

Learning Design based on Graphical Knowledge-Modelling

Gilbert Paquette, Michel Léonard, Karin Lundgren-Cayrol, Stefan Mihaila and Denis Gareau

Center for Interuniversity Research on Telelearning Applications
CIRTA (LICEF), Télé-université, Université du Québec
4750 avenue Henri Julien, bureau 100, Montréal, Québec, Canada
Tel: +1 514 840-2747 Ext 2292
gpaquett@licef.telug.quebec.ca
licef@licef.telug.quebec.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 our taxonomy of knowledge models as a 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 modelling, Knowledge-based systems, Graphic languages, Knowledge modelling, Competency-based learning design, IMS-LD, Learning design repositories.

Introduction

Building high quality learning designs is a very important and demanding task. It is also a difficult task that we started to address already 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 modelling editor (Paquette, 1996; 2002).

In this on-going work and for the present discussion, the point of view is taken 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.

In the next section of this article, a generic graphical modelling language is defined, MOT (Modelling using Object Types) which was developed as the backbone of our instructional design methodology. Our taxonomy of knowledge models will be presented and learning designs will be characterized according to this taxonomy as collaborative multi-actor process models.

The third section will present the MOT+LD editor, as a Specialized Graphical Modelling 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 modelled as a multi-actor process model in order to analyze and improve 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.

Graphical Knowledge Modelling

Graphical knowledge modelling is a way of representing knowledge structures or domains by linking concepts, procedures and principles in a way that describes the phenomena at hand. In the case of Learning Designs, the

basic structures can be likened to a workflow model containing information on who does what, when and with what type of resources.

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 best achieve knowledge and competency acquisition? To help designers solve these types of questions, we have developed a graphical knowledge modelling method and tools. In this section, we briefly present the basis for a modelling language to provide operational support to designers by discussing and explaining its goals, syntax and semantics as well as types of models and examples.

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 modelling, using an integrated graphical symbol vocabulary. In Paquette (1996; 2002; 2003), we have shown that different kinds of models can be modelled 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 be used both at a specialized domain knowledge level and at a meta-knowledge level, such as cognitive skills and competencies. Finally, the graphical MOT+ editor exports its models to different kinds of XML formats, including IMS-LD and OWL, for machine processing.

The benefits of graphical knowledge or cognitive modelling (Ausubel, 1968; Dansereau, 1978; Paquette, 2002) can be summarized as follows: it

- illustrates relationships among components of a complex phenomena
- makes evident the complexity of actors interactions
- facilitates the communication of the reality studied
- ensures the completeness of the studied phenomena
- helps scanning for a general idea because it minimizes use of text.

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*.

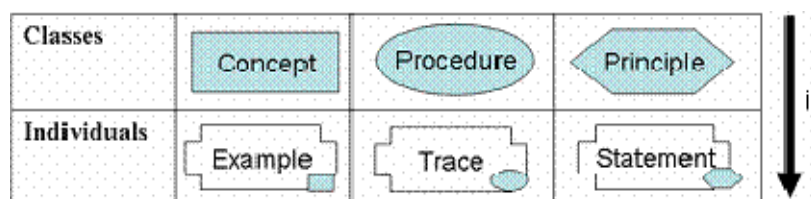


Figure 1. Types of knowledge units in MOT

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 in 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.

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

Table 1. Interpretation of various types of knowledge
Interpretations and Examples

<i>Type</i>	
Concept	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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 ...

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 *instantiation 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 objects 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 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 modelling.

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.

