From a Conceptual Ontology to the TELOS Operational System

Gilbert Paquette and François Magnan

gilbert.paquette@licef.ca; francois.magnan@licef.ca

CICE Research Chair, LICEF Research Center, Télé-université

Abstract. In the last four years, within the LORNET research network, the LICEF team has been designing and developing TELOS, an innovative operation system for eLearning and knowledge management environments. This communication will present the main steps that have led to the actual system, as a contribution to the general software engineering methodology. We will first briefly present the background and initial requirements for TELOS. Then, we will summarize an ontology that captures the conceptual architecture of the system. Finally, we will present the transition of this conceptual ontology towards the technical ontology that actually drives the system. In conclusion, we will underline the main advantages of this method for building ontology-driven systems such as TELOS.

Introduction

At the beginning of the present decade, new sets of concepts had emerge from various fields such as Web-based layered and programmable learning portals, service oriented frameworks, model-driven and ontology-driven architectures, multi-actor scenarios and workflows. These main technological trends have deeply influence our work to produce more flexible, powerful, yet user-friendly elearning environments.

One level up: aggregating custom-made platform or portals. Just as integrated suites of generic software have been replaced by integration mechanisms at the operating system level, we aimed to design TELOS on the same interoperability principles. Similarly, the TELelearning Operting System (TELOS) architecture aims to extend the portal assembly mechanisms to enable technologists to built their own platforms (or eLearning/Knowledge Management desktops), creating a variety of distributed learning environments or models such as electronic performance support systems (EPSS), communities of practice, formal on-line training and technology-based classroom, and different forms of blended learning or knowledge management environments.

As the project was starting, *Service-oriented frameworks* [5] such as ELF [26] or OKI [27] were proposed to lower the costs of integration, and to encourage more flexibility and simplification of software configurations. Such a framework could also create a broad vocabulary that could be extended to an ontology. The TELOS conceptual framework presented in section 2 would also be designed as a service oriented framework, facilitating the aggregation of services to create custom-made platforms and applications.

This has led us naturally to a *model-driven, ontology-driven architectures* [8]. The main gain of model-driven architectures is the generation of the code from the model in successive layers, the model being reusable in other contexts with few adaptations. Ontology-driven architectures [6, 7] add to this paradigm an explicit ontology structuring of the objects processed by the system, acting as its executable blueprint. They therefore tend to maximize the platform independent model (PIM), minimizing the platform specific (PSM) and Code models. This programming style follows a pattern analogous to the Prolog programming

language. Here the declarative part is encoded in the ontology, in our case through OWL-DL statements. The execution part is encoded in parameterized queries prepared for an inference engine that processes the queries. The result of a query is to trigger the execution of some of the services.

Another key architectural idea is the concept of *multi-actor learning designs and workflows,* as the main structure of the various environments produced using TELOS. We have pointed out elsewhere some weaknesses of our initial virtual campus models [**] and most commercial platforms, where actors can interact within mono-actor environment that do not really take in account collaborative processes. This question is now solved partly in workflows modelling languages such as BPMN, the Business Process Modeling Notation [18], and in eLearning design specifications like IMS-LD [16]. Multi-actor learning designs and workflows provide a central aggregation mechanism grouping actors, the operation they perform and the resources they use or produce from or for other actors. A multi-actor scenario editor and execution engine was planned as a central piece of TELOS [9, 10].

In the first section we will revisit our initial methodology and its main products. The second section presents the conceptual framework of the TELOS system, and the conceptual ontology derived from it. The third section will summarize the technical architecture of the system. The fourth section will present some use cases that illustrate the exploitation of the technical ontology. The last section will summarize the whole process and discuss its main features.

1. Rational Unified Process and Rapid Prototyping

Initially, we have tailored the Rational Unified Process (RUP) [ ] to the needs of the project. RUP is an adaptable process framework that describes how to develop software effectively using proven techniques. While the RUP encompasses a large number of different activities, it is also intended to be tailored to select development processes appropriate to a particular software project or development organization.

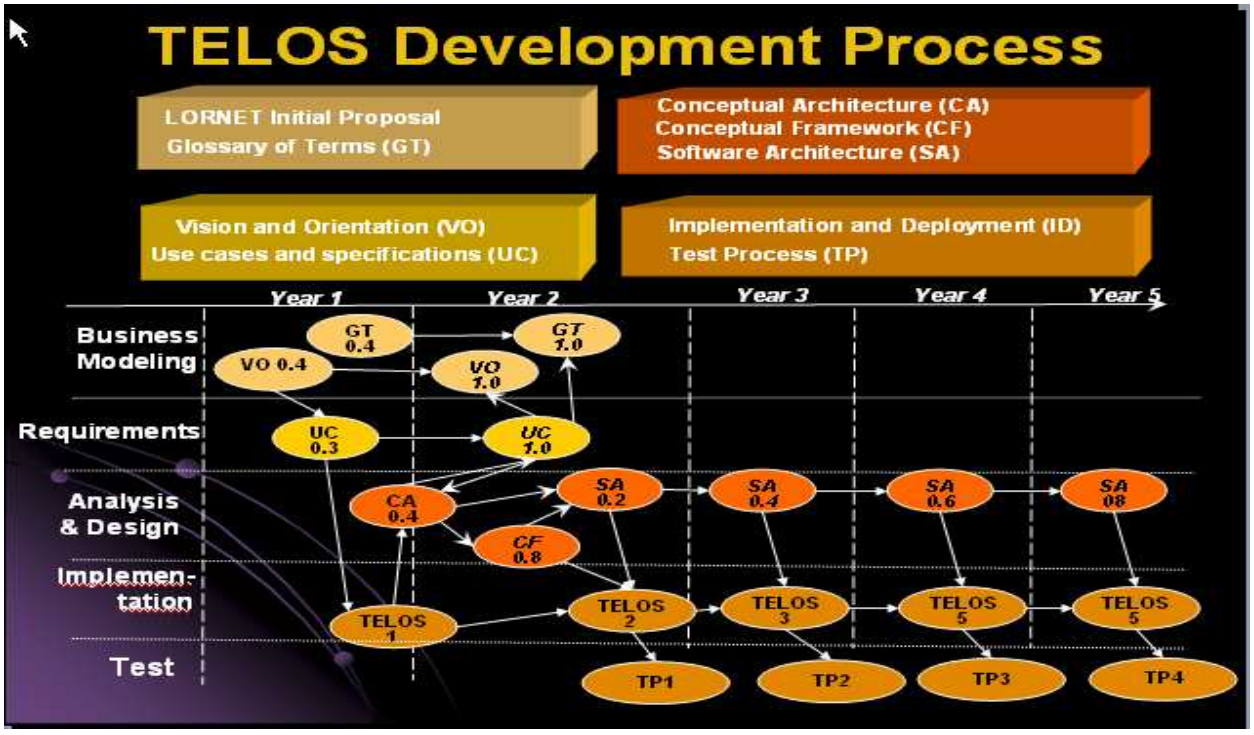


Figure 1 – TELOS development process.

