CoLearn: Real Time Collaborative Learning Environment

George Stylianakis, Polyxeni Arapi, Nektarios Moumoutzis, Stavros Christodoulakis
Laboratory of Distributed Multimedia Information Systems and Applications (TUC/MUSIC)
Technical University of Crete 73100 Chania, Greece
{gstylianakis, xenia, nektar, stavros}@ced.tuc.gr

***Abstract*—Computer Supported Collaborative Learning has become an important part of e-learning providing interactivity and accessibility to learning resources either synchronously or asynchronously among users. In order to develop an e-learning environment that supports collaborative learning one should 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. In this paper we present a Computer Supported Collaborative Learning platform which includes a collaboration script authoring tool based on the Business Process Modeling Notation as the model for the graphical representation of scripts. Collaboration scripts are parsed and transformed to IMS Learning Design Level C learning flows. The second important component of the platform is the run time environment which is based on the Copper Core Engine that executes the IMS Learning Design learning flows and a Message Oriented Middleware which is built over the Extensible Messaging and Presence Protocol. This middleware layer enables group management and social interaction among participants. The platform integrates external learning tools by implementing the IMS Learning Tool Interoperability (IMS LTI) specification. IMS LTI is used to enable the dynamic integration of external tools through services.**

Keywords- Computer supported collaborative learning, Collaboration scripts authoring, IMS Learning Design, Business Process Modeling Notation, Extensible Messaging and Presence Protocol, Message oriented Middleware, IMS Learning Tool Interoperability

I. INTRODUCTION

In collaborative e-learning environments where group members interact with each other there is some degree of uncertainty [2]. Uncertainty exists due to the fact that participants are not sure how others are feeling, reacting and thinking in online interaction situations. To reduce the uncertainty about coordination efforts, scripted cooperation has been proposed in learning research [17]. Collaboration scripts have been considered as facilitators of collaborative learning activities by structuring both face-to-face [18] and computer-mediated collaboration [20].

Collaboration scripts are scaffolds that aim to improve collaboration through structuring the interactive processes between two or more learning partners [15]. In [6] 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 (co-located vs. remote, synchronous vs. asynchronous etc.);and (e) timing of the phase.

In computer settings collaboration scripts must be represented in a formal way. A computational representation of a collaboration script for learning purposes is called Computer Supported Collaborative Learning (CSCL) script. Despite the flexibility of CSCL scripts in structuring collaborative learning, a general modeling language for formalizing CSCL scripts is still missing [16]. To provide a machine processable format for CSCL scripts the IMS Learning Design (IMS LD) [11] specification is proposed by several researchers. In [9] Hernández et al. presents the educational design requirements of CSCL scripts and how they can be expressed with IMS LD. The IMS LD can formally describe any design of teaching/ learning processes for a wide range of pedagogical approaches.

Although IMS LD is widely accepted in several aspects, it doesn't support sufficiently the modeling of group-based, synchronous collaborative learning [18], and additionally it does not offer a standardized graphical notation to use. Some of the most important aspects of the learning process that should be supported in any CSCL environment are capabilities for modeling groups as well as capabilities for enabling synchronous communication/collaboration among participants. IMS LD provides capabilities for specifying personalized asynchronous learning environments but lacks support for specifying group-based synchronous collaborative learning activities. In addition, IMS LD specification insufficiently supports the modeling/management of the outcome produced during the learning process. In [18] Miao et al. explain why IMS LD fails to address the aforementioned challenges that are highly connected with collaborative learning. Finally, although there are several high level authoring tools conforming to IMS LD, these tools are not using a standard graphical notation [14]. Moreover, the majority of these tools do not support complex learning flows- learning flows that combine splitting and synchronization of activities.

In this paper we describe the solutions proposed and developed to remedy the aforementioned challenges while staying compatible with the IMS LD exchange formats. Specifically, we present the architecture/implementation of a platform that better supports authoring and execution of real time collaborative learning. The platform (at the authoring level) is based on modeling concepts and graphical symbols coming from the Business Process Management Notation (BPMN) [3]. A BPMN graphical profile is used to formalize CSCL scripts that are computationally described with IMS LD level C. The graphical symbols inherit their name from the IMS LD specification in order to avoid the introduction of a new naming schema over the IMS LD. In addition, the platform (at the run time level) incorporates specific tools for the support of group management and social interaction among participants. These tools are developed upon the Extensible Messaging and Presence Protocol (XMPP).

