A Visual Ontology-Driven LD Editor and Player: Application to the Planet Game Case Study

Gilbert Paquette and Michel Léonard

LICEF Research Center, CICE
Research Chair, Télé-Université
100 rue Sherbrooke Ouest,
Montréal
Québec, Canada
www.licef.teluq.uquebec.ca/gp

Abstract: This article first summarizes our previous work on Educational Modelling and Visual Editors, to provide the context for our more recent work within the LORNET [1] research network that has led to the development of TELOS, an ontology-based system to support the design, development and delivery of Semantic Web learning environments. Within that system, a central piece is the Scenario Editor that enables the aggregation of resources (actors, activities, operations, documents, tools) into a visual multi-actor workflow/learnflow, which is the central model defining the environment. Another piece is the Ontology Editor that is used to associate execution and domain knowledge semantics to entities in the scenario. Using these tools, a new version of the Planet Game scenario has been build as a case study including an IMS-LD design at level A, B, C. We will analyze the case study to underline how it addresses some difficulties in the use of the IMS-LD specification. Finally, we will discuss the generality of this ontology-driven approach and this contribution to the field of educational modelling.

Keywords: Educational Modelling, Visual Scenarios, Instructional Engineering, IMS-LD specification, Ontology-Driven Tools and Systems.

1 Visual Learning Design

Our work on Educational Modelling Languages has started with the design of an instructional design support system called AGD (Paquette, Crevier and Aubin, 1994). From it, an instructional engineering method and some visual scenario (learning design) modelling tools MOT and MOT+ (Paquette 1996, 2003) were built based on large consensus in education and applied cognitive science (Merrill 1994, Romiszowski 1981, Tennyson and Rasch 1988, and West, Palmer and Wolff 1991). In the last two years we have moved to a another stage with the TELOS Scenario Editor that will be presented here, coupled with a player for IMS-LD scenarios at the three levels of the specification and an Ontology Editor to associate semantic references to scenario entities.

Building pedagogically useful learning scenarios is a difficult task per se. Using a complex specification like IMS-LD adds new difficulties to the average professor, trainer or instructional designer. Our interest in Visual Design has stemmed from our field experience in many learning contexts and projects. We soon found out that a visual modelling language was needed and would be closer to instructional designer's needs, than text-based XML editors or form-based editor like RELOAD [2] or software engineering visual languages like UML that have been proposed in the initial IMS-LD specification.

Our visual language and tools have evolved to the MOT+LD specialized visual language that we have presented in conferences two years ago (Paquette, Léonard, Lundgren-Cayrol, Mihaila and Gareau, 2006). MOT+LD uses a set of graphical symbols covering all the IMS-LD primitives at level A. Most of these primitives are shown in figure 1, together with Act 2 of our initial version of a Planet Game learning unit, presented in figure 2.

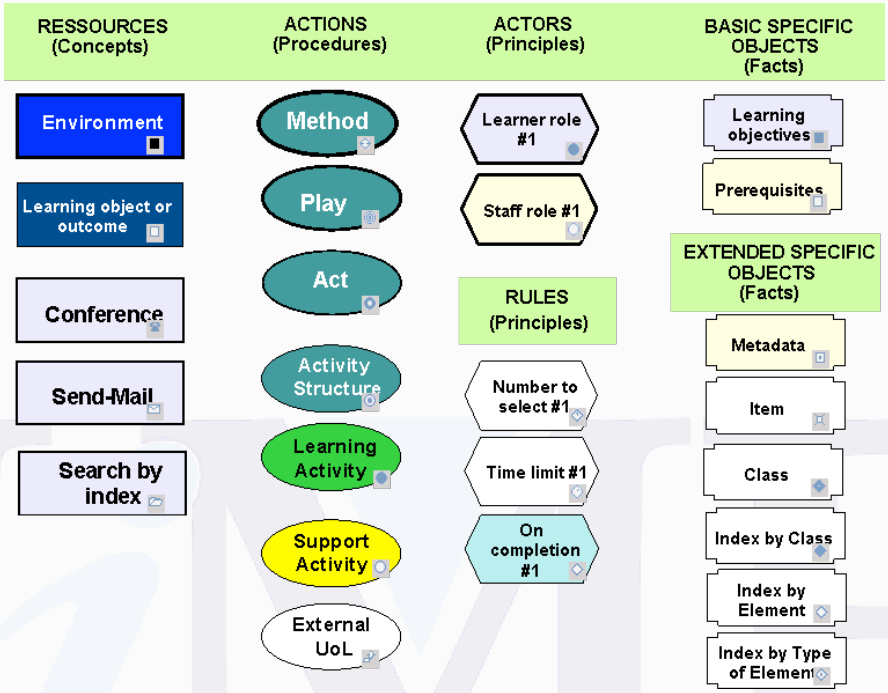

Figure 1: The MOT+LD Basic Set of Graphic Primitives.

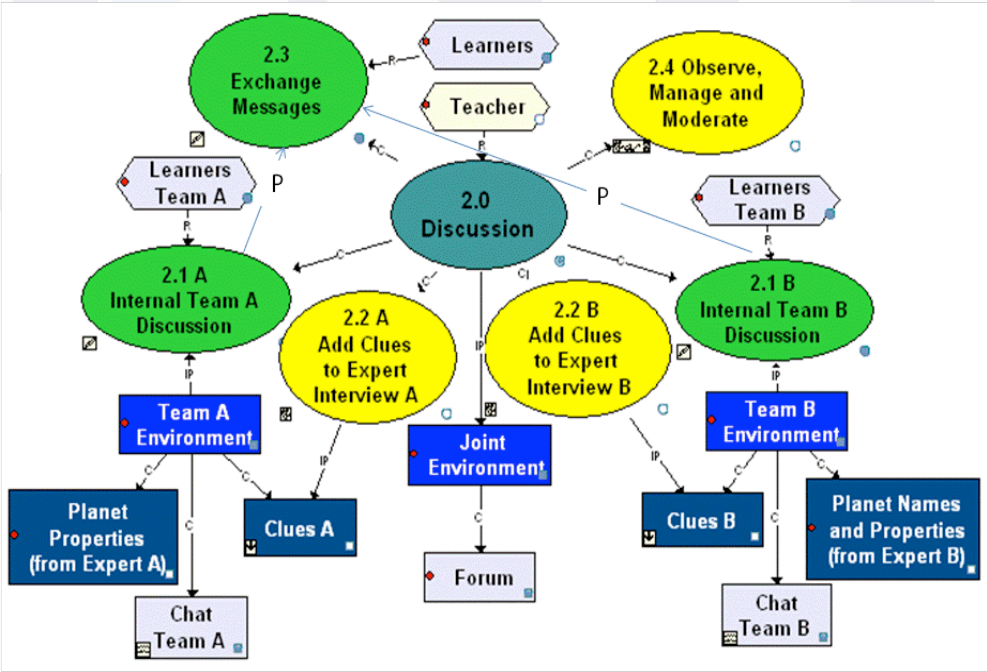

Figure 2: The MOT+LD version of Act 2 of the Planet Game Unit-of-learning.

This second figure has the following meaning, Act 2 is a discussion on planet properties managed or ruled (*R link*) by a teacher. This is composed (*C link*) of parallel learning activities 2.1.A and 2.1.B where Learners in team A and B both use (*IP links*) a different environment composed of a chat tools and different documents to undergo a team discussion to discover planet properties. These activities are supported by staff activities 2.2.A and2.2.B performed by the teacher where he observes both team discussions and provides to the participants, if needed, additional information (Clues A and Clues B). When these activities end, the flow moves to Activity 2.3 where all the learners in both teams exchange messages to share their discovery of planet properties uncovered in team discussion. This activity uses a forum managed and moderated by the Teacher.

