

Instructional Engineering for Learning Objects Repository Networks

Gilbert Paquette

▶ **To cite this version:**

Gilbert Paquette. Instructional Engineering for Learning Objects Repository Networks. CALIE04, International Conference on Computer Aided Learning in Engineering education, 2004, Grenoble, France. hal-00190193

HAL Id: hal-00190193
https://telearn.hal.science/hal-00190193

Submitted on 23 Nov 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTRUCTIONAL ENGINEERING FOR LEARNING OBJECTS REPOSITORIES NETWORKS

Gilbert Paquette (gpaquett@licef.teluq.uquebec.ca)
Centre de recherche CIRTA-LICEF, Télé-université
4750 avenue Henri-Julien, Montréal, Québec
http://licef.teluq.uquebec.ca/gp

KEYWORDS: Learning Object Repositories, Educational Modelling Languages, Instructional Engineering, eLearning Standards

Abstract

Knowledge management in organizations, the learning objects paradigm and the advent of a new Web generation, the "Semantic Web", are major actual trends that reveal a new potential for a renewed distance learning pedagogy, but at a certain number of conditions. The first and foremost is the use of education modeling languages and instructional engineering methods to help decide how to assemble learning objects in meaningful learning and knowledge management environment. This article proposes a set of tools and some Instructional Engineering principles to help use learning object repositories to create learning/training designs that respond to pedagogical needs.

THE CENTRAL ROLE OF LEARNING OBJECTS

The learning objects paradigm is deeply rooted in the evolution towards a knowledge society, which entails the need for lifelong learning and for more flexible, adaptive learning systems, inside and outside the public education system. More and more people are knowledge workers involved in knowledge management activities. This situation produces a huge demand for Web-based resources for work and learning and thus for the reusability and interoperability of digitized information materials.

Adopting knowledge management, an increasing number of education and workplace organizations have started to formalize their knowledge, re-engineering work and training processes using Internet technologies and, more recently, learning objects repositories accessible through the Web. These two trends converge and become integrated into yet another wider trend aiming at the next Web generation, the "Semantic Web " (Berners-Lee et al 2001). This starts by adding metadata to information resources (learning objects), thus representing the knowledge they embody, as a component of an ontology or a semantic description. The ultimate goal is to enable us to search and exploit Web resources in more intelligent ways.

Instructional engineering plays a central role in knowledge management and in the use of Web-based learning objects. Knowledge management aims at the identification and sharing of the available knowledge in an organization and at the increase of the competencies of their personnel. It is based on two main processes: the acquisition of knowledge from experts, and the construction of learning environments helping other persons to gain knowledge and skill through their use of learning objects.

EDUCATIONAL MODELING

The fast evolution of learning technologies has multiplied the number of decisions a designer must make to create a distributed learning system. The term “Educational Modeling Language (EML)” was first introduced in 1998 by researchers at the Dutch Open University. It came out of a strong preoccupation for Instructional Design and pedagogical concerns. The work on Educational Modeling Languages, and the its subsequent integration in the IMS Learning Design Specification (IMS-LD 2002a), is the most important initiative to date, to integrate Instructional Design preoccupations in the international eLearning Standards movement. In particular, it describes a formal way to represent the structure of a *unit of learning* and the concept of a pedagogical method that specifies the roles and the activities that learners and learning facilitators can play using learning objects repositories.

Educational Modeling Languages (EMLs)

The EML concept challenges the over importance devoted to learning objects seen solely as information packages. As Rob Koper (2001) puts it: “A lot of learning does not come from knowledge resources at all, but stems from the activities of learners solving problems, interacting with real devices, interacting in their social and work situation. A lot of research about learning processes provides evidence for this stance that learning doesn’t come form the provision of knowledge solely, but that it is the activities of the learners into the learning environment which are accountable for the learning.”

The emphasis on learning designs is also justified from a reusability perspective. For example, Michael Feldstein (2002) asserts that “content is harder to recycle than design”, that “recycling design can give bigger gains than recycling content” and, on the other hand, that “reusability (of learning objects) breaks some instructional designs”.