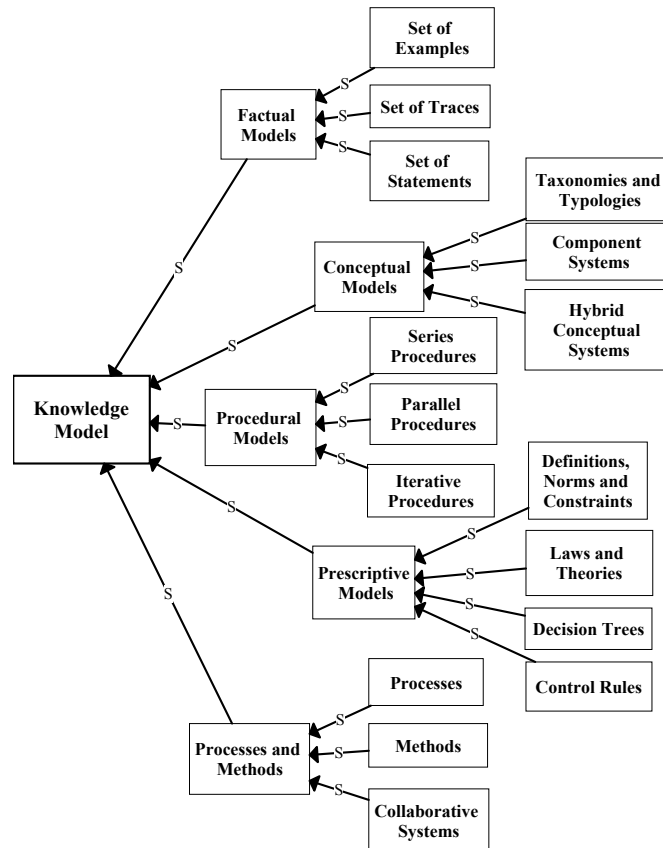


Figure 2 –Taxonomy of Knowledge Model Categories

Learning Designs as Collaborative Systems

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

- In “*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 modelling. SCORM-based scenarios for example are sometimes simple processes, and sometimes (very rarely in practice) methods where simple sequencing (IMS-SS, 2003) of activities is done by formal rules defined in the system.

IMS Learning Design, because it favours 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.

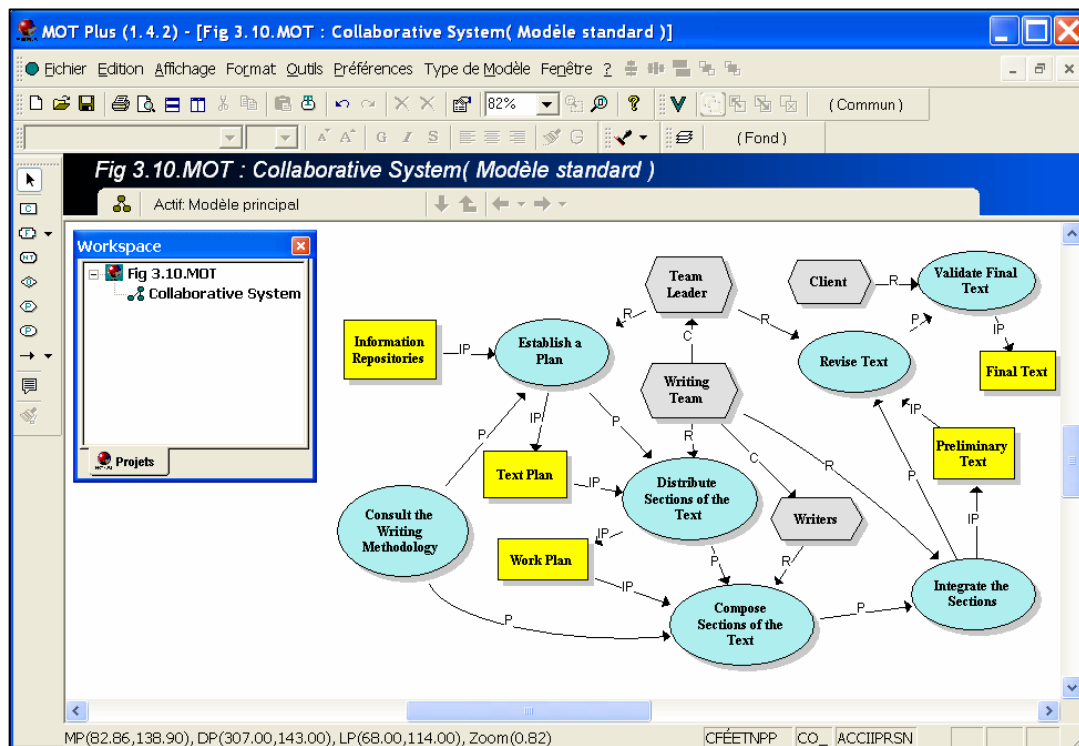


Figure 3. An example of a MOT collaborative system model

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” means that the general MOTplus editor is used. This general modelling tool has served as the basis for the development of the MOT+LD editor, described in the next section.

MOT+LD: a Graphical Learning Design Editor

In this section, our graphical learning design editor MOT+LD is 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 knowledge object can be decomposed into a sub-model on any levels. Each object can be associated to OLE compliant files, enabling a concrete walk-through of a model. Moreover, a standard feature of the MOT+ model editor makes it possible to associate components from co-models, such as a domain knowledge model. This feature is also available in the LD version of the software.