Last but not least in order to cope with the challenge of integrating external learning tools in a transparent way, we have expanded the IMS LD to adopt the IMS Learning Tool Interoperability (IMS LTI) [12] specification. In addition, IMS LTI specification facilitates the exploitation of the outcome of the learning process.

II. COLEARN ARCHITECTURE

COLearn architecture is presented in Fig. 1. The main components of the platform are: A) The authoring tool that provides the means to design CSCL scripts that represent the learning process in a machine processable manner, and B) the run time environment where CSCL scripts are enacted. The following sections describe in detail the components and the technological infrastructure of the COLearn architecture.

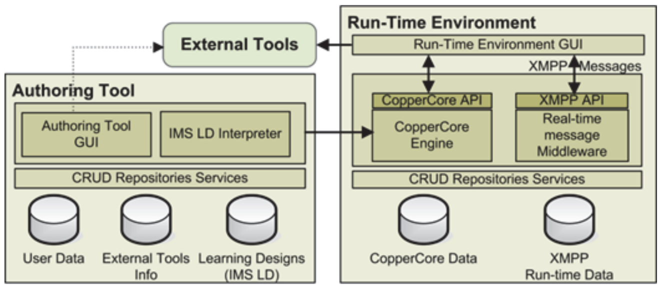


Fig. 1. CoLearn Architecture

A. *Authoring Tool: Modeling complicated learning flows - Supporting learning through Business Process Management*

A collaborative learning experience can be described by a collaboration script. Many collaboration scripts have been designed, tested, and embedded in CSCL applications [10]. The absence of a modeling language to formalize CSCL scripts led several researchers to adopt IMS LD as their computational representation [9]. Despite the wide adoption of the IMS LD specification, a common language for graphically representing learning flows is missing. On the other hand there are standards for human understandable graphical representations of processes (e.g BPMN) that are very flexible, they have very detailed formal descriptions and they have been widely used in a large number of diverse collaborative scenarios in the industry.

The COLearn Authoring Tool is a graphical tool that allows users to create/manage CSCL scripts. The Authoring Tool uses BPMN as the representational notation of CSCL scripts. BPMN is a standard for business process modeling, and provides human understandable graphical notations for specifying business processes in Business Process Diagrams (BPD). In [8] Denis Helic proposes business processes management as an approach to support complex learning processes. He points out that there are close connections between collaborative learning processes and business processes at many levels. At the communication/collaboration level human participants in both processes try to achieve a certain goal by working closely together in a particular social context. Furthermore, at the procedural level, both business and learning processes deal with activities structured in a certain way and executed by following a set of rules, i.e. business or pedagogical rules respectively. Also, at the content level both of the processes work with standardized and interoperable content in electronic form.

The machine processable counterpart of COLearn CSCL scripts is based on IMS LD. The IMS LD interpreter of the platform provides the mechanisms to automatically transform the BPMN representation of CSCL scripts to IMS LD level C, according to specific rules that are described below.

The adoption of BPMN supports the design process without requiring pre-existing knowledge of the details of the IMS LD model. Additionally BPMN minimizes the complexity of designing composite learning flows (e.g learning flows where specific conditions must meet in order to follow a specific path or flows consisting of learning paths that merge to a single output) which otherwise would require deep understanding of IMS LD model. Fig. 2 displays the mapping of BPMN core graphical elements to IMS LD components.