This approach also integrates well with the IEEE LTSC (2000) broad definition: “A learning object is any entity, digital or non-digital, that can be used, re-used, or referenced during technology-supported learning”. The unit of learning itself and all its components are embedded learning objects, including learning objectives, prerequisites, learners’ or trainers’ roles, activity assignment, information objects, communication objects, tools and questionnaire objects.

But identifying the learning objects associated to a learning unit and the interrelations between them is not sufficient from an technical perspective. The IMS-LD information model needs to be expressed in a standard XML binding enabling computer processing by any compliant eLearning system. It should then be possible for any Learning Content Management System (LCMS) to interpret and use the unit of study, reuse the learning objects composing the unit in new contexts, as well as adapt, distribute and archive units of learning and all the learning objects they contains.

A *unit of learning* refers to any delimited piece of education or training, such as a course, a module, a lesson. When activating a unit of learning, the *method* element is central. It is located within the unit of learning set of XML files. This central element and its sub-elements control the behavior of the unit of learning at runtime, coordinating the activities of the actors in the various roles they play and in their use of learning objects.

A method is composed of *plays* that provide alternative *scenarios* for the same unit of study, to adapt to different target populations or to different delivery models such as distance or classroom learning. Each play unfolds in a series of one or more *acts* which are always run in sequence. An act

brings together one or more *role-parts* , each role-part associating exactly one role (learner, trainer, tutor, manager, etc.) with exactly one activity, associated or not to a set of learning objects. At every level within a method, it is possible to specify rules when a role-part, act, play or unit-of-learning is completed.

Implementing IMS-LD within the eduSource infrastructure

The IMS-LD specification has been retained as a major eduSource component. The central goal of the eduSource project (www.edusource.ca) is to enable existing Canadian learning object repositories, or future repositories in Canada or elsewhere, to communicate seamlessly so that their learning objects can be found and aggregated in units of learning.

The eduSource software system is based on a Web service architecture composed of an *eduSource communication kernel.* This infrastructure comprises the eduSource Communication Language connector and gateway, the eduSource Registry of Services and the Common Search Component enabling Federated, Harvesting or Distributed Peer-to-peer (PtoP). It processes searches for learning objects and passes its results to a display component that presents the metadata to the user and launchs the corresponding learning object. On top of this infrastructure, an extendable *eduSource suite of tool* enables operations on the metadata (reference, search, view), on digital rights and on the learning objects (transfer, aggregate, design, package), providing support to learning design.