Griffiths et al. (2005) survey of learning design tools includes other graphic editors, which shows the interest and adequacy of graphical modelling to express learning scenarios or learn flows. In the IMS-LD best practice documents (IMS-LD 2003), the UML modelling system includes activity diagrams and others that can be used to represent certain learning design concepts and activity flows, but not all. 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 because it does not include the resources needed to carry out an activity, nor the outcomes. Another proposal is the LAMS software, which is not LD-compliant, and which simplifies the learning designer’s tasks by providing a drag and drop mechanism for assembling a limited set of learning design components (flows of activities and resources in the environment). We believe that this approach is interesting, but not powerful enough to support the whole LD specification. The advantage of MOT+ models is that they allow the illustration of all levels of the LD specification, including a simple Method model as well as the details of each act, including environments with its resources, the role-parts and the rules.

The MOT+LD graphical editor enables designers to fully describe the structure and concepts inherent in Level A unit-of-learning and to produce an instance of a standard LD XML schema. Work is on-going to extend the editor to levels B and C. In Griffiths et 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 structures 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 time automatically producing an IMS-LD conformant XML manifest file derived from the graphs.

MOT+LD Graphic Vocabulary

Basically, all the MOT objects and links applicable to LD models 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 (rectangles), the LD method components (actions) are represented by seven kinds of procedures (ovals), whereas actors and rules are represented by five kinds of principles (hexagons). Individual objects are represented by clipped rectangles (called “facts” in MOT+) representing learning objectives and prerequisites, metadata, items, and four other types of objects needed to describe conference, send-mail and index-search services.

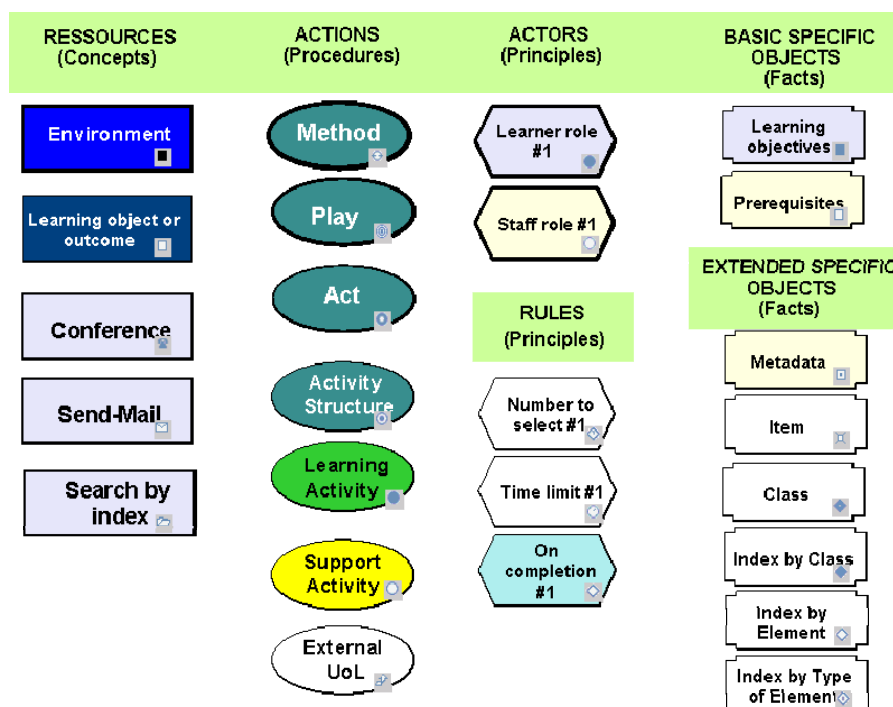


Figure 4 . MOT+LD basic vocabulary

The same basic links as in the general MOT language can be applied, however a number of new constraints on links between subtypes were added in order to comply to those specified in the IMS Information and Binding model and 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 composition link (C link). An environment is composed of other environments recursively or of other types of resources, such as learning objects, outcomes and/or services. Learner and staff roles, and also items can be organized in sets of component hierarchies. Methods are decomposed into plays, which are decomposed into acts, which are decomposed into role-parts, represented in our model by a role associated to an activity at any depth; finally terminal activity structures are decomposed into learning or support activities or a reference to an external unit of learning (UoL).