As shown on figure 1 our initial [1]use of RUP was first focused on the business modeling and the requirements processes, each with a few cycles including phases of inception, elaboration, construction and transition. This has led to a set of architecture documents, the main one being the Use cases and software requirements documents (UC 1.0). Then the focus has moved to the Analysis and Design process with the construction of the Conceptual Architecture (CA 0.7) and the Conceptual Framework (CF 0.8) documents, which includes the TELOS Conceptual Ontology. The first two years followed the RUP quite closely but with long iteration cycles resulting in a set of architecture documents and throw-away prototypes TELOS-1 and TELOS-2.

In the last three years, the team has reduced the length of the iterations, adopting a process closer to Rapid Prototyping in order to achieve workable prototypes. A number of Software Architecture (SA) documents have been written to support the implementation of the TELOS prototypes. TELOS-3 was the first evolutionary prototype on which we could build the following ones. Each year, a test bed was conducted where users would interact with the available prototype within a carefully planned test process. Prototypes 2, 3 and 4 were demonstrated at the LORNET annual conferences. The last one, TELOS-5, is demonstrated at the ITS-08 conference. This evolution reflects the fact that TELOS is does not follow traditional software development processes, being considered as innovative, risky and ambitious by many persons, in other words, a research project.

We will now briefly summarize the Use Cases Specifications and Requirements (UCR 1.0), and the TELOS Conceptual Framework (CF 0.9). These have been described more extensively in [10,11]. The UCR has undergone 10 iterations, from June 03 to December 04. It groups 30 use case diagrams and descriptions that are packaged as shown on figure 2.

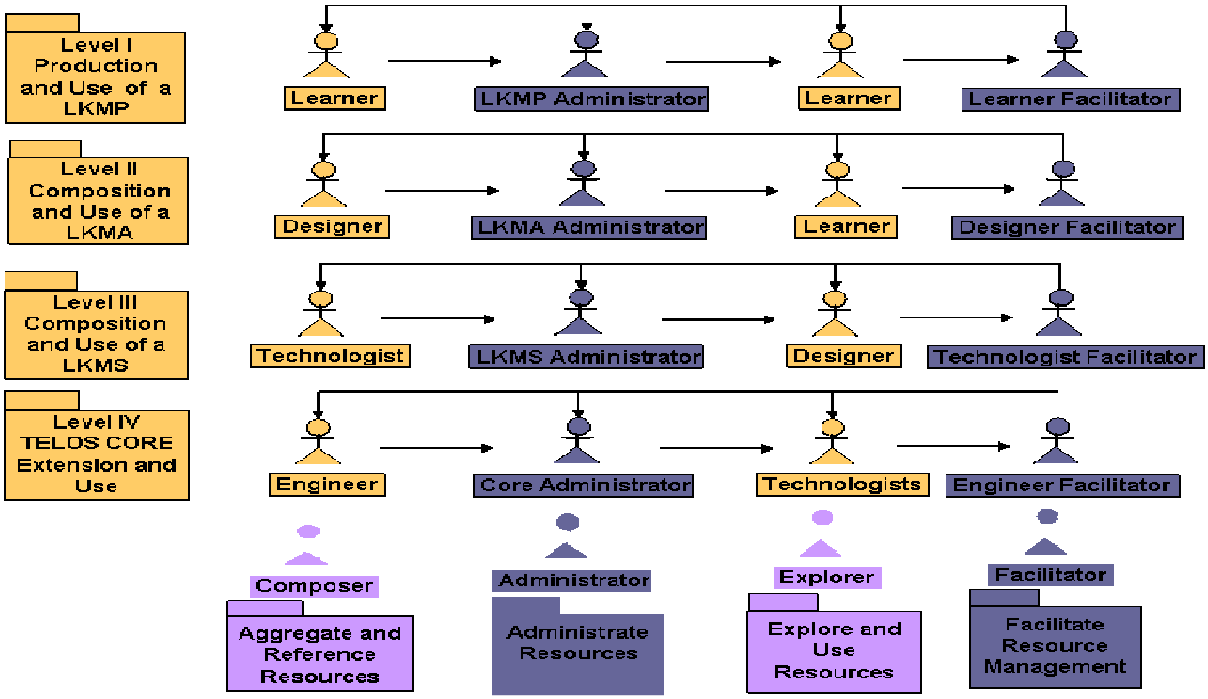


Figure 2: Resource life cycle and system cascade actors

[1] LORNET is a five-year project ending in October 2008 grouping six Canadian research centers and over a hundred researchers, professionals and graduate students. TELOS is the main integration production produced within this project.

The use cases at the four levels of the system describe how to build, administrate, use and support a Web-based environment, each being used at each cascade level (rows on the figure 2). Level IV concerns mainly an engineer extending the TELOS Core that will be used by technologists. At Level III, a technologist uses the TELOS Core to produce a platform, technically called a Learning and Knowledge Management System (LKMS). At level II, a designer uses a platform, to build one or more Learning and Knowledge Management Applications (LKMA): courses, learning events, knowledge management workflows, etc. Finally at level I, using one of these applications, a learner will acquire knowledge and produce results (homeworks, documents, performance) that can be grouped in a portfolio or Learning and Knowledge Management Products (LKMP).

Generic resource life cycle use cases (columns on figure 2) correspond to four sub-operations (phases) that traverse the cascade levels. In these, a resource is composed, managed (prepared) for use, used by its intended actors, and analyzed to provide assistance. These operations are generally performed in sequence at each of the cascade levels by corresponding actors called respectively composers, administrators, explorers (resource users) and facilitators (acting as analysts to provide assistance and feedback). These operations are generic, being applicable at any cascade level. For example a learner will have to search for resources in much the same ways as a designers, technologists or engineers, when they act as composer to produce respectively a LKMP, a LKMA, a LKMS or TELOS Core

We can use different metaphors to describe these general processes. In a manufacturing metaphor, the resource life cycle corresponds to a process where a product passes through different productions operations. Within the system generation cascade, the TELOS Core is like a factory that produces machine components or complete machines; the products of this first factory are used to build machines that will be used in other factories (LKMSs) to build cars, aircrafts, etc. These transportation machines, will finally be used by their clients to produce some outcome (e.g. to travel). As a manufacture, the TELOS Core itself starts with a complete set of components to produce LKMS factories, but it will also be open to improvement, adding new processes and operations, to produce more versatile machines.

Starting with this elaborated set of use cases, the conceptual architecture [12] and the conceptual framework [10] were build in the form of a service-oriented framework, bringing it closer to a possible implementation. Figure 3 present the main classes of services.

- *Kernel Communication services.* The Virtual Campus model is a distributed architecture on the Internet. To become a node in the Virtual Campus, each user installs a kernel on his machine that provides basic communication services with other nodes where resources are distributed. These services include for example a service registry, the location of resources on the nodes of the network, connectors to provide communication with resources built using different technologies, protocol translation and so on.
- *Resource interfacing services* .Basic resources are documents in a variety of media formats, tools to process documents, operations that can perform a process automatically and finally persons managing a set of activities on the network. All these resources usually will required to be interfaced in different way (by a communication agent for format translation, through encapsulation for tracing, etc.) in order to be reached and to participate in the learning and/or knowledge management processes
- *Resource life cycle services.* These services provide a number of editors for a composer to build, adapt and aggregate resources, thus producing a model of the resource. Tools for an administrator to produce instances of the model, as well as interfaces to help users and facilitator interact with an environment instance.
- *Aggregate' management services.* These services provide management functionalities for the main aggregates (or Web portals) used in the Virtual Campus: Core, LKMSs, LKMAs and LKMPs portals. For example, they will help in the storage, modification, display, evolution and

maintenance of versions of TELOS Core, the interoperability between platforms (LKMSs), the management of courses (LKMAs) and the LKMPs such as Portfolios.

