Integrating Learning Management Systems and Practical Learning Activities: the case of Computer and Network Experiments

Amine Bouabid, Philippe Vidal, and Julien Broisin
Institut de Recherche en Informatique de Toulouse
{bouabid,vidal, broisin}@irit.fr

Abstract

Even if virtualization technologies are mature enough to deliver remote experiment in a distance-learning context, they are not well-integrated into existing learning environments; as a consequence, they do not provide teachers with any mechanisms to get feedbacks on actions handled by learners. To tackle these issues, this paper presents an approach based on the Web-Based Enterprise Management (WBEM) standard. Full integration of learning and experiment contexts is ensured by an intermediate layer acting as a bridge, while tracking mechanisms stand on (1) an abstract modeling of experiments and learners’ information from an observation point of view, and (2) some components able to extract, store and retrieve data specified within the model. We demonstrate how this approach can be extended to satisfy additional requirements, and present an implementation based on existing open source software that validates our works.

1. Introduction

Hands-on lab works represent important learning activities for computer engineering education [1], especially when users aim at reaching specific pedagogical objectives [16]. In the Technology Enhanced Learning (TEL) context, there is great demand on remote hands-on activities to benefit from their pedagogic and economic advantages [17].

For the specific computer engineering education area, virtualization technologies are gaining popularity over classical ones [11]. They allow easily building realistic, complex, controllable and repeatable emulated computer networking experiments.

Even if these technologies satisfy most expectations from a technical point of view, providing learners and tutors with a remote access to these environments is not sufficient to reach learning effectiveness: remote practical learning activities should be integrated within instructional units where “learning activities should be designed with clear learning outcomes in mind” [16]. Also, both learners and tutors are not aware of actions performed over the experiment; as a result, it is very difficult, even nearly impossible, for distant tutors to guide and help distant learners when they encounter problems or blocking situations [10].

In this paper, we present a framework that tackles issues mentioned above. Our approach stands on (1) the introduction of an intermediate layer between the learning and experiment contexts in order to provide a transparent integration of these environments, and (2) the widely adopted Web-Based Enterprise Management (WBEM) initiative to model, gather and share tracking information. The WBEM standard [4] brings a solution to unify observation (even management) of distributed and heterogeneous computing environments, and includes a set of models to describe users and computer environments to supervise. The next section presents our hands-on lab context by defining network and system experiments, and identifies requirements that must be taken into account in order to reach our objectives. Section 3 presents our modeling of network experiments and learners activities, while section 4 introduces the distributed architecture supporting both integration of learning and experiment contexts, and management of tracking data described within our modeling. Finally we present benefits of the approach before concluding and presenting some further works.

2. Experiment definition and requirements

An experiment in our context is a network of various computer systems, protocols and distributed applications more or less complex according to the pedagogical objectives of the learning activity. Figure 5 illustrates a typical example including various equipments and networks.

Current LMS do not support effective learning and teaching activities using remote labs because they lack of (1) awareness about experiments’ evolutions, (2) control over the experiment, and (3) communication with tools supporting experiments. Therefore, the
LTSC’s Agent/Tools communication work group defined a generic communication protocol between learning tools (such as experiments) and instruction agents (such as LMS) aiming at [12]: (1) allowing agents to observe learner’s actions on tools, (2) allowing agents to provide teachers and learners with feedbacks, (3) allowing communication between agent and tools, and (4) allowing agents intercommunication in order to share information and to ensure learners’ assessment and guidance. In order to take into account these recommendations, our approach stands on:

- The modeling of hands-on labs in order to describe experiments from an observation point of view. This modeling also takes into account learners’ activities, thus facilitating and enhancing analysis of learners’ behavior during hands-on lab works.
- The introduction of components within LMS responsible for interactions with experiments hosted by remote testbed systems on one hand, and able to collect and externalize tracking information described within the model on the other hand.
- Additional components integrated within testbed systems in order to collect and store states of nodes and networks part of experiments.
- An intermediate layer acting as a bridge between learning and experiment contexts.