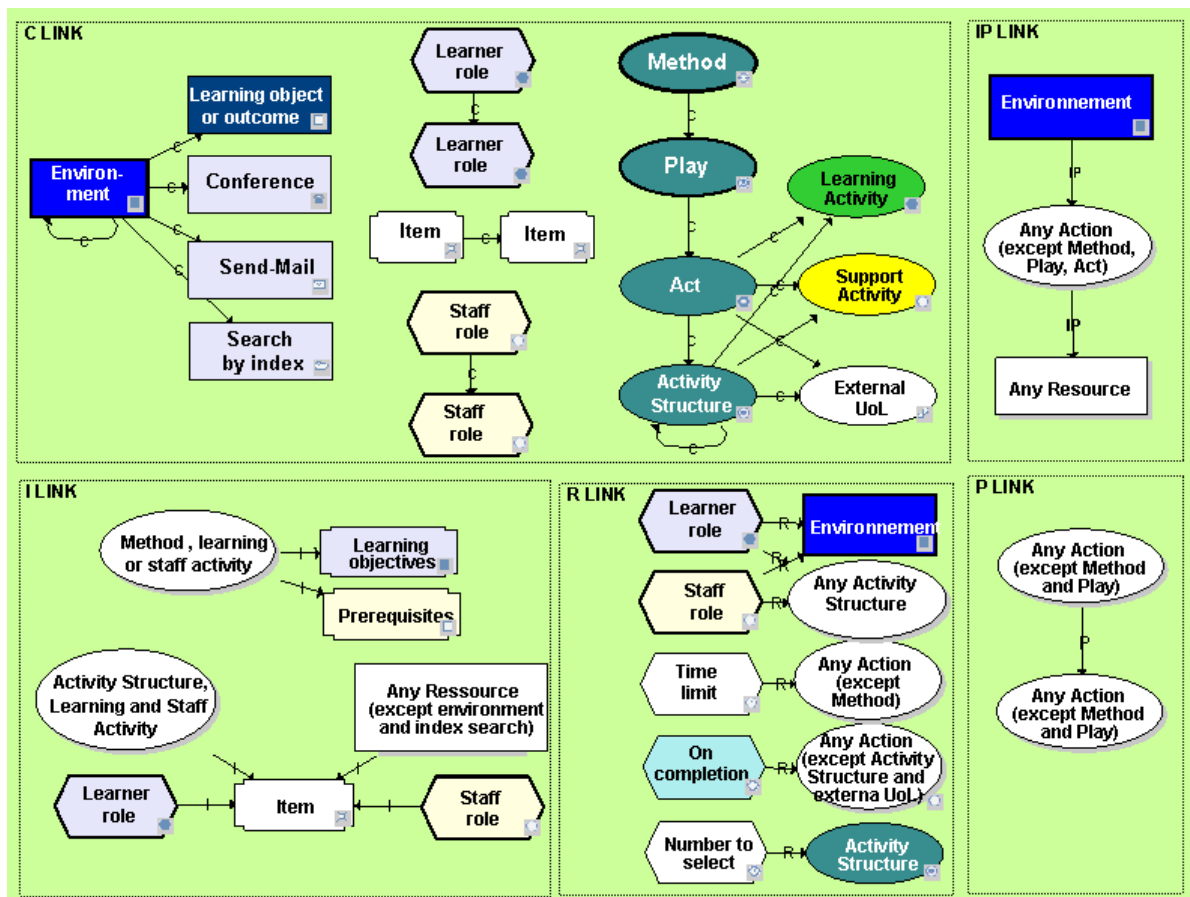


Figure 5. MOT+LD link constraints

The use of input/product (I/P-link) and precedence (P-link) links is clear and unambiguous. The precedence link is used between procedures only below the Play level, for example to show the order the acts are to be played. The I/P link is used only below the Act level, from an input resource to a procedure (LD Activity), that is to indicate resources in the environment of an activity, or conversely, from a procedure (LD Activity) 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 link (R-link) associates learner and staff roles to an environment or activity structures, learning or support activity, or it may associate a time limit to any action except the method. It is also used to associate a completion rule to an action except the activity structure and UoL. The number to select rule is R-linked to an activity structure when options are proposed.

Technically, to represent all IMS-LD concepts, subtypes of the original MOT+ object types as well as new graphical symbols with standardized labels (as shown in figures 4 and 5) were developed. The most difficult and time consuming part was to extend the native MOT XML schema and to parse it into a valid IMS-LD XML schema.

A post-validation mechanism was built into the parser informing the designer whether an IMS-LD rule has been violated and where to find it in the model. The number of possible violations was reduced while designing the model by limiting the choice of possible links between sub-types according to the constraints shown in figure 5. Finally, all the IMS-LD (IMS-LD, 2003) examples were modelled and tested, including the well-known and complex Versailles example (displayed in figure 6) by uploading them into the RELOAD editor (RELOAD, 2004), a form-based LD editor. This exercise resulted in very small discrepancies between our analysis of the

the UoL and its corresponding knowledge model. Finally, as discussed in the last section, target knowledge and competency statements help orient designers on what type of learning strategies and activity structures to select. It is well known that conceptual and procedural knowledge are not learnt in the same way, for example to acquire the competency to apply an administrative procedure is less demanding than acquiring the competency to build and adapt such procedures.