- *Semantic services.* These services enable users to query or edit semantic resources, for example ontologies or metadata, and to reference resources. Resource publication services enable users to package resources with their semantic references, enabling various kind of resource search, retrieval and launching. With these services a user can call upon federated or harvested search operations to jointly display documents, tools, operations (including activities and units-of-learning) related to some domain knowledge and competencies.
- *Common services.* We have grouped in this category all the lower level services that are called by the services in the preceding categories. They correspond to operations that all the actors need to make or called upon while participating in the Virtual Campus.

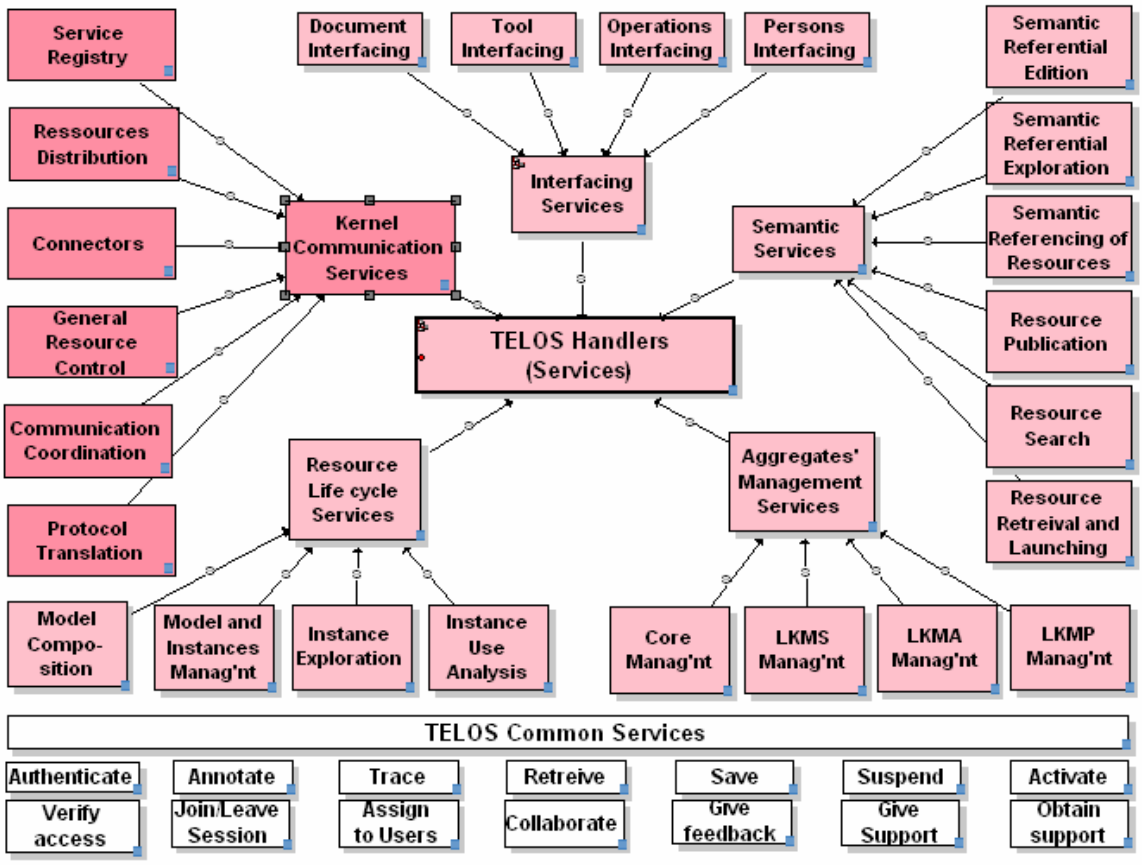


Figure 3: The Virtual Campus Service Oriented Framework

2. The TELOS Conceptual Ontology

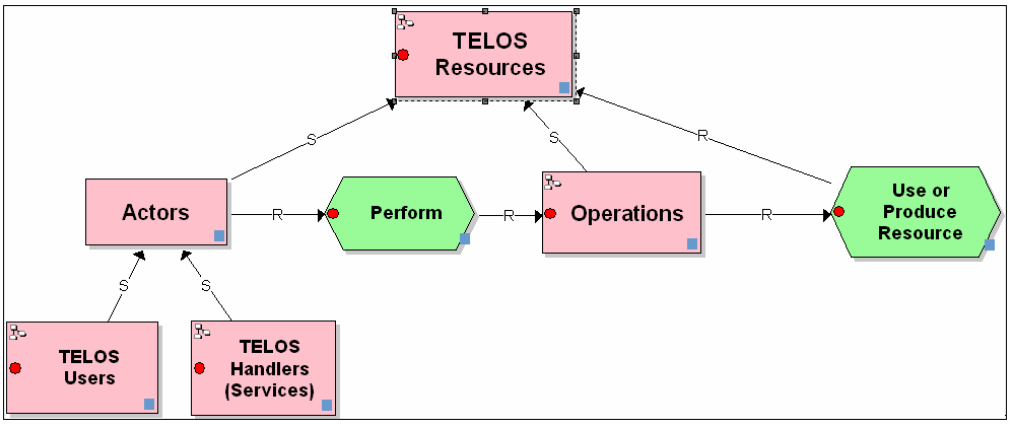
An important goal is to embed in the system technology-independent models, to help the system survive the rapid pace of technology evolution. For that purpose, the conceptual specifications of TELOS, expressed as an ontology, should not be kept apart from the code of the system as is usually done in software engineering. The TELOS system should be able to reuse ontologies as "conceptual programs". In this vision, the conceptual models are not just prerequisite to the construction of the TELOS system; they

are part of the system, as one of its most fundamental layer. These considerations motivated the need for an ontology-driven architecture (ODA).

We have translated the use cases and the above service-oriented framework into an OWL-DL ontology. We have selected to use OWL-DL ontologies [13] for a number or reasons. It is one of the three Ontology Web Languages that are part of the growing set of World Wide Web consortium recommendations related to the Semantic Web. Of these three languages, OWL-DL has a wide expressivity and its foundation in Description Logic guarantees its computational completeness and decidability. *Description Logic* [14] is an important knowledge representation formalism unifying and giving a logical basis to the well known traditions of frame-based systems, semantic networks, object-Oriented representations, semantic data models, and formal specification systems. It thus provides an interesting logical framework to represent knowledge. On a more practical side, a growing number of software tools have been designed to process OWL-DL XML files and to put inference engines at work to query the ontology in order to execute processes in a system.

