finally, to be able to act on an ILE, it is necessary to be able to modify files allowing personalization. It is therefore necessary to have technical information on the ILE, as the presence of an exercise generator, the place and content of configuration files, etc.

A. *Why a Meta-Model?*

The AKEPI meta-model (Acquisition of Knowledge Enabling Personalization of ILEs) is designed to facilitate the expert's complex task of identifying knowledge needed to the outsource personalization of an ILE. This meta-model defines the type of information needed to personalize an ILE and is made up of two parts: the pedagogical description of an ILE and the technical description (see Fig. 4). We combine two processes with this meta-model: the first one, AKASI, instantiates the AKEPI meta-model with specific knowledge to an ILE α to get the OKEP/α model (Operational Knowledge Enabling Personalization of the ILE α), and the second one, OPIKSI, uses the OKEP/α model to personalize the ILE α.

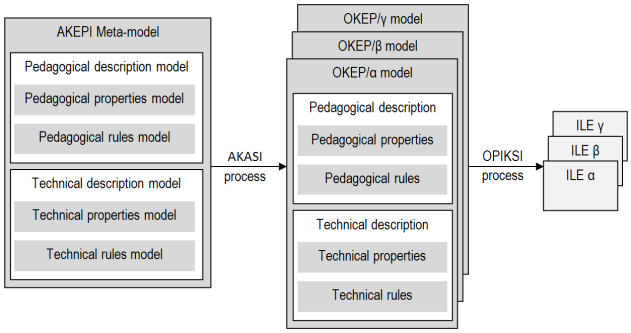

Figure 4. Overview of the AKEPI meta-model and associated processes.

B. *Knowledge of the Meta-Model*

The AKEPI meta-model defines the type of information an expert must provide to create, for an ILE α, the OKEP/α model allowing its outsourced personalization. This information should cover the pedagogical description and the technical description of ILEs (see Fig. 4). The pedagogical description aggregates the customizable properties of a system, with the associated competences, and the rules to manage these properties. The technical description aggregates all the necessary information to act concretely on the system: localization of the system, folder of the exercises generator or exercises database, place and content of configuration files and rules for fill in these files. In its implementation, the AKEPI meta-model is defined using the XML Schema formalism [21].

In the remainder of this section, we describe the ILE pedagogical description model of the AKEPI meta-model. This model is composed of the model of pedagogical properties and the model of pedagogical rules.

1) The Model of Pedagogical Properties

Fig. 5 shows the part of the AKEPI meta-model defining types of pedagogical properties. We can see that these properties are organized into five targets of personalization that we identified in our study (see Section III-A). Formally, the *PedagogicalProperties* model will be defined for an ILE, identified by *NameOfILE*. This model must contain a description of the *PedagogicalContent* for this ILE. Then, depending on the ILEs, we can add a description of the *PedagogicalOrganization*, the Functionalities, the Feedback and the Customization of the interface.

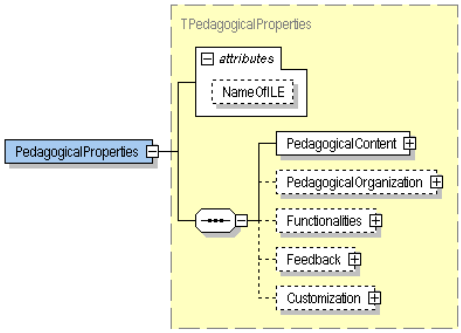

Figure 5. Meta-model AKEPI: pedagogical properties.

The *PedagogicalContent* allow describing how different types of activities will be selected or generated. It is composed of description of one or more *TypeOfActivity*. A *TypeOfActivity* is defined, as we can see in Fig. 6, by Name; the list of *Parameters* for choosing an activity of this type in a database, or generating it with the generator in the ILE; and possibly the parameters of *AssociatedFunctionalities*, *AssociatedFeedback* and *AssociatedCustomization* that can be associated with this type of activity. Every *TypeOfActivity* have a unique ID.

