Learning Design based on Graphical Knowledge-Modeling

By Gilbert Paquette, Michel Léonard, Karin Lundgren-Cayrol,

Stefan Mihaila and Denis Gareau

LICEF-CIRTA Research Center and CICE Research Chair

Télé-université, gpaquett@licef.teluq.uquebec.ca

Abstract:

This chapter states and explains that a Learning Design is the result of a knowledge engineering process where knowledge and competencies, learning design and delivery models are constructed in an integrated framework. We present a general graphical language and a knowledge editor that has been adapted to support the construction of learning designs compliant with the IMS-LD specification. We situate LD within a taxonomy of knowledge models, namely the multi-actor collaborative system. We move up one step in the abstraction scale, showing that the process of constructing learning designs can itself be viewed as a unit-of-learning (or a "unit-of-design"): designers can be seen as learning by constructing learning designs, individually, in teams and with staff support. This viewpoint enables us to discuss and compare various "design plays". Further, the issue of representing knowledge, cognitive skills and competencies is addressed. The association between these "content" models and learning design components can guide the construction of learning designs and help to classify them in repositories of LD templates.

Keywords: learning design, educational modeling, knowledge-based systems, graphic languages, knowledge modeling, competency-based learning design, IMS-LD, learning design repositories.

1. Introduction

Building high quality learning designs is a very important task but a demanding one. It is a difficult task that we have started to address a decade ago by progressively building an instructional engineering method (Paquette et al. 1994, 2005a; Paquette 2003), a delivery system (Paquette et al, 2005b) and a graphical knowledge editor (Paquette 1996, 2002).

In this on-going work and in for the present discussion, the point of view that a Learning Design is the result of a knowledge engineering process is put forth, where knowledge and competencies, learning design and delivery models are constructed in an integrated framework.

In the next section of this article, a generic graphical modeling language is defined, MOT (modeling using object types) which was developed as the backbone of our instructional design methodology. A taxonomy of models will be presented and learning designs will be characterized in this taxonomy as multi-actor process models.

The following section will present the MOT+LD editor, as a Specialized Graphical Modeling Tool for IMS Learning Designs, as well as some examples and a process to engineer learning designs. We advocate

that this construction process can also be modeled as a multi-actor process model in order to analyze and improve the learning design methodology.

The last section presents other types of MOT models which represent domain knowledge and competencies that can be used to plan, support staff roles and evaluate the quality of learning designs. Finally, we propose that the domain and competency models can provide a classification scheme for a library of learning design templates.

2. Graphical Knowledge Modeling

When designers start building a Learning Design, two basic questions arise: "Which knowledge must be acquired and what are the target competencies or educational objectives for that knowledge?" and "How should the activities and the environment be organized to achieve knowledge and competency acquisition? To help designers solve these questions, we have developed a graphical knowledge modeling method and tools. In this section, we briefly present the basis for a modeling language to provide operational support to designers.

Goals of the MOT graphic language

It is often said that a picture is worth a thousand words. That is true of sketches, diagrams, and graphs used in various fields of knowledge. *Conceptual maps* are widely used in education to represent and clarify complex relationships between concepts to facilitate knowledge construction by the learners. *Flowcharts* are graphical representations of procedural knowledge or algorithms, composed of actions and decisions that trigger series of actions in a dynamic rather than static way. *Decision trees* constitute another form of representation used in various fields, particularly in decision-making expert systems, establishing influence or cause/effect relations between various factors. Building a decision tree is equivalent to building a series of rules which will constitute the knowledge base of the expert system.

In the last ten years, our main goal has been to generalize and consolidate various forms of graphical representations, which are useful for educational modeling, using an integrated graphical symbol vocabulary. In (Paquette 1996, 2002, 2003), we have shown that different kinds of models can be modeled more precisely using the same graphical language (syntax and semantics) by utilizing typed objects (concept, procedures, principles) as well as typed links. With this set of primitive graphic symbols, it is possible to build very different graphic models, from simple taxonomies to ontologies, more or less complex learning designs, delivery process, decision systems, methods etc. Besides its generality, the MOT graphical representational language has been proven sufficiently simple and friendly to be used by persons with non-technical background in many different contexts through the years. Modelling facilitates thought organization and communication between humans about the knowledge as the graphic representation, model, evolves. As will be seen, it can also be used both at a specialized domain knowledge level and at a meta-knowledge level, such as cognitive skills and competencies. Finally, the graphical MOTplus editor exports its models to different kinds of XML formats, including IMS-LD and OWL, for machine processing.

Syntax of the MOT Graphic Language

Concepts (or classes of objects), *procedures* (or classes of actions) and *principles* (or classes of statements, properties or rules) are the primitive objects of the MOT graphical language. Other primitive objects are instantiations of these three kinds of classes that correspond to single individuals. These individuals are respectively called *examples*, *traces* and *statements*.

MOT models are thus composed of up to six types of objects or knowledge units. The *object* type is represented by a geometrical figure as shown on figure 1, where each class or individual is represented by a name within the figure. Classes can be related to corresponding types of individuals by an instantiation (I) link.

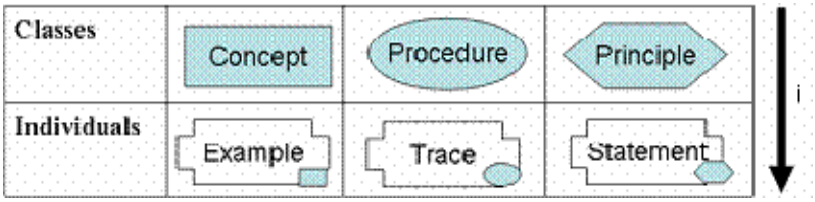

Figure 1 – Types of knowledge units in MOT

Table 1 presents various possible semantic interpretations of these graphic symbols.

Type	***Interpretations and Examples***
Concept	• Object classes: country, clothing, vehicles, … • Types of documents: forms, booklets, images, … • Tool categories: text editors, televisions, … • Groups of people: doctors, Europeans, … • Event classes: floods, conferences, …
Procedure	• Generic operations: add up numbers, assemble an engine, … • General tasks: complete a report, supervise production, … • General activities: take an exam, teach a course, … • Instructions: follow a recipe, assemble a device,… • Scenarios: the unfolding of a film, of a meeting,…
Principle	• Properties: the taxpayer has children, cars have four wheels, … • Constraints: the task must be completed within 20 days, … • Cause and effect relationships: if it rains more than 5 days, the harvest will be in jeopardy,... • Laws: any metal sufficiently heated will stretch out,… • Theories: all of the laws of the market economy,… • Rules of decision: rules to select an investment, … • Prescriptions: principles of instructional design principles, … • Regulating agent or actor: the writer who composes a text,…

Table 1

– Interpretation of various types of knowledge

The *relations* we use between objects are represented by links bearing a letter that specifies the type of relation. There are six basic types of relations or links that connect the various types of objects to form more complex models.