BPMN Graphical Element	Description	BPMN Graphical Notation	IMS LD Component
Lane	Used to organize and categorize activities according to function		Act
Task & Data Object	A task represents a single unit of work.		Role Part
Sequence Flow	Used to show the order that activities will be performed		Flow represented using if conditions
Gateway	Used to control the divergence and convergence		Properties and if conditions
Ad hoc Process	Group of activities that have no required sequence relationships. The sequence and number of performances for activities are determined by the performers		Activity Structures
Data objects	Data objects show the reader which data is required or produced in an activity.		The elements of activity/ activity structure
Annotation	Give the reader of the model/diagram an understandable impression		Metadata placeholder
Start, end	Start, end of the process		Conditions

Fig. 2. BPMN graphical elements to IMS LD core components mapping

To represent the learning flow (the sequence of activities assigned to each role) the IMS LD model uses the Act element. An Act element organizes and synchronizes set of activities which are assigned to specific roles. The combination of an activity and the role performs it; is called Rolepart. An Act element can be represented by a Lane in BPMN model. A Lane is logically organizing and categorizing Tasks according to specific function. Therefore, an IMS LD Rolepart element which represents a unit of work can be represented by a Task notation in conjunction with Data objects. A Data object that is attached to each Task incorporates the information about the roles that are assigned to perform the specific IMS LD Activity. In [14] Karampiperis et al. proposes the Lane as a representation counterpart to Rolepart and BPMN Task as a representation of IMS LD Activity. The objection to this pairing is that an IMS LD Activity can be assigned to several roles, meaning that multiple Roleparts can refer to the same

Activity. Thus, in the corresponding BPMN representation the same Task/Activity should be rendered in multiple Lanes. It is complex to represent such a case in BPMN. On the contrary in our approach such complexity is transcended.

Furthermore, a learning process should describe the order of the activities to be performed. In the BPMN model this is described with sequence flows (structural approach). In the IMS LD model it can be defined by “If” expressions (procedural approach).

Additionally, BPMN Gateways are used to describe complex business processes. A Gateway determines forking or merging of paths. COLearn authoring tool uses Inclusive and Parallel Gateways in order to describe complex learning flows. Gateways are transformed to IMS LD “If” expressions, similar to the case of sequence flows we described above. However, in case of Inclusive Gateways where we need to evaluate specific conditions, we additionally introduce IMS LD Property and Condition elements.

Furthermore, a BPMN Sub-Process is used to represent an IMS LD Activity Structure which is expanded in nested learning flows. The attributes of IMS LD Activities/ Activity Structures are represented graphically with BPMN Data objects. Data objects incorporate appropriate forms where users complement/update these attributes.

In addition, we use BPMN Annotations to attach LOM Description Metadata to specific Activities, giving to the participants of the CSCL script; at the run time level, more information about the activity.

Finally, the End BPMN notation denotes when a learning flow ends, and is represented by IMS LD “If” expression.

```
Procedure Transform:
  Create a queue Q with the acts included in script
  While Q is not empty
     node <- Q.dequeue()
     If node == end point
        createCondition();
        continue;
     Else If node is task:
        la = Create learning activity(name, description, objec-
tives,  prerequisites, environments, completion rules, feed-
back,notifications, metadata)
        roles = GetRolesAssignedToActivity();
        CreateRolePart(la, roles);
     Else If node is gateway:
       If node is inclusive gateway:
          expressions = GetDefinedExpressions();
          Create LD condition (expressions)
       If node is parallel gateway:
          Create LD condition()
       Else If node is activity structure
          Q.enqueue(activity structure)
          Continue;
```

Fig. 3. Pseudocode describing the transformation of BPMN workflows to IMS LD Level C learning flows

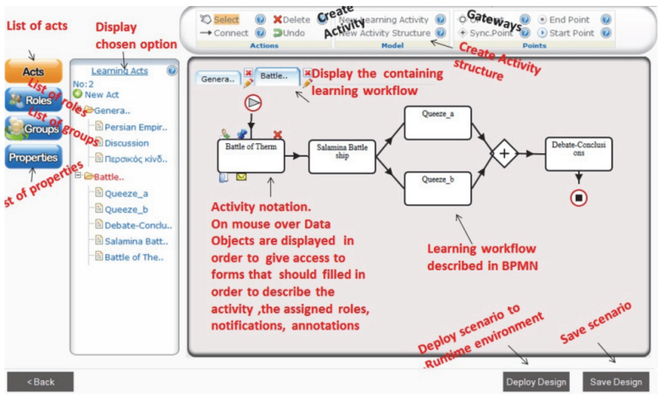
Fig. 3 describes in pseudocode the algorithm for transforming a BPMN graph to the corresponding IMS LD Level C representation. The algorithm essentially implements a depth-first traversal of all the BPMN trees (one for each Lane) rooted at BPMN Start elements. During this traversal the correspondence between BPMN elements and IMS LD Level C elements (depicted in Fig. 2) is used to compute the final transformation. Note that each BPMN tree traversed corresponds to a specific learning workflow represented, after the transformation, as an IMS LD Act. In more details the algorithm is as follows:

- If the node is a task: a) Create a learning activity with name, description, learning objectives, prerequisites, environments (learning objects and tools), completion rules, feedback, notifications, metadata. The mentioned data is compliant to the IMS LD model and is provided by the users using a form-based interface. b) Create a role-part element referring to the learning activity created and to the roles assigned to perform the activity.

- If the node is a gateway: The CoLearn authoring tool includes inclusive and parallel gateways. In case of an *Inclusive* gateway the user additionally defines properties which form expressions that are computed at the run time level in order to specify the path to be followed. The transformation algorithm creates LD conditions that are defined by the activities converging to the gateway and the properties’ expressions. *Parallel* gateways are used to combine or to create parallel flows. The algorithm creates LD conditions which describe that the outgoing task node will be triggered when all incoming tasks are completed.

- If the node is an activity structure: An activity structure according to the IMS LD specification can contain as sub-elements any combination of activity references and activity Structure references. Consequently, an activity structure is also a workflow container just like an act element. Therefore, the algorithm uses the same mechanics as in case of an act, in order to transform the graphical notations which represent activity structures.

- If the node is an end point: An IMS LD condition is created which specifies that when the finishing learning activity is completed then the containing Act or Activity Structure is completed as well.



Activity notation. On mouse over Data Objects are displayed in order to give access to forms that should filled in order to describe the activity ,the assigned roles, notifications, annotations

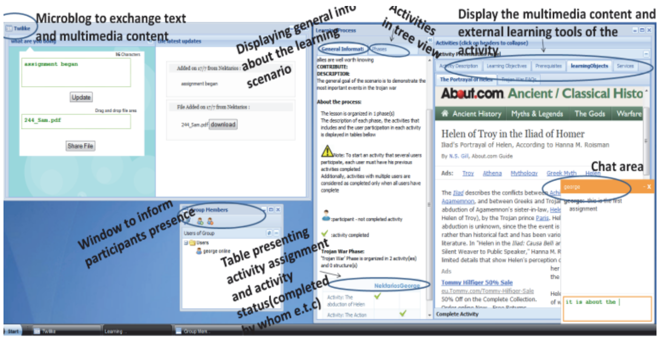
Fig. 4. The COLearn AuthoringTool

The COLearn authoring tool is presented in Fig. 4. It provides users the functionality to model create/manage CSCL scripts in a flexible and transparent way. These scripts are transformed to the IMS LD Level C counterpart document and are saved in author’s portfolio. CSCL scripts can be copied by others, enabling sharing of learning scenarios. Additionally the generated scripts can be deployed to the run time environment.

Before a CSCL author deploys the LD document (CSCL script of the learning scenario), she must choose the participants (learners/teachers registered as users in the platform) to compose the working groups. Each participant is assigned to specific roles and each group performs the learning scenario at the run time environment. The deployment of a learning scenario is a two-step process. Initially, the IMS LD document is published to the CopperCore engine [24]. 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 run time environment.

B. *Run Time Environment*

The COLearn Run-time 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. Fig. 5 presents the COLearn run time environment and displays its functionality (which is described in the following sections).