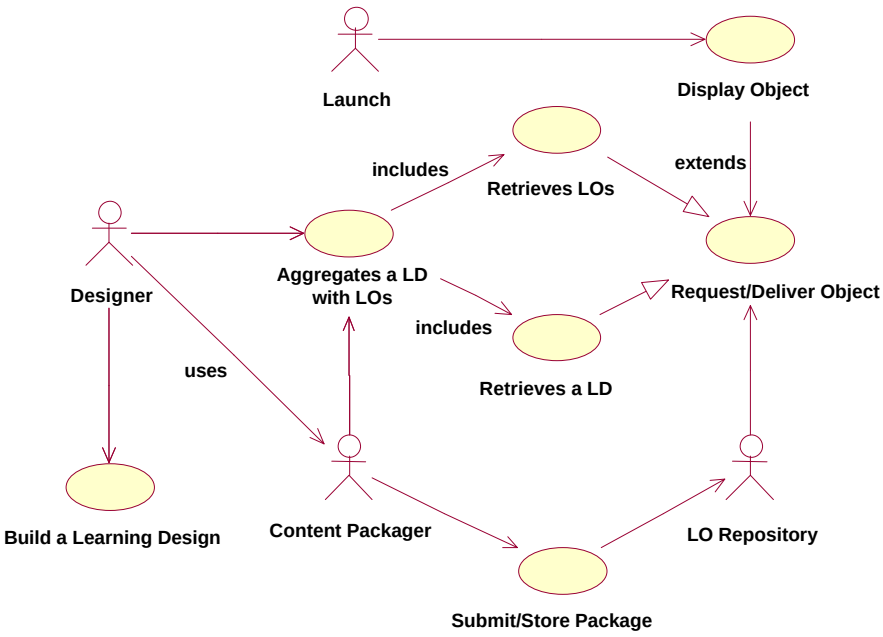


Figure 1 – Learning Design Use Cases in the eduSource project

The UML use case diagram on figure 1 shows the Designer, as the main actor involved in building a learning design. Basically, he/she aggregates a LD structure with content learning objects, for example, associating one or more resource or object with the learning activities where they are used or produced. The designer has to retrieve a LD and a number of content LOs from learning objects

repositories, by making a request/deliver (search) operation. This operation will be extended by a display-object operation.

If we go deeper in the analysis, we see that, the designer controls four main use cases.

- He/she creates a new Learning Design with the help of a LD Editor. To complete his design, the Designer has to aggregate the LD structure with learning objects holding content retrieved from one or more LO repositories.
- The Designer can also modify an existing LD using the LD Editor by first retrieving a LD-XML file, a LD manifest or LD content package from a LO repository.
- When a LD is created or modified, the Designer has the option to publish either the XML file, the LD content package manifest with pointers to the resources, or a complete content packaging zip file containing also the learning objects with the LD file. In the same way as the publication of other learning objects, this is done, in all three cases, by adding metadata for the LD-XML file, the LD-manifest or the LD-CP, in a Metadata Repository so that it becomes accessible through the eduSource network as a new learning object.
- Finally, the Designer will test the result of his/her design using a LD enabled viewer or player. Such a tool can be a simple standalone XML/XSLT viewer that can display an LD-XML file, or de-package a LD manifest or a LD content package. It can also be a complete LD player with dynamic LD viewing, embedded in an LD-aware LCMS.

A Learning Design Set of Tools

The preceding discussion leads us to define a possible set of tools, the first being *an IMS-LD editor*. Such a tool enables a Designer to build the specification of a Unit of learning. It must produce an XML file structured as defined by the IMS-LD XML binding document (IMS-LD 2002b). There are many ways to provide a user interface for LD edition. It is possible to use a Web-based form editor, a tree editor or a graphic editor.

- Figure 2a is extracted from the IMS-LD document. It shows the possiblity of a Web-based dynamic form editor. Of course, functionalities would have to be added to associate resources to the activities. Also, it has to be generalized to enable the definition embedded activity-structure, potentially at any number of levels.
- Figure 2b is a part of the Explor@-2 Activity Editor (Paquette 2001). It presents a method identical to the one in figure 2a but in a tree form to facilitate the embedding at any depth. Actually, this editor has the possibility to associate resources obtained from a search in multiple learning objects repositories, and also to add other properties to activities. It is possible to define separate activity structures (scenarios) for any number of actors, but Explor@-2 actually lacks functionalities to process multi-actor activities in the same scenario such as proposed by IMS-LD.
- Figure 2c is produced using the MOT+ knowledge editor (Paquette 1999), a mature and general graphic tool used by many organizations. It presents in a graphic way a method identical to the one in figure 2a. The MOT+ Editor has functionalities to embed activity structures as sub-models at any number of levels. It can associate resources to activities as OLE attachments. A

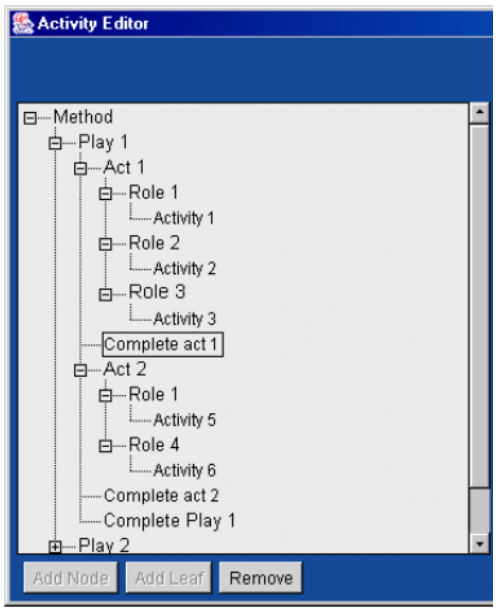
MOT+ translator module produces an XML file that has a different format than the IMS-LD XML binding.