- The i*nstantiation link* (I), connects abstract knowledge (classes) to corresponding types of individuals

- The *composition link* (C) connects a class to other classes, either component attributes or constitutive parts of concepts, sub-procedures of procedures or component principles of more complex principles or set of principles; the C-link can also connect an individual to component individuals.
- The *specialization link* (S) connects two abstract knowledge of the same type, in which one is a sub-class of the other one; in other words, the second class is more generic or more abstract than the first one.
- The *precedence link* (P) connects two procedures or principles of which the first one must be completed or evaluated before the second starts; in a trace, it also connects individual actions of statements to other subsequent individual actions or statements.
- The *input-product link* (I/P) connects a concept and a procedure, from an input concept to the procedure (examples of the concept are possible inputs) or from a procedure towards to an output or produced concept (examples of the concept are possible outputs of the procedure).
- The *regulation link* (R) connects a principle to another class; in the case of a concept, the principle defines the concept by properties to be satisfied (sometimes called “integrity constraints”), or it establishes a law or a relationship between two or several concepts (for example rules); the regulation link from a principle towards a procedure or another principle means that the principle controls the execution of the procedure or the selection of other principles, for example a rule-based system controlling the execution of a process from the outside.

Types of Models: Ontologies and Learning Design

These basic classes or individual objects can be combined into increasingly complex systems of structured knowledge. For example, it is possible to represent conceptual maps, flowcharts (iterative procedures) and decision trees, and also other types of models useful for educational modeling.

Figure 2 presents five main categories of MOT models which are subdivided into sub-types. (see Paquette 2002 for more details).

Of particular interest here is the class "processes and methods" within which learning design is included, and "laws and theories" composed of concepts that can be organized in specialized hierarchies or part-whole hierarchies, and principles defining their properties and relationships. Particular cases are ontology models describing knowledge domains and competencies.

In (Paquette et al 2005a) the relationship between both types of models is presented as the foundation of the MISA method, which will be discussed further.

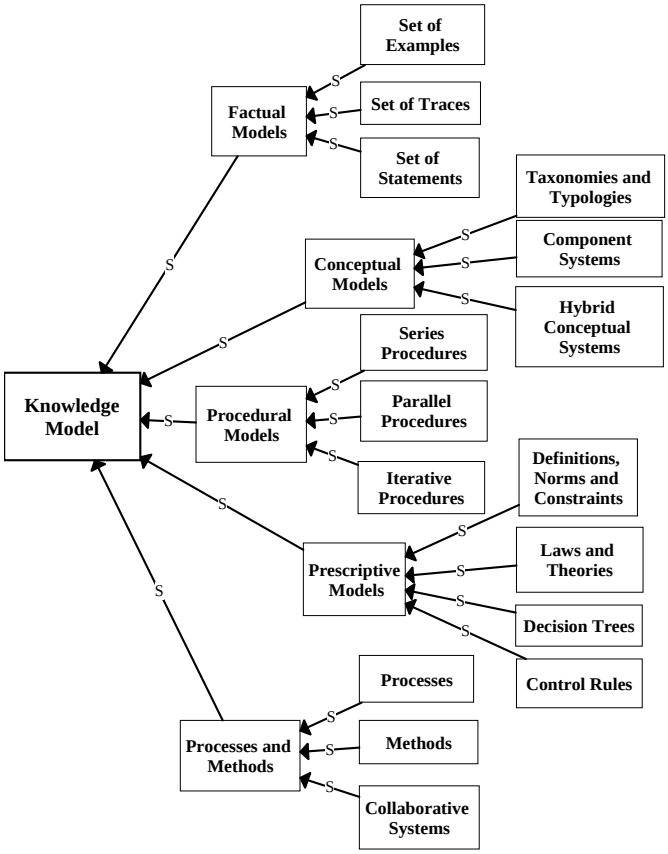

Figure 2 –Taxonomy of Knowledge Model Categories

Learning Designs as Collaborative Systems

The Processes and methods class of knowledge modesl, shown in figure 2, is a class that groups models mainly composed by procedures, where complex procedures are decomposed into simpler ones, each with their inputs and products. Three sub-categories can be discerned:

> In "*simple processes*" the execution of procedures is achieved by simple decision principles; the flow of control is embedded within the procedures in an algorithmic way.
>
> In "*methods*", the execution of the procedures is controlled by a set of principles; these principles can be heuristic rules governing the flow of control from outside the procedures that compose the model.
>
> In "*collaborative systems*" the execution of procedures is controlled by collective/collaborative decision principles; the control is distributed between formal rules embedded and described within the model, and actors personified by human participants that apply control to the process based on evaluations made at run-time.

From these definitions, it is possible to characterize the innovation that learning design brings to educational modeling. SCORM-based scenarios for example are sometimes simple processes, and sometimes (very rarely in practice) methods where simple sequencing (IMS-SS 2001) of activities is done by formal rules defined in the system.

IMS Learning Design, because they favour collaborative systems, adds a new dimension to simple sequencing systems. Activities are controlled by a combination of actors (making decisions at run-time) and formal rules: simple on-completion rules in LD level A, more or less elaborated rule-based systems (conditions) in LD level B, and rule-based systems mixed with actor notification in LD level C. Notifications request actors to exercise some control on the learning process according to the activation of certain conditions.

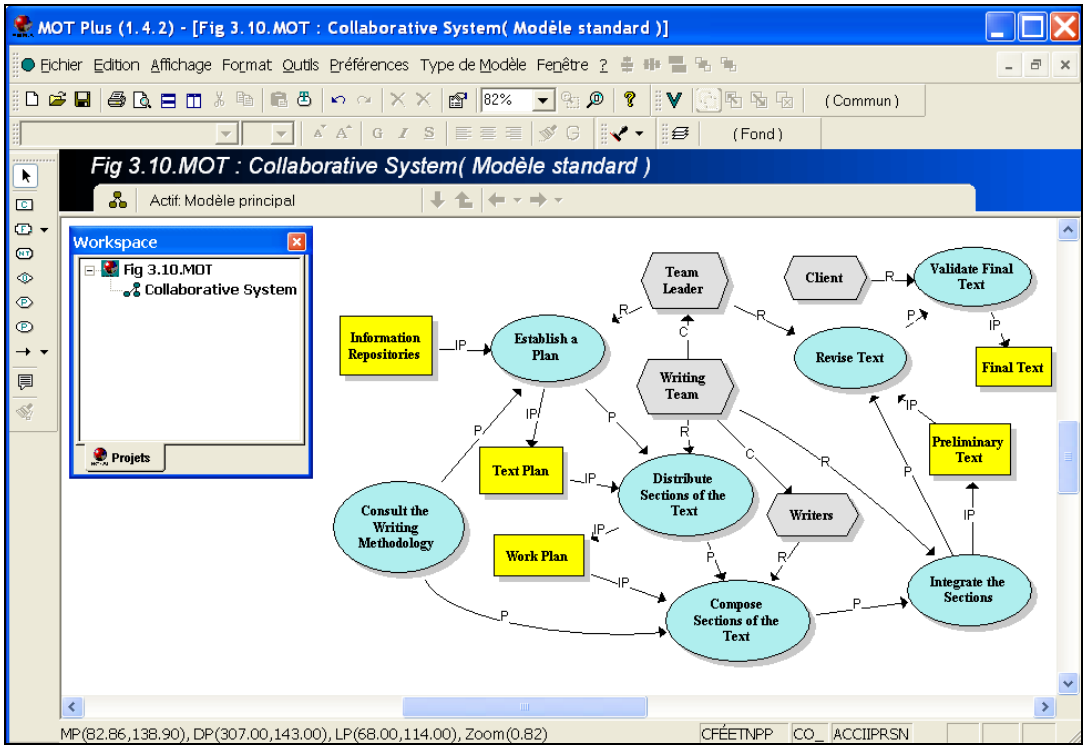

Figure 3 – An example of a MOT collaborative system model

Figure 3 offers a MOT model of a collaborative system very similar to learning design where activities are represented as procedures (ovals), input and output resources as concepts (rectangles) and actors by principles or control objects (hexagons). "Modèle standard" mean that the general MOTplus editor is used, This general modeling tool has served as the basis for the development of the MOT+LD editor, described in the next section.