A couple of years ago, the MISA Instructional Engineering Method, its operations, products and principles were modelled 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 7 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 to 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.

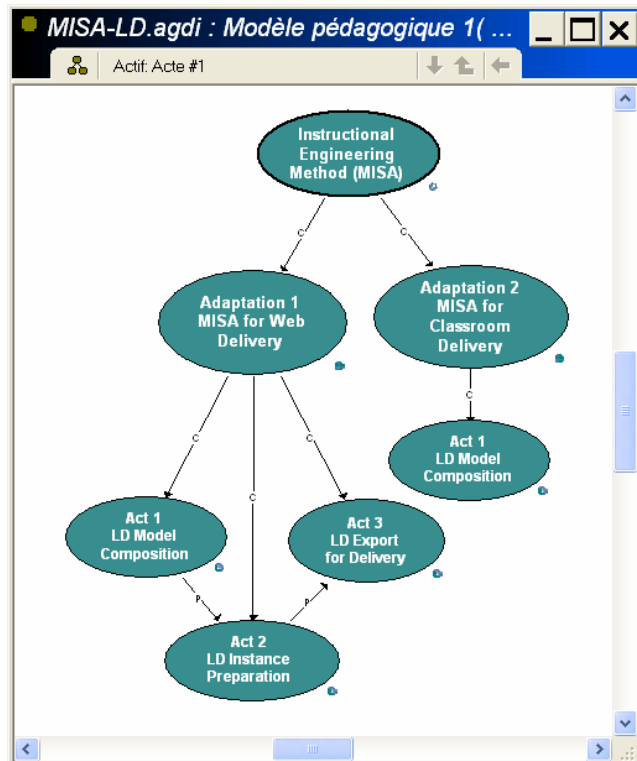


Figure 7. MISA as an LD (meta)-method

A general instructional engineering method like MISA can be adapted to many different situations. The preceding discussion opens the way to investigate a variety of ways to adapt MISA as an LD construction method described as alternate “design plays”.

Figure 8 shows a partial model of Act 1, where the main Activity Structure is called “MISA for Web delivery” including the role-parts for the designers as learners and IMS-LD facilitator as staff. The flow shows the design team’s preliminary analysis of training needs, target population, available resources, delivery and cost constraints, etc. followed by four processes, again modelled as activity structures, starting in parallel. These activity structures correspond to the design team’s role-parts for each of the content expert, the instructional designer, the media designer and the delivery specialist as Learner Roles. In figure 8, the designer role-part is derived from the R-linked Instructional Designer Role (hexagon) to the Instructional Modelling Activity (oval). The other role-parts are derived in a similar manner, although not developed here.

The instructional modelling activity structure corresponds directly to the engineering of the learning design. This activity is supported by a Staff Role where an IMS-LD facilitator coaches designers using an IMS-LD guide and an 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 (see Paloma LO Repository Manager <http://www.cogigraph.com>) and the RELOAD tool. Then knowledge units and competencies are associated to learning activities and to resources (using metadata).

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 (IMS-RDCEO,

2002). In Paquette & 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, and a way to represent them as procedural (meta-) knowledge models is explained. Further, we show how competency modelling can contribute to the guidance of the learning design engineering process.