Note that the control flow between activities is not shown totally on the figure. The sequencing of the activities is shown in some cases by precedence (*P links*) between 2.1.A and 2.1.B activities to 2.3. Other precedence links are there but are not visible on the figure to improve readability. Still events that trigger an activity stop are not present here because some are level B conditions. When there are no precedence links, it is considered that the activities run in parallel, which is the case for the set 2.2.A, and 2.2.B, and also for 2.3 and 2.4.

We were able to execute that scenario by first exporting the visual model to an XML file compliant with IMS-LD. This file was then read into the IMS-LD RELOAD editor to edit the missing level B and C conditions, and the result was executed by the RELOAD Player.

The usability of the MOT+LD Visual Editor at a conceptual level was further validated by the development of a learning design repository by designers of four different universities. Around 50 IMS-LD scenarios can be found on the IDLD site [3]. This project has shown, that even with a more user-friendly visual editor such as MOT+LD, creating IMS-LD scenarios is not that easy, even for experienced designers. All of them succeeded with little help, even though they were not initially familiar with IMS-LD, but they reported improvements to be made to the Editor (Lundgren-Cayrol, Marino, Paquette, Léonard and De la Teja, 2006). Three tools, MOT+LD, RELOAD and PALOMA for LOM referencing, had to be used which added some more complexity.

Before extending the MOT+LD editor to level B and C, we decided to revise the computer basis of the design environment to increase its usability and friendliness, and also its generality and efficiency. The new TELOS environment is Ontology-Driven so it embodies a metadata referencing scheme for scenario entities and scenarios as well. The new Scenario Editor covers all levels of the IMS-LD specification using visual symbols and links. The resulting scenario can be exported to an IMS-LD file or executed within the TELOS environment. TELOS includes a player that provides a visual tool, the task manager, to enables participants to realize their activities and interact with each others and the environment. The later is produced automatically from the scenario visual design.

In the next section, we will briefly present these TELOS tools, the visual symbols used in the Scenario Editor, and the way they are semantically referenced. In section 3, we will present the case study on the Solar Astronomy environment using these TELOS tools. In section 4, we will analyze this case study to discuss how it solves difficulties encountered with the previous editor and some aspects of the specification. In section 5, we will address the generality issue and propose some improvements to the specification.

2 Ontology-Driven Visual Scenarios

Before presenting the case study, we need to introduce the main tools of the TELOS system from a user's viewpoint. We will present the graphic forms and links that are used in the new TELOS Scenario Editor to underline how we cover level B and C of the IMS LD specification. Then, we present the TELOS Ontology Editor visual forms and links that help build ontologies that can be made available as semantic referencing of resources within the system.

2.1 The Global TELOS Interface

The TELOS user interface is available to all kinds of actors through a Web browser. They can download a kernel that will open the TELOS interface shown on figure 3 where we see four main tools open: the *Resource Manager*, the *Scenario Editor*, the *Ontology Editor* and the *Task Manager*.

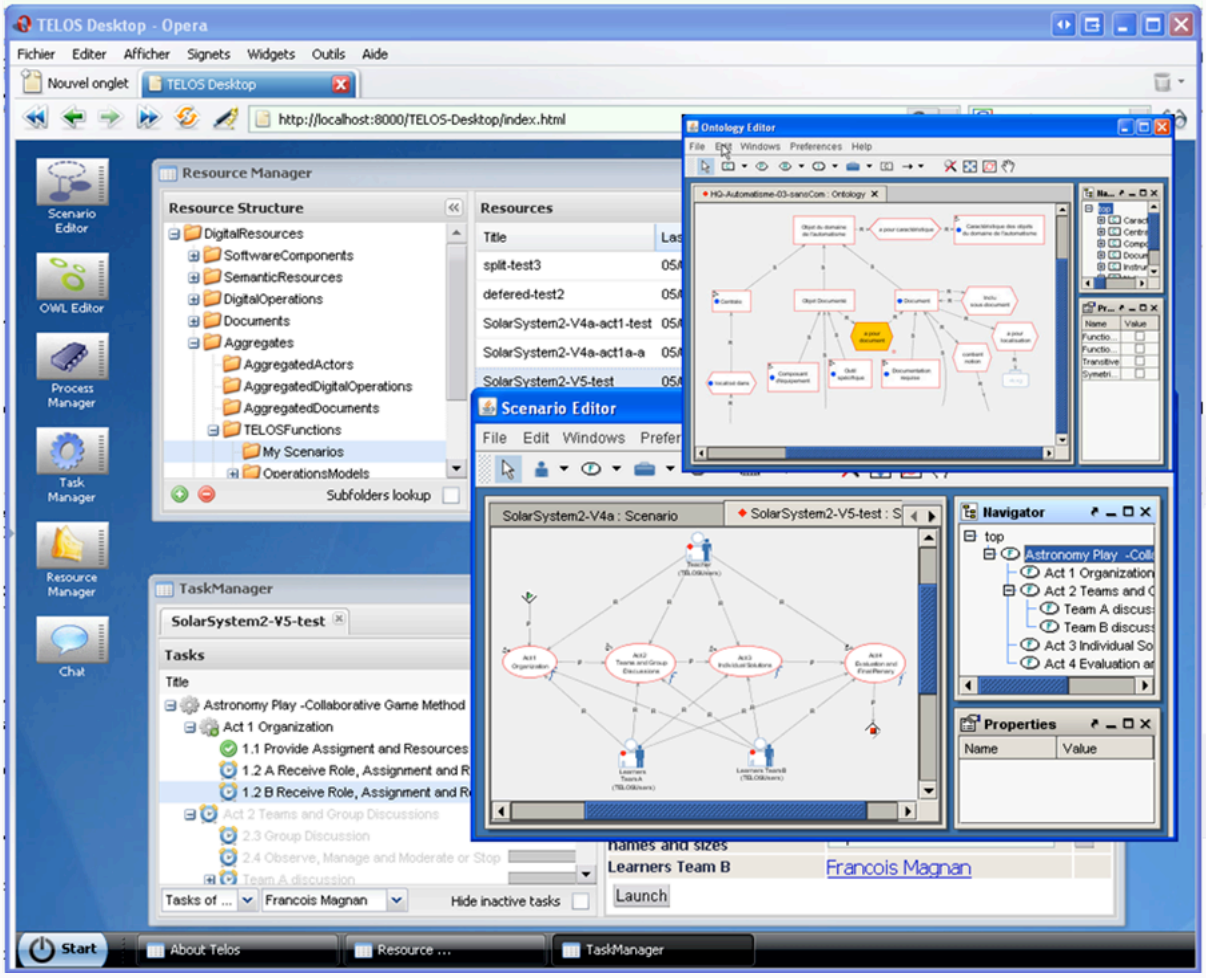

Figure 3– A view of the TELOS interface and the main TELOS tools

The *Resource Manager* gives access to all the resources known by TELOS, whether they are documents, tools, operations, functions or scenarios, actors and ontologies. The resource folders on the left side of the corresponding window are not just folders; they are classes of the TELOS technical ontology that has been built with the *Ontology Editor*. This editor helps also to build domain ontologies that describe the knowledge embodied in other resources; these are also stored in the Resource Editor classes of the TELOS ontology. The action of putting a resource in a class folder means that it is declared as an instance of this class. This action semantically references that resource in terms of the class properties defined in the ontology, which provides its execution metadata. For example the semantic reference will inform the system that the resource can be displayed either as document, launch as a software component, be use to notify a person or be used as a metadata source, depending on its type.