3. MOT+LD, a Graphical Learning Design Editor

In this section, our graphical learning design editor MOT+LD described. It is based on the same graphical language explained in the previous section. This development stems from MOT's sophisticated and mature graphical capabilities that were already in place and ready to be adapted. Any object can be decomposed into a sub-model on any number of levels. Each object can be associated to OLE compliant files, enabling a concrete walk-through of a model. Moreover, a standard feature of the MOTplus model editor makes it possible to associate to learning design components, components from other co-models, such as a domain knowledge model.

In Griffiths and al. (2005) a survey of learning design tools can be found including other graphic editors, showing the interest and adequacy of graphical modelling. In the IMS-LD best practice documents (IMS-

LD 2003), the UML modeling system includes activity diagrams and others that can be used to represent parts of learning designs. Although UML is now a standard in software engineering, and widely used, the different diagrams are not very well adapted to the task of building learning designs. Another proposal is the LAMS software, which is not LD-compliant but simplifies learning designer's tasks, providing a drag and drop mechanism for assembling a limited set of learning design components. We believe that this approach is interesting, but not powerful enough to support the whole LD specification.

The MOT+LD graphical editor enables designers to fully describe the structure and concepts inherent in lLevel A unit-of-learning and produce a standard LD XML schema. Work is on-going to extend the editor to levels B and C. In Griffiths and al. (2005), this approach is considered "significant, not only because it provides an example of a powerful and expressive high-level LD editor, but also because the structure of LD are mapped onto a graphical language which appears to be very remote from the specification". Our aim is to provide a way closer to instructional designer's needs for building Learning Designs, alleviating the designer from having to deal with XML, but at the same itme producing automatically a completely IMS-LD conformant XML manifest file from the graphs.

MOT+LD Graphic Vocabulary

Basically, all the MOT objects and links applicable to LD were used and interpreted with much of the same general semantics. Figure 4 shows the resulting equivalences and symbolism. Resources are represented by five kinds of concepts, the LD method components (actions) are represented by seven kinds of procedures, whereas actors and rules are represented by five kinds of principles. Individual objects are represented by individual symbols (also called "facts") representing learning objectives and prerequisites, metadata, items, and four other types of objects needed to describe the conference, send-mail and index-search services.

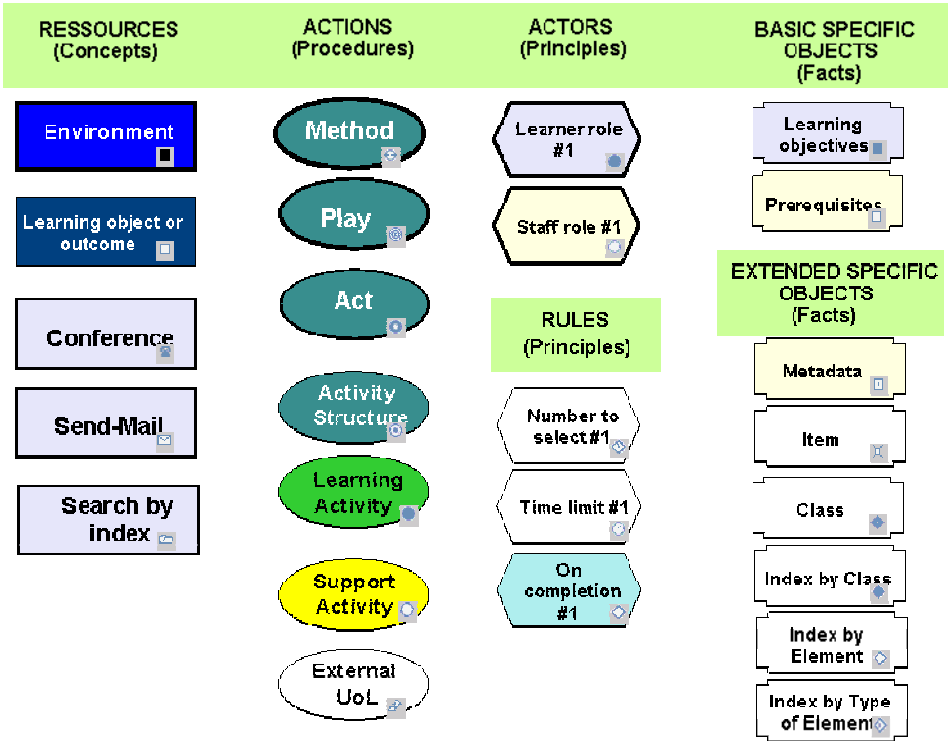

Figure 4 - MOT+LD basic vocabulary

The same basic links as in the general MOT language can be applied, however we had to consider a number of new constraints on links between subtypes specified in the IMS Information and Binding model in order to produce a valid XML manifest file.

Figure 5 underlines the relative complexity of the LD information model (IMS-LD, 2003) but helps to understand it better. It shows a rather straightforward use of the component C-link. An environment is composed of other environments recursively or of other types of resources, learning objects, outcomes and/or services. Learner and staff roles, and also items can be organized in sets of components hierarchies. Methods are decomposed into plays, which are decomposed into acts, which are decomposed into role-parts, represented in our model by role associated to the activity at any depth; finally terminal activity structures are decomposed into learning activities, support activities or a reference to an external units-of-learning (UoL).