Microblog to exchange text and multimedia content
Displaying general info about the learning scenario
Activities in tree view
Display the multimedia content and external learning tools of the activity
Chat area
Window to inform participants presence
Table presenting activity assignment and activity status(completed by whom e.t.c)

Fig. 5. The COLearn run time Environment

a. *Copper Core Engine*

In the run time environment the user must be provided with an up-to-date view of her/ his learning process. For example, let’s assume 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 run time environment should check this constraint and synchronize access to the second activity. Such checking, synchronization and personalization can be developed by integrating the CopperCore engine into the run time environment. The engine fully supports IMS LD levels A, B, C and provides the persistence storage of the application 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 run time environment is. COLearn run time environment provides the appropriate graphical components to exploit the CopperCore functionality and deliver the IMS LD learning scenarios.

b. *Real Time Message Middleware:Modeling varied forms of social interaction*

CSCL applications should provide support to three essential aspects of collaboration, namely coordination, collaboration and communication; with communication being the base for reaching coordination and collaboration [4]. Collaboration and communication might be synchronous or asynchronous. However, student–teacher and student–student interactions in purely asynchronous distance learning courses are much lacking compared to similar interactions found in face-to-face teaching, causing learners to experience feelings of isolation, thus reducing motivation and increasing dropout rates [2].The adoption of a real time protocol as the communication channel provides the infrastructure to assist in fostering a sense of belonging to one supportive learning community among distant learners. We developed the message oriented middleware by integrating Openfire server, which is a real time collaboration server that implements the XMPP protocol. XMPP is an open technology for real-time communication, using the XML as the base format for exchanging information. In essence, XMPP provides a way to send pieces of XML from one entity to another in real time. Additionally, XMPP is used to establish peer-to-peer media sessions supporting voice/video chat, file transfer, and other real time interactions. XMPP provides great potentials in terms of team management and communication.

To provide the communication/collaboration among participants the COLearn platform adopts the following strategy: When a CSCL script author deploys a CSCL script to the run time environment, a contact list is created for each member of a specified 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 working group. In addition, for each specific working group one publish-subscribe communication channel node is established and each member of the working group is subscribed. The delivery of messages to and from these specific channels-nodes provides the option to organize the management of working groups as well as the social interaction among group members. The XMPP publish-subscribe extension provides a framework to implement notification, multi-party messaging and file transfer in COLearn platform. More specifically, COLearn run time environment incorporates a calendar in order to assist members of the working group to work in an organized way, a microblog in order to exchange text and multimedia content among the participants of each group and a one to one chat.

In any case, the adoption of real time middleware is managing the need of implementing different ways of message addressing such as point-to-point, multicast and broadcast. Additionally, 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, every time a participant completes an activity, the other members of the group are notified through the messaging system. A more complex situation would be a case of a collaborative activity with two participants. COLearn platform considers that a collaborative activity is finished when both users have completed the specified activity. In such a case, when a user completes the activity, he waits the other to finishes it to. When the second participant finishes, the messaging system will trigger the CopperCore engine and the next activity will be enacted automatically for both participants.

Finally, organizing participants as subscribers to a specific node which essentially represents a group, offers a significant advantage: It can be considered as a piece of software that can be shared by multiple tools. This is to be highly configurable so that tools compliant to instant messaging mechanisms can be incorporated in the platform and perform the way they want without changing the underlying code or data structure.

III. Adopting IMS LTI: Integrating External Tools & Managing Learning outcomes as digital artifacts

For a CSCL platform, learning tools and learning outcomes are very crucial. Both the technological infrastructure and tools support several aspects of the learning process. The infrastructure should include (but is not limited to) content presentation tools, synchronous/asynchronous communication tools, annotation tools, quizzes etc. In addition, artifacts are very essential in many occasions (for example in learning adaptation). In order to deal with educational tools and artifacts COLearn platform supports IMS Learning Tool Interoperability (IMS LTI). The principal concept of IMS LTI is to establish a standard way of integrating rich learning applications (often remotely hosted and provided through third-party services) with platforms like learning management systems or other educational environments that present them to users. In IMS LTI these learning applications are called Tool Providers and the LMSs/platforms, are called Tool Consumers.

When users are passed from the Tool Consumer application to the Tool Provider, the following data is transferred: a) User details (e.g. name, email); b) details about the institutional context; c) details of the specific context from which they are coming from (such as an online course); d) the assigned role in the external tool (e.g. 'learner', 'teacher', 'moderator').

In COLearn run time environment, each external tool is launched as a service. The learning outcomes of the tool usage are kept and managed by the external tool, providing the ability to overcome the obstacles of modeling/ managing the outcome produced during the learning process [18].