The *Scenario Editor* will be described in the next paragraph. Basically, it is a visual resource aggregator. It helps to orchestrate interactions between actors in activities where they use, process and produce other resources. A scenario player acts behind the scene to execute the scenario and make the appropriate processing of the resources as the scenario unfolds.

The *Task Manager* is the runtime user interface of the scenario player provided to the participants in the scenario. It is a multi-actor player interface with both a tree view and a local visual view on an actor's and other actor's activities. It provides a form of tele-presence that evolves as the scenario is undergoing.

2.2 TELOS Visual Scenario Editor

The Visual Scenario Editor is the central piece of the TELOS architecture (Paquette, Rosca, Mihaila and Masmoudi, 2005, Magnan and Paquette 2006). To design it, a comparative analysis has been made between business workflows and IMS-LD learning designs (Marino, Casallas, Villalobos, Correal and Contamines 2006). It has lead to the identification of 21 control situations for workflows encountered in the software engineering literature (Correal and Marino, 2007) that subsume the properties and conditions in IMS-LD, level B and C. Based on the MOT visual language, the Scenario Editor uses four kinds of objects (Actor, Function, Resource, Condition) with subtypes related to the TELOS technical ontology that drives the system. These symbols are not necessarily in one to one correspondence with IMS-LD terms. For example, the Function symbol can represent Methods, Plays, Acts, as well as Activity Structures. Differentiations between such terms can be made either in the property sheet of the object or be deduced by the Export-to-LD parser when it translate automatically a graph into the IMS-LD XML file.

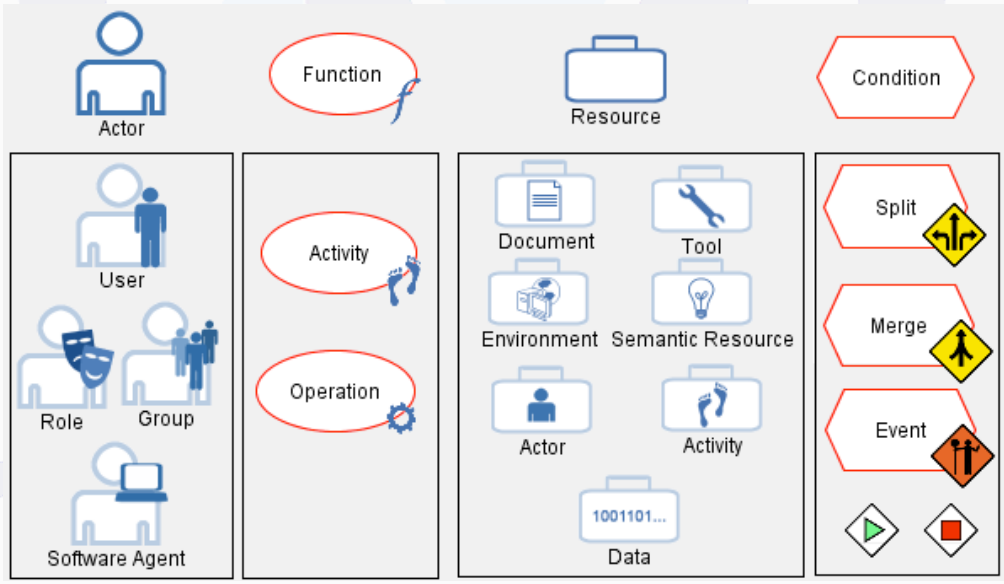

Figure 4– TELOS Scenario Editor (TSE) Visual Symbols

These symbols are shown on figure 4. MOT Concept symbols serve to represent all kinds of *Resources*: documents, tools, semantic resources, environments, resource-actors, resource-activities and datatypes. MOT Procedure symbols represent *Functions* (groups of resources that together achieve a function) that can be decomposed into other functions at any depth level down to activities enacted by humans, or operations performed automatically by the system. Finally, MOT Principles serve to represent actors as well as conditions. The *Actors'* symbols represent users, groups, roles or software agents, seen as control objects that enact the activities using and producing resources as planned by the scenario model. The *Condition* symbols represent control element inserted within the basic flow to decide on the following activities or operations. The diamond shapes defines the start and end point for activities.

On figure 5, we see a scenario that combines some of these symbols. A coordinator writes the plan of a document in the first activity. Then three activities are performed in parallel by different writers to produce the three sections. When these are all terminated, a Web site grouping the different parts is built by the coordinator using a Web editor, and this site is annotated by the group to describe it using metadata and/or ontologies, in order to produce the annotated Web site. This example shows a split condition after the first activity. Later on, the flow from the activities merges through the merge condition before the next activity takes control. According to the merge condition properties, the "Assemble sections" activity will wait for all the incoming flows to terminate before it is executed.

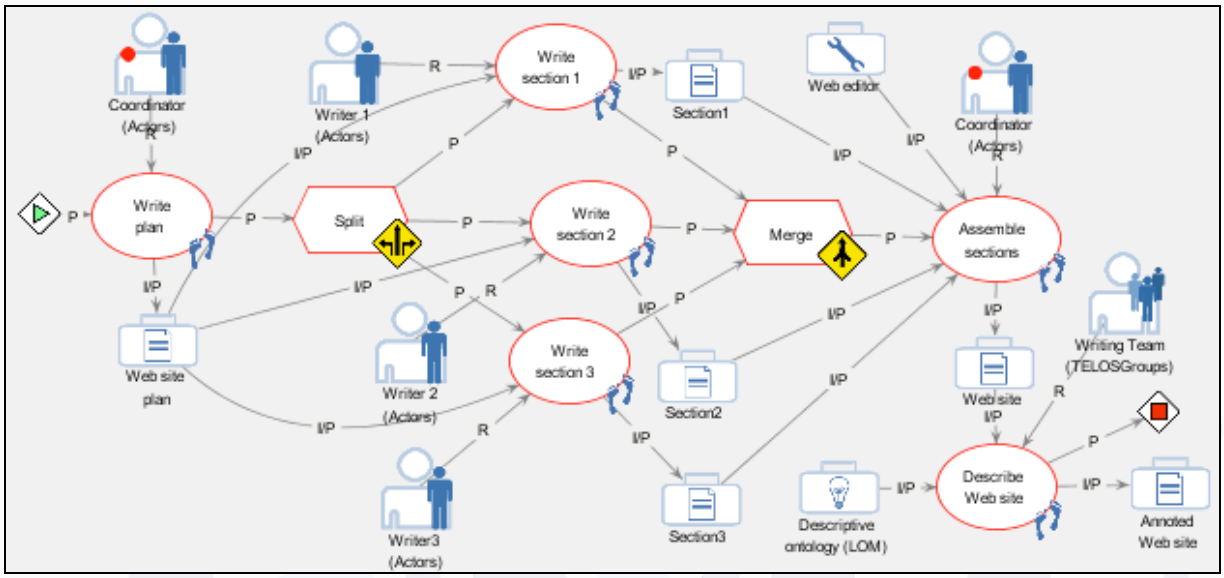

Figure 5– A simple scenario model

In the Scenario Editor, we see a combination of a control flow and a data flow. The control flow is modeled using the MOT basic P and R links. P links are used for the basic line of execution to indicate the sequencing between Functions, Activities and Operations. R links identify the source of an event (from a user or from the system) that triggers a condition that will alter the basic flow of control. MOT I/P links serve to model the data flow, either from resources to activities where they are consulted, used or processed or from activities to outcome resources.