The first graph of figure 4[2] presents the upper level of the TELOS Conceptual ontology. In TELOS, the actors, the operations they perform and the resources they use or produce are all TELOS resources (shown on the graph by S "is-a-sort-of" links. They are represented as classes (rectangles) linked together with properties (hexagons) such as "perform" and "use or produce".

Some classes are further defined in sub-models that present sub-taxonomies of classes and their properties. The second graph of figure 4 shows the taxonomy of TELOS users and the corresponding operations they perform. The taxonomy of operation is further define in an Operations' subgraph (not shown here) where the operations are linked with the services (handlers) presented on figure 3. Another sub-graph describes the taxonomy of resources, including the very important concept of "content" package, which is redefined as resources having semantic descriptions. This sub-graph is presented partly as the third graph on figure 4.



[2] These graphs have been constructed using our MOT+OWL editor that exports to OWL-DL XML schemas [ ]

Figure 4: Part of the Virtual Campus Conceptual Ontology

3. From the Conceptual Ontology to the Technical Ontology Driving TELOS

We now explain how the conceptual ontology was revised, simplified or expanded to build the TELOS technical ontology that is integrated as code to drive the operation of the system

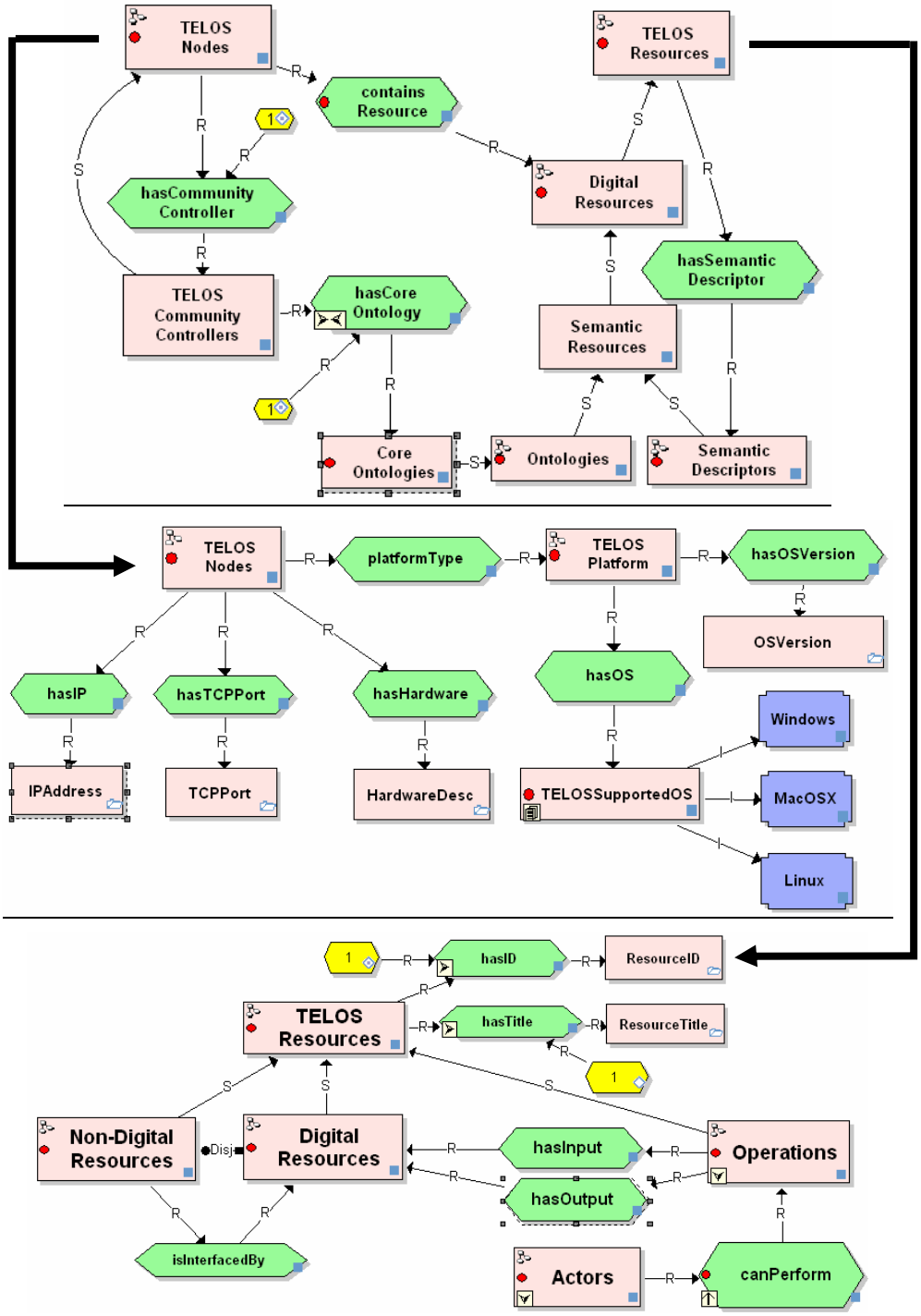


Figure 5: The upper layer of the TELOS technical ontology

First, we had to capture the distributed aspects of TELOS by adding the concept of a "TELOS Node" that was not present in the Conceptual Ontology but defined in the Conceptual architecture. The TELOS Global bus enables the interoperability between different TELOS nodes abstracting their particular physical platform and their network configuration. By connecting the Global Bus of many TELOS nodes we form a dynamic peer-to-peer network. This network may contain special nodes called community controllers which are basically centralized repositories for resources.

The first graph of figure 5 is the upper level of TELOS Technical ontology. It presents these concepts in relation to the concept of a TELOS resource which was the root of the former conceptual ontology. The second graph on figure 5 present a sub-model defining the concept of a TELOS Node, and the third one presents a more precise, upper level definition of a TELOS Resource very similar to the one in the Conceptual ontology. In these graphs, cardinality axioms (hexagons with numbers), disjoint axioms (Disj link) and functional property axioms have been added for more precision. This is essential because this ontology will have to respond to queries using an inference engine that do not tolerate ambiguity.

In these upper layer graphs, we find the interrelations between the main concepts of *Actors*, *Operations* and *Non-digital resources*. The Actors' and the Operations' sub-ontologies are directly imported from the Conceptual ontology presented on figure 4. The Non-digital resources' sub-ontoloy has undergone more important changes. It is presented on figure 6. We see that implementation concepts have been added such as *productID* and *locationURL*. Each of five main sub-class: *Documents*, *Atomic Resources*, *Actors*, *Aggregates* and *Digital Operations* is detailed in other OWL-DL models.