The process of launching external tools that support the IMS LTI is as follows. In the COLearn authoring tool, a CSCL script author can choose to use an external tool from a list of available ones. The list of the available tools is maintained and updated by the administrator of the system. More specifically, 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 run time level) is using the Open Authentication protocol. When the CSCL script author selects the desired external tool, she additionally submits the LTI specific attributes that are described in the specification and are mentioned in the previous paragraph. At the process of the transformation of the script from the BPMN to its IMS LD pair the LTI provided values are stored in the attribute "parameter" which is included in the element "service" of the IMS LD specification. At the run time environment, when the activity that defines the usage of an external tool is activated then the tool is launched. One of the immediate benefits of the IMS LTI is the provision of a single sign-on mechanism for user connections from the consumer to the learning tool provider. The COLearn platform integrates the MediaWiki, a free open source wiki package written in php, and Jforum, a powerful discussion board system implemented in Java. In order to integrate these tools, LTI tool provider compliant code in php and in java was developed and attached to MediaWiki and JForum respectively.

IV. Related Work

Several authoring tools for designing learning activities exist. The most popular are MOT+LD [23], LAMS [5] ASK LDT [25], RELOAD Editor [21], CopperAuthor [26], ReCourse [7] and Open Graphical Learning Modeler (OpenGLM) [19].

RELOAD Editor, CopperAuthor and ReCourse 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 [14]. On the other hand MOT+LD editor, LAMS, ASK-LD and OpenGLM are tools that provide graphical-based interfaces for the definition of learning scenarios. Note here that LAMS' underlying model is not IMS LD compliant, but it can export scenarios to IMS LD compatible format. 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. COLearn authoring tool adopts the approach of graphically design learning scenarios. In contrast to the aforementioned tools, COLearn is based on the BPMN model in order to provide a standard means for creating human understandable graphical representations of processes. 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. An important advantage of adopting BPMN is the ability of designing complex learning flows in a higher level of abstraction keeping the design process as simple as possible. The COLearn's BPMN to IMS LD interpreter, transforms Inclusive and Parallel Gateway notations to complex expressions consisting of IMS LD conditions and properties.

Moreover, there are several tools that execute learning scenarios. The most popular ones are LAMS, SLeD player [22] and the CopperCore based player included in the Reload Project [13]. SLeD and Reload players are lacking group coordination and collaboration. Despite the fact that several roles participate and perform activities, those roles are working in parallel without affecting one another during the progress of the scenario. Additionally, no communication tools are available. On the contrary LAMS is a platform for designing, managing and delivering online collaborative learning activities. Activities in LAMS can include a range of individual tasks and group work. LAMS functionality is extended through the integration of external tools. In order to integrate an external tool one should cope with a complex LAMS Tool contract [1]. 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 running environment should account the absence of physical interaction by providing synchronous collaboration among participants [2].

The COLearn run time environment, as opposed to aforementioned run time environments, focuses on supporting 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 learning 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, as we already have described, the CopperCore engine is integrated to the COLearn platform. Furthermore, the adoption of the XMPP real time protocol offers to the users of COLearn platform a messaging system to notify group members about the changes in the progress of the scenario in real time. Moreover, XMPP-based tools are provided by the COLearn platform to support synchronous and asynchronous collaboration and communication. Note that XMPP can provide asynchronous communication as well. When a user is not present, the messages that are addressed to him can be delivered when he enters the platform. Finally, external learning tools are integrated to the COLearn platform in a transparent way by implementing the IMS LTI specification. By adopting IMS LTI, we can also assign a specific role for each group user to the external tool which is selected in the corresponding activity that the user participates in. This is of high importance in case of tools that incorporate the means to orchestrate a work process, such as wikis.

V. CONCLUSIONS

We presented CoLearn, a software platform for a real time computer supported collaborative learning environment. The CSCL authoring tool produces collaboration scripts employing the BPMN model for their graphical representation. These scripts are internally represented using the IMS LD specification. The run time environment consists of the CopperCore engine, which supervises the execution of the design. The group management, social interaction and collaboration among the participants in learning flows are made feasible with the adoption of the XMPP real time protocol and the integration of the Openfire server implementing XMPP. The architectural decisions allow for better support of groups and group interactions, easier description of complex learning processes and learning artifacts, 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 and integrate different types of internal/external learning applications.