In TELOS, the Scenario Editor can be used at all levels of the system. It enables engineers to combine software components into larger one, technologists to build platform workflows (instructional methods) for designers, and finally designers to build courses or work scenarios. Examples of these are presented in (Paquette and Magnan, 2008).

2.3 Ontology Editor and Semantic Referencing

The Scenario Editor is ontology-driven in the sense that the execution of a scenario depends on the association of its objects to a TELOS technical ontology described in the OWL-DL format (W3C 2004). All the pieces of TELOS must fit in this ontology, which is the logical blueprint of the system. The TELOS technical ontology is not only a conceptual representation of the architecture of TELOS with all its main components. It is an internal way to guide the execution of TELOS. Here we have used a software development strategy in which core functionalities are programmed in ontologies and in the queries we send to the inference engine processing this ontology in order to make deductions that produce the system's services. In the above sense TELOS is an Ontology Driven Architecture (ODA) [4]. This approach combines very well with the concepts of Service Oriented Architectures [5], which is also a key orientation of the system. These architectural principles are explained in (Paquette and Magnan 2008, Magnan and Paquette, 2006).

In the example, on figure 6 (a version of the Planet Game UoL, Act 2) we see a selected document used in the activity labeled"team Adiscussion". On the property sheet at the right lower corner, an execution semantic has to be chosen if we want the scenario to be played. Such a selection corresponds to telling the system what kind of object this is and what are its properties according to the TELOS technical ontology. Selecting that property opens the window shown at the centre of the figure. This window is a view of the TELOS ontology where we can select a class of resources (to be instantiated by the actors at run time) or a specific instance to be displayed at run time. In this example, the resource is an instance, a specific powerpoint presentation on planet properties labeled "GroupA-ini", also shown on the figure. By this association, we tell TELOS that this is not an actor, an activity, an operation or a condition, but a document that the system should open at run time when the time comes.

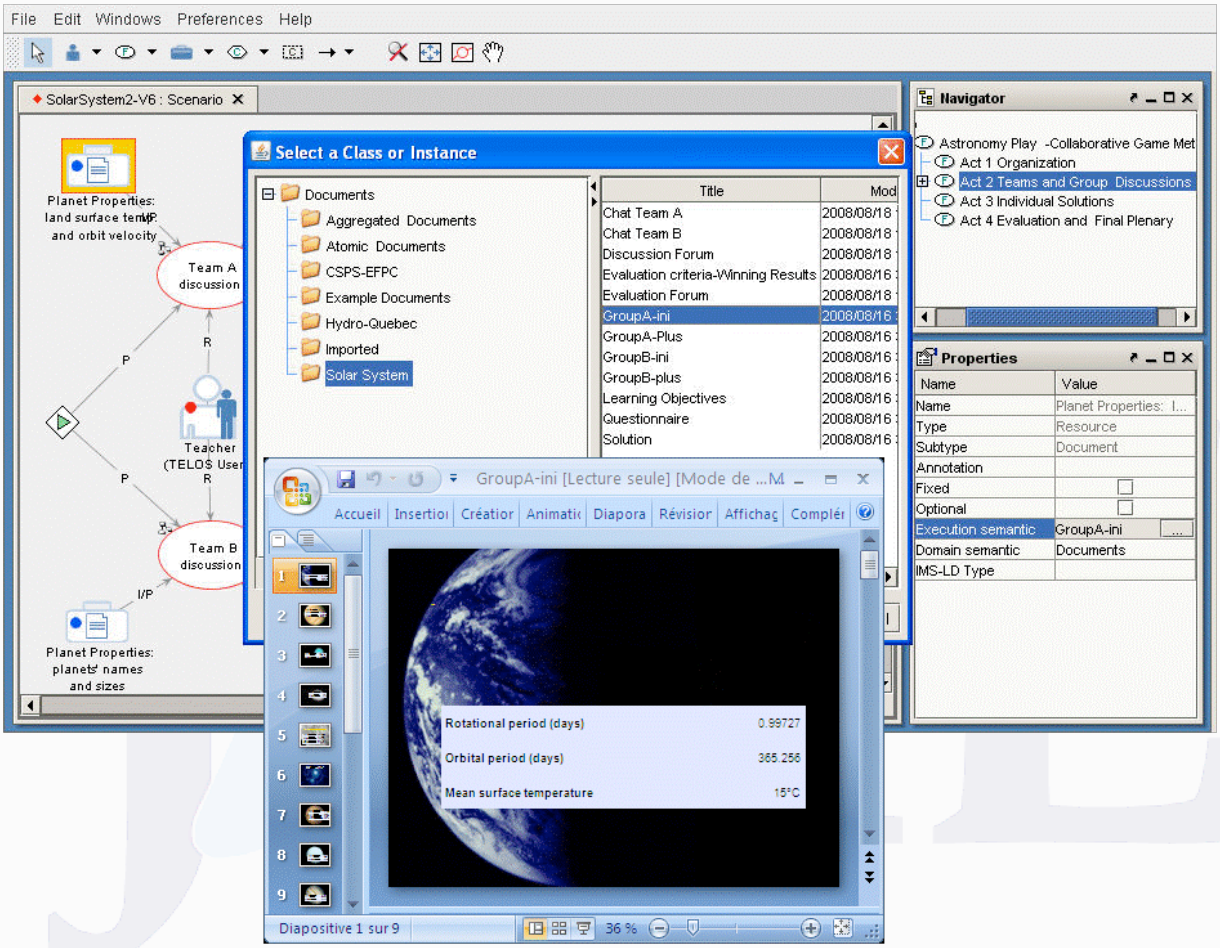

Figure 6– A view of a scenario and the semantic referencing of a resource.

3 The "Planet Game" Use Case

We will now illustrate the TELOS Scenario Editor in action. On figure 3 of the TELOS interface, an upper layer of the scenario is shown that contains four acts ordered sequentially. These acts involve three actors, the teacher and two groups of learners called "Team A" and "Team B". The overall objective of the learners is to discover planet properties. For this, the two teams start with different information. Team A is provided with values of properties such as land surface temperature and orbit velocity, while team B is provided with planet names and sizes. The goal is to reconcile all this data, ordering the planets according to their distance from the sun. This is achieved first by internal team discussions and a forum of the whole group in Act 2. This act starts after the learning objectives, the assignments and a document on planet properties have been provided to both teams in Act 1. In Act 3, each learner will write an individual document and send it to the teacher. In the final Act 4, another forum will be held based on these individual works and the annotations made by the teacher, in order to conclude on the best solution to the initial problem.

3.1 Unfolding the Design

The most difficult Act to model is the second one. Figure 7 presents the sub-model for Act 2 in the TELOS Scenario Editor. On the left side, we see the upper model of Act 2, where the flow splits into team discussions. The execution waits for both discussions to end and then moves to the group discussion forum where members from both teams and the teachers will all participate.