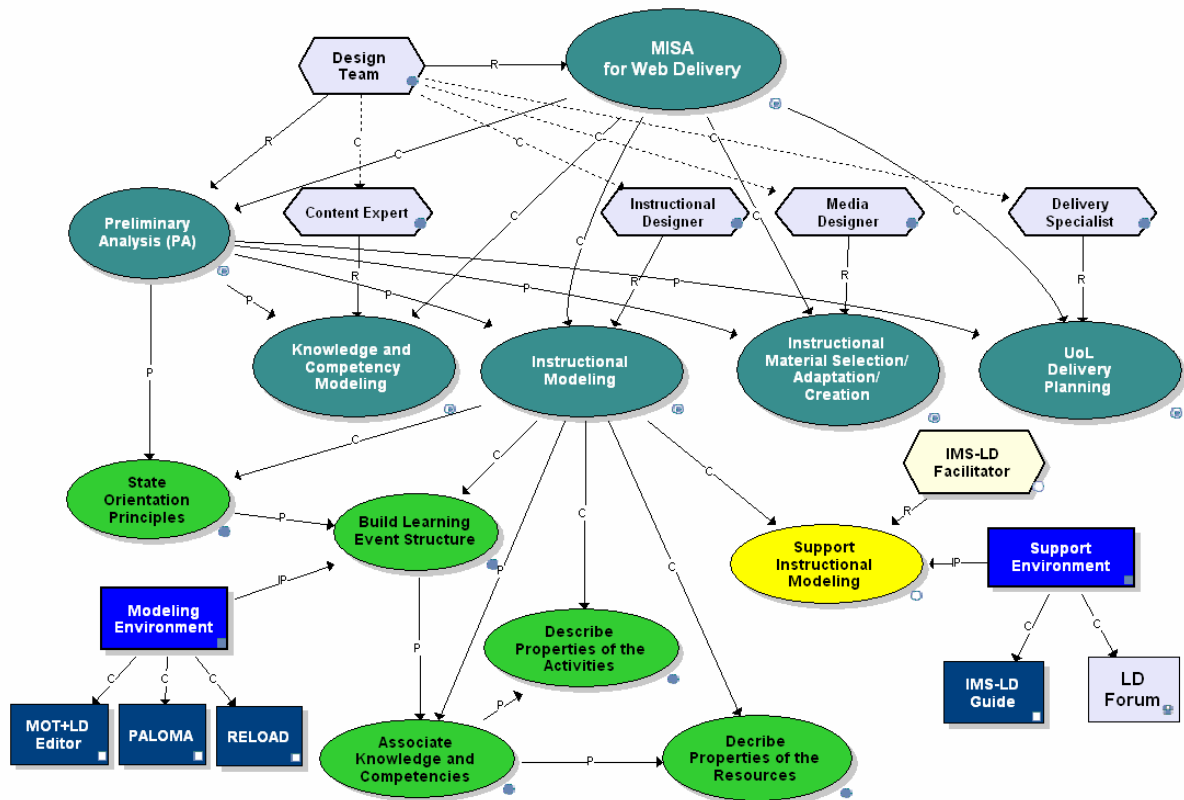


Figure 8. MISA for Web delivery Act 1 – Main activities

Competency: Cognitive Skills Applied to Knowledge

To say that a person knows something (prerequisite) or that a person must acquire some particular 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, transpose, analyze, synthesize, evaluate, self-control and so on, which can be applied in different knowledge domains. If more precision is needed, a degree of *performance* can be added by specifying the situational context where the cognitive skill is to 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 unit, or an actual state of the knowledge unit that someone possesses. 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 (Chandrasekaran, 1987; McDermott, 1988; Steel, 1990). For example, if a learner possesses a diagnostic or classification skill, it

implies that this learner is 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 & Briggs, 1986). Integrating all three viewpoints will enable us to provide a cognitive skill taxonomy that might prove useful in producing effective and efficient learning designs by identifying the gap between prerequisites (entry competencies) and learning objectives (target competencies).

Table 2. Taxonomies of Cognitive Skills

Cognitive Skills Taxonomy Levels			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 Explain	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, Modelling	Synthesize	
Re-invest	9. Evaluate		Knowledge Acquisition		Evaluate	Evaluation
	10. Self-manage	10.1 Influence 10.2 Self-control				Initiation, Continuation, Control

A Skill Taxonomy

Table 2 presents an overview of the proposed the skills taxonomy. This taxonomy combines and adapts an artificial intelligence taxonomy (Pitrat, 1990), a software engineering taxonomy (Breuker & Van de Velde, 1994; Schreiber 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, skills are here viewed as knowledge objects that can be described, analyzed and evaluated, by themselves or in relation to various knowledge domains.

The taxonomy shown in the left part of table 2 portrays three levels, from left to right, from the generic to the specific term. It could be expanded to more levels for additional precision. The first two levels 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.

Representation of a Cognitive Skill

Every cognitive skill in the taxonomy can be represented as a MOT process model by a main procedure in the meta-knowledge domain, which is the domain that categorizes knowledge and describes 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” skill, a sub-class of the level “5-Apply skill”, is compared to the level “8.3-Construct a process” skill, which is a sub-class of the “8-Synthesize” skill in the taxonomy.

Table 3. Comparison of two generic skills

Skill	Input	Product	Process Flow
Simulate a process	A <i>process</i> , its procedures, inputs, products and control principles.	A <i>trace</i> of the procedure : set of facts obtained through the application of the procedures in a particular case	<ul style="list-style-type: none"> - 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.	<ul style="list-style-type: none"> - 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.

From the descriptions of these two generic skills, we can easily see that a learning 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 demonstrations and exercises of the process will probably be sufficient, while in the second case, a project-based scenario where learners are 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 MOT+ graph on the left side in figure 9 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 from each exercise. On the graph, four groups of principles are added to constrain concepts and/or control procedures in the learn flow. Note that this model is totally generic, applicable to any specific knowledge domain, such as Internet processes, manufacturing processes, or others.