3. Modeling of computer and network experiments

Within the network and system domain, the Distributed Management Task Force (DMTF) elaborated the WBEM standard [4] to unify management and supervision of distributed computing environments. Ubiquitous operating systems such as Microsoft™ Windows, UNIX or some Linux distributions natively integrate a WBEM-compliant tool to ensure their remote management. Our approach consists in reusing this initiative to achieve our objectives. Indeed, the Common Information Model (CIM) specifies a metamodel providing a unified view of heterogeneous computing environments while taking into account extensibility through a high level of abstraction so that additional pedagogical requirements could be taken into account.

3.2. Modeling of experiments

Our experiment model has been built according to the following methodology:

- Defining common requirements of computer and network experiments (see section 2).
- Studying existing CIM schemas and selecting those meeting our needs.
- Building a model based on existing CIM schemas and introducing new classes specific to TEL domain. This model must ensure extensibility through a high level of abstraction so that additional pedagogical requirements could be taken into account.

![Generic experiment modeling](image)

**Figure 1.** Generic experiment modeling

The class `TEL_ComputerExperiment` illustrated on Figure 1 is the basis for designing an experiment. It represents an aggregation of two components: nodes represented by the class `ComputerSystem`, and topologies modeled by the class `TopologyGraph`. As it appears on Figure 2, the modeling of the experiment is divided into three parts (for readability reasons, classes’ properties are not included):

- Computer systems’ description includes main hardware devices, operating system, file system and pre-installed software. These classes are defined in the CIM Computer System model [8].
- Networks’ description defines physical and logical networking properties defined in the CIM Network model [5].
- Topologies’ description models the physical and logical network architecture [5].
Since CIM is natively dedicated to system and network management, existing classes are suitable to elaborate a model describing experiments from an observation point of view. However, none of the models describe activities performed on experiments. To fill this lack, we suggest in the following section a CIM extension to describe learners’ activities.

3.3. Modeling of learners’ activities

As mentioned in section 2, there is a need for observing learners’ activities in order to provide tutors with data that can help them in the process of guiding and assessing learners involved in an online experiment.

The CIM User model [6] identifies a user with the class Identity (see Figure 3), and specifies other classes dedicated to administration information such as the user name or given name, the mail address or the home phone number; it also provides details related to users’ preferred language or organization name.

Based on the native User model, Figure 3 introduces classes and relations making it possible to record activities performed by users on any element of the global model. A dependency relation is established between a user and an element, and this pair is bound to the abstract class TEL_Activity that represents any TEL activity. The class TEL_ExperimentActivity specializes the root activity class in order to represent activities related to experiments. At this time, our model only takes into account activities performed through command line interface.

The whole modeling presented in this section offers teachers the opportunity of specifying systems, networks and software to deploy within the experiment, while describing information about activities handled by learners over experiments.

4. A distributed architecture for integrated experiments

Figure 4 illustrates the global framework allowing for both integration of experiments within learning context, and tracking of learners’ activities defined within our modeling.

4.1. Support systems

The learning context is composed of an LMS integrating additional components to transparently access to hands-on lab works on one hand, and others elements able to collect observation data on the other hand. The experiment interface allows teachers to design experiments, and provides learners with a web-based terminal to handle operations on remote experiments. The agent is responsible for capturing and propagating users’ activities to both components of the intermediate layer, so that actions are stored into a WBEM-compliant tool and also executed on the remote experiment. The visualization interface is a dashboard used to display feedbacks about experiment’s evolutions and learners’ information.

The experiment context comprises a testbed system composed of two main components. The WBEM-compliant tool manages tracking information by providing both a repository to store CIM and TEL classes and instances, and a tracking manager ensuring safety and integrity of data. The virtualization tool runs network and system experiments, and forwards experiments’ status to the tracking manager. The CIM provider, a component
specific to environments being supervised, achieves this task.

The intermediate layer integrates two services and acts as a bridge between learning and experiment contexts. By communicating with the tracking manager, the WBEM interface of the tracking service creates or updates CIM and TEL instances matching with tracking data sent by LMS agents and received through the SOAP interface. The experiment service processes actions handled through the experiment interface of the LMS by executing these actions on the virtual experiment.

### 4.2. Current implementation and technologies

The framework has been implemented to set up a basic network experiment: teachers had to (1) design the network topology illustrated on Figure 5 from a LMS, and (2) deploy the experiment on a remote testbed system. Pedagogical objectives behind this hands-on lab-work is to learn basic networking principles by configuring IP parameters and routing.

Moodle [13] has been deployed within our architecture and extended with the experiment GUI illustrated on Figure 5. An agent collects learners’ profiles from Moodle data, whereas learners’ activities are gathered through the web-based terminal.

![Figure 5. The Moodle experiment editor](image)

The Open Pegasus software [15] has been installed on a Linux-based testbed system, and the experiment model designed in section 3 has been compiled within the tracking repository; this scalable and platform-independent environment performs better than others existing tools [18]. The testbed system executes Manage Large Network (MLN) environment [14], a virtualization tool designed to manage virtual networks based on the well-known and mature XEN, UML and VMWARE technologies.

The SOAP interface of the tracking management service has been developed as a C++ web service, and implements several methods dedicated to learners’ activities. Treatments associated to these methods consist in querying the tracking manager to create/update or retrieve CIM and TEL instances of the model. The CIM Query Language [7] elaborated by the DMTF and implemented by Open Pegasus is exploited to achieve this task.

The experiment service currently exposes two methods. The first one is dedicated to teachers and (1) accepts WBEM-formatted files transferred by the Moodle experiment interface, (2) ensures the mapping of these files towards the MLN format, and (3) deploys the experiment on the MLN virtual environment through its management interface. The second method receives commands handled by learners through the terminal embedded into Moodle, and effectively performs operations on the MLN experiment.

Some works are still under progress. Since the CIM provider specific to MLN is not implemented yet, status of virtual equipments integrated within the experiment are not dynamically extracted; instead, they are deduced from learners’ actions. However, the DMTF’s Virtualization model [9] provides a standard format for packaging and describing virtual machines and applications for deployment across heterogeneous virtualization platforms. Some open source implementations of CIM providers already exist (especially for the XEN technology [2]) and constitute a solid basis for implementing the MLN provider.

On the learning context side, the visualization dashboard has just been specified and technologies have been identified. Its development represents our main concern, so that teachers and learners can benefit from a user-friendly feedback interface.

### 5. Conclusion and future works

This paper presents an approach based on the WBEM standard to setup a full integration of online computer networking experiments within a learning context composed of LMS. Our proposal ensures observation of experiments and learners activities, thus providing teachers and learners with feedbacks helping them to enhance teaching and learning processes. The framework stands on the modeling of information to collect and store, and introduces a distributed architecture to support learning and experiment contexts and able to manage and share tracking data.

Existing models specified by the WBEM standard have been studied and reused to design an abstract experiment model describing common computer and networking equipments, but also learners’ activities. Thanks to its high abstraction level, our modeling can be widely extended to satisfy others specifications. The computing environment model offers the opportunity to integrate additional elements or requirements, since it aggregates entities presenting very high levels of abstraction. Moreover, activities can be specialized from abstract classes, thus allowing for description of extra users’ actions.
The distributed architecture ensures (1) transparent communication between learning and experiment contexts, (2) observation of testbed systems running online experiments, (3) observation of learners’ activities within learning context, (4) storage of tracking data within a central repository, and (5) delivery of feedbacks through a graphical user interface. An implementation based on open source software validates our approach: Moodle offers the opportunity to handle operations on a remote experiment running on the MLN virtualization environment, and to store tracking data into the Open Pegasus WBEM application.

This 3 tiers architecture can be easily upgraded to N tiers architecture to provide scalability and availability: the intermediate layer can be duplicated on several physical servers, whereas the WBEM standard ensures manager-to-manager communications: tracking data stored into various repositories can be exchanged from one storage system to another. In addition, several physical hosts running MLN can cooperate to setup distributed virtual experiments; this may be useful for complex and resource-consuming practical activities.

Finally, to promote sharing and reusing of experiments, we plan to exploit some previous works [3] related to learning object virtualization to transparently index experiments’ models as learning object into the ARIADNE learning object repository. Models could be easily retrieved and integrated within any WBEM-compliant tool, thus promoting adoption of our approach.

6. References


