Effectively Integrating e-Assessments in the Design of Virtual Learning Scenarios

Jorge Torres  
DASL4LTLD  
Tecnológico de Monterrey  
Querétaro, México

Juan Manuel Dodero  
Escuela Superior de Ingeniería  
Universidad de Cádiz  
Cádiz, Spain

Ignacio Aedo  
Departamento de Informática  
Universidad Carlos III de Madrid  
Madrid, Spain

Abstract— The design of e-learning oriented assessment poses some technical issues when it has to be designed and integrated with the elements that describe a learning process to be deployed in a virtual learning environment. Current web service-based integration solutions are not completely satisfactory from the authoring perspective, in order to provide an integrated approach that seamlessly assembles the design, deployment and enactment stages of a learning process. This work summarizes the integrated authoring approach of LPCEL and explains how its editing environment can be used to design a service-based assessment along with the activities, resources, and user roles in a project-oriented learning course case study.

Keywords - learning design; web services; e-assessment

I. INTRODUCTION

Educational modeling is concerned with the design and delivery of activities and resources that assemble the scenario in which a virtual learning experience can take place. Educational Modeling Languages (EMLs) provide the framework to formally design such learning scenarios, including the tools needed to represent all the activities, resources, services, and user roles [16]. EMLs such as IMS LD [17] are extensively used to represent Complex Learning Processes (CLP) integrating dynamic resources and collaborative learning methods in the learning process definition (Dodero et al., 2005). The smallest unit of work of a CLP is known as Unit of Learning (UoL), which is formed by the events that students need to complete in order to achieve specific learning objectives. The IMS LD specification can fulfill a wide spectrum of educational modeling situations and learning scenarios [33]. Some situations, however, can be hardly represented with IMS LD, either because of the expressiveness of the language [31] or the under-developed authoring tools that are necessary to design advanced UoLs [22]. For example, the design of elaborate flows of learning activities [31] and transactional web service-based activities [32] present drawbacks if using only IMS LD. The definition of learning assessments has also been a target of extensions to IMS LD [28]. To overcome the former issues, the Learning Process Composition and Execution Language (LPCEL) provides a framework to design CLP-based courses [30]. The LPCEL can be used to specify complex and dynamic flow structures, and it can describe the execution of a learning process as well as the resources and services needed to complete it. The objective of this paper is to describe how the web service integration facilities of LPCEL can be used to define and deploy effective e-learning assessments (e-assessments) that are seamlessly integrated with the overall design of the CLP.

II. E-ASSESSMENT: MOTIVATIONS AND ISSUES

Effective learning assessment aims at empowering students in the development of advanced assessment capabilities, on the basis of applying self-assessment and collaborative (e.g. peer-to-peer) assessment procedures between students. Students have to use assessment instruments designed by the teachers to assess the learning outcomes [1] and must follow assessment procedures that are seamlessly combined with the rest of learning activities. Learners can thus develop their own skills of evaluation and assessment, which are acquired by making judgments on their work or their peers’ work, and by providing explanations for those judgments [26]. Effective assessment equips learners to study and perform to their best advantage in a CLP and to progress with confidence and skill as lifelong learners [15].

Procedures and instruments for an effective e-assessment are based on inspecting the learning outcomes as explicit deliverables created by the learners in the Virtual Learning Environment (VLE), and then filling in an assessment instrument (e.g., rubrics and valuation scales) with their appraisals [36]. Students find it difficult to learn how to apply such assessment instruments without a set of guidelines of how to conduct themselves in the procedure. When using a CLP in an e-learning environment, autonomous peer-to-peer and self-assessment of the learning outcomes is especially difficult. For instance, in a Project-Oriented Learning (POL) experience every project should encompass an explicit set of outcomes as multiple products that must be developed to give students opportunities to demonstrate their learning and that form the basis for the assessment plan and rubrics as the assessment instruments [19]. The teacher can successfully manage the process of learning and assessment by using tools and strategies that bring structure to the process, and help students to learn to assess the outcomes of the CLP.