Figure 9 provides an example of 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.

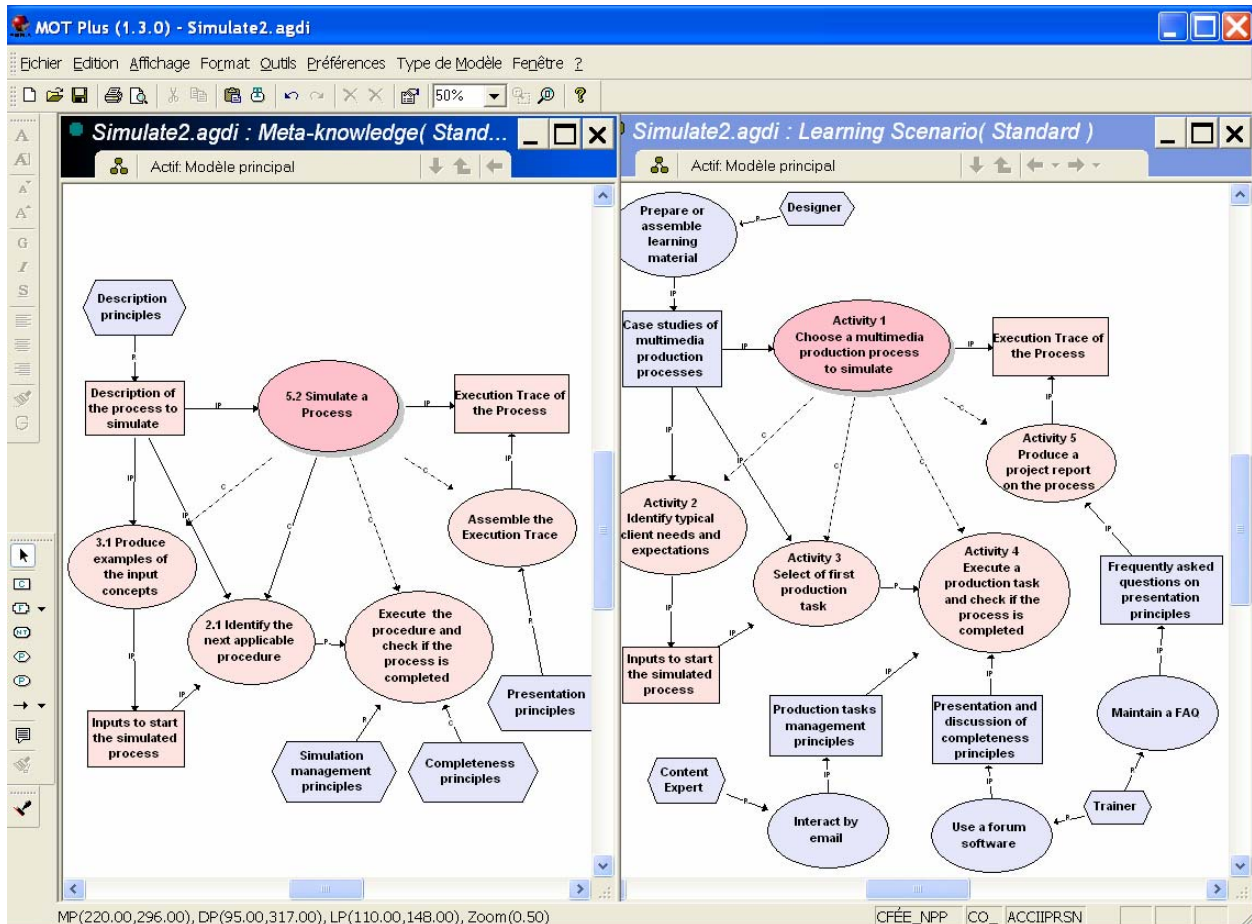


Figure 9. A learning scenario model simulating the “Search the Internet” process

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 request structure 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 model’s set 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 in its entirety.

But the important thing here is that the generic process becomes the founding principle for the learner’s assignments. In that way, it is possible to make sure that the learner exercises at the right skill level, 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 Modelling Languages and the IMS Learning Design specification (IMS-LD, 2003), 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.

We propose 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 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 competencies organized according to a skill taxonomy which 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 targeting 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 make 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 modelling 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

Ausubel, D. (1968). *Educational psychology: A cognitive view*, New York: Holt, Rinehart, & Winston.