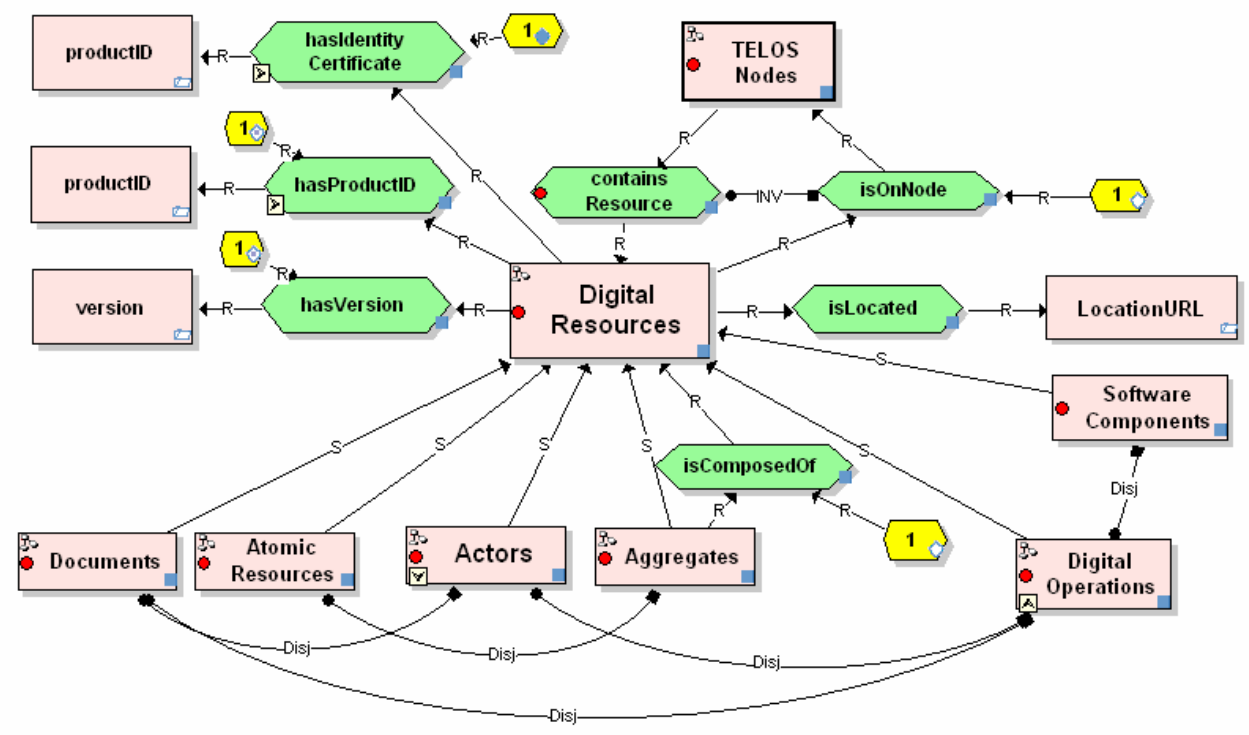


Figure 6: Part of the sub-ontology for Digital Resources

The resources are the persistent data of the TELOS node. The "Aggregates" sub-class is particularly important. This type of resource enables users to create new eLearning tools by gluing existing software components and other resources. It also enables users to model collaborative workflows or scenarios aggregating actors, the activities they perform and the documents, the software components they use or produce.

The semantic layer is defined as presented in the Conceptual ontology on figure 4. It is where all TELOS concepts are declared and related together through logical constraints in the Technical ontology. It defines the global behavior of TELOS. The semantic layer also contains the domain ontologies created by users that later permits semantic classification of the resources involved in the resource libraries using semantic descriptors. The semantic layer is the foundational element of the ODA (Ontology Driven Architecture) behind TELOS.

We will now examine how the actual TELOS prototype uses the Technical ontology for its operations. Figure 7 displays the TELOS interface in a Web browser, with the three main tools open: the Resource Manager, the Scenario Editor and the Task Manager.

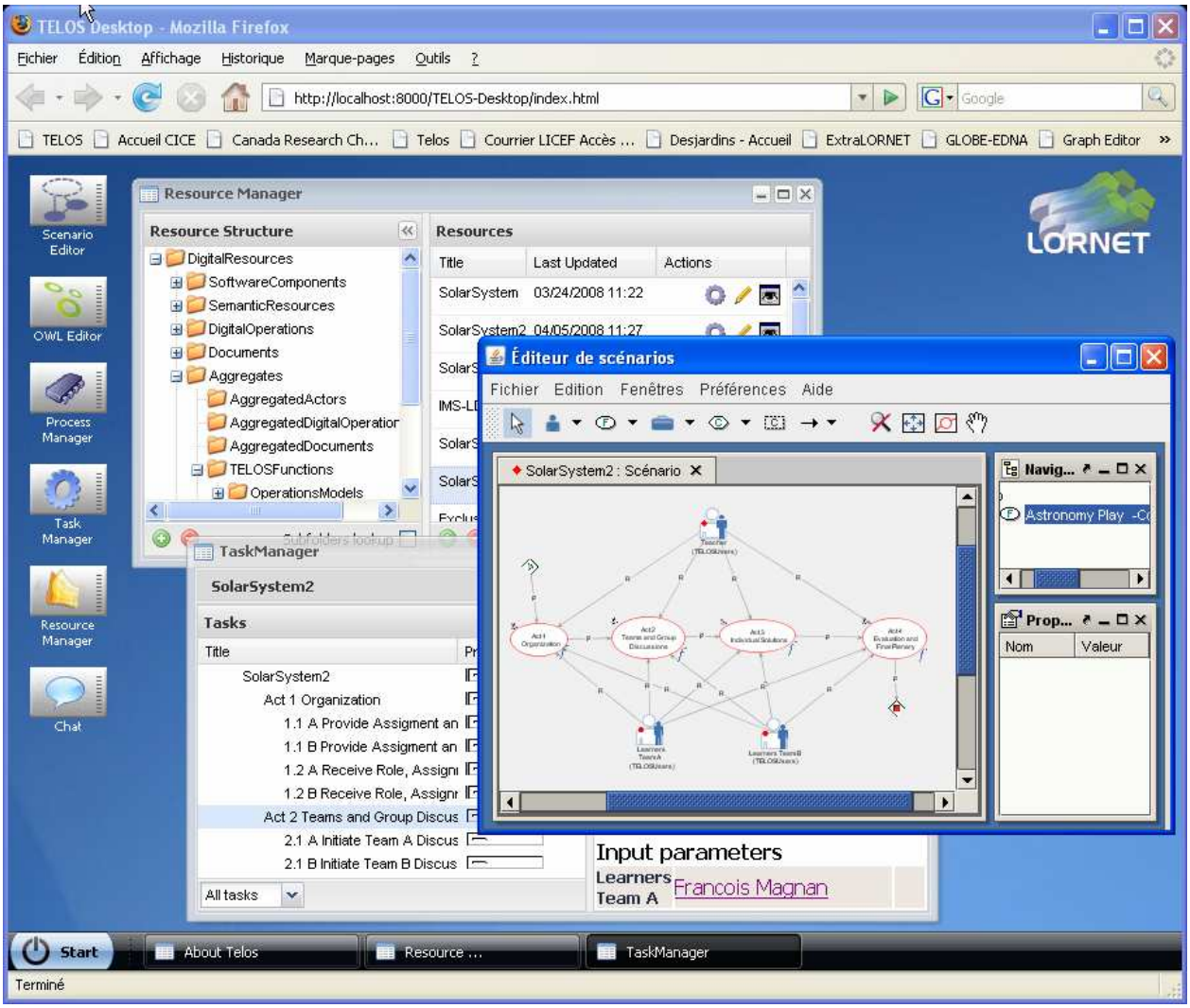


Figure 7: TELOS main interface

The *Resources Manager* gives access to all the available resources classified according to the technical ontology. On figure 7, the aggregates' class is expanded and the "My scenarios" class is selected showing the available scenarios on its right side. One scenario has been selected and is displayed in the *Scenario Editor* as a graphic interface showing a learning workflow involving three actors and four learning units organized sequentially. In the resource manager interface, it is possible to execute, edit or view any of the resources. The scenario editor provides the functionalities to link graphic objects in the scenario to the technical ontology and when this is done, users can run the scenario o by following the activity flow in the *Task Manager*.