Method

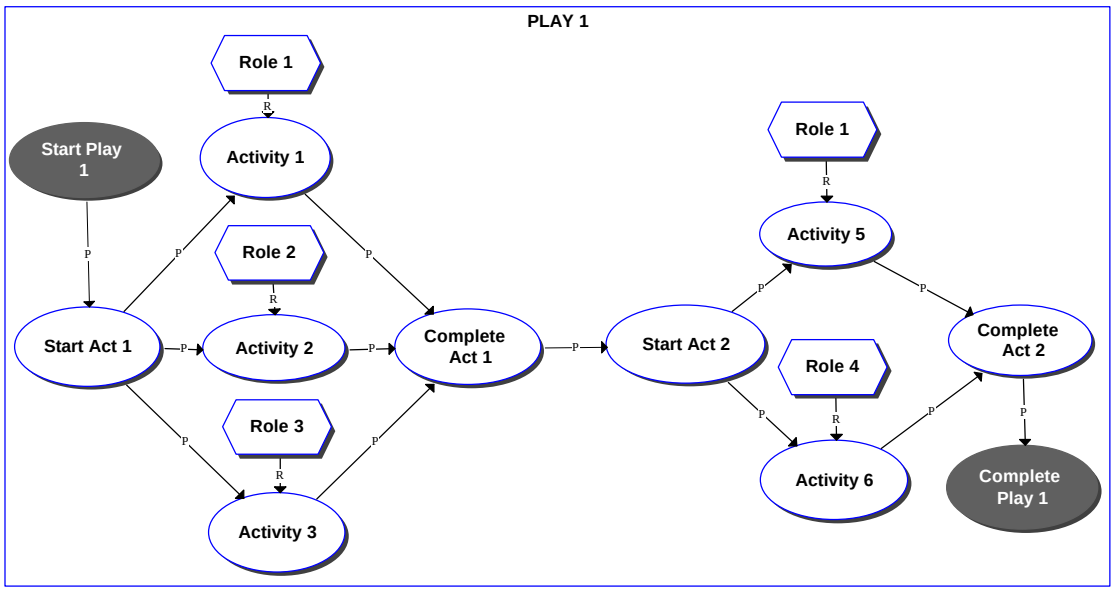
Play 1	Act 1	Role 1	Activity 1
		Role 2	Activity 2
		Role 3	Activity 3
	complete act requirements		
	Act 2	Role 1	Activity 5
		Role 4	Activity 6
	complete act requirements		
complete play requirements			
Play 2	Act 3	Role 1	Activity 9
		Role 3	Activity 10
		Role 4	Activity 11
	complete act requirements		
	Act 4	Role 1	Activity 3
		Role 2	Activity 1
		Role 3	Activity 2
	complete act requirements		
complete play requirements			

complete method (unit of learning) requirements

2a- A form-based LD editing interface



2b- A tree-based LD editing interface



2c- A graphic LD editing interface

All these three possibilities are interesting, but they require quite different development efforts starting from our actual set of tools at CIRTA-LICEF or in the eduSource consortium.

The Web-form Editor would probably require the most work because we have to start from scratch and build embedding and resource association capacities. A tree Editor could be more rapidly produced from the Explor@-2 activity editor. This editor's code could be the basis of a new version facilitating IMS-LD edition. Actually, this Editor's output is stored in a relational SQL-2000

database that also generates XML files. An analysis has to be made whether it is simpler to translate this XML in IMS-LD format, or to directly generate IMS-LD XML format. The later is most probably the best choice.