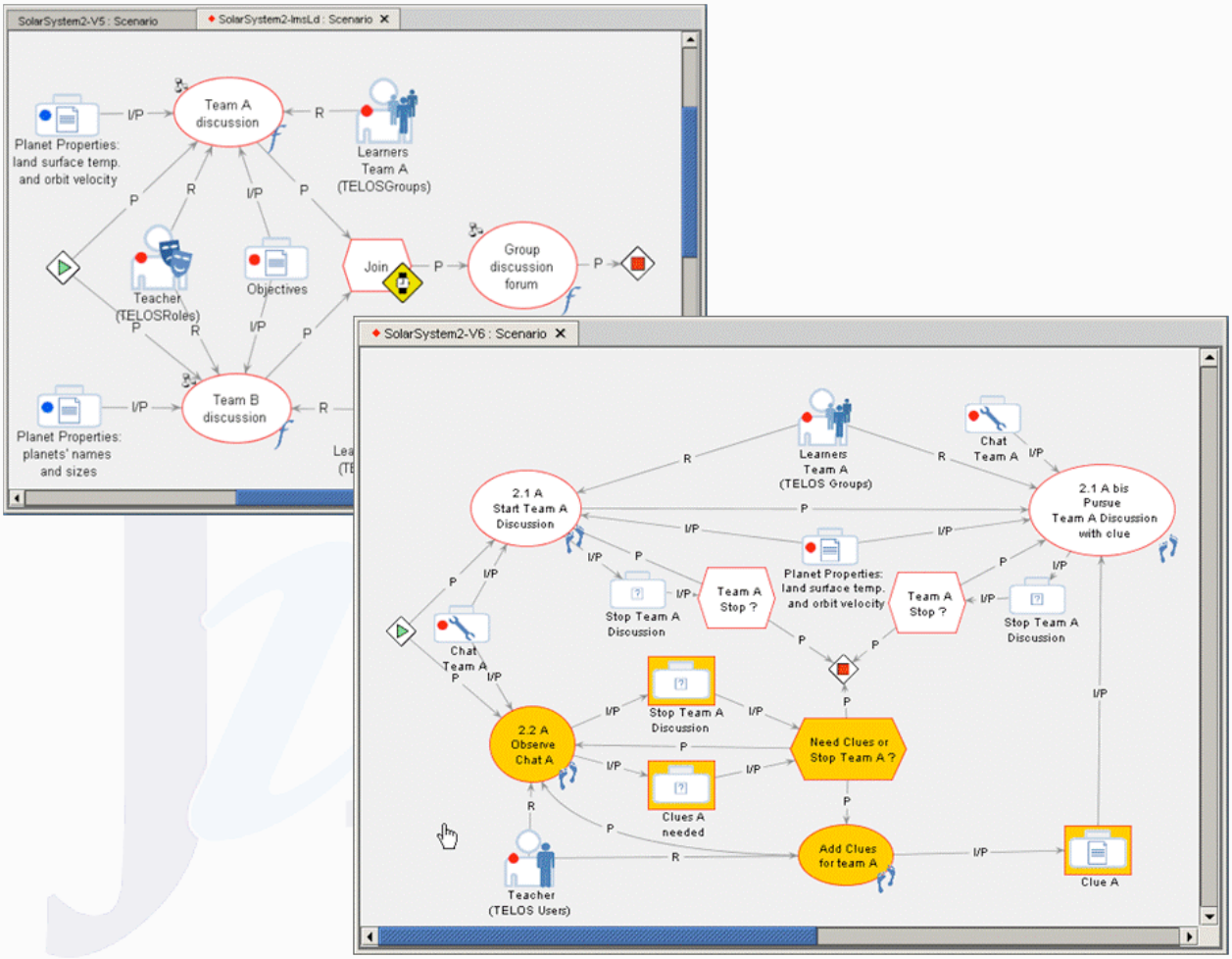

Figure 7– A view of a scenario for Act 2 of the Planet Game unit-of-learning

In the right hand part of figure 7, we see a sub-model for the team A discussion. It starts with opening the chat service for team A that we will detail later. Then, the control splits between the learning activity 2.1.A, where team A learners discuss on the planet properties that have been provided to them, and the support activity 2.2.A performed by the teacher where he observes the team A discussion. The teacher's part is highlighted on the figure. After a certain time in activity 2.2.A, the teacher can either stop the discussion or provide additional information (Clue A) to help the learner solve the problem. The learners can also decide to stop, either before or after they have received this additional information. A similar pattern rules the discussion for team B, with the same teacher acting as a facilitator for both teams, each with a different set of planet properties as additional information.

3.2 Visual Level B Properties and Conditions

The conditions shown on figure 8 represent level B properties of the IMS-LD specification. The decision "Need Clues or Stop Team A?" depends on its input data and the value "true" or "false" that a teacher action will produce in activity 2.2.A for these input variables of the condition. If the value "Stop team A discussion" is true, then the flow of control goes to the end symbol, after which the flow goes up to the main act 2 model to the Join condition. If the value of "Clues A needed" is true, the flow will proceed to the teacher's activity 2.3.A to select a document named "Clue A". If both input variables are false, the flow will come back to activity 2.2.A.

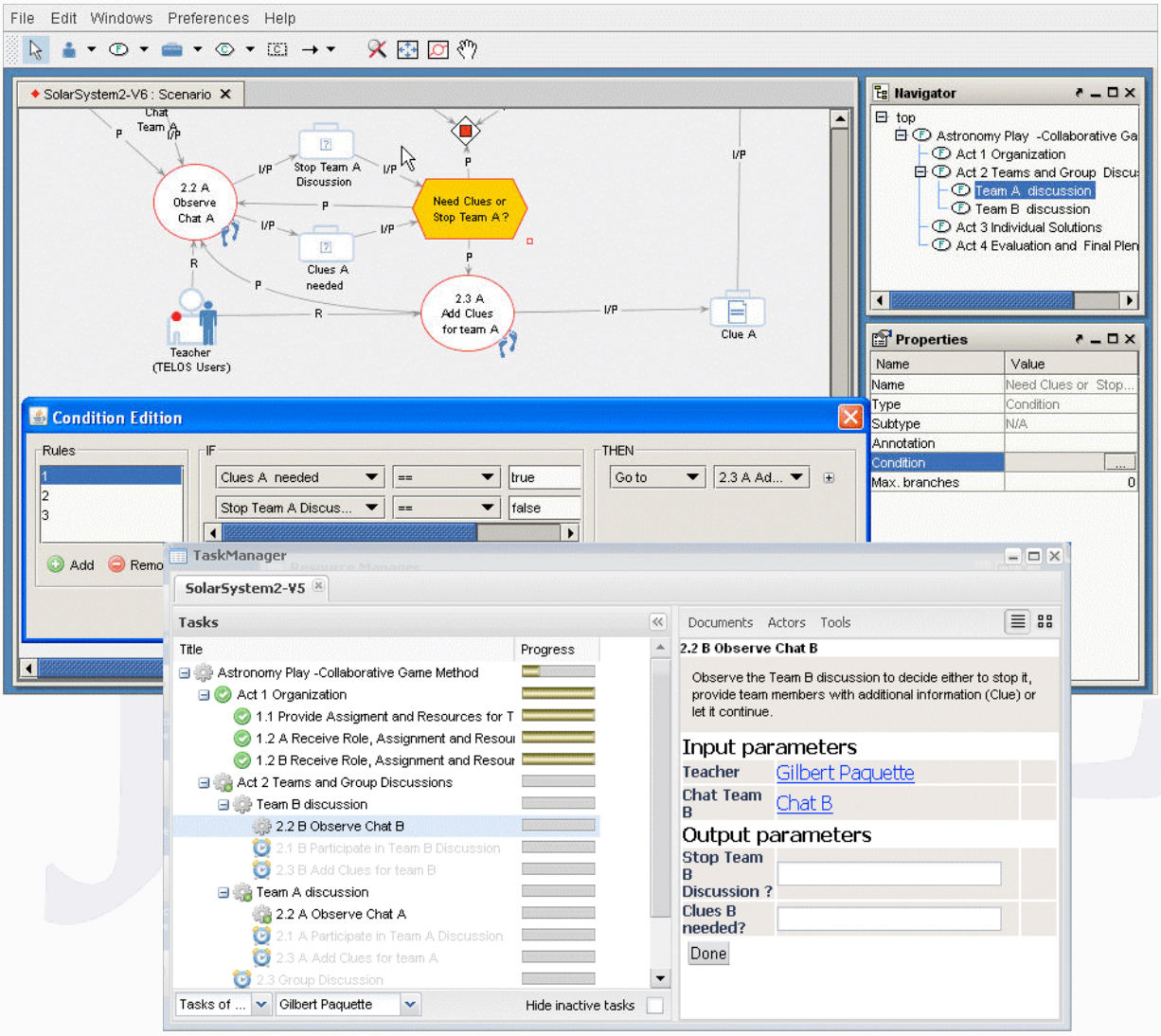

Figure 8– The execution of two teacher's conditions in the scenario.