4. Scenario Aggregation and Semantic Referencing.

Scenarios provide a high-level programming language for TELOS. This generic language is designed to be user-friendly to all TELOS users, including students/workers, teachers/designers or technologists and programmers/engineers, when they act, respectively, as composers at different levels of the cascade of systems. Figure 8 presents the Scenario Editor used to define and maintain scenario aggregates. The *Scenario Editor* aims to generalize both IMS-Learning Design methods as well as multi-actor business workflows.

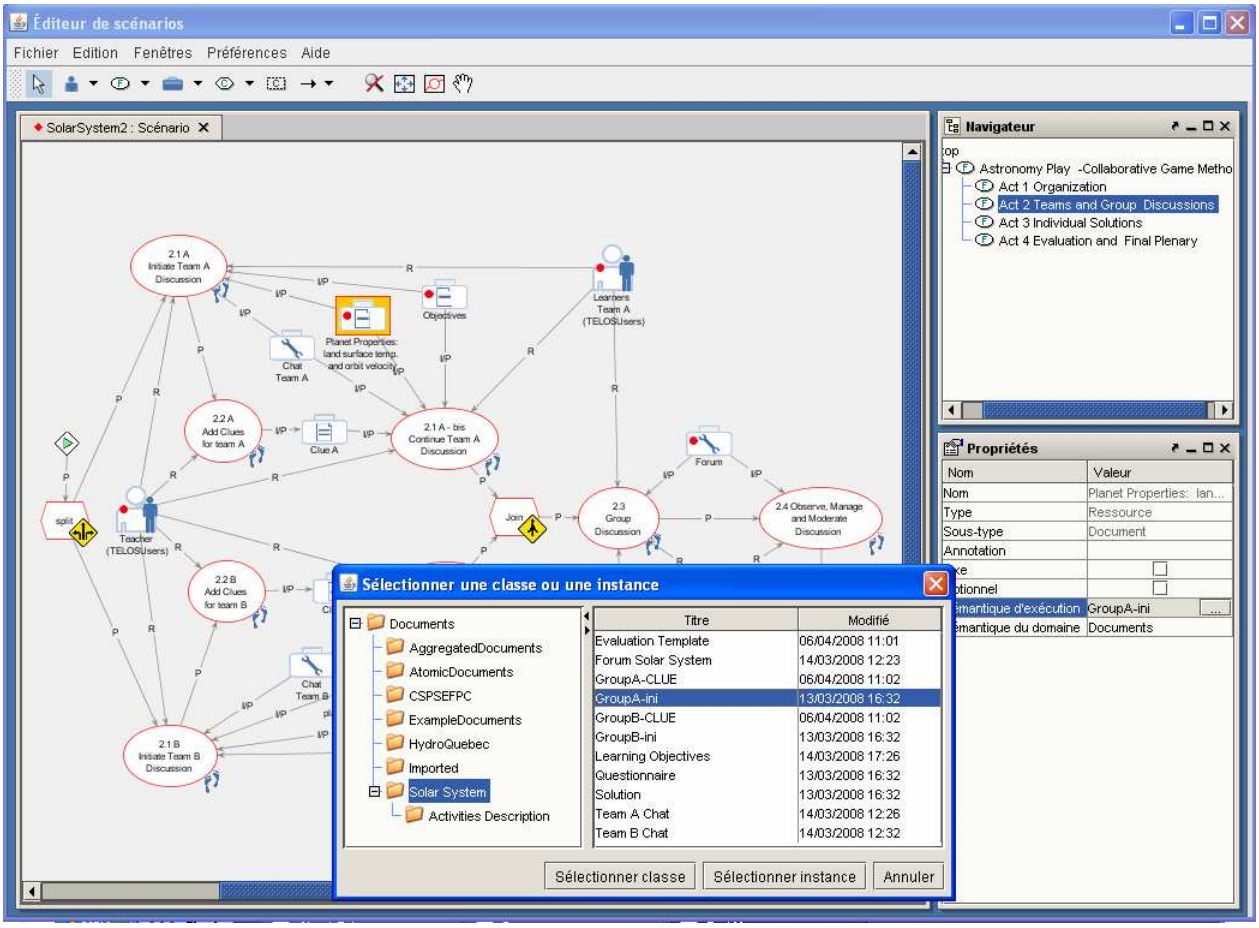


Figure 8: Linking scenario objects to classes or instances of the Technical Ontology

Various graphic symbols serve to represent actors, activities and operations, digital resources and conditions that influence the flow of control. All these objects are endowed with a semantic by linking them to the technical ontology. On figure 9, a document is selected and its execution semantic is given by selecting a class or an instance (GroupA-ini) of the technical ontology. In this way, TELOS knows what to do at the runtime, so the graphs can be processed by an execution engine called the *Scenario Evaluator*. The Scenario Evaluator will assure the coordination of actors, activities/operations/task and other resources in the scenario at delivery time. It enables engineers to combine resources into larger ones, technologist to built platform workflows and designers to build course designs or work processes for end users.

The scenario on figure 8 was built by a designer that has constructed part of a course involving a teacher and two student teams. The following examples are less common. The next example shows (figure 9) how a technologist combines an existing platform with TELOS. The central component of the extended platform is a scenario for the designer to produce courses. This design scenario corresponds to the central tasks of the MISA instructional engineering method [2, 4]. Figure 9 shows part of this scenario that involves using Concept@, Télé-université's actual course design platform, augmented with the TELOS scenario editor and other components.