The Graphic MOT+ Editor needs to be constrained to standardize the form of the graphs.. Since the corresponding XML file exported by MOT+ is not in IMS-LD format, we would also need to build a parser to translate this XML output to the IMS-LD XML format. Because of its ease of use by instructional designers and content expert and the short time frame of the eduSource project, the MOT+ graphic editor has been retained as the basis for a LD Editor in the eduSource project. Other forms of graphic editors might be considered in the future

Packaging and De-packaging for a LD Viewer and Player

It is interesting to separate the editing tool from the content packaging tool because a designer might want to store only the LD-XML file in a repository for further reuse, without the associated resources or resource pointers. A LD content-packaging tool should offer the choice to either store this file, to store it with pointers to associated resources (the manifest of a content package) or together with the associated resources (the zipped content package). Since some of the eduSource teams have already build non LD content packagers, it should be straightforward to adapt one of them to a LD content packaging tool with the proper functionalities.

Conversely, once a LD content package is retrieved, there are many ways for a designer to see its content through a specialized *LD viewer*. Such a viewer can be seen as a part of the content packager, as a preview function that displays the LD-method as a tree structure, enabling the user to select a tree node and display any associated resource.

An alternative to such a package viewer is of course to use a *LD-aware LCMS player* that can deliver the unit of learning to learners and staff, with all its associations to learning objects to be displayed or launched. This requires that the LCMS be made LD aware. We need to experiment how this can be done and the Explor@-2 system is a good candidate for this research, since it already possesses the ability to define any number of actor's environments.

Actually, in Explor@-2, each actor or role has its own activity structure (which is not multi-role) and its own resource environment, so additional functionalities will have be built to exploit the new multi-actor capabilities proposed by the IMS-LD specification. Basically Explor@ would read, as an option to its actual functionalities, an IMS-LD package in an activity editor such as the one presented on figure 2b. It could also import a graphic multi-actor function model, built with the MOT+ editor, such as the one presented in Paquette and Rosca (2002).

INSTRUCTIONAL ENGINEERING METHODOLOGY

We have proposed a new approach to Instructional Design, founded on cognitive science labeled as Instructional Engineering (Paquette 2003). It is defined as *a method that supports the analysis, the design and the delivery planning of a learning system, integrating the concepts, the processes and the principles of instructional design, software engineering and cognitive engineering.*

Today, it seems necessary to renew the instructional design methodology to support the creation of distributed learning environments. The instructional design models and theories have been built

since the sixties on solid foundations, and they present an impressive body of work. The problem lies elsewhere, at the level of the operationalization of these foundations in the new technological and pedagogical context.

In an instructional engineering method, the modeling processes are at the forefront. They help designers define content and objectives, instructional scenarios, instructional materials, as well as the delivery processes of a learning system. Figure 3 presents the main tasks in the MISA instructional engineering method developed at CIRTA-LICEF since 1992. After the preliminary problem definition phase, the other tasks are grouped around four axis: knowledge modeling, instructional modeling, materials (or learning objects) modeling, delivery modeling. In each axis, the modeling tasks (in italic) are preceded by orientation principles and followed by the description of the properties of the components of the models. The scope covered by educational modeling languages such as IMS-LD corresponds well to the Instructional Modeling processes in MISA.

Problem definition		
100 Organization's training system	102 Training objectives 104 Target populations	106 Actual situation 108 Reference documents

Knowledge Modeling	Instructional Modeling
210 Knowledge modeling principles	220 Instructional principles
212 Knowledge model	222 Learning events network
214 Target competencies	224 Learning units properties
310 Learning unit content	320 Instructional scenarios
410 Learning instrument content	322 Learning activities properties
610 Knowledge and comptency management	420 Learning instruments properties
	620 Actors and group management

Learning Materials Modeling	Delivery Modeling
230 Media development principles	240 Delivery principles
330 Development infrastructure	242 Cost-benefit analysis
430 Learning mateirials list	340 Delivery planning
432 Learning materials models	440 Delivery models
434 Media elements	442 Actors and user's materials
436 Source documents	444 Tools and telecommunication
630 Learning system and ressource management	446 Services and delivery locations
	540 Assessment planning
	542 Revision decisions log
	640 Maintenance and quality management

Figure 3 – Main Tasks of the MISA Instructional Engineering Method

Instructional Engineering and Education Modeling Languages have much in common. They put the main emphasis on pedagogy and Instructional Design. They share a software engineering approach; EML being represented using the UML software modeling methodology, while MISA uses the MOT representation system.