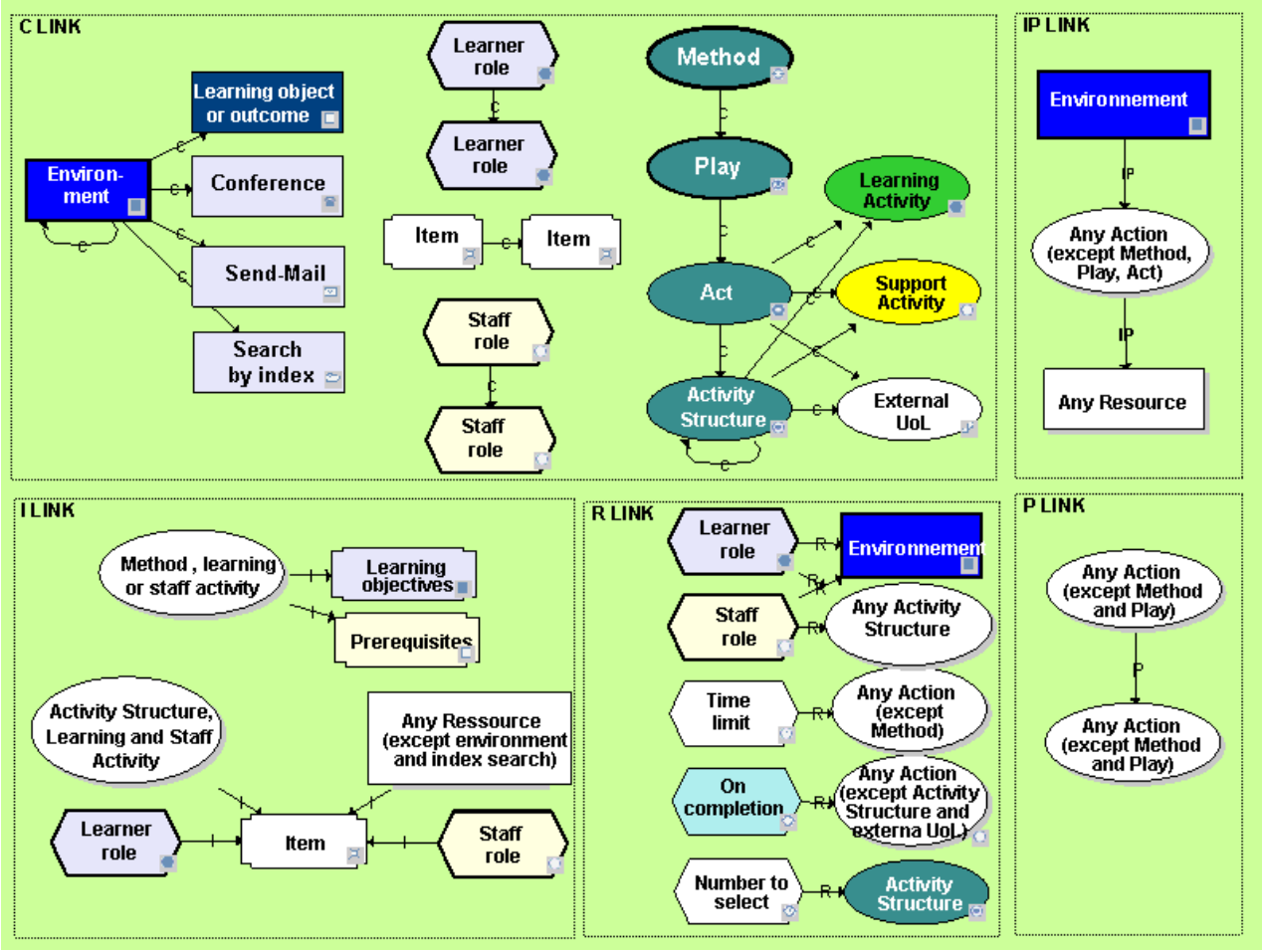

Figure 5 - MOT+LD link constraints

The use of input/product I/P-link and precedence P-link is clear and unambiguous. The precedence link is used between procedures only below the Play level. The I/P link is used only below the Act level, from an input resource to a procedure or conversely, from a procedure to its resource outcome. This is more precisely put than the specification itself, since the LD XML file does not distinguish between input resources and outcomes, whereas the outcome is a necessary ingredient of a Learning Design from a designer's point of view.

The instantiation I-link associates learning objectives and prerequisites to a method or to learning activities. Activity structures, learning and support activities, learning and staff roles or resources (except environment and index search) can be associated to items pointing to a location where the physical file of the objects are found.

Finally, the regulation R-link associates learner and staff roles to any environment or activity structure, learning or support activity, or it may associate a time limit to any action except a method. It is also used to associate a completion rule to any action except an activity structure and a UoL. The number to select rule is R-linked to an activity structure where options are proposed.

Technically, subtypes of the original MOTPlus object types were added and new graphical symbols with standardized labels (as shown on figures 4 and 5) to distinguish each subtype from the others of the same type had to be developed. The most difficult technical part was to extend the native MOT XML schema and to parse it into the IMS-LD XML schema.

A post-validation mechanism was built into the translation program informing the designer whether a rule of the IMS-LD specification is violated and where to find it in the model. The number of possible violations is reduced while designing the model by limiting the choice of possible links between sub-types according to the constraints shown on figure 5. Also, some of the constraints for metadata association and the description of the services not presented here have been covered. Finally, many examples were tested, including the wellknown complex Versailles example (see IMS 2003 Best Practices) modelled in figure 6. The MOTPlus XML manifest files were uploaded in the RELOAD editor (RELOAD 2004), a form-based LD editor. This exercise has shown very small discrepancies between our analysis of the specification and theirs. Minor corrections were made to the MOT+LD editor to arrive at the present version.[1]

As illustrated in figure 6, the upper window shows that the Method is composed of one Play, composed of 8 Acts. Act 6 has been decomposed by a graph not shown in the figure, composed of activity structures describing the negotiation day for each country, similar to the “France Negociation Day” model presented in the second model in figure 6. Finally, each of the learning activities within this activity structure is structured the same way as the smaller model in the bottom right hand corner. This model presents the France-Serbia side-room discussion in an environment composed of a conference service and the discussion activity as well as their items pointing to corresponding resources.

[1] A version of the MOT+LD editor is available on the CICE Web site (www.cice.org) or on the Unfold Web site (http://www.unfold-project.net:8085/UNFOLD/)

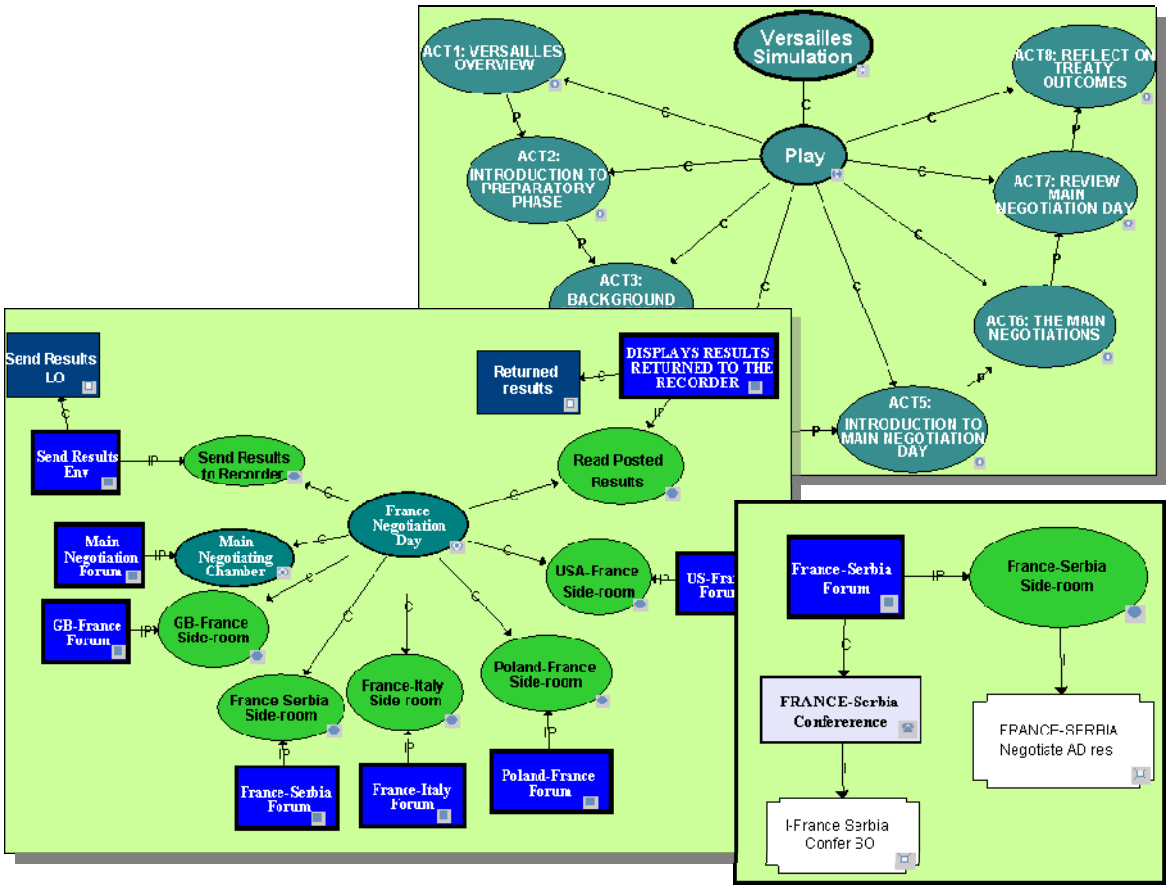

Figure 6 - MOT+LD link constraints

LD Engineering Processes and Meta LD Models

A simple design process is provided in the MOT+LD user guide. Seven steps indicate the main tasks involved in engineering an IMSLD Unit of Learning : 1- Open an LD template, 2- Add prerequisites and learning objectives linked to the Method object to guide the engineering of the UoL method, 3- Specify actor roles and hierarchies, specifying minimum and maximum for each role, 4- Develop the instructional structure (Method, Plays, Acts and Role-parts) as defined by the LD Information Model, 5- Add items to resources, activities, roles, add appropriate metadata to learning objects and services; 6- Save the model as a LD Manifest and revise, if necessary, 7- Export the manifest to a LD Player.