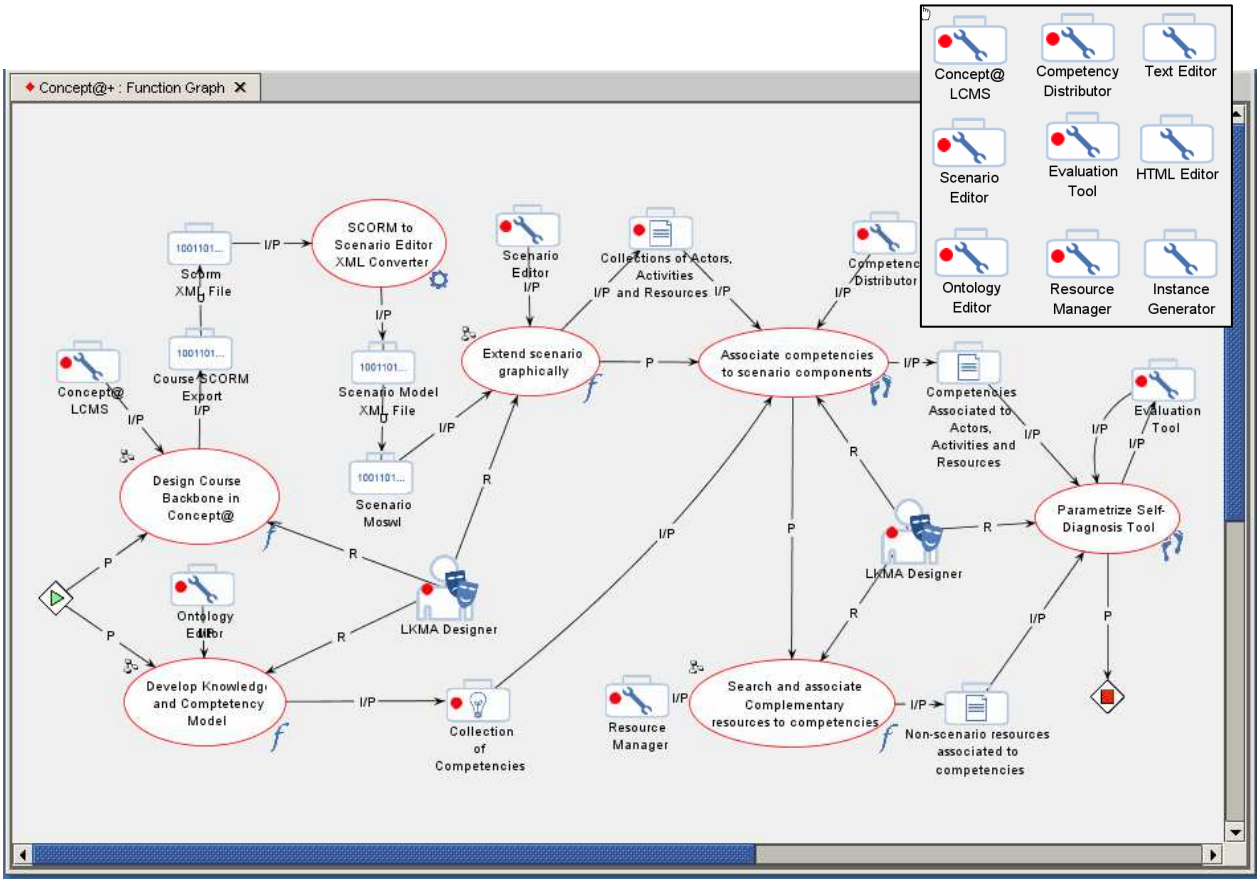


Figure 9: **Technologist constructing an augmented LKMS platform for designers**

The design scenario starts with two parallel functions performed by an LKMA designer: design of a course backbone using the Concept@ LCMS and development of a knowledge and competency model for the course using the TELOS ontology editor. Let us note that actually, Concept@ helps produce an activity tree representing the course plan with the subdivision of the course into modules and activities.

This structure can be exported to a SCORM package. Many roles can be defined in Concept@ but this exceeds SCORM's mono-actor capabilities. So information about roles/actors is lost when we open the corresponding graph in the TELOS scenario editor, produced by the TELOS operation that translates the SCORM file of the course structure to a graphic scenario. The next design phase proceeds graphically in the TELOS scenario editor to add the actors designed in Concept@ manually which is not a big task. There, more advanced flow of control can be added to better personalize learning based on knowledge and competency model.

Another example is the scenario on figure 10. It has been built by an engineer aggregating a new service embedded in an operation called the "Batch LOM Extractor". This operation takes a set of keywords, a number of LOM to find and the name of a destination folder in a repository of learning objects managed in the PALOMA software. The aim of this aggregated operation is to search Google with the given keywords, do some text mining on the resulting websites to extract some metadata according to the LOM standard, insert those LOMs into the requested PALOMA folder and open this folder into the PALOMA software interface to show the results to the user.

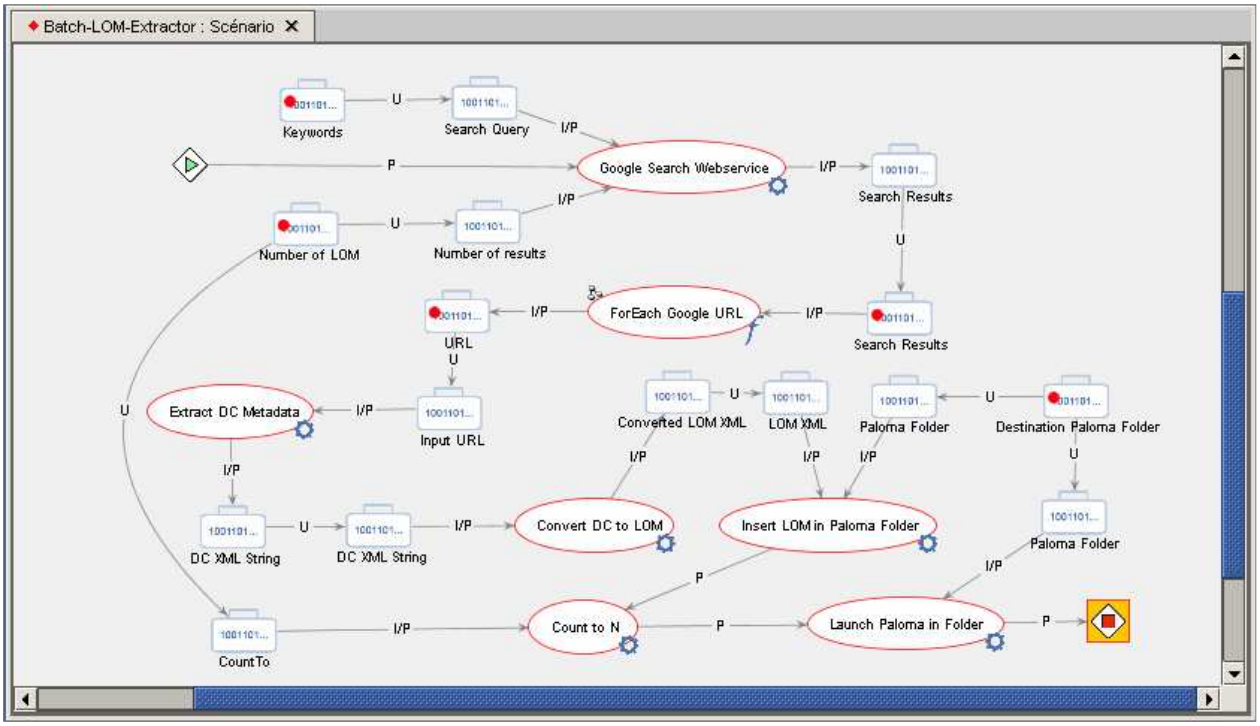


Figure 10: **Engineer constructing an operation aggregating services**

What we see here is the aggregation of software components built by different groups using different technologies that transfer data from one to another. The Google Search Service is launched using a SOAP Web service connector provided by the TELOS kernel. The Metadata Extractor is a C# component linked to the TELOS kernel by a C# connector. The DC to LOM conversion is a Scheme component linked through a Scheme connector. PALOMA is a Java applet linked through a Java connector.