Figure 8 shows the Condition Edition Interface provided by the Scenario Editor and the resulting situation in the task manager interface where the teacher is asked for a decision in the team B activities for both variables that are input of the condition.

3.3 Differentiated Views

The task manager provides various views to the users depending on the role they have to play in the scenario. Figure 9 shows three views of the same run of the Planet Game scenario. The one on the left is the teacher view selected at the bottom of this window. Since the teacher takes part in all the activities, we see a tree view of all the activities. Some of the activities are Xed because they have not been completed. The reason for this is that in the team A discussion, the learners have put an end to the chat without having to use any clue. We see this on the second window that shows what team A members see of scenario. On the contrary, in the team B discussion, the team members have continued the discussion with a Clue provided by the teacher and later on, the teacher has put an end to the team B discussion, thus completing his own activities. The last window on the figure shows the view provided to team B members.

Another feature of the task editor helps any user see where the others are in the scenario during its execution. Some of the documents may be made not visible to others.

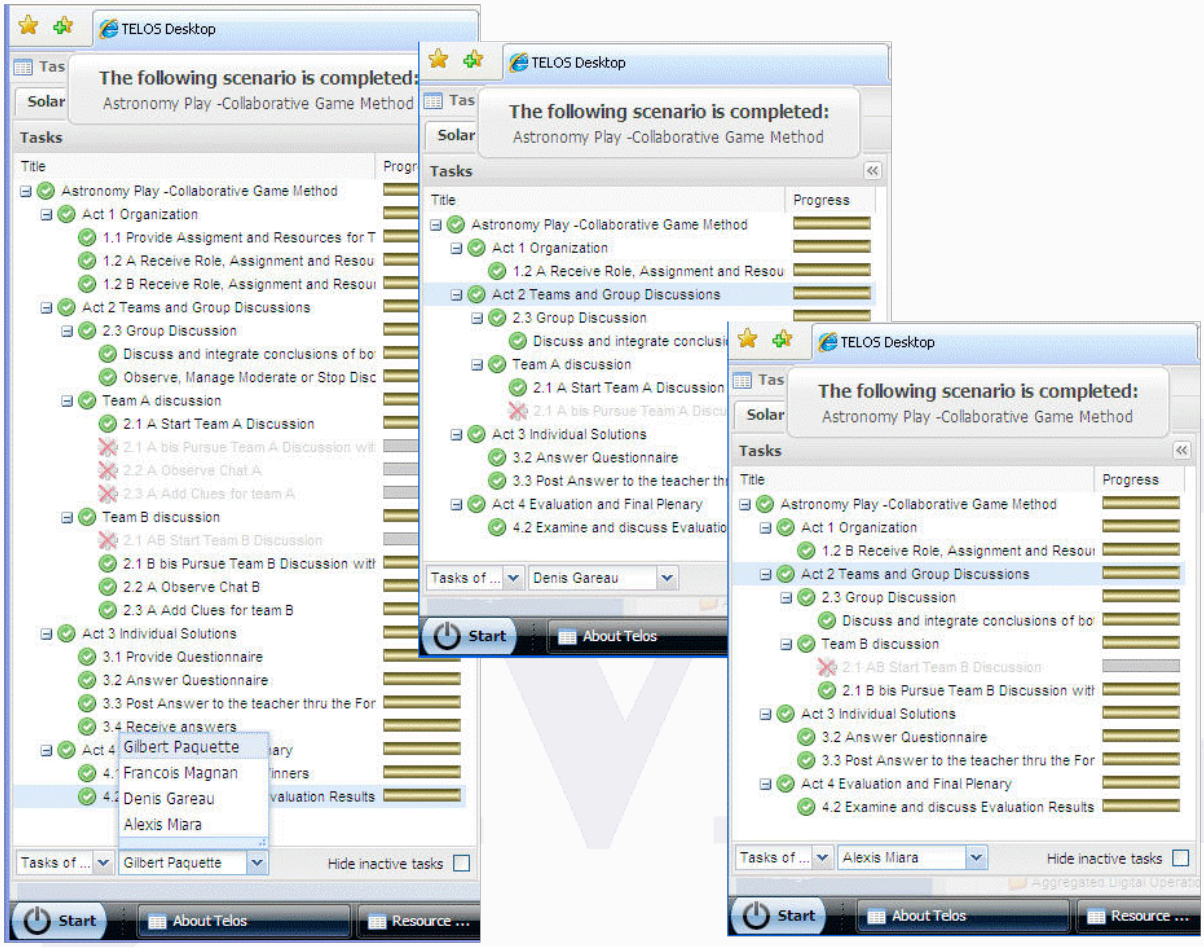

Figure 9– The multi-view aspect of the Task Manager

3.4 Service Visualisation

In figure 10, we see that two services need to be included in Act 2, a synchronous chat, one for each team discussions and a forum for the whole group. The first part of figure 10 shows how we intend to model the chat service visually within the TELOS Scenario Editor. We represent that service as an operation pattern. Around the operation symbol, we see four roles (participant, moderator, manager and observer) and a chat tool that will have to be declared at design time or at runtime, depending if it is associated to an instance or a class of the TELOS ontology. Then the three actors in the Planet Game scenario are associated to roles of this operation. For the team A discussion, learners and teacher are "unified" (using U links) to participants and the teacher is also unified with the three other roles. The forum service is represented in a similar way, the difference between the two is made in the property sheet of each operation. The forum is declared as "asynchronous" while the chat is declared as "synchronous".

