Representing CSCL macro-scripts using IMS LD lessons learned

Representing CSCL macro-scripts using IMS LD: lessons learned

Davinia Hernández-Leo, Daniel Burgos, Colin Tattersall and Rob Koper

Davinia Hernández-Leo is a teaching assistant at the Department of Signal Theory, Communications and Telematics Engineering, University of Valladolid, Spain. She is currently writing her PhD thesis, within the GSIC/EMIC multidisciplinary group, related to design and computational representation of CSCL scenarios using patterns and technical specifications. During the spring of 2006 she was working as a visiting Research Fellow at the Educational Technology Expertise Centre of the Open University of the Netherlands, where she has collaborated with Daniel Burgos, Colin Tattersall and Rob Koper. Address for correspondence: Davinia Hernández-Leo, School of Telecommunications Engineering, Camino del Cementerio s/n, 47011, Valladolid, Spain; Email: dherleo@ulises.tel.uva.es

Abstract

CSCL (Computer-Supported Collaborative Learning) macro-scripts are machine-readable collaboration scripts that structure the activities making up a learning process. The support of IMS Learning Design (LD) specification to implement these scripts is an unsolved research question marked by controversy. This paper is provides a systematic analysis to answer this question. With this purpose, we point out the requirements of CSCL macro-scripts for their representation using LD. These requirements include common collaborative learning mechanisms (group composition, role and resource distribution and coordination) and flexibility demands (such as flexible group composition). Each of these needs is described and illustrated by means of two significant examples that reflect the identified requirements well: Universanté and ArgueGraph Scripts. These scripts are used in the article to present and exemplify the realization of the requirements using LD. The paper collects the lessons learned from this analysis differentiating two angles – that of the LD notation itself and also related tools and specifications. The paper positions related work and discusses the possibility of generalizing the lessons learned to the representation of CSCL micro-scripts.

Acknowledgement: The work on this publication has been partially sponsored by the TENCompetence Integrated Project that is funded by the European Commission's 6th Framework Programme, priority IST/Technology Enhanced Learning. Contract 027087 [http://www.tencompetence.org].

Introduction

Computer software for supporting scripted Collaborative Learning (CL) is designed with the aim of scaffolding social interactions among participants (Dillenbourg, 2002). Scripts structure CL scenarios by defining the composition of groups, the distribution of roles and resources as well as the coordination of the activities that make up the learning process (Kobbe et al., 2005). Computer-Supported Collaborative Learning (CSCL) practice and research propose specific structuring mechanisms of the scripts and empirically study their effectiveness and risks (Dillenbourg & Tchounikine, in press).

The purpose behind the structuring mechanisms can be of two types (Dillenbourg et al., in press). The first type refers to processes that should be internalized by the learners. They usually describe the fine-grained actions that each participant should accomplish within activities. This is the reason why they are called micro-scripts. (Weinberger, Fischer, & Stegmann, 2005) present two examples of micro-scripts that guide argumentation processes. Their goal is that students learn how to argument in order to construct knowledge together. In contrast, macro-scripts aim to organize situations that encourage productive targeted interactions that lead to learning outcomes (eg the script arranges fruitful discussions by grouping students with different results in previous activities). Macro-scripts characteristically denote CL strategies defining flows of coarse-grained activities (Hernández-Leo et al., 2005).

Two different approaches have been employed to script collaboration using learning technology up to now. In the first approach, the teacher is in charge of scripting the collaboration (without computer support) by scaffolding the students while working with existing tools. However, this approach involves high workloads for teachers. The second approach aims at providing automatic computer support for the scripting process: a new CSCL application is developed to implement each specific script. This approach has clearly many drawbacks mainly related to time and cost efforts in development. The implemented scripts are neither resistant to technological changes nor easily tailorable to support another learning situation. To overcome these problems, a promising approach is to formalize the scripts so that they are automatically interpreted by a script engine. The computational representation of a script is called a CSCL script (Hernández-Leo et al., 2005; Miao, Hoeksema, Hoppe, & Harrer, 2005). This paper is focused on CSCL macro-scripts (hereafter referred to as "CSCL scripts" or simply "scripts" to achieve a more readable text).

In order to computationally represent the scripts we propose the use of IMS Learning Design specification (LD). LD is broadly accepted as de facto standard to formally model interoperable Units of Learning (UoL). The specification was designed so that UoLs can describe any teaching-learning process (Koper & Olivier, 2004). CSCL is an important field within Educational Technology and CSCL scripts are widely used teaching-learning processes (Kobbe et al., 2005). Surprisingly, the LD support for implementing CSCL scripts is not clear. The motivation of this problem is twofold. Firstly, since LD is a recent specification (IMS, 2003), there is a lack of significant examples and efforts that show the possibilities of LD for CSCL. Secondly, there is a lack of clarity regarding which characteristic of the scripts should be expressed by the notation itself as opposed to which requirements can be supported by tools or even other related specifications. Although partial work has been already accomplished (Gorissen & Tattersall, 2005; Hernández-Leo et al., 2005; IMS, 2003; Koper & Burgos, 2005), a more complete and systematic analysis is needed. As a consequence, some researchers are pointing out the limitations of LD and alternative EMLs (Educational Modelling Languages) proposals to describe CL scenarios are emerging (Miao et al., 2005; Vignollet, David, Ferraris, Martel, & Lejeune, 2006).

In this paper we systematically analyse the support of LD to implement the main requirements of CSCL macro-scripts. Therefore, next section identifies the educational design requirements of CSCL scripts. Then, the requirements are illustrated by means of two significant examples proposed in the CSCL literature and which reflect the identified requirements: Universanté and ArgueGraph Scripts (Dillenbourg, 2002). These examples are used in the article to present how the requirements can be implemented using LD. This problem is approached confronting the differences between the needs that can be satisfied by the LD notation itself and the needs that can be solved using tools or related specifications. Before concluding, we discuss the possibility of generalizing the lessons learned to the realisation of CSCL micro-scripts and position related work in this area.

Requirements of CSCL macro-scripts

This section presents the educational design requirements of CSCL scripts, which we have identified in the CSCL literature. The main sources have been the review of CL available at (NISE, 1997), the Collaborative Learning Flow Patterns (CLFPs) that collect good practices when structuring flows of CL activities (Hernández-Leo et al., 2005), several CSCL script examples and, especially, the current research on framing their components and mechanisms (Kobbe et al., 2005) as well as their flexibility demands (Dillenbourg et al., in press). Based on these sources, we classify the requirements for computationally representing the scripts into four different types of requirements, namely group composition, role and resource distribution, coordination and flexibility:

A. Group composition

Appropriate and effective implementation of groups is crucial in collaborative learning. Depending on the scenario, a CL practitioner needs to consider the composition of the involved group (or groups) according to different, sometimes interrelated, criterions:

- Hierarchy of groups: CSCL scripts typically make use of groups forming hierarchies, ie, groups may be composed of other (smaller) groups or different (individual) roles.
- Group size: defining the desired number of group members is perhaps the most common suggestion of scripts regarding group composition. They usually recommend keeping group size small for short activities

because, for example, there is not enough time for the group to become effective. However, larger groups are adequate in long scenarios (eg a course).

- Amount of groups: many scripts require a certain number of groups or at least a minimum or maximum amount so that the dynamics they propose are afforded.
- Group formation policies: depending on the scenario groups should be heterogeneous or homogenous to be more effective. The groups can be formed either by the students themselves or by the teacher by referring to existing common features (eg gender, age) or simply using a random assignment policy.
- Dynamic group formation: some CSCL scripts need some groups to be formed at runtime. That is to say, the assignment of a person to a group (and to a role) may depend on the result of a previous activity.

B. Role and resource distribution

With the aim of fostering (positive) interdependence, CSCL scripts often make use of distributing "tasks", being directly connected to allocating roles and resources (Kobbe et al., 2005):

- Role distribution: in a CSCL script participants may assume one or more roles at the same time (eg one of the students in a group is assigned to the role "scribe"). In addition, participants can switch their roles with other participants (eg rotation of roles, scribe vs. spokesman).
- Resource distribution: the amount of required resources and their selected distribution may depend on the number of groups, roles and participants.

C. Coordination

Coordination is an inherent characteristic of CSCL scripts. We distinguish three different requirements regarding coordination:

- Flow of (collaborative) learning activities: the main problem of the activity coordination falls into the synchronization of groups and roles through the activities: a person may belong to a group in a certain activity and to another group in the following one (then she probably needs to wait for the rest of the members of her second group in order to start the second activity).
- Floor control: while working together in the same activity, learners' actions are sometimes guided or constrained according to floor control mechanisms (eg a model of turn-taking when modifying an artefact).
- Flow of artefacts: artefacts (eg a document) are often created by an individual or a group. They may be used in different activities and by different individuals or groups of the same script.

D. Flexibility

The main drawback of scripts is their associated "risky" flexibility restrictions. An example is the decision related to the duration of activities. Even after defining it according to the course calendar or the class duration, unexpected situations may happen (eg teachers' strike, students' trip…) and the time structure of the CSCL script should be consequently modified (otherwise it will be impossible to finish it).

Some of these restrictions are intrinsic constraints of the script that justify their effectiveness. However, inflexible extrinsic constraints, such as the duration of activities, can spoil a satisfactory enactment of the learning scenario (Dillenbourg et al., in press). This requires modifications on the fly regarding the time structure, the resources or even the activities themselves and their order. Nevertheless, this need is actually present in any learning situation. That is why we focus our analysis on a common flexibility-demanding characteristic that significantly appears in CSCL scripts:

- Flexible group composition: a typical problem of CL is the variability of students' participation. It is often impossible for a teacher to guess the precise amount of participants that are attending a particular session, if they will be an even or odd number, whether some of them will join the class afterwards or cannot participate in a specific moment. These situations require unexpected group composition modifications.

Each requirement implies a different challenge. However, they are all shaped around the fact that they involve groups and multi-participant characteristics: specification of groups (hierarchy, size, amount, formation policy and dynamic formation), distribution of roles and resources (according to groups, roles and participants), coordination of activities, users' actions and artefacts (assuming that there are several participants, etc.), and flexible group composition. Next section tackles these challenges by implementing the manifestation of the requirements in two CSCL scripts well-known in the CSCL community.

Implementing the requirements using IMS LD

The problem of implementing the requirements of CSCL scripts using LD is approached by examining the needs that can be satisfied by the LD notation itself and the needs that can be supported using tools or related specifications. This section emphasises the role of the LD notation to express the requirements while next section discusses how tools and specifications complement the implementation of the requirements. To accomplish this analysis we make use of the examples described below.

Examples: Universanté and ArgueGraph Scripts
From a number of scripts reported and tested in the CSCL literature, we have selected two that feature all of the aforementioned requirements well. These examples are ArgueGraph, used in a master course on the design of educational software, and Universanté Script, which exploits socio-economics and cultural differences for teaching community health to students of different countries (Dillenbourg, 2002; Kobbe et al., 2005). Table 1 summarizes the scripts along with illustrating the requirements.

Table 1: Requirements of CSCL macro-scripts. Universanté and ArgueGraph Scripts

Expressing the requirements using IMS LD notation
Table 2 and Table 3 present the lessons learned when expressing the requirements of the Universanté and the ArgueGraph scripts with LD. The tables also include selected excerpts of suggested coding. They refer to the characteristics of the examples that illustrate well the use of LD elements and attributes for computationally representing each requirement.

Table 2: Computationally representing the requirements using LD, Universanté Script

Table 3: Computationally representing the requirements using LD, ArgueGraph Script

In addition, Figure 1 and Figure 2 show screenshots of specific runtime executions (runs) of the UoLs representing the examples using CopperCore LD engine (Vogten et al., 2006).

Figure 1: Screenshot of a run corresponding to the UoL that represents the Universanté Script. In this activity the members of the same thematic group (cancer) and the same country group (Switzerland) create a shared fact sheet concerning the thematic status in their country

Figure 2: Screenshot of a run corresponding to the UoL that represents the ArgueGraph Script. In this activity the tutor forms pairs of students according to their answers so that each student will argue with somebody having opposite opinions

Although modelling the examples was not easy and demanded a deep knowledge of the specification and several weeks of devoted work, these lessons learned show the many possibilities of LD for representing the requirements of CSCL scripts. Nevertheless, there are some detailed issues that cannot be formally expressed using the notation but which should be supported by tools or other specifications, which we now discuss.

Regarding group composition, the use of the create-new attribute provides flexibility since a specific number of groups is not predefined at design-time. However, the desired amount of groups cannot be explicitly defined in the same way as it can be done with the group size. In addition, the relationship between some roles to describe group arrangements (see Table 2) as well as other types of group formation policies cannot be formally specified. That is to say, it cannot be described in such a way that automatic binding of persons to roles according to these characteristics is achieved.

With regard to role distribution, rotation of roles can be implicitly represented by rotating activities and defining "neutral" roles as proposed in (Hernández-Leo et al., 2005). Moreover, in the case of the conference service four different types of system roles (participant, observer, conference-manager, and moderator) can be distributed. On the other hand, when creating a new occurrence of a role (allowed by the create-new attribute), new instances of the associated local role properties are also created. This facilitates the distribution

of resources, which in this case could not be accomplished by referencing different environments to different role instances (these instances are not available at design-time). Furthermore, when relying on the create-new attribute, the conditions used in Table 2 for checking if a person is a member of two groups at the same time should not be used. Instead, a new role modelling this situation should be declared and referenced in the corresponding role-parts.

To sum up, many of the requirements are addressed by the LD notation whereas some are not fully supported. Adding new constructs to LD would increase even more the complexity of the specification, what it is not desirable. At this point it is important to be aware of the difference between the LD specification itself and its relation to other (IMS) specifications and tooling.

Discussion

LD relies on other specifications and tools that conveniently integrated envisage an extensive support for the implementation of any teaching-learning process and, particularly, CSCL script. For example, to implement the questionnaire of ArgueGraph Script there are three possibilities: using LD properties (see Table 3), interoperating with IMS Question and Test Interoperability specification (Vogten et al., 2006) or referencing an external questionnaire tool (Hernández-Leo et al., 2006a). In this section we discuss how requirements of CSCL macro-scripts can be supported using related tools or specifications. In addition, we reflect on the possibilities of LD for representing CSCL micro-script and position related work on specifying collaborative learning processes.

In the discussion we will refer to four different types of tools:

1. authoring tools devoted to the creation of the scripts (eg *Collage* (Hernández-Leo et al., 2006b)),
2. players, whose main component is an engine (eg *CopperCore* (Vogten et al., 2006)) for running the script,
3. administration tools in charge of managing roles (and groups) and assigning participants to roles (eg *Clicc* de CopperCore),
4. and supporting tools, which refer to any kind of tools used to support an activity or part of an activity (eg a chat (Hernández-Leo et al., 2006a)).

According to the computational representations exposed in the previous subsection, administration tools should show to the users the textual information of the roles so that they are aware of the suggested group formation policies or amount of groups. It is also possible to envisage the possibility of having a dedicated specification to represent this type of characteristics. The resource referenced in the information element as well as the administration tool should be thus compliant to this specification. In this sense, administration tools should also adopt the IMS Enterprise Services specification (IMS, 2004), which defines how systems manage the exchange of information that describes people, groups and memberships within the context of learning.

Administration tools are also needed at runtime either as supporting tools, such as a "grouping service" providing the functionality to (automatically, semi-automatically or manually) assign people to roles (Burgos, Tattersall, & Koper, 2006), or just as a utility of the player. For example, in the Problem-Based Learning example included in (IMS, 2003) students dynamically decide who is going to be the chairperson during runtime. Once decided, he should be bound to the corresponding (previously defined) role using, for instance, a utility of the player. In the case of the ArgueGraph script an alternative to the use of properties is defining the Pairs as roles and using a grouping service to bind the students to the roles according to their responses. Of course, a "dedicated activity service" devoted to analyse students' answers and automatically form the pairs (by setting properties or binding users to roles) could be used as well.

Rotation of roles can be also realized if the player provides a mechanism that allows roles switching. In the Literature Circles example (IMS, 2003), each session, which implies role rotation, is viewed as a different run of the UoL. Regarding role distribution, supporting collaborative services may also define their own roles, which should contain references to roles in the LD. If the number of groups is not known at design-time, the requirement of resource distribution can be also supported using a "shared repository service". A different instance of the service (eg a different folder) is provided to each occurrence of the role participating in the service. An example of this solution is presented in (Hernández-Leo et al., 2006a). This idea is also employed

for providing a different instance of a "synchronous conference service" (a chat) to the members bound to each occurrence of a role. In addition, supporting collaborative tools may implement more sophisticated floor control mechanisms than the effect achieved using properties (see Table 3). Creating the fact sheet can be accomplished with a "collaborative text editor service" or a "collaborative whiteboard service" making use of, for instance, turn-taking policies.

Since UoLs can be designed considering adaptation issues (Koper et al., 2005), various flexibility issues can be pre-defined (eg activities to be added or removed by the teacher on the fly). Besides, the design of LD runtime systems meets flexibility requirements (Tattersall et al., 2005). The main idea behind this design relies on the distinction between an abstract description (UoL) and its specific instantiations (runs). Consequently, the same UoL can be executed in different settings with different participants. Administration tools should be also available at runtime for managing unexpected group composition variations. Moreover, minor modifications to runs are also possible by changing the details in the related UoL. Furthermore, (Zarraonandia, Dodero, & Fernández, 2006) propose an alternative to introduce slight variations on a run in progress without modifying the UoL. Finally, it is worth mentioning that another important supporting tool for CSCL scripts is an "awareness service". This kind of tool should provide information updated in real time by the engine about the progress of the participants (and groups) in the learning flow. This is not only essential for managing flexibility but it is also fundamental to provide an adequate context for each own work by understanding the work of others. The LD monitor service facilitates this functionality if it is previously specified in the UoL (Gorissen et al., 2005).

Supporting tools may also implement finer-grained scripts (micro-scripts) so that interaction processes within activities are scaffolded too. In fact, tools implementing particular floor control mechanisms (eg turn-taking polices) also aim at structuring interactions at a micro level. At this point the following question arises. Can LD be used for computationally representing micro-scripts? To answer this question, similar research to the detailed analysis contained in this article concerning macro-scripts needs to be carried out. Nevertheless, a positive answer is envisaged if we consider this example related to argumentative micro-scripts. The example is devoted to guide the construction of a specific argumentation sequence within discussion activities (Weinberger et al., 2005). The first message of a discussion thread is labelled "argument". The answer to an argument is categorized as "counterargument" and a reply to a counterargument is labelled as "integration". This script could be easily expressed with LD by means of grouped properties with titles "argument", "counterargument" and "integration" and coordination elements (mainly acts and conditions). Alternatively, other languages specifically dedicated to formalize particular types of micro-scripts (eg. argumentative micro-scripts) may be developed. In any case, the interoperability between macro and micro-scripts should be afforded.

In addition to IMS consortium, ISO/IEC JTC1/SC36 develops standards in the field of information technologies for learning. Particularly, WG2 is devoted to collaborative technology (ISO/IET, 2002). WG2's efforts are focused on standardizing the Collaborative Workplace (data collection and reuse for collaborative environments), Learner to Learner Interaction Scheme (peer-to-peer and group), and Agent/ Agent Communication (agent-based interfaces in collaborative environments). However, its progress in the last four years has been limited and no official outcome has yet become available. In fact, its proposal regarding Learner to Learning Interaction Scheme has similarities to LD and the WG has recently agreed to re-name this project to "Data Model for Text-based Communication by the Study Group".

Moreover, other EMLs for specifying collaborative learning processes are emerging as alternatives to LD. Some approaches explicitly target describing the characteristics of CSCL towards a domain specific language (Miao et al., 2005). Recently several EMLs were compared in a workshop (Vignollet et al., 2006). One of the conclusions of the workshop is further confirmed in this article: LD specification does support the implementation of CSCL macro-scripts. Only few details of the other approaches cannot be formally expressed using the LD notation itself, such as the automatic random allocation of participants to groups (LAMS, 2006). However, as noted above, this is a particular group formation policy which should be instead supported by administration tools.

Conclusion

The lessons learned described in this paper show the capacity of the LD notation to express CSCL macro-scripts. We have also discussed how the requirements can be supported by tools and related specifications. CSCL macro-scripts aim at structuring collaborative learning processes of coarse-grained activities. Their requirements are shaped around the fact that they involve groups and multi-roles characteristics.

These possibilities of LD to support the identified needs have been tested and illustrated by means of two examples (Universanté and ArgueGraph Scripts) in the article. Returning to the different types of generalized requirements, we summarize now how they are addressed by the notation itself and/or by other specifications and tools:

- The LD roles component and its related elements and attributes together with the joint use of properties and conditions provide constructs to computationally represent several group composition requirements. These requirements refer mainly to hierarchy of groups, group size and dynamic group formation. The notation provides limited support for the formal specification of the number of groups and group formation policies. However, these requirements as well as an enhanced realization of the others can be supported by related administration tools (and also supporting tools such as grouping services or player utilities) in combination with exhaustive group composition specifications.

- Similarly, role distribution relies on the constructs offered by the roles component, in this case complemented with the coordination of role-parts, in each of which a participant may play different roles. In addition, supporting tools may define specific roles implying different privileges when using the tools. Rotation of roles can be realized by rotating activities or by using mechanisms provided by the players. The distribution of resources is facilitated by the coordination of role-parts but also through the possibility of referencing resources to different elements of LD such as activity-descriptions or environments. The use of properties or supporting tools also provides another means of resource distribution.