ACKNOWLEDGMENT

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

REFERENCES

[1] C. Alario-Hoyos, M.L. Bote-Lorenzo, E. Gómez-Sánchez, J. I. Asensio-Pérez, G. Vega-Gorgojo, and A. Ruiz-Calleja, "GLUE!: An Architecture for the Integration of External Tools in Virtual Learning Environments." Computers & Education (2012).

[2] 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): 583-593.

[3] BPMI.org, "Business Process Modeling Notation (BPMN) – V1.0",

[4] S. Caballé, T. Daradoumis, and F. Xhafa, "A Generic Platform for the Systematic Construction of Knowledge-Based Collaborative Learning Applications." *Architecture Solutions for e-Learning Systems* (2007).

[5] J. Dalziel, "Implementing learning design: The learning activity management system (LAMS) ." (2003): 2-1.

[6] 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.

[7] 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.

[8] D. Helic, "Technology-supported management of collaborative learning processes." *International Journal of Learning and Change* 1, no. 3 (2006): 285-298.

[9] D. Hernández-Leo, J. I. Asensio-Pérez, Y. Dimitriadis, M. L Bote-Lorenzo, I. M. Jorrín-Abellán, and E. D. Villasclaras-Fernández, "Reusing IMS-LD Formalized Best Practices in Collaborative Learning Structuring. " *Advanced Technology for Learning; 2 (3): 223-32* (2005).

[10] H. U. Hoppe, and R. Ploetzner, "Can Analytic Models Support Learning in Groups." *Collaborative-learning: Cognitive and Computational Approaches (1999)*: 147-168.

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

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

[13] JISC-funded project [Online] Available at: http://www.reload.ac.uk/

[14] 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.

[15] L. Kobbe, A. Weinberger, P. Dillenbourg, A. Harrer, R. Hämäläinen, P. Häkkinen, F. Fischer, "Specifying CSCL Scripts." ijCSCL 2, no. 2-3 (2007) :211-224.

[16] I. Kollar, F. Fischer, F.W Hesse, "Collaboration scripts–a conceptual analysis." *Educational Psychology Review* 18, no. 2 (2006): 159-185.

[17] K. Mäkitalo, A. Weinberger, P. Häkkinen, F. Fischer, "Uncertainty-reducing cooperation scripts in online learning environments." In *1st joint meeting of the EARLI SIG 6 Instructional Design and SIG 7 Learning and Instruction with Computers, Tübingen, Germany*. 2004.

[18] Y. Miao, K. Hoeksema, H. U. Hoppe, and A. Harrer: " CSCL scripts: modelling features and potential use." In *Proceedings of th 2005 conference on CSCL: learning 2005.* pp. 423-432. International Society of the Learning Sciences, 2005

[19] S. Neumann, and P. Oberhuemer, "User evaluation of a graphical modeling tool for IMS Learning Design." In *Advances in Web Based Learning–ICWL 2009*, pp. 287-296. Springer Berlin Heidelberg, 2009.

[20] A. M. O'Donnell, D.F. Dansereau, "Scripted cooperation in student dyads: A method for analyzing and enhancing academic learning & performance." The theoretical anatomy of group learning, 120-141('92).

[21] B. Olivier, "Learning Design Update." (2006).

[22] OUUK. Sled player (2005) (cited April 8th, 2007) Available from: http://sled.open.ac.uk

[23] 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.

[24] H. Vogten, H. Martens, P. Rosmalen, and E. J. R. Koper, CopperCore. Retrieved January14, 2005, from SourceForge: http://coppercore.org

[25] P. Zervas, D. Sampson, "Designing & Sharing Inquiry based Learning Designs: The PATHWAY ASK LD Toolkit.", *Never Waste a Crisis!* (2011): 173.

[26] W. Van der Vegt, "CopperAuthor." *Retrieved January* 3 (2006): 2008.