Using IMS LD to describe a CLP, the collaborative assessment procedure can hardly manage the interaction with the data and information of a structured assessment instrument such as a rubric. IMS LD allows creating interactive UoLs by injecting the UoL with special user interface elements that
allow its users to get and set values of single properties, which assemble the IMS LD implementation of the data exchange with an assessment instrument. This procedure has been used with simpler assessment instruments such as QTI [34]. Generic service-based approaches [8] can be used to integrate third party assessment services in the scripting of the CLP. Authoring the connection between parts of the CLP and a collaborative assessment service is still difficult on the basis of IMS LD linguistic primitives.

III. DESIGN OF VIRTUAL LEARNING EXPERIENCES

When designing a virtual learning experience, the resources and services in which it is based on have to be also provided or described. After deploying the course, if a given resource is not available, some adjustments should be done in the design phase to replace the resource. In general, the author of a learning experience should be aware of three stages in the lifecycle of the CLP, namely design, deployment and enactment [9]:

- The design phase involves defining the didactical method and the structure of activities within which learning resources and services are going to be run, and to package all in a self-contained UoL. Web services cannot be packaged, but described to prepare its execution, sometimes out of the virtual learning environment. Visual instructional design languages are frequently used for authoring [3][18]. Design tasks are more difficult if the designer only has scattered design items as activities, resources, services, and roles, which must be combined and assembled to build a UoL. That is the rationale after using learning design patterns [23], which provide reusable collections of elements that avoid the implementation details of the UoL, especially for collaborative learning [12].

- In the deployment phase, space and resources of the VLE is allocated to host all the elements contained in the UoL. Activity structures are defined according to the course format and facilities of the environment, which is prepared to run the flow of activities defined by the UoL and associate its user roles with actual users. For this stage it is especially relevant the data representation of the UoL and how it is connected with external applications and services. These are aims of exchange EMLs [21]. A deeper analysis of existing approaches to exchange EMLs can be found in [7].

- The last phase is to enact the course in a VLE, such that participants can start interacting among themselves and with the learning resources. In the case of interacting with web services, the environment has to configure the execution context with information that is not part of the UoL, such as the actual number of participants per activity, the number of instances of each application or service (e.g. forums, wikis, etc.) that deploys the learning activities, and the association of actual users to service instances according to the didactical method of the UoL. Such course enactment issues require a too tight coupling between an IMS LD UoL and its learning services [35]. The generic service integration method [8] facilitates enactment by means of a thorough description of IMS LD language primitives, such as properties, which are provided by service adaptors.

These phases of a CLP lifecycle are not completely independent from each other. Therefore, an integrated approach for learning design tools is needed that takes such dependencies into account. Despite the focus on the design phase for this work, we must also cater to the dependencies on deployment and enactment phases, particularly considering web service interactions as an essential part of the design tasks. In the following we study the related work on the former issues, which have relevance in designing a CLP for a virtual learning experience.

A. Editing the learning process

A number of visual editing tools have simplified the design of CLPs, which are delivered often as IMS LD-compliant UoLs. The ReCourse editor [11] can display the flow of activities of a UoL, but it has not visual support for the level B of the IMS LD specification. CompendiumLD [2] resembles a mind-mapping tool where sequences of tasks can be designed, but it doesn't translate designs to any standard EML. Other tools such as MOT+ [25] and ASK LDT [27] enable to visually edit only IMS LD level A-compliant UoLs. The LAMS system [4] provides a group of tools to design, manage and deploy learning scenarios, using its own graphical representation and EML. The LAMS editor provides a pre-defined catalog of tools that package the interaction with off-the-shelf applications and services, but does not provide a generic web service interaction editing facility. The FlexoLD visual editor [6] manages the IMS LD level B properties through high-level abstractions (e.g. branches and loops) that generate a UoL ready to run in an IMS LD engine or in a specific VLE (e.g. Moodle), but has no visual facilities to editing the web service interactions.

The LPCEL Editor uses a subset of the Business Process Modeling Notation (BPMN) 2.0 [24] as the visual layer to create an abstraction for the LPCEL primitives. The BPMN is a graph-oriented language where the flow of the nodes can be specified in a simple fashion. The subset used by the LPCEL Editor can be divided into three groups: Basic Elements, Simple elements and Artifacts. Using these elements, the user can be more focused on the pedagogical intention and the learning objectives of a given learning course, without being aware of the XML implementation details for the same scenario.

B. Configuring the use of web services

A web service is a software system designed to support interoperable machine-to-machine interaction over a network that is normally described using the WSDL standard [37]. There have been proposals of integrating web services in a CLP. The IMS Tools Interoperability Specification gives a set of rather abstract guidelines to integrate third party tools in a VLE. The CopperCore Service Integration (CCSI) is a concrete IMS LD-based solution that considers services as a type of functional concept supporting a user in the learning process [35]. CCSI provides an ELF-based API for interacting with e-learning services but tightly couples the UoL with the service,
not facilitating the design and deployment of the UoL in different VLEs or with different services. Instead of unrealistically requiring all these services to be present at design time, the Generic Service Integration (GSI) approach aims at orchestrating all the functions remotely provided by external services [9]. GSI is conceived as an extension of the IMS LD functionality, so the VLE in which the UoL is to be deployed must be IMS LD-compliant.

The LPCEL Editor supports the integration and testing of web services independently of the deployment platform. It provides the tools needed to include and test educational web services for learning scenarios. The objective of this functionality is to facilitate the inclusion of educational web services by means of a simple wizard instead of asking for XML tags and attributes. Basically, the user only makes use of a wizard where some of the key components of the web services are requested (the URL, service, operation, etc.) while the remaining information is gathered automatically. Before the process is completed, the editor provides the mechanism to test the web services so the user can be sure that the selected service is still useful. The objective of testing a web service during the design stage is to verify if a particular web service or operation is still available or functional, this way the user can check if the service is still operational in order to avoid runtime errors in the execution stage of the learning scenario.

IV. EDITING AN ASSESSMENT IN A LEARNING PROCESS

The LPCEL Editor has been subject to a case study evaluation [13] in a course that follows a POL strategy [20]. During the course, the students must deliver an incremental version of the system that must be assessed; each milestone consists of the following tasks: the leader manages the requirements and assigns them to the other team mates; the programmer implements the requirements as planned; the tester verifies the work done by the programmer; and the documenting one writes the software documentation. Once these stages are done, it is first evaluated first by the client and later by teachers, who assess students according to the specific criteria of an evaluation instrument (i.e. a rubric). Finally the teachers decide the final grade for each one of the students in their respective course. The following steps describe the method of authoring a CLP that uses an external web service to assess the students’ work.

A. Choosing a pattern

The use of patterns for the LPCEL editor can be achieved by using the tools it provides to store, group and reuse the patterns preloaded within the tool, but also for those created by the user. To illustrate the different levels of detail for the pattern classification presented before, we make us of the same case study.

The modality presented in the case study is based on the Project Oriented Learning (POL) method. The POL method consists of a set of high-level sequence of activities and roles [20]. This high-level detail of activities can be seen in Fig. 1, and it presents a set of complex activities such as the parallel, sequence, and sub-process. Notice that this is only a part of the complete POL process, but is a good example of the nature of the high-level classification, because it only presents some high-level activities but not the specific tasks.

B. Visually editing the learning process

Using the visual elements provided by the LPCEL Editor can facilitate the editing task. The same course design and sequence of activities mentioned in the case study can be represented using the LPCEL Editor visual approach as depicted in Fig. 2. One of the main properties of the visual representation is how easy it presents the pedagogical intention of the sequence of tasks, and how intuitive and understandable it can be for the teacher. It is important to notice how the learning scenario structure (sequence of activities, the roles and which tasks every role performs) can be understood just by analyzing the visual representation. Using this visual abstraction the user can focus on the learning objectives and how the complete learning scenario must be executed rather than the low-level implementation details.

If the user needs a more advanced low-level arrangement of tasks for the sequence activity, it can be facilitated using the LPCEL Editor. For instance, if the individual activity for the learner is a very complex set of resources and activities, then it is easier to use the editor rather than write the XML tags and structure, i.e. if it was difficult to divide a basic activity into a complex sequence then it would be more complicated to create a complex structure like the one depicted in Fig. 3, where an external resource is specified for the activity A.
On the other hand, there is also an intermediate representation, which is based on JSON format. The file can serve several purposes, e.g. as an exchange file (other than XML) and it can also be read and understood by the user. Because of the human-readable property of the intermediate representation, the user can understand the complete learning scenario without uploading it first into the tool. This property becomes important when the user only needs to do minor adjustments to the intermediate representation.