5. Conclusion – The Main Process and the Expected Benefits

We now summarize the benefits we expect from this design and development process and more generally from ontology-driven systems.

1. *Fidelity from Requirements to Code:* The Capturing in an ontology of the main use cases and conceptual architecture concepts seem to improve the fidelity of the architecture to the requirements. Transforming the Conceptual ontology to a technical ontology embedded in the system ensure that the code will respect the architecture requirements. Also, the ontology-driven aspect of TELOS eases its evolution when new concepts will need to be integrated in the system.
2. *Global Systemic View.* The technical ontology is a Virtual Campus model (VC). It provides a global view to support cohesion of the activities, from the upper level where an institution can create a global workflow to coordinate the major processes to run a virtual campus, to the next levels of a design scenario-based platform and scenario-based applications.
3. *Extended set of actors.* Compared to the commercial LCMS in operation this new global approach leads to an unlimited set of actors. At any level, in principle, any number of actors can be defined and supported.
4. *Better process coordination.* The fact that the system holds a model of the VC processes and support resources leads to better process coordination. Especially in distance universities, this provides better assurance that the quality of services will be maintained when the personnel changes, especially when it must provide products of his activities to other actors.
5. *Visible scenarios and workflows*. Learning scenarios or workflows can always be consulted in a Web portal interface, changes to components or actors can be seen right away. Each user taking an actor's role can visually see the context of the activities he has to perform, what resources to use or produce and with whom he is to interact with.
6. *Flexible and adaptable environments*. Each environment operating according to a technical ontology which is part of the system enables very flexible and adaptable environments. If a new kind of actor, activity or resource needs to be introduced, this is done simply by modifying the instances or classes of the ontology, without changing the main operations of the system.
7. *Resource reusability* is a goal pursued by many advocates of learning object repositories, but it is not that easy to achieve. Using ontologies to annotate each resource within the same framework, and adding connecting operations to take care of possible technology mismatches brings solutions to resource many reusability problems.
8. *System interoperability*. With this new VC model, it is possible to bring different technologies and different platforms to work together. For example, a designer could built a course using a scenario editor in one platform, and transfer it to another platform to add new functionalities, for example personalized assistance. This process can be designed by defining the aggregation scenario between platforms at the LKMS level.
9. *Modeling for all*. Modeling is not an easy task but it is important enough to made it accessible not only to engineers and technologists, but also to instructional designers, learners and trainers.
10. *Focus on learning and work designs*. Finally, we hope the proposed approach to Virtual Campus modeling and operation will reduce the technology noise that is often present in eLearning applications when too much time is devoted to solving pure technology problems, instead of focusing on learning problems. We shope the activities will be more focused on pedagogy and quality of educational services.

These approaches offer new possibilities but also pose additional challenges. The LORNET five year research project ending, some considerable refinements will happen. We will also need to ensure a transparent use of the tools that will be novel to most users. But our hope is that the results achieved in this project will lead the way to future research and developments and fruitful applications to Web-based learning and knowledge management systems.

References

[1] Paquette, G. (1995) Modeling the virtual campus. in Innovating Adult Learning with Innovative Technologies (B. Collis and G. Davies Eds), Elsevier Science B.V., Amsterdam

[2] Paquette, G. (2001) Designing virtual learning centers. In H. Adelsberger, B. Collis, J.P.E., ed.: Handbook on Information Technologies for Education & Training. International Handbook on Information Systems. Springer-Verlag 249–272

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

[4] 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 and Proceedings of the UNFOLD-PROLEARN Joint Workshop, Valkenburh, The Netherlands, September 2005 on Current Research on IMS Learning Design.

[5] Wilson, S., Blinco, K., Rehak, D.: Service-oriented frameworks: Modelling the infrastructure for the next generation of e-learning systems. White Paper presented at alt-i-lab 2004 (2004)

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

[7] Davies, J., van Harmelen, F., Fensel, D., eds.: Towards the Semantic Web: Ontology-driven Knowledge Management. John Wiley & Sons, Inc., New York, NY, USA (2002)

[8] Kleppe, A. G., Warmer, J. B., & Bast, W. (2003). MDA explained : the model driven architecture : practice and promise. Boston: Addison-Wesley.

[9] Paquette, G., Rosca, I.: Modeling the delivery physiology of distributed learning systems. Technology, Instruction, cognition and Learning 1-2 (2003) 183–209

[10] Rosca, I. (2005) TELOS Conceptual Architecture, version 0.5. LORNET Technical Documents, LICEF research centrer, Télé-université, Montreal, 2005

[11] Paquette, G., Rosca, I., Mihaila, S., Masmoudi, A.: 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 (2006 in press)

[12] Paquette, G., Rosca, I., Masmoudi, A., Mihaila, S.: Telos conceptual framework v0.8. Lornet technical documentation, Télé-Université (2005)

[13] W3C (2004) OWL Overview Document (http://www.w3.org/TR/2004/REC-owl-features-20040210/)

[14] Baader, F., D. Calvanese, D.McGuinness, D. Nardi, P,Patel-Schneider, editors (2003) The Description Logic Handbook. Cambridge University Press.

[15] Paquette G. and Rogozan D. Primitives de représentation OWL-DL - Correspondance avec le langage graphique MOT+OWL et le langage des prédicats du premier ordre. TELOS documentation. LICEF Research Center. Montreal, Québec, 2006.

[16] IMS-LD (2003). IMS Learning Design. Information Model, Best Practice and Implementation Guide, Binding document, Schemas. http://www.imsglobal.org/learningdesign/index.cfm, last retrieved October 3, 2003

[17] Magnan, F.: Distributed components aggregation for elearning: Conducting theory and practice. ILOR2005 Conference, http://www.lornet.org/presentation i2lor 05/papers/i2lor05-03.pdf (2005)

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

[19] Marino O. et al., Bridging the Gap between e-learning Modeling and Delivery through the Transformation of Learnflows into Workflows , in S. Pierre (Ed) E-Learning Networked Environments and Architectures: a Knowledge Processing Perspective, Springer-Verlag.

[20] W.M.P. van der Aalst, A.P. Barros, A.H.M. ter Hofstede, B. Kiepuszewski, Advanced Workflow Patterns, 7th International Conference on Cooperative Information Systems (CoopIS 2000)

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

[22] W. M. Johnston, J. R. P. Hanna, and R. J. Millar. Advances in dataflow programming languages. ACM Comput. Surv., 36(1):1--34, 2004.

[23] Germain G., Feeley M. and Monnier S. Concurrency Oriented Programming in Termite Scheme, Proceeding of Scheme and Functional Programming 2006, Portland

[24] Armstrong J., Virding R., Wikström C. and Williams M. Concurrent Programming in Erlang. Prentice-Hall, second edition, 1996.

[25] Boley H. Functional RuIeML: From Horn Logic with Equality to Lambda Calculus, Upgrade Vol. VI, issue no. 6, December 2005, CEPIS

[26] ELF – eLearning framework, Web site at http://www.elframework.org/, last consulted June 14, 2007.

[27] OKI – Open Knowledge Initiative, at http://www.okiproject.org/, last consulted June 14, 2007