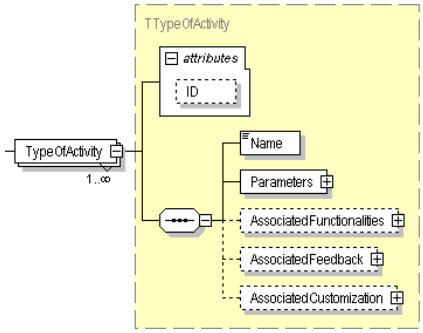

Figure 6. Meta-model AKEPI: pedagogical content.

The list of parameters contains one or more parameters. Each Parameter (see Fig. 7) is defined by a *Name*; the *AccociatedCompetence* on which the modification of parameter can influence; a *Scale* respecting a one of six types of proposed scales covering various possibilities encountered in different systems studied; and possibly the *Category* or *Subcategory* to which it belongs. This precision allows organizing the parameters so as to consolidate the settings that links them. Moreover, as the *TypeOfActivity*, each *Parameter* has a unique ID.

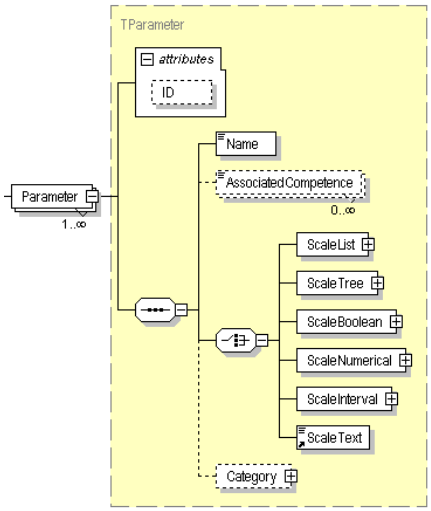

Figure 7. Meta-model AKEPI: parameter.

The scale type *ScaleList* allows defining enumerated lists of values (see Fig. 8). It includes two boolean, *Variable* and *MultipleSelection*, which allow, for the first, to indicate whether the enumerated list defined by the expert for a given parameter can be increased by the values defined by the teacher, and for the second, to indicate if the teacher can select multiple values for this parameter. Then it includes one or more values. A *Value* includes a *Name* and possibly *AssociatedCompetences*.

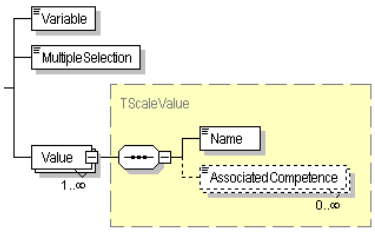

Figure 8. Meta-model AKEPI: scale List.

The scale type *ScaleTree* used to define values organized in a tree (see Fig. 9). It includes, as for the type *ScaleList*, the booleans *Variable* and *MultipleSelection*. Then it contains a set of Node. Each Node consists of a unique ID, a Name, and possibly the *AssociatedCompetences*. Each Node can contain one or more Node to represent the tree of values.

The scale type *ScaleBoolean* may be specified indicating the competences which may be associated with each of the values, *AssociatedCompetenceIfTrue* and *AssociatedCompetenceIfFalse*.

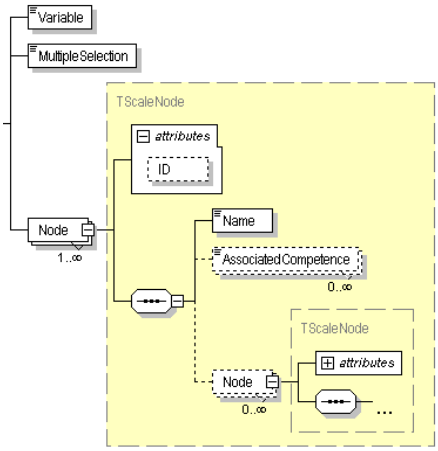

Figure 9. Meta-model AKEPI: scale Tree.