- Bloom, B. S. (1975). *Taxonomy of Educational Objectives: the Classification of Educational Goals*, New York, USA: D. McKay.
- Breuker, J., & Van de Velde, W. (1994). *CommonKads Library for Expertise Modelling*, Amsterdam: IOS Press.
- Chandrasekaran, B. (1987). Towards a Functional Architecture for Intelligence Based on Generic Information Processing Tasks. *Paper presented at the IJCAI-87 Conference*, August 23-28, 1987, Milan, Italy.
- Dansereau, D. F. (1978). The development of a learning strategies curriculum. In O'Neil Jr, H. F. (Ed.), *Learning Strategies*, New York, USA: Academic Press, 1-29.
- Griffiths, D., Blat, J., Garcia, R., Vogten, H., & Kwong, K.-L. (2005). Learning Design Tools. In Koper, R. & Tattersall, C. (Eds.), *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*, Berlin: Springer, 109-136.
- IMS-RDCEO (2002). *IMS Reusable Definition of Competency or Educational Objective – Best Practice*, 1.0 Final Specification, IMS Global Learning Consortium, Inc., retrieved November 2, 2005, <http://www.imsglobal.org/competencies/index.html>.
- IMS-LD (2003). *IMS Learning Design. Information Model, Best Practice and Implementation Guide, Binding document, Schemas*, retrieved November 2, 2005, from <http://www.imsglobal.org/learningdesign/index.cfm>.
- IMS-SS (2003). *IMS Simple Sequencing*, 1.0 Final Specification, IMS Global Learning Consortium, Inc., retrieved November 2, 2005 from <http://www.imsglobal.org/simplesequencing/index.html>.
- Koper, R. (2005). An Introduction to Learning Design. in Koper, R. & Tattersall, C. (Eds.), *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*, Berlin: Springer, 3-20.
- Krathwohl, D. R., Bloom, B. S., & Masia, B. B. (1964). *Taxonomy of educational objectives: The classification of educational goals*, Handbook II: Affective domain. New York, USA: Longman.
- Martin, B. L., & Briggs L. (1986). *The Affective and Cognitive Domains: Integration for Instruction and Research*, New Jersey, USA: Educational Technology Publications.
- McDermott, J. (1988). Preliminary steps towards a taxonomy of problem-solving methods. In Marcus, S. (Ed.), *Automating Knowledge Acquisition for Expert Systems*, Boston, MA, USA: Kluwer Academic, 225-255.
- 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 Koper, R. & Tattersall, C. (Eds.), *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*, Berlin: Springer, 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 Koper, R. & Tattersall, C. (Eds.), *Learning Design – A Handbook on Modelling and Delivering Networked Education and Training*, Berlin: Springer, 311-326.
- Paquette, G., Marino, O., De la Teja, I., Lundgren-Cayrol, K., Léonard, M., & Contamines J. (2005). Implementation and Deployment of the IMS Learning Design Specification. *Canadian Journal of Learning Technologies*, retrieved October 15, 2005 from <http://www.cjlt.ca/content/vol31.2/paquette.html>.
- Paquette, G., & Rosca, I. (2004). An Ontology-based Referencing of Actors, Operations and Resources in eLearning Systems. *Paper presented at the 2nd International Workshop on Applications of Semantic Web Technologies for E-Learning*, August 23-26, 2004, Eindhoven, The Netherlands.
- Paquette, G. (2003). *Instructional Engineering for Network-Based Learning*, San Francisco: Pfeiffer/Wiley Publishing Co.
- Paquette, G. (2002). *Modélisation des connaissances et des compétences, un langage graphique pour concevoir et apprendre*, Québec, Canada: Presses de l'Université du Québec.
- Paquette, G. (1999). Meta-knowledge Representation for Learning Scenarios Engineering. In Lajoie, S. & Vivet, M. (Eds.), *Proceedings of AI-Ed'99 in AI and Education - Open learning environments*, Amsterdam: IOS Press, 29-37.
- Paquette, G. (1996). *La modélisation par objets typés: une méthode de représentation pour les systèmes d'apprentissage et d'aide à la tâche*, France: Sciences et techniques éducatives.
- Paquette, G., Crevier, F., & Aubin, C. (1994). ID Knowledge in a Course Design Workbench. *Educational Technology*, 34 (9), 50-57.

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

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

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

RELOAD (2004). *RELOAD Project*, retrieved November 2, 2005 from <http://www.reload.ac.uk>.

Romiszowski, A J. (1981). *Designing Instructional Systems*, London: Kogan Page.

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

Steels, L. (1990). Components of Expertise. *AI Magazine*, 11 (2), 29-49.