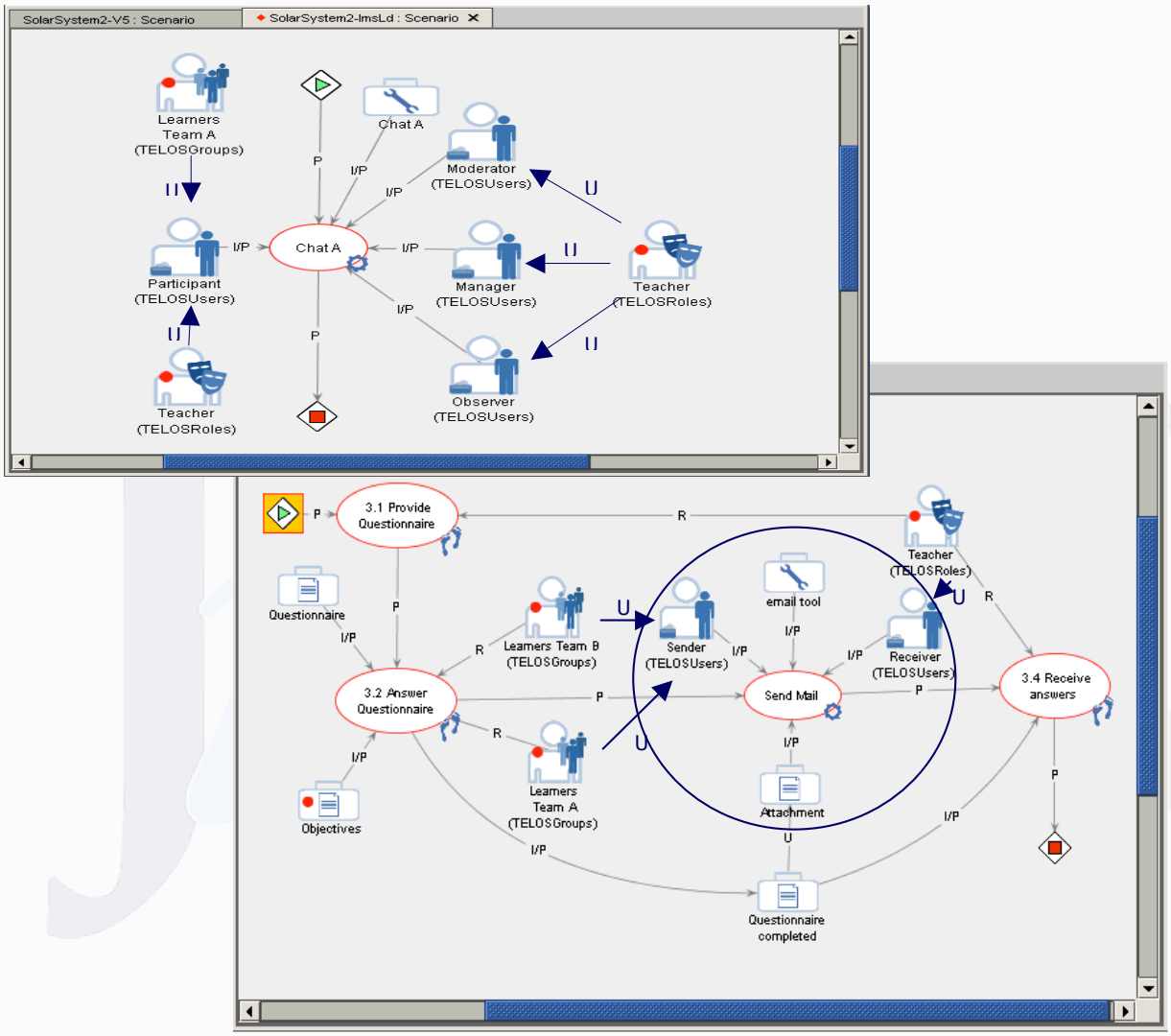

Figure 10– Visual representation of IMS-LD services

The second part of figure 10 presents another service. In Act 3, the teacher provides a questionnaire to all the learners that will send him/her back their individual answers. The Send Mail service used for that operation is displayed here as a TELOS operation with two roles (sender and receiver), an email tool to be instantiated and a document to be attached. Learners in team A and B are unified to the "sender" role (again using U links) and the teacher is unified to the "receiver" role.

3.5 Comments on the Implementation

In this section, we review the various steps proposed for the case study.

In *Step 1*, the modelling of this activity was quite straight forward, but we had to do several executions of the scenario to make sure that it could run properly in the TELOS Player and display adequately in the Task Manager. The fact that the TELOS Player was being built by the TELOS developers at the same time we were designing the scenario added some difficulties, but it helped building a more useful player.

More fundamentally, the authors acting as designers had to adapt to the fact that they were not just designing, but also programming, although they were using a rather friendly visual language. For example, in Act 2, the introduction of local level B conditions in the team discussions was a bit "tricky". How can we specify properly the addition of information ("Clues") by the teacher at run time? If the new resource is simply designed as an input to a team's discussion, then the player would

wait after this resource to be provided and the discussion would not start without it. Our solution was to have two steps in the discussion, the first without the "Clue", the second being optional if a "Clue" was made available to the team by the teacher when it triggers the condition accordingly. Then to make everything "clean", another condition was to be added to close the first part of the discussion if the second one was being activated. With these question settled, the TELOS player could execute the scenario properly and the Task Manager would display the adequate interfaces to the actors according to their roles.

In *Step 2,* at run time, we found out that the Task Manager interface provided adequate observation facilities of the learners' performance. For example in Act 2, the teacher could look at the messages in the chats of each team and decide what to do, stop the discussion to move to the general forum activity, or add a clue if a team seems to have some difficulties with the content. He could also use the communication services at any time to send messages to the learners and interact with them to monitor their progress.

The tree view of the Task Manager presented on figure 9 shows at any time the learners' progress at a glance by means of the progress bar and the symbols in front of the name of the activities. On the last window of figure 8, we see a more complete view of this user interface. The teacher can select the view of any user involved in the scenario. The documents, actors and tool menus show the list of all the resources in the scenario as soon as they are added. These display features are based on traces registered during the activity's performance. These traces could serve to trigger pieces of advice that could be displayed automatically to users or be used to replay and analyse different sessions. These are possibilities that we plan to exploit in the future.

The scenario we have built for this case study could be re-used or adapted in many ways.

- We could change only the activity assignments to propose different learning strategies on the same subject. For example the learners in each team could be asked to find resources by themselves on the Planet, instead of starting with the one provided by the design. Then they could proceed with the team discussion and the group forum in Act 2. In Act 3, the teacher would provide a different individual assignment.
- We could also keep the structure and replace most of the resources to provide an environment in a completely different subject area such as geography or management. For this we need to simply import the new resources from the Web, a LAN or computer into the TELOS resource manager. Then, using the TELOS Scenario Editor, we change the name of resource and link it to the appropriate class or instance where the resource is located.
- We could also adapt the structure in the TELOS Scenario Editor to add more teams for a larger class; this would require copying some graphs with slight modifications. We could also decompose the scenario in the four acts and store them in the "Aggregates" section of the Resource Manager for future retrieval and recombination in a different order or with other "nuggets" extracted from other scenarios.

4 TELOS Scenarios and the Usability of IMS-LD

A number of considerations have been taken into account while building the TELOS Scenario Editor, the Scenario Player and the Task Manager.

First of all, we aimed to *represent all levels of IMS-LD visually*, in order to simplify the design process, a result that is badly needed according to many authors and to our own experience in our previous projects. Unlike the proposals in the IMS-LD specification, the conditions are not defined and accessible at the method level. In our case, this would mean that they would all appear on the main graph of the scenario. This would blur the whole design that would look sometimes like huge spaghetti of links and figures. More important, that would complicate the design process and the validation of an author's intended flow of control. The fact that the *conditions are inserted locally, where they are needed*, provides a clear picture of the author's intentions and simplifies the validation of the scenario

flow. We let our IMS-LD translator gather all the conditions and bundle them at the method level to comply with the specification while freeing author's from the burden.

Even at level A of the specification, the IDLD project mentioned earlier has pointed out a number of improvements that needed to be made to our initial MOT+LD visual editor to simplify the authors' work.