Obviously, these are only the main processes. They are insufficient to guide effectively the process, but they summarize the fundamentals of engineering a LD Model. Many elements are missing. Prerequisites and learning objectives could be obtained by modeling the domain knowledge and associating it to target competencies. Also, the gap between entry and target competencies give designers clues on the scope of the UoL and its corresponding knowledge model. Finally, as discussed in the last section, target knowledge and competency statements help orient designers on the types of learning strategies and activity structures to select. For example, conceptual and procedural knowledge are not learnt in the same way; to acquire the competency to apply an administrative procedure is less demanding than acquiring a competency to build and adapt such procedures.

A couple of years ago, the MISA instructional engineering method, its operations, products and principles were modeled using an early version of the MOT software. Presently, a new model of MISA using the MOT+LD software is being developed within the framework of the IMS-LD information model.

Figure 8 represents the MISA method as one of many possible engineering methods to create a "Unit-of-Learning". This MOT+LD model shows two plays, one for Web delivery and the other for classroom delivery. Many other plays are of course possible. In the Classroom play, only the first act is needed since the UoL will be delivered directly by the professor. In that case, only the steps 1-2-3-4 of the above engineering method are required.

In the Web delivery play, the designer (or the design team) will have add two more acts besides the LD model composition. Act 2 is where the components are itemized to be assigned to concrete resources, activity assignments or participants, and also where services are described more precisely. Act 3 simply produces a validated LD XML file for delivery purposes.

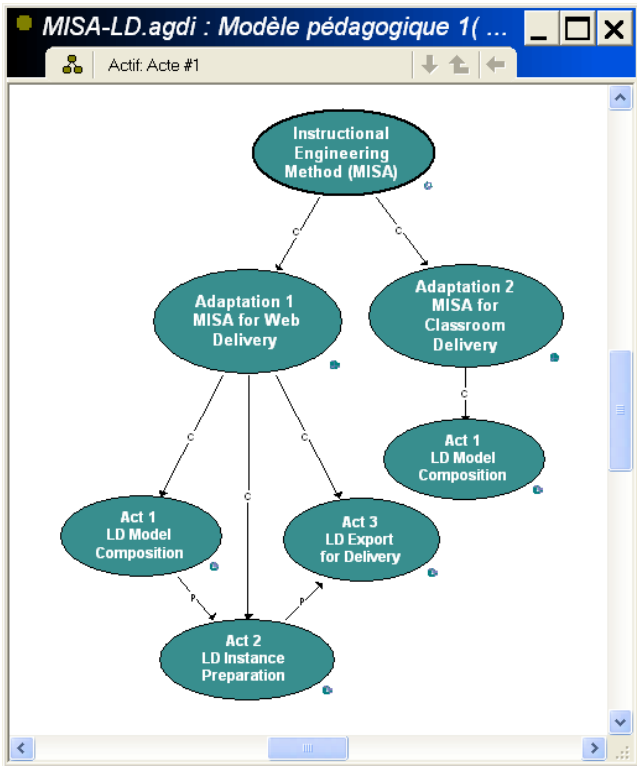

Figure 8 – MISA as a LD (meta)-method

A general instructional engineering method like MISA needs to be adapted to many possible situations. The preceding discussion opens the way to investigate a variety of ways to adapt MISA as a LD construction method described as alternate "design plays". In the model in figure 9, a fragment of one of the possible models for Act 1 is modelled. Here "learner roles" are replaced by "designer roles" and "staff roles" by "IMS-LD facilitator roles".

"MISA for Web delivery" is presented in figure 9 as the main activity structure in Act 1. It starts by the design team's preliminary analysis of training needs, target population, available resources, delivery and cost constraints, etc. followed by four processes, again modeled as activity structures, start in parallel. These correspond to the design team's role-parts, such as the content expert, the instructional designer, the media designer and the delivery specialist. In figure 9, one of the role-parts is the combination of the Instructional Designer Role and the Instructional Modeling Activity, the only Activity Structure developed in this model. Note that the other activity structures on figure 9 need to be expanded in a similar way.

The instructional modeling activity structure is the one that corresponds directly to the engineering of the learning design. It is supported by a staff role where an IMS-LD facilitator coaches designers using an IMS-LD guide and a LD forum included in a community-of-practice environment. Designers start by stating instructional orientation principles and proceed to develop the UoL using an environment composed of the MOT+LD editor, the PALOMA learning object manager[2] and the RELOAD tool. Then knowledge units and competencies are associated to learning activities and to resources (using metadata).

[2] See Paloma LO Repository Manager http://www.cogigraph.com

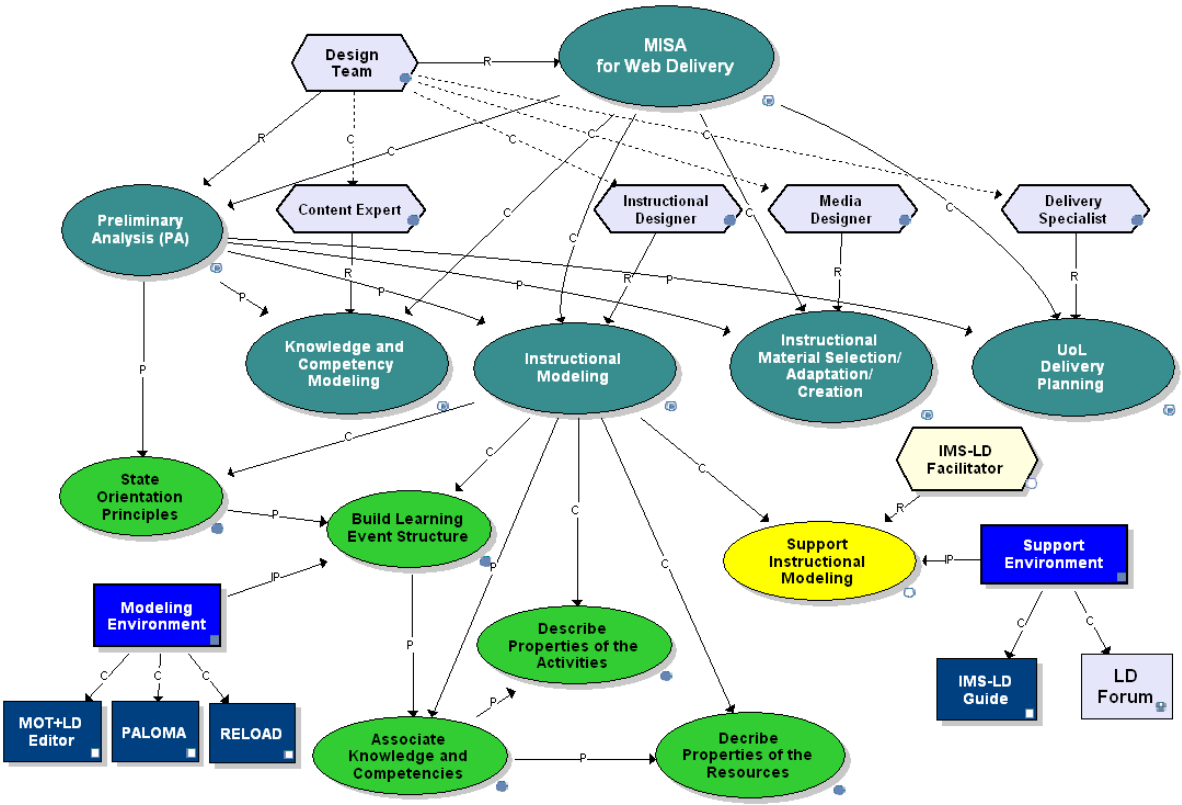

Figure 9 – MISA for Web delivery Act 1 – Main activities

4. Generic Skills and Learning Designs

The relationship between a learning design model and a knowledge and competency model is critical. In IMS-LD, prerequisites and learning objectives can be defined using the IMS-RDCEO specification. In (Paquette and Rosca 2004) we have shown that using unstructured text to define competencies or learning objectives is not sufficient to help guide the learning design engineering. Furthermore, competencies should be linked to knowledge units in the learning domain, where both should be associated to actors, activities and resources at any level of the learning design. In this section, the notion of competency specification is elaborated by relating cognitive skills to knowledge, our taxonomy of cognitive skills is defined, a way to represent them as procedural (meta-) knowledge models is explained. Further, we show how competency modeling can contribute to the guidance of the learning design engineering process.

Competency: Cognitive Skills Applied to Knowledge

To say that a person knows something (prerequisite) or that a person must acquire such or such knowledge (learning objective) is not sufficient. What is needed is to specify a degree or a level of knowledge mastery. Thus, we define a *competency* as a statement that an "actor" has the ability to apply to a certain knowledge unit, a precise *cognitive skill*, with a specific degree of “performance” in a certain context.

We define a *cognitive skill*, as a generic intellectual, socio-affective or psycho-motor ability, such as to memorize, to transpose, to analyze, to synthesize, to evaluate, to self-control and so on, which can be

applied in different knowledge domains. If we need more precision, we can add a degree of *performance* specifying in which situational context the cognitive skill can be applied: in familiar or new contexts, in a persistent or sporadic way, in simple or complex situations, etc.

Competencies state objectives to be reached in relation to some knowledge, or an actual state of the knowledge that someone possess. They also identify the cognitive skill that must be applied by a learner or that can be developed or acquired through learning activities. Finally, by specifying a performance context, competency statements help designers build useful learning activities, environments and assessment tools to help learners and trainers test their knowledge and cognitive skill, which in turn is one way of ensuring some quality control of the learning design.

Possessing a cognitive skill means that a learner can solve a corresponding class of problems (Chandrasekarann 1987, McDermott 1988, Steel 1990). For example, if a learner possesses a diagnostic or classification skill, he should be able to solve some diagnostic or classification problems to a certain performance level prescribed by the context. Another view is to see cognitive skills as active procedural meta-knowledge (generic procedures) applied to knowledge (Pitrat 1991, 1993). A third view considers the association between cognitive skills and application knowledge as objects to be learned together, such as educational objectives principles and statements (Bloom 1975, Krathwohl et al 1964, Reigeluth 1983, Martin and Briggs 1986). Integrating all three viewpoints will enable us to provide a cognitive skills taxonomy that might prove useful for learning design.

A Skill Taxonomy

Table 2 presents an overview of the skills taxonomy proposed. This taxonomy combines and adapts an artificial intelligence taxonomy (Pitrat 1990), a software engineering taxonomy (Breuker and Van de Velde, 1994; Scheiber et al. 1993) and two educational taxonomies (Bloom 1975 ; Romiszowski 1981). Although the terms are not in direct correspondence, table 2 distributes them onto ten levels that lay the foundations for our taxonomy (Paquette 1999, 2003)

In this taxonomy, cognitive skills can be viewed according to three perspectives: as a generic problem solving process, as procedural meta-knowledge acting on knowledge or as a learning objective related to a knowledge processing task. Contrary to the traditional view on learning objectives, here skills are viewed as knowledge objects can be described, analyzed and evaluated, by themselves or in relation to knowledge domains of various fields.

The taxonomy showed in the left part of table 2 portrays three layers, from left to right, from the generic to the specific term. It could be expanded to more layers for additional precision. The first two layers are ordered from simple to complex. A detailed discussion of the validity of this ordering can be found in (Paquette 2002) together with precise definitions and examples of each skill.

Cognitive Skills Taxonomy Layers			Active meta-knowledge (Pitrat)	Generic problems (KADS)	Cognitive objectives (Bloom)	Skills cycle (Romiszowski)
1	2	3				
Receive	**1. Acknowledge**					Attention
	2. Integrate	**2.1 Identify** **2.2 Memorize**			Memorize	Perceptual acuteness and discrimination
Reproduce	**3. Instantiate / Specify**	**3.1 Illustrate** **3.2 Discriminate** **3.3 Explicitate**	Knowledge Search and Storage		Understand	Interpretation
	4. Transpose/ Translate					Procedure Recall Schema Recall
	5. Apply	**5.1 Use** **5.2 Simulate**	Knowledge Use, Expression		Apply	
Create	**6. Analyze**	**6.1 Deduce** **6.2 Classify** **6.3 Predict** **6.4 Diagnose**	Knowledge Discovery	Prediction, Supervision, Classification, Diagnosis	Analyze	Analysis
	7. Repair			Repair		Synthesis
	8. Synthesize	**8.1 Induce** **8.2 Plan** **8.3 Model/ Construct**		Planning, Design, Modeling	Synthesize	
Re-invest	**9. Evaluate**		Knowledge Acquisition		Evaluate	Evaluation
	10. Self-manage	**10.1 Influence** **10.2 Self-control**				Initiation, Continuation, Control

Table 2– Taxonomies of Cognitive Skills

Representation of a Cognitive Skill

Every cognitive skill in the taxonomy can be represented as a MOT process by a main procedure in the meta-knowledge domain, which is the domain that categorize knowledge and describe processes and principles to transform and acquire knowledge. The main procedure is broken down into sub-procedures, to as many levels as needed, until terminal procedures are found that do not need further decomposition. For each procedure, there is also a description of input or product concepts that feed them or are generated by them, as well as principles that regulate the transfer of control between the generic procedures. Cognitive skills or processes are thus structured sets of generic cognitive actions that can be instantiated to different knowledge domains called application domains.

In table 3, the "5.2-Simulate a process" skills, a sub-class of the level "5-Apply skills", are compared to the level "8.3-Construct a process" skills, which is a sub-class of the "8-Synthesize" skills in the taxonomy.

Skill	Input	Product	Process Flow
Simulate a process	A *process*, its procedures, inputs, products and control principles.	A *trace* of the procedure : set of facts obtained through the application of the procedures in a particular case	- Choose input resources objects (data) - Select the first procedure to execute - Execute it and produce a first result - Select the next procedure and execute it - Use the control principles to control the flow of execution
Construct a process	Definition constraints such as relations between inputs and products of the process and/or required steps in the process.	A description of the process: its inputs, products, sub-procedures with their input and output, and the process control principles.	- Assign a name to the procedure to be constructed - Relate this main procedure to a specific input and product resource, respecting the definition constraints - Decompose the procedure, respecting the definition constraints - Continue to a point where well understood small procedures are defined.

Table 3 – Comparison of two generic skills

From the descriptions of these two generic skills, we can easily see that a lerning design aiming at the acquisition of procedural knowledge such as “Information search on the Internet” will be very different if the goal (the learning objective) is to simulate that process or to construct it. In the first case, a number of walk-throughs of the process will probably be sufficient, while in the second case, a project-based scenario where learners and engaged in a more complex problem-solving activity is a better suited learning strategy. The description of both processes is however just a summary example to illustrate the potential use of competency statements.

From Cognitive Skill Models to Activity Structures

The cognitive skills are processes, which are easily represented as MOT models. The MOTplus graph on the left side in figure 10 entitled “Meta-knowledge Model”, provides a more precise definition of the “Simulate a process” skill. This cognitive skill is described by its main procedures with its input (the process to simulate) and its product (a trace of the process). These main procedures are decomposed into sub-procedures, each being associated with less complex cognitive skills that provide intermediate products, which are reused by other sub-procedures, until the process is completed. The resulting trace can be produced by collecting the individual products of the walk-through. On the graph, four groups of principles are added to constrain concepts or control procedures in the process. Note that this model is totally generic, applicable to any specific knowledge domain, such as Internet processes, manufacturing processes, or others.

Figure 10 provides an example on how to build an activity structure based on such a cognitive skill model. In this activity structure, learners will simulate the process “Search information on the Internet” performing learning activities similar to the sub-procedures of the “simulate a process” skill. To build the activity structure shown on the right part of the figure labelled “Learning Scenario”, a graph similar to the generic process is modelled, however, taking a “learning activity” viewpoint. The specific domain vocabulary is used, and the five activities are formulated in an “assignment style” format. As in the cognitive skill model, the activity structure starts with a description of the process to simulate and ends by producing a trace report of the simulation.

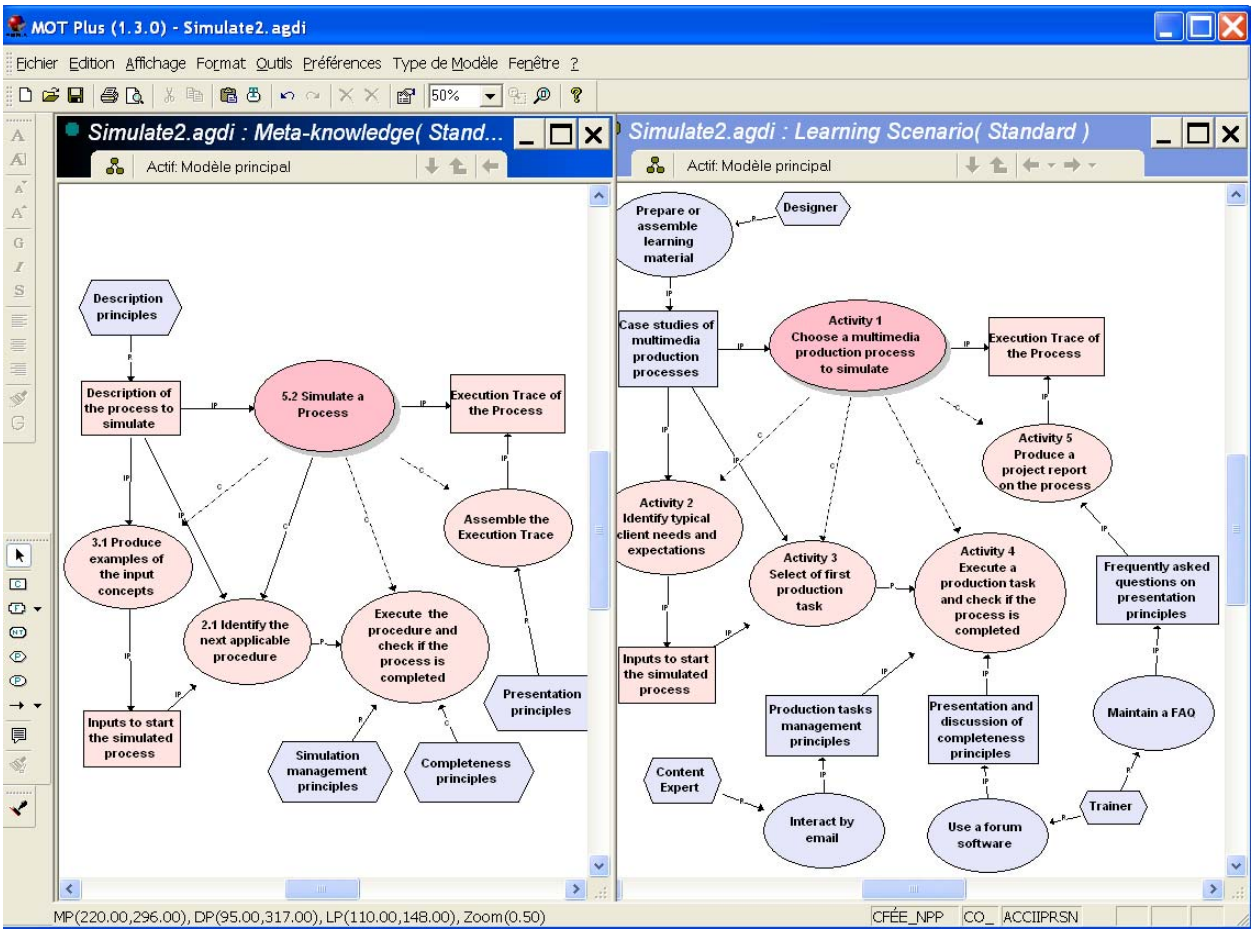

Figure 10– A learning scenario model simulating the "Search the Internet" process

Of course the learning design is not yet complete. For example, resources that help learners achieve their tasks can be added, such as a tutorial on the structure of a request or on a final report form. Also, we might specify some collaborative assignments and maybe a description of the evaluation principles that will be used to assess the learner's work. All these additions should be guided by the skill's models' sets of principles in order to ensure instructional quality. For example the "completeness principles" can become a check-list for the learner, or a guide for a trainer to help learners execute the simulation completely.

But the important thing here is that the generic process becomes the founding principle for the learner's assignments. In that way, we make sure that he exercises the right cognitive skill, in this case "simulating a process", while working on the specific knowledge domain, thus building specific domain knowledge and meta-knowledge at the same time.

Metadata for Learning Design Repositories

Another use of the skill taxonomy is to help identify important metadata for learning design repositories. Recently, while working on documents to support the use of Educational Modeling Languages and the IMS Learning Design specification (IMS-LD 2002), it was stated that "To support reusability of good learning designs, it is essential that libraries of learning designs can be made available as learning objects in one or more repositories" (Paquette et al 2005). In (Koper 2005), similar preoccupations are expressed and discussed.

It is proposed that learning object repositories under construction in different countries should distinguish between “content object”, “tool objects” and “process objects”, the latter including generic and specific learning designs (or scenarios). If a growing library of these learning designs is available, then reuse by adaptation to to particular knowledge domains can increase. New learning design templates could be built by abstracting generic processes from a large body of existing scenarios, situating the resulting abstraction in the framework of a generic skills’ taxonomy.

The preceding discussion opens a door to organize repositories of generic learning design templates related to cognitive skills that can provide a way to classify learning designs or scenarios by their association to generic graphic knowledge-based models. In the beginning of the development of our Instructional Engineering methodology, we first developed a set of such templates that have been used to start the construction of learning scenarios in different domains, further enhanced with a small advisory system assisting the designer in selecting proper scenarios in different situations (Paquette et al, 1994). In the MISA documentation, later on, and in field applications carried out since, we have collected a large set of designs that need to be systematically organized as a kind of learning scenario repository or handbook. A more comprehensive collection is being created on the corpus of distance learning courses at Télé-université.

These learning design templates can be organized as a hierarchy indexed by the main cognitive skill they exercise and other metadata can be added to further identify the type of knowledge (concept, procedure, principle, facts) or knowledge model involved in the LD template. For example, it is quite different to synthesize or construct a taxonomy, or a process, or a decision tree thus demanding clarifications explaining the performance context of the LD template.

Conclusion

The systematic interpretation of competencies using the cognitive skills taxonomy creates a bridge between competency profiles and instructional engineering in many ways. For each main knowledge unit, the gap between the entry or actual competency and the target competency of the learner can guide the construction of knowledge models; if the gap is large, for example starting at a simple memorizing skill targetting an evaluation skill, then the knowledge model will be quite complex, more so then if the goal is just to increase the performance level within an evaluation skill.

As discussed, target competencies and their associated cognitive skill process model provide a solid foundation to engineer effective and efficient learning scenarios ensuring some type of quality control as well as serving as criteria for classifying learning design templates. Competency models also makes it possible to create activities for other actors in the learning design aiming to improve coordination between roles and to offer appropriately adapted resources in each case .

In this paper, we have advanced a new strategy, competency based design based on a knowledge model, describing a design process that facilitates designer’s tasks to create learning designs which are multi-actor learning processes. An instructional engineering method is itself a multi-actor process used to engineer other multi-actor processes for learners and staff. We believe this novel use of LD can shed light on alternative methodologies that will assist in implementing the IMS-LD specification more easily and with a solid instructional design foundation.

Learning design based on graphical knowledge modeling is the basis of all the discussion carried out here. It helps situate the components and the levels of knowledge involved in a more precise and transparent way. Our goal is now aimed at providing user-friendly and powerful tools to educators and designers to increase the production of higher quality learning designs.

References

Bloom, Benjamin S. (1975) *Taxonomy of Educational Objectives: the Classification of Educational Goals*. New York: D. McKay.

Breuker J. and Van de Velde W. (1994) *CommonKads Library for Expertise Modelling*. IOS Press, Amsterdam, 360 pages.

Chandrasekaran B. (1987) Towards a Functional Architecture for Intelligence Based on Generic Information Processing Tasks. *IJCAI-87 Proceedings*, Milan Italy, pp 1183-1192

Griffiths, D., Blat, J., Garcia, R., Votgen, H., & Kwong, KL. (2005). Learning Design Tools, in R. Koper & C. Tattersall (Eds.). *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*, Springer Verlag, pp. 109-136

IMS-RDCEO (2002) *IMS Reusable Definition of Competency or Educational Objective – Best Practice* (2002). 1.0 Final Specification, IMS Global Learning Consortium, Inc. Revision: 25 October 2002, http://www.imsglobal.org/competencies/index.html

IMS-LD (2003) *IMS Learning Design. Information Model, Best Practice and Implementation Guide, Binding document, Schemas.* Retrieved October 3, 2003, from http://www.imsglobal.org/learningdesign/index.cfm

IMS-SS (2003) *IMS Simple Sequencing* 1.0 Final Specification, IMS Global Learning Consortium, Inc. Revision: 03 March 2003 http://www.imsglobal.org/simplesequencing/index.html

Koper, R. (2005). An Introduction to Learning Design, in R. Koper & C. Tattersall (Eds.). *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*, Springer Verlag, pp. 3-20

Krathwohl D.R., Bloom, B.S., and Masia, B.B. (1964) *Taxonomy of educational objectives : The classification of educational goals.* Handbook II: Affective domain. New York: Longman, 1964

Martin, B.L. & Briggs L. (1986) *The Affective and Cognitive Domains: Integration for Instruction and Research.* Educational Technology Publications, New Jersey, 494 pages, 1986

McDermott J.. (1988) Preliminary steps towards a taxonomy of problem-solving methods. In Marcus, S. (Ed), *Automating Knowledge Acquisition for Expert Systems*, pp. 225-255. Kluwer Academic Publishers, Boston, Mass.

Paquette, G., De la Teja, I., Léonard, M., Lundgren-Cayrol, K., & Marino, O. (2005a). How to use an Instructional Engineering Method and a Modelling Tool, in R. Koper & C. Tattersall (Eds.). *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*, Springer Verlag, pp. 161-184

Paquette,G., Marino, O., De la Teja, I., Léonard, M., Lundgren-Cayrol, K., (2005b). Delivery of Learning Design: the Explor@ System's Case. In R. Koper & C. Tattersall (Eds.). *Learning Design – A Handbook on Modelling and Delivering Networked Education and Training*, Springer Verlag, pp. 311-326.

Paquette G., O. Marino, I. De la Teja, K. Lundgren-Cayrol, M. Léonard, and J. Contamines (2005) Implementation and Deployment of the IMS Learning Design Specification, *Canadian Journal of Learning Technologies* (CJLT), *http://www.cjlt.ca/*

Paquette, G. et Rosca. I. (2004) *An Ontology-based Referencing of Actors, Operations and Resources in eLearning Systems.* SW-EL/2004 Workshop. Einhoven, August 2004

Paquette, G. (2003) *Instructional Engineering for Network-Based Learning*. Pfeiffer/Wiley Publishing Co, 262 pages.

Paquette, G. (2002) *Modélisation des connaissances et des compétences, un langage graphique pour concevoir et apprendre*, 357 pages, Presses de l'Université du Québec, mai 2002.

Paquette, G. (1999) Meta-knowledge Representation for Learning Scenarios Engineering, in S. Lajoie et M. Vivet (Eds), *Proceedings of AI-Ed'99 in AI and Education - Open learning environments*, IOS Press, 1999.

Paquette, G. (1996) *L*a modélisation par objets typés: une méthode de représentation pour les systèmes d'apprentissage et d'aide a la tâche. *Sciences et techniques éducatives* , France, avril 1996.

Paquette, G., F. Crevier, C. Aubin. (1994) ID Knowledge in a Course Design Workbench. *Educationnal Technology*, USA, volume 34, n. 9, pp. 50-57, November 1994

Pitrat J. (1993) *Penser l'informatique autrement*. Hermès, Paris, 1993.

Pitrat J. (1991). *Métaconnaissance, avenir de l'Intelligence Artificielle*. Hermès, Paris, 1991.

Reigeluth C. (1983) *Instructional Theories in Action: Lessons Illustrating Selected Theories and Models.* Hillsdale, NJ: Lawrence Earlbaum, 487pages.

RELOAD (2004) RELOAD Project. Retrieved Janduay 23, 2004 from http://www.reload.ac.uk

Romiszowski A. J. (1981) *Designing Instructional Systems* Kogan Page London/Nichols Publising, New York, 415 pages.

Schreiber G., Wielinga B., Breuker J. (1993) *KADS – A Principled Approach to Knowledge-based System Development*. San Diego : Academic Press. 457 p.

Steels L. (1990) Components of Expertise. *AI Magazine*, vol. 11, no 2, Summer 1990.