Most of the differences between the two approaches lie in their position in the design-development-delivery process. MISA is a design method similar to software engineering in the field of computer science. It describes processes and that help define design elements for any kind of Learning System. Thus, a IMS-LD unit of learning is one of the possible outputs of a method like MISA. Conversely, IMS-LD is not a method but a standard specification for the products resulting from the application of an instructional engineering method such as MISA (or another one). It bridges the gap between design tools and the delivery of instruction through a LCMS.

INSTRUCTIONAL ENGINEERING PRINCIPLES

The preceding discussion shows the importance of that good instructional engineering for the quality of learning designs and learning objects aggregated in an IMS-LD unit of learning. We now turn to some of the necessary conditions to enhance the quality of Web-based learning by the use of learning object repositories. We present here a summary of 18 instructional engineering principles for that purpose.

Self-Management and Metacognition

Self-management interactions are where the learner has to analyze what she knows, should know or want to know, building a model of her own knowledge and skills. Here, the structure of the learning design and activities is critical to enable learners to self-manage their activities and trigger metacognitive processes that are essential to learning.

1. *The knowledge model in a learning unit must be of sufficient grain size and structured using precise links***.** A learning unit should group a sufficient number of interrelated knowledge units: not a single small concept, but a concept with its main components and the procedures where it is used; not a single small procedure but also its inputs, products and control principles; not a single principle, but principles linked to procedures they control or concepts they define.
2. *Knowledge in a learning unit must be related to skills in target competencies.* Knowledge specific to a domain and generic skills are being constructed at the same time (Romisowski 81, Pitrat 91). They are what define human competency. A learning unit with just knowledge and no associated generic skill is like a set of data without any cognitive process acting on it. Such an association in a target competency brings dynamic and deeper interaction of the learner with the knowledge in a unit of learning. The explicit association of a skill to a set of knowledge units is also essential to orient the choice of the learning and evaluation methods.
3. *The learning scenarios or plays should be described as a generic process, corresponding to target competencies.* The learning design must propose problem to solve, tasks or projects to achieve, instead of static knowledge to contemplate. In other words, it must involve the learner in information processing activities directly related to the generic skills in a target competency. For example, if we want to develop generic skills like classification, diagnosis, induction or modeling, we should propose classification, diagnosis, induction and modeling problems or projects to the learner.
4. *Learning designs must be open to alternative learning paths. The information process in plays should not be too linear or detailed to a point where the learner cannot make any choice.* If the sequencing is too tightly designed, the learner will not have a chance to think about his learning path and strategy. For instance, the activities can be structured in the form of a network where the learner can follow different paths. Then, an important interaction will consist in the learner managing the sequence of his activities within the design.
5. *Self-management tools and resources must be provided.* Open learning designs encourage the learner to evaluate regularly his progress, to self-diagnose his results and sometimes, to radically redefine his strategies. This metacognitive activity is a difficult task. Tools like reports on progress in the activities, success in knowledge acquisition, time and task management are essential and must be designed in a way not to interfere with the learning activities.

Information Processing

Lets us now examine the interactions of a learner, with the information resources (or learning objects), learning materials or on-line content experts. This type of interaction iw where the learner, through his processing of information, transforms it into internalized knowledge or, conversely communicate information to others from his own knowledge.

6. *Learning designs must propose rich and diversified information sources.* It is important that the learner be offered a sufficient diversity of information sources to be able to choose the information most related to the work to be done in the learning activities. This implies enough resources with a certain level of redundancy and a variety of media. A good way is to put the learner in command of search operations in learning object repositories.
7. *Information resources must include bi-directional communication tools or services.* A good example, besides forums and chats is an asynchronous on-line service called "frequently ask questions", generally managed by one or more content experts.
8. *Learning designs must define clear information goals.* The huge number of available information sources can demobilize a learner. To counter-act this "lost in space" effect, it is essential that the learner has a clear view of the information he/she is looking for, either in learning activity assignments, by a non-restrictive list of information sources or through interactive help or advice agents.
9. *Learning designs must offer tools for information search, annotation, and structuring.* Such tools can be associated to activities where they will be particularly useful. Search engines help the learner filter the right information according to the goals of the search. Annotation tools help complement and assess the usefulness of learning objects. Information structuring tools such as conceptual maps or modeling editors can synthesize the information gathered from different locations.
10. *Learning designs should offer production tools well adapted to the task involved by each learning activity.* The most crucial part of the learner's interaction with information is the one where he/she processes the information to build a product of a learning activity. Text editors, spreadsheets, presentation, planning or modeling tools are example of these. The choice of tools associated to a learning unit depends very much on the main skill, the generic process on which the activity is based. For example, a planning process will necessitate a spreadsheet or a project management tool, while a taxonomy construction will make good use of a graphic editor.

Collaboration Between Learners

The interaction of a learner with his/her co-learners adds an essential collaborative dimension to a Telelearning system. From a multi-agent perspective, these interactions concern essentially the management of different agents in a learning activity, the distribution of responsibility between them and their coordination.

11. *In a learning design, collaborative and individual activities must sustain and build on one another.* Learning is social as well as an individual process. Through collaboration communication with equals, learners can express and clarify their evolving ideas, validate their emerging hypotheses and enrich mutually their growing solutions, models and theories. These collaborative interactions will be richer if they are well prepared by each participant, and if they

are extended by other activities where each learner investigates more thoroughly the group's conclusions.

12. *The collaboration modalities must be adapted to the generic process that characterizes a learning unit.* The mode of team interaction and coordination depends strongly on the generic process retained as a basis for a play in a learning design. For example a generic process like law induction can start by a collaborative activity where teams of learners gather parts of the data, followed by a synthesis by the whole group. In general collaborative problem-solving activity should match the general resolution process: it could start by a group analysis of the problem statement, followed by a brainstorming for possible solutions. Afterwards, each proposed solution could be evaluated by different teams and a group discussion would finally helps sort the adequate solutions to the problem

13. *Collaboration must use well-coordinated synchronous and asynchronous interactions.* Asynchronous interaction between learners should be the default mode, punctuated by a certain number of synchronous meetings at strategic moments. Real time interaction, in presence or at a distance is useful to start a collaboration process and, from time to time, to manage the learning activities or to exchange ideas using all the sensorial attributes of a face-to-face meeting. However, synchronous interactions consume a lot of time during which many participants can become passive or unmotivated. More fundamentally, very few are able to engage in the interactions with the very rapid pace and the limited amount of time given to analyze the others' statements. So the learning designs should assign the largest amount of time to asynchronous activities where the learners communicate with each another over a longer period of time, when they are ready for interaction, using different tools like email, forums, audio and video mail.

14. *The collaboration model should include management activities and tools for team coordination.* Fruitful interaction between learners cannot occur without well-defined management activities and tools such as group agendas, communication software, groupware, individual and collaborative work plan viewers or self-presentation materials made available to all learners. Assistance from the facilitators must also be well planned in the system to support collaboration. For example, a trainer acting as a group discussion animator, in synchronous as well as in asynchronous meetings, is absolutely essential for meaningful interaction. A manager can also help learners form subgroups and synchronize their individual and collaborative activities.

Assistance from Facilitators

Facilitators are special resources or learning objects providing help and assistance to the learning process. They can be human tutors, trainers, animators, coaches, or digitized help files, advisor agents or tutoring components integrated in the learning design.

15. *Assistance interactions in a learning design must correspond to principles regulating the corresponding generic process.* The assistance resources should be based on the principles regulating the generic skill process on which a learning design is constructed. For example, if the design proposes a diagnosis project, facilitators should be provided at first to assist the learner in the analysis of the component structure of the faulty system under scrutiny; later on, other facilitators can help systematize the evaluation of all possible failures in the system. As another example, if the design proposes a planning activity, facilitators could first help learners identify the goals and constraints of the plan.

16. *Provide assistance scenarios with multiple facilitators.* It can be counter-productive to rely on a single assistance mode. For example, a unique human trainer can become omnipresent or not well synchronized to the needs of certain learners. At the other extreme, a completely computerized assistance system can frustrate some learners who will not necessarily find the level of help they need. A combination of assistance from human facilitators with help from the computerized environment is preferable in most cases. The help or intelligent advisor system can offer a first level assistance readily available that can lead to deeper, but less frequent, interaction with a human trainer or manager. In a distance learning system, it is often fruitful to give different roles to human tutors working in teams.

17. *Assistance should be given carefully, mainly at the learner's initiative.* In the planning of a learning design, it is important to design the assistance sub-system in a way that the facilitators will not constantly disrupt the learning activity. A facilitator should intervene, most of the time, when being requested by the learner. But also, a set of intervention principles should trigger the facilitator's interventions, to help identify extreme cases where the learner must be supported.

18. *The type of guidance from the assistance system should be mainly heuristic and methodological.* An algorithmic guidance tries to detect every possible move that could lead the learner to error, thus preventing the learner from making him/her own choices. At the other extreme, a heuristic form of guidance orients the facilitator's intervention towards suggestions to the learner in terms of generic methods instead of solutions clues. Typically, the human trainer or the intelligent advisor system will suggest to build tables or graphs, decompose the problem or, more generally, propose an analysis process based on his knowledge of the generic process on which the learning scenarios are based.

CONCLUSION

The future solution of the major interoperability technical problems, through the implementation of international eLearning standards, will shift the focus from media development to instructional engineering and pedagogical concerns. The greater availability of reusable digitized content, together with the larger set of instructional decisions to be made by instructional designers, will hopefully push forward the agenda for innovative instructional engineering methods and tools. The goal of this contribution is to serve that purpose.

REFERENCES

BERNERS-LEE ET AL (2000) Berners-Lee, T., Hendler J., Lassila,O. *The Semantic Web*. Scientific American, May 2001, Feature article.

IEEE-LSTC (2001) Standard for Information Technology - Education and Training Systems - LOs and Metadata. LTSC

IMS-LD (2002a) *IMS Learning Design Information Model*.
http://www.imsglobal.org/specificationdownload.cfm Last consulted, January 2003.

IMS-LD (2002b) *IMS Learning Design XML Binding*
http://www.imsglobal.org/specificationdownload.cfm Last consulted, January 2003.

FELDSTEIN (2002) Standard for Information Technology - Education and Training Systems - LOs and Metadata. LTSC

KOPER (2001]) Koper R. *Modeling units of study from a pedagogical perspective – The pedagogical metamodel behind EML*
http://www.eml.ou.nl/introduction/articles.htm Last consulted, March 2002.

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

PAQUETTE (2001) Paquette G. *Designing Virtual Learning Centers.* In H. Adelsberger, B. Collis, J. Pawlowski (Eds) Handbook on Information Technologies for Education & Training within the Springer-Verlag series "International Handbook on Information Systems", pp. 249-272

PAQUETTE AND ROSCA (2002) G. Paquette and I.Rosca. Organic Aggregation of Knowledge Objects in Educational Systems, Canadian Journal of Learning Technologies, Volume 28-3, Fall 2002, pp. 11-26

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

PITRAT (1991) Pitrat, J. *Instructional Enginnering for Network-based Learning*. Pfeiffer/Wiley Publishing Co, 257 pages.

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