- In MOT+LD we had to integrate many symbols for activity structures in an artificial way to distinguish visually between sequential ordering of activities and selection between parallel activity. In the TELOS scenario editor, this need is completely eliminated. Selections are modeled using condition symbols and sequences by activities related by precedence links. Activity structures are modeled as functions and their sub-models can be a mix of other activity structures (also modeled as functions), activities and operations.
- The designer should not have to use the "Environment" symbol, except in limited cases. These symbols make the graphs unnecessarily heavy and are a nuisance when completing certain basic transformations to a scenario. It is implicit that an activity's environment is composed of all the resources around it, whether they are used or produced during the activity. In the TELOS Scenario Editor , the translator to IMS-LD creates environment symbols automatically by gathering all the resources having input-output links with an activity.
- The introduction of item symbols carrying the concrete resource locations makes graphical models a bit crowded. In the TELOS Scenario Editor, there is only one symbol for a resource (if necessary duplicated in the same or in different models). We give to the designer the freedom to associate a symbol to an instance in the ontology (which corresponds to an item or concrete resource) or to assign to it a class of resources to be instantiated at run time.
- The graphical representation of the IMS LD services, send-mail, conference and index-search, was particularly complicated and made corresponding portions of the graph difficult to read and understand. In TELOS, services will be modeled as generic operations where the resources involved in the operation (actors, documents and tools) are specified as an operation's port. To insert a service in a scenario, we only need to "unify" resources in the scenario, including actors, to the operation's ports. The detailed structure required by IMS-LD XML is automatically created by the translator to IMS-LD and added to the manifest file.
- In the IDLD project, the MOT+LD editor and the PALOM@ learning object manager were used respectively to build the design and annotate the resources with metadata, without being directly linked to each other. This complicated the designer's task. In TELOS, metadata referencing is replaced by semantic annotations performed within the Scenario Editor by associating to a resource a class or an instance of an ontology, a much simpler operation for the author. The resource then acquires all the properties in the ontology. To reference resources by the LOM or DC metadata, one way is to include LOM or DC classes into the TELOS ontology..

5 Generality Issues

Even though TELOS scenarios and IMS Learning Design are both essentially multi-actor process models, there are important differences between them for very good reasons. We have decided to keep these differences and add graph parsers to provide import/export facilities from and to standard IMS-LD XML files.

First, we wanted to be able to use the Scenario Editor as the main aggregation tool in TELOS. TELOS is a multi-layer system. Engineers have to aggregate existing software components (built possibly with different technologies) to create new ones, in order to extend the capabilities of the system. Technologists have to create or extend a Web platform by building scenarios for designers (instructional design methods) that includes a variety of design tools and documents (Paquette and Magnan, 2008). The constraints introduced in the method structure of IMS-LD do not take these use cases into account.

A second goal was to encompass business workflows as well as instructional scenarios for learners within the same Scenario Editor to enlarge the domains of application. For workflows, Business Process Model Notations such as the BPMN specification [6] are more restricted than learning designs on certain aspects, but they provide a larger set of conditions to control the flow of activities. Unlike IMS-LD level B and C where all the conditions are declared at the method level, BPMN conditions are visually located at the point where they are used, thus given a more transparent view of the execution flow. Also, they make important distinctions that are not present in IMS-LD such as multi-activities that are to be done separately by each actor in a group, compare to activities to be achieved by all group member cooperatively. Some of these features are also useful for learning designs, providing a solid foundation to the TELOS Scenario Editor.

6 Contributions and Future Work

With regard to Educational Modelling, our goal was to provide a visual language that could be simple enough to support a wider use of specifications like IMS-LD and, at the same time, be powerful enough to produce efficient and operational environments for the variety of situations encountered in learning and knowledge management.

We have produced a solution that encompasses larger considerations, but sometimes addressing a more general question which seems to simplify many aspects of the solution. Furthermore, to produce an IMS-LD scenario, a designer does not have to be familiar with the details of the specification; an export tool will take care of the tedious task of creating the IMS-LD XML file corresponding to the model build graphically with the TELOS Scenario Editor.

Even though TELOS is a mature prototype, our next tasks will be to make it a robust industrial system and implement it in a variety of applications. We have the feeling that we have solved most of the scientific problems, but real-life applications have the ability to impose new and sometimes surprising research problems. Investigating such problems is a stimulating challenge that lies ahead.

7 References

Correal. D., Mariño O. (2007) Software *Requirements Specification Document for General Purpose Function's Editor (V0.4)*. LORNET Technical Documents, LICEF research centrer, Télé-université, Montreal.

Karin Lundgren-Cayrol , Olga Marino, Gilbert Paquette, Michel Léonard, Ileana de la Teja. *Implementation and Deployment of IMS-LD - Outcomes of the IDLD Project*, ICALT-06 conference, Kerkrade, The Netherlands, June 2006

Magnan, F. and Paquette, G. (2006) *TELOS: An ontology driven eLearning OS*, SOA/AIS-06 Workshop, Dublin, Ireland, June 2006

Mariño O., Casallas R., Villalobos J., Correal D., Contamines J. (2006) Bridging the Gap between e-learning Modeling and Delivery through the Transformation of Learnflows into Workflows. In Pierre, S., ed.: *Elearning networked. Environments and Architectures: A Knowledge Processing Perspective.* Springer Verlag.

Merrill M.D (1994). *Principles of Instructional Design.* Educational Technology Publications, Englewood Cliffs, New Jersey, 465 pages.

Minski M. (1975) A framework for reprsenting knowledge. In P. H. Winston (ED.), *The psychology of computer vision.* New York: McGraw-Hill

Newell A & Simon H. (1972) *Human problem solving.* Englewood Cliffs, NF: Prentice-Hall.

Paquette, G. and Magnan F., (2008) An Executable Model for Virtual Campus Environments in H.H. Adelsberger, Kinshuk, J.M. Pawlowski and D. Sampson (Eds.), *International Handbook on Information Technologies for Education and Training*, 2nd Edition, Springer, Chapter 19, pp. 365-405, June 2008

Paquette, G. M.Léonard, K. Lundgren-Cayrol, S. Mihaila and D. Gareau (2006) Learning Design based on Graphical Knowledge-Modeling , *Journal of Educational technology and Society* ET&S , Special issue on Learning Design, January 2006.

Paquette, G., Rosca, I., Mihaila, S., Masmoudi, A. (2006) Telos, a service-oriented framework to support learning and knowledge management. In Pierre, S., ed.: *E-Learning Networked Environments and Architectures: a Knowledge Processing Perspective.* Springer-Verlag.

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

Paquette G. (1996) La modélisation par objets typés: une méthode de représentation pour les systèmes d'apprentissage et d'aide a la tâche. *Sciences et techniques éducatives*, France, pp. 9-42, avril 1996

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

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

Tennyson, R. & Rasch, M. (1988) Linking cognitive learning theory to instructional prescriptions. *Instructional Science*, 17, pp. 369-385

W3C (2004) *Ontology Web Language (OWL) - Overview Document* (http://www.w3.org/TR/2004/REC-owl-features-20040210/)

West C. K., Farmer J. A., Wolff P. (1991) M. *Instructional Design, Implications from Cognitive Science.* Allyn and Bacon, Boston, 271 pages.

8 Footnotes

[1] LORNET is a five year pan-Canadian project led by the first author. Its mission is to research and develop new Semantic Web methods and tools for the design and delivery of learning and knowledge management environments (www.lornet.org) . The TELOS system presented here is still in development but an autonomous and previous version of the TELOS Scenario Editor can be consulted at http://poseidon.licef.ca/scenarioEditor/.

[2] RELOAD editor and player, http://www.reload.ac.uk/ last retrieved July 24, 2006

[3] The Portal for IDLD (Implementation and Deployment of IMS-LD) is at www.idld.org

[4] See the paper by Tetlow, P., Pan, J., Oberle, D., Wallace, E., Uschold, M., Kendall, E. (2001): Ontology driven architectures and potential uses of the semantic web in systems and software engineering. http://www.w3.org/2001/sw/BestPractices/SE/ODA/051126/

[5] See the paper by Sprott D. and Wilkes L. (2004) Understanding Service-Oriented Architecture, Microsoft Architecture Journal, http://msdn.microsoft.com/en-us/library/aa480021.aspx

[6] See Business Process Modeling Notation (BPMN) Specification, Final Adopted Specification 06-02-01, at http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf