COLearn and Open Discovery Space Portal Alignment

A case of enriching open learning infrastructures with collaborative learning capabilities

George Stylianakis, Nektarios Moumoutzis, Polyxeni Arapi, Manolis Mylonakis, Stavros Christodoulakis
Laboratory of Distributed Multimedia Information Systems and Applications
Technical University of Crete (TUC/MUSIC)
Chania, Greece
{gstylianakis, nektar, xenia, manolis, stavros}@ced.tuc.gr

***Abstract*—*In this paper we present COLearn platform which can be seen as a shell, easily integrated on top of existing open learning infrastructures (such as LMSs and OER repositories) to enrich their capabilities by offering functionality to design rich learning activities and learning workflows. COLearn run-time environment is used to enact these activities and workflows. This way, COLearn leverages the power of the underlying infrastructures, provides structure to the groups of learners that participate in learning workflows, dynamically adapts the workflows during their enactment, monitors their evolution to facilitate assessment and provides feedback to the learners. COLearn employs an intuitive graphical representation exploiting the BPMN standard. As an internal representation and interoperability model it uses IMS LD thus offering effective sharing and remixing to realize the vision of open educational practices. This is an important aspect as it makes explicit the, otherwise tacit knowledge pertaining to the design of learning experiences.*

Keywords—Computer supported collaborative learning, IMS Learning Design, Business Process Modeling Notation, IMS Learning Tool Interoperability

I. INTRODUCTION

Traditional Learning Management Systems (LMS) have been proven successful in the management of learning contents of different granularities (learning resources, learning objects and courses), organization of learning materials by teachers, delivery of learning contents to learners, provision of core learning tools including communication services such as instant messaging and forums. Despite the unquestionable impact of LMSs on the evolution of e-learning and education in general, there are some extensions that could enrich their capabilities. In particular, there is room for improvement with respect to (a) contextualization; (b) informal and lifelong learning support; (c) personalization; (d) openness and decentralization; (e) knowledge pull; and (f) extension of the learners' role [10]. In the following paragraphs we describe in more detail the aforementioned issues, explain why an LMS in its traditional versions does not address them appropriately and present the solutions we propose in order to extend the functionality of an LMS to address these issues.

Contextualization: A traditional LMS follows a one size-fits-all approach to learning by offering a static system to teachers with predefined tools to support uniform courses to learners. Our approach, in contrast, takes into account the needs and design decisions of the teachers allowing them to organize learning materials, tools and learners within meaningful contexts (learning activities) that are structured within wider designed learning spaces (learning scenarios).

Each designed space can be reused or remixed as well, thus offering to teachers the possibility to share their pedagogical strategies with other teachers. To enable such an extension we provide a graphical authoring tool with which teachers (acting as educational designers) can describe learning scenarios in the form of collaboration scripts. In [4] Dillenbourg describes scripts as a sequence of phases each characterized by the following five attributes: (a) type of task to be accomplished; (b) group formation (and composition); (c) distribution of the task within and among groups; (d) type and mode of interaction; and (e) timing of the phase. COLearn scenarios are thus conceptualized as collaboration scripts, i.e. scaffolds that aim to improve collaboration through structuring interactive processes between two or more learning partners [7].

Informal learning and lifelong learning support: A traditional LMS does not support informal or lifelong learning, but it can be used in a formal learning setting, managed and controlled by the educational institution. In an LMS, learning stops when a course terminates. Our approach, however, can connect formal, informal, and lifelong learning opportunities within a context that is centered upon the specific learning context that the learning design is targeting at. It allows the learning designer and the facilitator (when the learning design is enacted) to capture her informal and lifelong learning objectives and develop her own learning design e-portfolio.

Openness and decentralization: A traditional LMS stores information on a centralized basis within a closed and bounded environment. Our approach goes beyond the boundaries of the organization and operates in a more decentralized, loosely coupled, and open context. Learners could be engaged in a distributed environment consisting of a network of people, services and resources. To provide such an environment COLearn platform at the runtime level incorporates specific tools for the support of group management and social interaction among participants, tools such as chat, micro-blogging, calendar etc. These tools are developed by using the Extensible Messaging and Presence Protocol (XMPP). In addition, in order to cope with the challenge of integrating external learning tools in a transparent way, we employ the IMS Learning Tool Interoperability (IMS LTI) [9] specification. This way the runtime environment can synchronize external tools that facilitate the learning process.

Knowledge-pull: A traditional LMS adopts a knowledge-push model and is concerned with exposing learners to content expecting that learning will then happen automatically. Usually, what actually happens is that the teacher organizes the learning process outside the LMS employing various ad-hoc facilities such as email messages or face-to-faces guidance in case of a blended learning approach. In our approach, however, we permit the teacher to adopt a knowledge pull model. Teachers could create their customized learning environments- at the authoring level, in the form of organized learning activities and at the runtime level those learning activities will guide the participants to pull knowledge that meets their particular needs from a wide array of knowledge sources.

Role of learner: In a traditional LMS a learner is considered as a consumer of pre-defined learning materials, dependent on the "creativity" of the teacher. In our approach, through content creation tools and their interlinking with collaboration and communication elements (such as micro blogging, forums, chat rooms etc.) in a highly contextualized way, the learner becomes active, self-directed, and, if the teachers wishes to, creator of content.

COLearn offers a technological shell that enriches the underlying infrastructure, exploit its power and address the above issues so that new services are offered (a) to teachers to embrace a culture of open educational practices; and (b) to learners to engage in meaningful learning within the context of rich learning activities supported by digital resources and tools.

II. COLEARN ARCHITECTURE

COLearn, is a web-based collaborative learning environment that supports the specification of collaboration scripts (these could be pedagogical scenarios or lesson plans), their deployment and enactment.

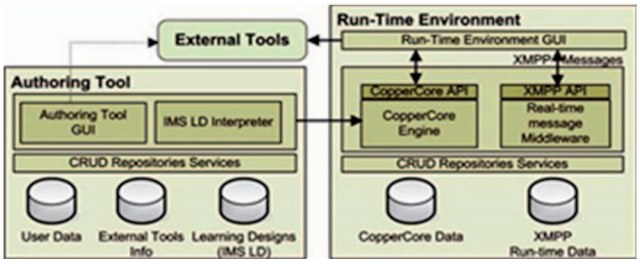

Fig. 1. *COLearn Architecture*

It consists of two main components: a) The authoring tool that provides the means to specify the collaboration scripts, representing the learning process within a scenario or lesson plan, in a machine processable manner; and b) the run-time environment where theses scripts are enacted. The following sections describe in detail the architectural components of COLearn as depicted in the figure above.

A. Authoring Tool

In order to develop and offer collaborative learning facilities on top of an open learning infrastructure one has to cope with learning content, communication and collaboration facilities, technological infrastructure, and run-time execution (i.e. learning process). Collaboration scripts are employed to specify the components of the learning process including collaboration and interaction patterns within groups of participants. COLearn offers a collaboration script authoring tool based on the Business Process Modeling Notation (BPMN) [2] as the model for the graphical representation of scripts. The COLearn authoring tool, as seen from the user, is presented in Fig. 2. The figure depicts the main pane of the tool, which is used to specify the learning activities that constitute a collaboration script.

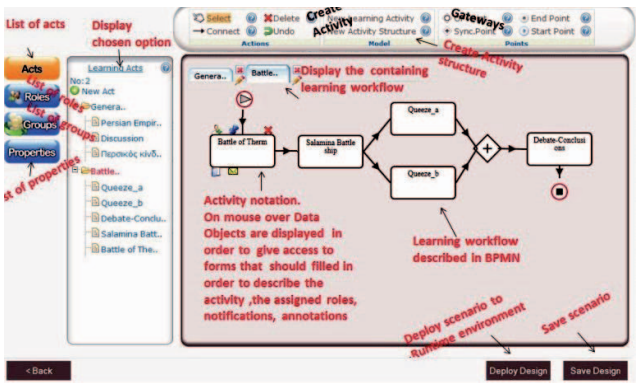

Fig. 2. *COLearn Authoring Tool.*

Collaboration scripts are parsed and transformed to IMS Learning Design Level C learning flows. The platform integrates external learning tools by implementing the IMS LTI specification. IMS LTI is used to enable the dynamic integration of external tools through services. Details regarding the internals of the Authoring Tool have been given in Stylianakis et al. [11] along with the rationale for differentiating the user interface representation model from the internal representation model and their mapping.

One critical issue that we describe here is the incorporation of external repositories hosting Open Educational Resources (OER's). One such repository that has been interfaced is the repository of the Open Discovery Space portal project (http://portal.opendiscoveryspace.eu/), which aggregates millions of quality OER's from several thematic and national educational repositories and offering them to a European-wide community of teachers in primary and secondary schools. The incorporation of this repository exploits an available ODS Search Plugin. This plugin is a JavaScript component that has been developed to offer functionality to search the ODS repository using the SOLR search engine operating on top of it. The SOLR search engine provides powerful searching mechanisms including full-text search and faceted search through REST-like HTTP/XML and JSON APIs that make it easy to use virtually any programming language. The aforementioned functionality is provided to authors when they create or edit collaboration scripts in the COLearn authoring environment. There are several cases when the ODS search is applicable: When describing a learning activity, when specifying its learning objectives, learning objects, learning prerequisites or its description.

Through this functionality an author has the opportunity to include materials of acknowledged learning quality. Figure 3

shows the "Search ODS" that activates the ODS search service. As soon as the user presses the "Search ODS" button, the search pane depicted in Figure 4 below is shown. In this search pane, the user can specify the search string and additional search parameters such as the desired language and the repository where the returned resources should come from.

In a similar way, other external repositories could be interfaced by COLearn to enable the creation of designed learning experiences that exploit the wealth of digital content that is available or will become available in digital repositories that conform to a minimum set of interoperability guidelines.

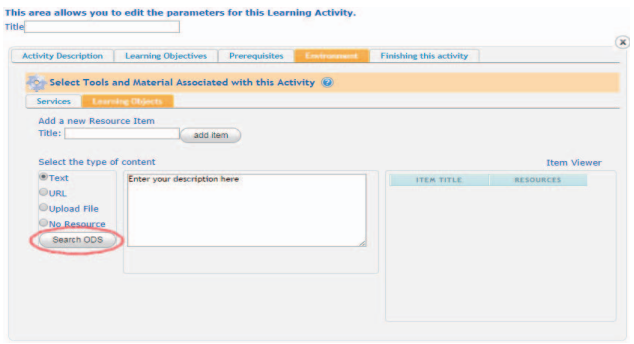

Fig. 3. *New Activity Description*

The search results are shown in the right column of the pane depicted in figure 4 and the author is able to browse them and see more details for each one of them by clicking on the resource title. All this information is retrieved from the external repository. In case more information is requested, the author is redirected to the repository where the actual content resides. After the author inspects the results and decides to use any of them, she may fill the scenario placeholders that are relevant with the URL that points to the actual content.

Fig. 4. *Searching the ODS repository*

B. Deployment of collaboration scripts

The deployment of a collaboration script refers to the identification of the participants and their roles for enacting it. This is a two-step process: Initially, the IMS LD document is published to the CopperCore engine. Then, the appropriate communication channels among the members of each group are established. The latter, is based on the XMPP protocol and made feasible by integrating the Openfire Server into COLearn runtime environment.

Several Rest Services are provided by COLearn platform in order to make available the functionality described above to external learning environments/ systems. The list of the services is as follows:

- create_group: Post the data of a group in json format (e.g {"title":" ", "persons_in_group":[{"username":"", "mail":"", "gender":"", "full_name":"", "organization": ""},…]}). The outcome will be a new group resided in the COLearn infrastructure.
- add_student: Post student's data (e.g {"username":"", "mail":"", "gender":"", "full_name":"", "organization": ""}). The outcome is a new student available in the COLearn platform.
- add_student/{studentId}/{groupId}: Adds a specific student in the specified group.
- student_group/{groupId}: Retrieves the data of all students that are members to the specific group.
- student/{studentId}/{groupId}: Retrieves the data of a specific user that is member of the specified group.
- delete_group/{groupId}: Deletes a specific group
- delete_student/{studentId}/{groupId}: Removes a specific student from the specified group.
- get_group/{groupId}: Retrieves the data of the specified group in json format (e.g {"title":" ", "persons_in_group":[{"username":"", "mail":"", "gender":"", "full_name": "", "organization":""},…]}

An external learning system could organize groups of students as separate web pages. In each page scenarios (collaboration scripts) relevant to each group can be resided. Each scenario may or may not, have been deployed to the COLearn runtime environment. Based on these requirements, the following services, necessary for addressing the requirements, have been defined and implemented:

- save_scenario/{groupId}: Post the information that a collaboration script is relevant to a specific group.
- scenario_status/{scenarioId}: Retrieve the information if a collaboration script has been deployed to the runtime environment or not.
- get_scenarios/{groupId}: Retrieve the ids of all the scripts that are relevant to the specified group.
- get_runId/{scenarioId}: Each script can be deployed multiple times in the runtime environment. In order to access a specific instance of the script we have to provide its running id before accessing the runtime environment.

C. Runtime Environment

Enactment is a term that refers to the actual implementation of a script through the COLearn runtime environment after its deployment, i.e. after the assignment of scenario-specific roles to particular users that will participate in the scenario implementation. Consequently, to enact a script it is necessary to deploy it first. Note that each scenario could be deployed

many times. Each distinct deployment corresponds to unique implementation of the scenario with specific participants. The COLearn runtime Environment consists of two main components: a) The CopperCore engine and b) the Real Time Message Middleware. These components are described in the next paragraphs. Figure 5 presents the COLearn runtime environment and displays its functionality.

Fig. 5. *COLearn runtime environment*

a) Copper Core Engine

In the runtime environment the user has an up-to-date view of her/ his learning process. For example, let's assume the deployment of a learning scenario, which specifies a group assignment where all learners need to complete a specific activity before they can proceed to the next one. The runtime environment checks this constraint and synchronizes access to the second activity. Such checking, synchronization and personalized view by each participant are supported by the CopperCore engine. The engine fully supports IMS LD levels A, B, C and provides the persistence storage and the APIs to manage the administration and delivery of IMS LD. CopperCore is designed to be used in a service oriented framework as COLearn runtime environment is. COLearn runtime environment provides the appropriate graphical components to exploit the CopperCore functionality and deliver the IMS LD learning scenarios.

b) Real Time Message Middleware / Social Interactions

COLearn incorporates Openfire server, which is a real time collaboration server that implements the XMPP protocol. XMPP is an open technology for real time communication, providing a way to send XML messages from one entity to another in real time. Additionally it provides the establishment of peer-to-peer media sessions supporting voice/video chat, file transfer, and other real time interactions.

COLearn adopts the following strategy for supporting collaboration: When an author deploys a Collaboration script to the runtime environment, a contact list is created for each member of each specific working group. The contact list is essentially, an instant messaging "friend list" providing the option for one-to-one messaging among members of the same group. Additionally, for each group a communication channel implemented as publish-subscribe node is established. Each member of the group is subscribed to this node. The XMPP publish-subscribe extension provides a framework to implement notification, multi-party messaging and file transfer in COLearn platform. In addition, it offers an extensible approach to integrate tools compliant to instant messaging mechanisms for the organization/ management of working groups as well as the social interaction among group members.

Finally, the message middleware interacts with the CopperCore engine, providing information and notifications among the users and triggering actions in the learning workflow process. For example, in a case of a collaborative activity with two participants, when a user completes the activity, he waits the other to finishes it to. When the second participant finishes, the messaging system triggers the Copper Core engine to enact the next activity for both participants.

III. ADOPTING IMS LTI: INTEGRATING EXTERNAL TOOLS

In COLearn runtime environment, each external tool is launched as a service employing IMS LTI. In the COLearn authoring tool, an author can choose to use an external tool from a list of available ones. This list is maintained/ updated by the administrator of the system. The administrator provides the URL of the external tool, a secret and a key that are needed in order to establish the connection. The secret and the key parameters are needed because the connection of the COLearn platform with the external tools (at the runtime level) is using the Open Authentication protocol. When the author selects the desired external tool, she additionally submits the LTI specific attributes described in the specification. During transformation of the script from the BPMN to its IMS LD counterpart the LTI provided values are stored in the attribute "parameter" of the element "service" of the IMS LD document. At the run-time environment, when an activity which defines external tool usage is activated, then the tool is automatically launched. Currently, COLearn platform integrates the MediaWiki, a powerful discussion board system implemented in Java. In order to integrate it, LTI tool provider compliant code in PHP was developed and attached to MediaWiki.

IV. COMPARISON WITH RELATED WORK

The most known platform that combines authoring and runtime aspects for learning scenarios is LAMS [3]. LAMS is a complete platform for designing, managing and delivering online collaborative learning activities. LAMS is not intended to complement an existing learning environment as it is the case for COLearn. Activities in LAMS can include a range of individual tasks and group work. LAMS functionality is extended through the integration of external tools. However, to integrate an external tool one should cope with a complex LAMS Tool contract. Despite the fact that activities can be assigned to groups of participants, LAMS lacks synchronous group support. LAMS provides some synchronous communication tools during the learning flow but these tools are initiated as activities as long as the author explicitly defines their use in the scenario. Also, these tools are not constantly active during the scenario execution. A teamwork environment should account the absence of physical interaction by providing synchronous collaboration among participants [1].

RELOAD Editor [15] and ReCourse [5] are form-based editors supporting IMS LD. Despite the fact that they provide direct control of the IMS LD model elements, they are extremely difficult to be used by less experienced designers. On the other hand MOT+LD editor [13], LAMS, ASK-LDT [14] are tools that provide graphical-based interfaces for the

definition of learning scenarios. One of the most notable advantages of tools that adopt the graphical-based interface approach is that knowledge of the IMS LD model is not a prerequisite in order to use them. As already stated COLearn authoring tool adopts the approach of graphically design learning scenarios and is based on the BPMN model in order to provide a standard means for creating human understandable graphical representations of processes [6]. Furthermore, the COLearn authoring tool maintains IMS LD naming over BPMN graphical notations contributing to the mapping of the IMS LD semantics to BPMN graphical notations.

SLeD [12] and Reload Player [15] are runtime environments that enable the enactment of IMS LD scenarios. Despite the fact that several roles participate and perform activities in these environments, those roles are working in parallel without affecting one another during the progress of the scenario. Consequently, they both lack group coordination and collaboration. In addition, there are no communication tools available during scenario enactment. In contrast, the COLearn runtime environment supports group-based learning. The learning scenarios are performed by group of users in a collaborative and synchronized manner. In the learning process each member is assigned to specific roles and performs specific tasks. There are cases (depending on the scenario) where a participant must wait others to complete their tasks in order to move forward in the learning process. To enable synchronization and personalization of the learning process, the CopperCore engine is integrated to the COLearn platform. Furthermore, the adoption of the XMPP real time protocol offers to the COLearn platform a messaging system to notify group members about the changes in the progress of the scenario in real time. XMPP-based tools are provided by the COLearn platform to support synchronous and asynchronous collaboration/ communication. Note that XMPP can provide asynchronous communication as well. Messages addressed to absent users are delivered to them when they enter the platform. External learning tools are integrated to the COLearn platform in a transparent way by implementing the IMS LTI specification. This specification enables the assignment of roles to group users in the external tools. This is of high importance in case of tools that orchestrate a work process, such as wikis.

V. Conclusions and Future Work

We presented COLearn, a platform acting as a shell on top of an existing open learning infrastructure to enrich its capabilities. An infrastructure (e.g. an LMS) that uses COLearn, could extend its functionality by offering to teachers the ability to design rich learning activities and learning workflows, enact them, and provide structure to the groups of learners that participate in these workflows, dynamically adapt the workflows during their enactment and monitor their evolution to facilitate assessment. The COLearn authoring tool produces collaboration scripts employing the BPMN model for their graphical representation. These scripts are internally represented using the IMS LD specification. The runtime environment consists of the CopperCore engine, which supervises the execution of the design. The group management, social interaction and collaboration among the participants are made feasible by integrating the Openfire server which implements the XMPP real time protocol. The architectural decisions allow for better support of groups and group interactions, easier description of complex learning processes, while maintaining compatibility with the IMS LD specifications. For the integration of external learning tools, COLearn implements the IMS LTI specification. The COLearn platform is an open platform that can collaborate with different types of internal/external learning applications.

Our future plans in enhancing COLearn services focus on supporting mobile learning scenarios including support for augmented reality games. To enable such scenarios it is necessary to model interactions with tools running on mobile devices and incorporating parameters for the specification of physical contexts on which learning activities take place. The learning domains that will be facilitated with these extensions include field trips for science projects and physical inquiries related to history, geography and language learning.

Acknowledgment

This work is partially funded in the scope of the Open Discovery Space project - http://portal.opendiscoveryspace.eu (FP7-ICT-PSP: 297229).

References

[1] M. N. K. Boulos, A. D. Taylor, and A. Breton, "A synchronous communication experiment within an online distance learning program: a case study." Telemedicine Journal & e-Health 11 no. 5(2005)

[2] BPMI.org, OMG: Business Process Modeling Notation Specification. Final Adopted Specification. Object Management Group, 2006, available at: http://www.bpmn.org

[3] J. Dalziel, "Implementing learning design: The learning activity management system (LAMS)." LAMS Teacher's Guide, 2006, V2.0.

[4] P. Dillenbourg, "Over-Scripting CSCL: The risks of blending collaborative learning with instructional design." Three Worlds of CSCL. Can We Support CSCL? (2002): 61-91.

[5] D. Griffiths, P. Beauvoir, O. Liber, and M. Barrett Baxendale, "From Reload to ReCourse: learning from IMS Learning Design Implementations." Distance Education, 30, no. 2 (2009): 201-222.

[6] P. Karampiperis, D. Sampson, "Towards a common graphical language for learning flows: Transforming BPEL to IMS Learning Design level A representations." In Advanced Learning Technologies, 2007. ICALT 2007. Seventh IEEE International Conference on, pp. 798-800. IEEE.

[7] L. Kobbe, A. Weinberger, P. Dillenbourg, A. Harrer, R. Hämäläinen, P. Häkkinen, F. Fischer, "Specifying CSCL Scripts." CSCL 2 (2007)

[8] IMS Learning Design Specification Version 1.0. [Online] Available at:. http://www.imsglobal.org/learningdesign/index.cfm IMS

[9] IMS Learning Tools Interoperability Version 1[Online] Available at http://www.imsglobal.org/lti/blti/bltiv1p0/ltiBLTIimgv1p0.html

[10] M.A. Chatti, A.L. Dyckhoff, U. Schroeder and H. Thüs (2012) 'A reference model for learning analytics', Int. J. Technology Enhanced Learning, Vol. 4, Nos. 5/6, pp.318–331.

[11] G. Stylianakis, P. Arapi, N. Moumoutzis, S. Christodoulakis, CoLearn: Real time collaborative learning environment. International Conference on e-Learning and e-Technologies in Education (ICEEE) (2013)

[12] OUUK. Sled player (2005) (cited April 8th, 2007)

[13] G. Paquette, O. Marino, K. Lundgren-Cayrol, and M. Leonard "Principled construction and reuse of learning designs." Handbook of Research on Learning Design and Learning Objects: Issues, Applications, and Technologies (2008): 869-890.

[14] D. Sampson, P. Karampiperis and P. Zervas, "ASK-LDT: A Web-Based Learning Scenarios Authoring Environment based on IMS Learning Design", International Journal on Advanced Technology for Learning (ATL) (ISSN 1710-2251), vol. 2(4), pp. 207-215, ACTA Press, 2005

[15] B. Olivier, "Learning Design Update." 2004, [online] Available at http://www.jisc.ac.uk/uploaded_documents/Learning_Design_State_of_Play.pdf, last visited 15 October 2014.