The scale type *ScaleNumerical* contains five optional fields (see Fig. 10). The first two, *LowerBound* and *Upperbound*, reduces the range of values initially defined from - ∞ to + ∞. The Step can specify the granularity of the parameter value (integer, defined to about 0.1, etc.). The last two fields allows to specify the associated competences, specifying what competences are associated with the lower bound, *AssociatedCompetenceLowerBound*, and those associated with the upper bound, *AssociatedCompetenceUpperBound*.

The scale type *ScaleInterval* contains six optional fields (see Fig. 11). The first two, *MinLowerBound* and *MaxLowerBound*, allow reducing the value of the lower bound of the interval. The two following, *MinUpperBound* and *MaxUpperBound*, allow reducing the value of the upper bound of the interval. The last two fields allow indicating the associated competences with each of the bounds of the interval, *AssociatedCompetenceLowerBound* and *AssociatedCompetenceUpperBound*.

Finally, the scale type *ScaleText* can be increased with *AssociatedCompetences*.

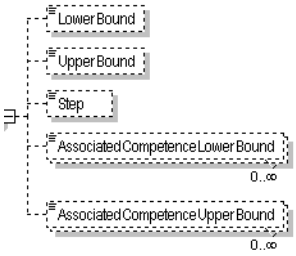

Figure 10. Meta-model AKEPI: scale Numerical.

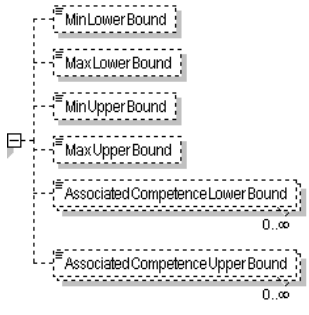

Figure 11. Meta-model AKEPI: scale Interval.

The *PedagogicalOrganization* indicates how the sequences of activities are constituted (see Fig. 12). Thus, a Sequence consists of one or more *TypeOfContent*. Each of these *TypeOfContent* refers to a previously defined *TypeOfActvity* and two integers, *MinNumberOfActvities* and *MaxNumberOfActivities*, which indicates the number of such activity may be provided consecutively to the learner. The boolean *OrdoredSequence* indicates if the *TypeOfContent* must be provided in order or they may be randomly provided to the learner. Finally, parameters describing *AssociatedFunctionalities*, *AssociatedFeedback* or *AssociatedCustomization* that can be addded with the *PedagogicalOrganization*.

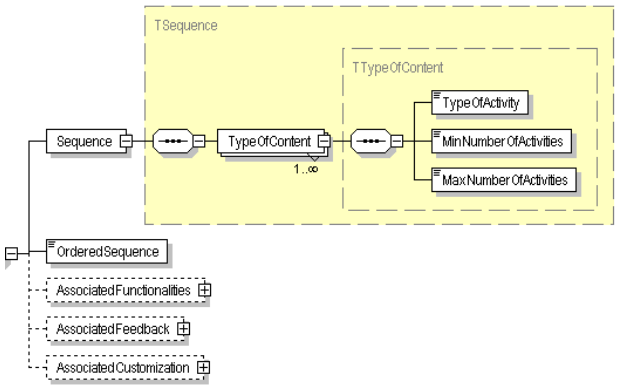

Figure 12. Meta-model AKEPI: pedagogical organization.

The elements *Functionalities*, *Feedback* and *Customization* contain a description of the parameters for the personalization (see Fig. 13). Each of these parameters satisfies the definition of a *Parameter* that we introduced previously.

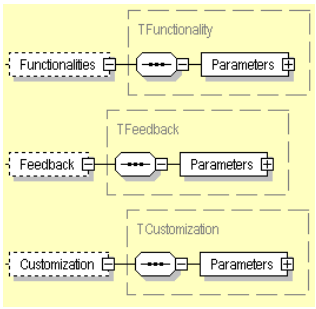

Figure 13. Meta-model AKEPI: functionalities, feedback and customization.

2) The Model of Pedagogical Rules

The pedagogical description contains thus the customizable properties of an ILE. The AKEPI meta-model combine these properties with meta-rules allowing the expert to formalize the links and the constraints that could link the properties he has identified. Thus, the expert can define constraints on the properties using rules whose conditions and conclusions respect the format shown in Table 2.

On the same principle, the technical description of ILEs in the AKEPI meta-model contains a set of properties associated with a set of rules.

TABLE II. FORMAT OF CONDITIONS (AT LEFT) AND CONCLUSIONS (AT RIGHT) OF PEDAGOGICAL RULES OF AKEPI META-MODEL

IF Value(parameter i_1) = X_1	**THEN** Value(parameter j_1) = Y_1
IF Value(parameter i_1) ϵ {X_1 .. X_n}	**THEN** the parameter j_1 will be inaccessible
IF Value(parameter i_1) not defined	**THEN** ValueDomain (parameter j_1) = {Y_a .. Y_b} with a≥m and b≤n where m and n are the initial bound
IF C_1 **and** C_2 with C_i is a constraint on a value of a parameter	**THEN** C_1 **and** C_2 with C_i is a constraint on a value or domain of value of a parameter

C. *Processes Associated With AKEPI Meta-Model*

We have presented the principles of AKEPI meta-model, as well as an overview of knowledge it can acquire. We will now indicate how this meta-model can be used by presenting two processes, AKASI and OPIKSI, that we associate with it (see Fig. 4).

The AKASI process, allowing defining ILE's specific knowledge in order to create an OKEP/x model, is decomposed into three steps (see Fig. 14). In the first step, the expert specifies, using an interface created dynamically from the AKEPI meta-model, the pedagogical properties of an ILE. In the second step, the expert defines the rules for the management of pedagogical properties respecting the format of pedagogical rules defined in the AKEPI meta-model. In the final step, the expert specifies the technical properties and technical rules for the ILE. By implementing these rules, the XSL file (eXtensible StyleSheet Language) converts the constraints made by teachers on an ILE to produce a configuration file.

The OPIKSI process, allowing the use of the OKEP/x model to personalize an ILE, is decomposed into four steps (see Fig. 15). In the first step, an interface is automatically generated from the pedagogical properties of OKEP/x model. In the second step, the teacher uses this interface to define constraints on the generation or on the selection of activities already generated, and defines constraints on the choice of parameters acting on the functionality and the interface of the ILE x. The pedagogical rules, in the form of Prolog predicates, associated with an inference engine implemented in Prolog, manage dynamically the interface offered to the teacher. In the third step, from the constraints specified by the teacher and the pedagogical properties of OKEP/x model, the system generates sequences of activities suitable to learners and defines parameters to customize the environment. In the last step, the system uses the technical description of the OKEP/x model to create the configuration files specific to the ILE x.

V. CONCLUDING REMARKS

For facilitating the personalization of an ILE, it is often useful to rely on a model describing its pedagogical and technical properties. However, designing such models is a complex task and requires specific knowledge on the properties of each ILE.

In this paper, we have presented the AKEPI meta-model and associated processes, AKASI and OPIKSI, aiming at supporting the acquisition of this specific knowledge and their use to personalize ILEs. Our approach aims at assisting the expert while defining pedagogical and technical descriptions of a specific ILE.

By following the AKASI process, the expert instantiates the AKEPI meta-model with knowledge related to the ILE x thus producing the OKEP/x model of the ILE. This model can then be used to support personalization.

Building a model for a given ILE takes some time and requires a certain involvement from the expert. However our approach ensures that this tedious task is done only once. Then, if the ILE evolves, its model evolves as well thus avoiding to rebuild a model from scratch.

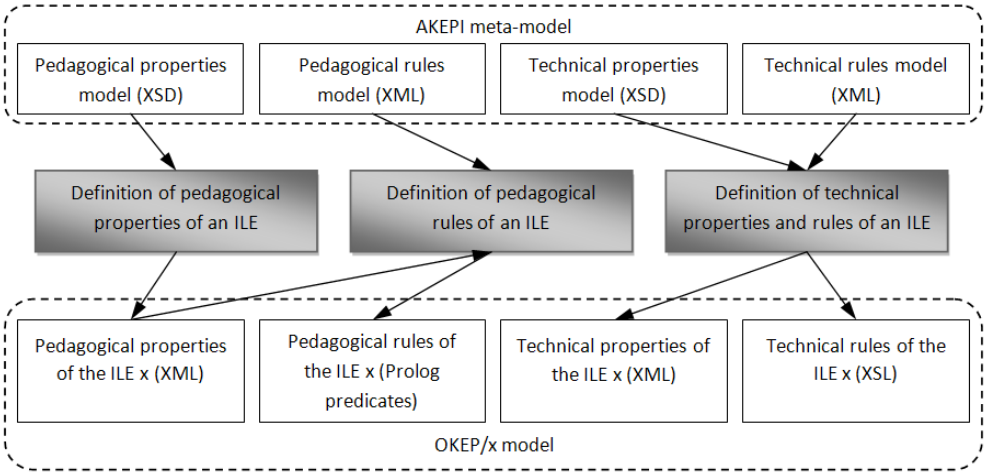

Figure 14. AKASI process

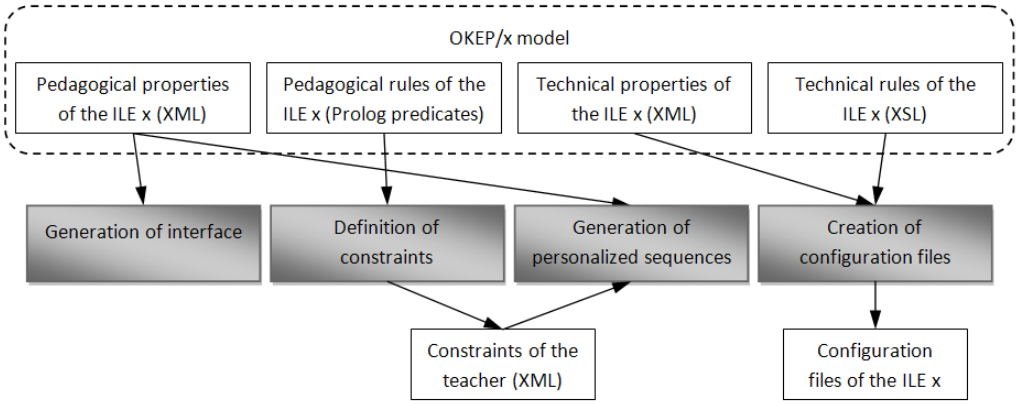

Figure 15. OPIKSI process

Thanks to this meta-model and its associated processes, we are now able to efficiently support personalization of a large variety of pedagogical situations regardless to the tools they involve. Indeed, we have implemented this approach (meta-model and processes) in the Adapte module belonging to the EPROFILEA environment. This module relies on ILE models to help teacher personalizing pedagogical session depending on information available in profiles of learners.

The meta-model has been built with regard to the results of a detailed analysis in which we have studied more than thirty ILEs (including ITS, microworlds and simulators) for identifying invariants relevant for personalization purpose. In order to validate this meta-model, we have applied our approach to define the models of the thirty initial ILE plus five new ones. We were able to build detailed models for each system. Ongoing experimentations consist in building models for new ILEs to identify the limitations of our approach and to make sure that no critical property was omitted in the meta-model.

Our future work is to carry on with experimentation in order to validate the approach and to identify better its limitations. For example, we have to study the impact of this process on feedback personalization. Indeed, we have not encountered yet any system allowing feedback customization. We must then confront the AKEPI meta-model to such systems to check if our proposals remain valid. Another future work is to experiment with the Adapte module, and consequently with the AKEPI meta-model, in the context of a classroom. Indeed, preliminary experimentations have shown the benefits of this approach from a theoretical point of view, however we need to study it in real conditions and to make sure that it is helpful enough to be accepted and used by teachers.

REFERENCES

[1] S. Jean-Daubias and C. Eyssautier-Bavay, “An environment helping teachers to track students' competencies”, Proc. Workshop LEMORE, AIED'2005, Pays-Bas, 2005.

[2] E. Bruillard, Les machines à enseigner, Hermès, France, 1997.

[3] E. Wenger, Artificial Intelligence and Tutoring Systems, Los Altos, CA, Morgan Kaufmann, ISBN : 0934613265, 1987.

[4] Projet PERLEA, http://liris.cnrs.fr/stephanie.jean-daubias/projets/p-perlea.html (last visited in August 2009).

[5] M. Lefevre, N. Guin and S. Jean-Daubias, “Adapte, a tool for the teacher to personalize activities”, Proc. ITS'2008, Montréal, 2008, pp. 699-701.

[6] K. VanLehn, C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein and M. Wintersgill, “The Andes Physics Tutoring System: Lessons Learned”, IJAIED. vol. 15, 2005, pp. 147-204.

[7] S. Nogry, S. Jean-Daubias and N. Duclosson, “ITS Evaluation in Classroom: The Case of AMBRE-AWP”, Proc. ITS'2004, Laceio, 2004, pp. 511-520.

[8] J.-F. Nicaud, D. Bouhineau, H. Chaachoua, T. Huguet and A. Bronner, “A computer program for the learning of algebra: description and first experiment”, Proc. 11th International PEG Conference, St. Petersburg, 2003.

[9] C. Vu Minh, V. Luengo and L. Vadcard, “A Bayesian Network Based Approach for Student Diagnosis in Complex and Ill-structured Domains”, Proc. TICE'2006, Toulouse, 2006.

[10] E. Melis, E. Andrés, J. Büdenbender, A. Frischauf, G. Goguadze, P. Libbrecht, M. Pollet and C. Ullrich, “ActiveMath: A Generic and Adaptive Web-Based Learning Environment”, IJAIED, vol. 12, 200, pp. 385-407.

[11] A. Mitrovic, “A Knowledge-Based Teaching System for SQL”, AACE, 1998.

[12] LiliMath, http://lilimath.free.fr/ (last visited in August 2009).

[13] Mathenpoche, http://mathenpoche.sesamath.net/ (last visited in August 2009).

[14] Exomatiks, http://exomatiks.free.fr/mb/ (last visited in August 2009).

[15] P. Brusilovsky, “Adaptive and Intelligent Technologies for Web-based Education”, Kunstliche Intelligenz, vol. 13, 1999, pp. 19-25.

[16] Dublin Core Metadata Initiative, http://dublincore.org/documents/dcmi-terms/ (last visited in August 2009).

[17] K. Keenoy, “SeLeNe-Preliminary Report: Learning Objects, Meta-Data and Standards”, 2003.

[18] LOM v1.0.. IEEE P1484.12.1-2002, http://ltsc.ieee.org/wg12/ (last visited in August 2009).

[19] SCORM Metadata set, http://www.adlnet.gov/scorm/index.aspx (last visited in November 2008).

[20] I. Rebaï, B. de la Passardière and J.-M. Labat, “To Store and Retrieve Software Components for Interactive Learning Environments: the ECR Repository”, International Journal of Advanced Media and Communication, vol. 2(1), 2008, pp. 73-95.

[21] XSD, http://www.w3.org/XML/Schema (last visited in August 2009).