Using the intermediate representation, it is possible also to generate courses compliant with a specific EML. The EML-specific transformations are to be implemented in the LPCEL editor, as they have been tested in the model-driven LDDSL approach [7]. Although that s out of the scope of this work, it must be noted that a one-to-one mapping between two different EMLs is not always possible without losing some expressiveness.

C. Adding an assessment by editing a web service

In the case study presented before, the students need to take an assessment activity at the beginning of the course in order to obtain their level of knowledge about the topics. After the test is completed, the students are presented with custom activities according to the grade obtained by the student in the assessment in order to customize the delivered content. Using the visual tools, this task was done using one simple icon (the resource element). However, the LPCEL specification requires some information about the web service that is being set for the activity. The user would also need a web service client (web form) in order to use the selected web service when the learning process is executed.

The following listing presents only the critical information (URL, Service name, port, namespace etc.) that has to be set to describe a web service in LPCEL.

```json
{
  "id":6,
  "elementType":"dataElement",
  "complexType":"basic",
  "name":"Assessment",
  "description":"EvalComix rubric assessment",
  "resource":{
    "type":"remote",
    "url":"http://avanza.uca.es/assessmentservice/wsdl",
    "port":"AssessmentSoap",
    "operation":"Get"
  },
  "serializable":true,
  "position":{
    "x":112,
    "y":172,
    "width":50,
    "height":60
  },
  "incomingConnections"::[ 2 ],
  "outgoingConnections"::[ ]
}
```
The listing depicts the JSON format as an alternative to the regular XML-based representation. Using XML tags, assigning web services to activities can be difficult for two reasons. One is that there is some information that the user needs to get from the WSDL file, therefore a non-expert user might not know where and how to get that information. The other is that the user must place this information in the right structure and using the appropriate LPCEL XML tags. Should other EML specifications support web services, the same task is also difficult when it comes to write the same web service description using that EML.

It is important to highlight how easy it is for the user to attach an external web service to a task, and because of the LPCEL capabilities to execute certain tasks based on conditions and to include external resources, the assessment activities can also be facilitated by using the EvalComix 3.0 web services because they provide different assessments tasks and procedures for different evaluation scenarios. In this case, the process of including web services for assessment can be accomplished using the Web Services wizard depicted in Fig. 5.

![Web services wizard](image)

Figure 5. Web services wizard

In this wizard, the user selects the web service and the desired operation, configures the web service client (web forms) and optionally she can test the web service. So all of the work needed to complete the XML description of the web service can be summarized in three simple steps where the user only select the same basic information, but in an intuitive and easy-to-use manner. After completing the steps of the wizard, the LPCEL editor gathers automatically all the relevant information about the web service and a web service client is automatically generated.

**D. Reusing the design as a new pattern**

The user is also able to store of complex structures as patterns if they are a good solution for a recurrent problem. The LPCEL Editor provides the tools to manage the patterns by requiring the user to fill specific fields about the new patterns. An example of some sections of the web form can be depicted in Fig. 6 and is completely based on the classification proposed by Gamma et al. [10] for software engineering patterns.

![Diagram](image)

Figure 6. LPCEL facility to save learning designs as a new pattern

There are two particular fields that are important to consider because of their relevance: The **Diagram** and **See also** fields. Although all of the fields are necessary and serve specific purposes for the user and the editor, these fields are especially useful from a design point of view. The **Diagram** field presents a preview of the pattern flow of activities. This is particularly useful when the user needs to see if a selected pattern is, in fact, a good solution to a given problem. The **See also** field gives the user the possibility to navigate among similar patterns in order to find one that better resolves a specific learning design situation.

**CONCLUSIONS**

This paper describes how the LPCEL Editor can effectively integrate and test an assessment web service in a complex learning process. The web service wizard allows the learning designer to set all the information about the assessment web service without the need to know the underlying representation required to describe it. The LPCEL editing framework can facilitate thus the definition of service-based assessments for learning activities. Thanks to the independence of web services with respect to the environment in which the UoL is to be played, different web services can be included according to the requirements of the learning design scenario.

**ACKNOWLEDGMENT**

This work is partly funded by a grant from the ASCETA project (P09-TIC-5230) of the R&D programme of the Government of Andalusia, Spain.

**REFERENCES**