- Coordinating the flow of CL activities is feasible using the LD method and conditions. The flow of artefacts can be attained by employing properties, global-elements and monitor services (as well as other specialised supporting tools) conveniently referenced by other LD elements. The consistency of shared artefacts is ensured by jointly held properties. Moreover, sophisticated floor control mechanisms can be described in the learning flow or simply realised by using supporting tools.

- Flexibility requirements are also tackled by both the LD notation and its implementation in tools. The main attributes of roles that enable flexible group compositions are min-persons, max-persons and create-new. Further flexibility is provided by the capabilities of LD to support adaptation as well as the distinction between abstract descriptions (UoLs) and specific instantiations (runs). This distinction affords new developments allowing the introduction of slight modifications to runs in progress.

Concluding, computationally representing CSCL macro-scripts using the LD interoperable notation provides the following benefits. Firstly, they can be repetitively and automatically processed. As a consequence, the scripts become resistant to technological changes. In addition, they can be reused in different settings and with different participants. And, furthermore, they can be easily adjusted to support other learning scenarios by using LD-compliant authoring tools. Current development in this area aim at teacher-friendliness and are focused on visual representations and the reuse of learning design solutions (Hernández-Leo, Harrer, Dodero, Asensio-Pérez, & Burgos, 2006). LD-based CSCL macro-script editors should hide the computer representational details laid out in this article.

Acknowledgments
The authors would like to thank the rest of the members of the OTEC centre and the GSIC/EMIC group. Davinia Hernández-Leo acknowledges partial financial support to the Spanish Ministry of Education and Science project TSI2005-08225-C07-04, Autonomous Government of Castilla and León project VA009A05, Kaleidoscope Network of Excelence FP6-2002-IST-507838 and, especially, the Research Fellowship Program of the University of Valladolid.

References

Burgos, D., Tattersall, C., & Koper, R. (2006). Representing adaptive eLearning strategies in IMS Learning Design. In R. Koper & K. Stefanov (Eds), *Proceedings of International workshop on Learning Networks for Lifelong Competence Development* (pp. 54-60) Sofia, Bulgaria.

Dillenbourg, P. (2002). Over-Scripting CSCL: the risks of blending collaborative learning with instructional design. In P. A. Kirschner (Ed), *Inaugural Address, Three Worlds of CSCL. Can We Support CSCL?* (pp. 61-91). Heerlen: Open Universiteit Nederland.

Dillenbourg, P., & Tchounikine, P. (in press). Flexibility in macro-scripts for CSCL. *Journal of Computer Assisted Learning.*

Gorissen, P., & Tattersall, C. (2005). A Learning Design worked example. In R. Koper & C. Tattersall (Eds), *Learning Design: a handbook on modelling and delivering networked education and training* (pp. 3-20). Berlin-Heidelberg: Springer Verlag.

Hernández-Leo, D., Asensio-Pérez, J.I., Dimitriadis, Y., Bote-Lorenzo, M.L., Jorrín-Abellán, I.M., & Villasclaras-Fernández, E.D. (2005). Reusing IMS-LD formalized best practices in collaborative learning structuring. *Advanced Technology for Learning, 2*, 4, 223-232.

Hernández-Leo, D., Harrer, A., Dodero, J.M., Asensio-Pérez, J.I., & Burgos, D. (2006). Creating by reusing learning design solutions. *Proceedings of the 8th International Symposium on Computers in Education*, León, Spain, in press.

Hernández-Leo, D., Villasclaras-Fernández, E.D., Asensio-Pérez, J.I., Dimitriadis, Y., Bote-Lorenzo, M.L., & Marcos-García, J.A. (2006a). Tuning IMS LD for implementing a collaborative lifelong learning scenario. *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies* (pp. 1160-1161) Kerkrade, the Netherlands.

Hernández-Leo, D., Villasclaras-Fernández, E.D., Jorrín-Abellán, I.M., Asensio-Pérez, J.I., Dimitriadis, Y., Ruiz-Requies, I., & Rubia-Avi, B. (2006b). COLLAGE, a collaborative learning design editor based on patterns. *Educational Technology and Society, 9*, 1, 58-71.

IMS (2003). *IMS Learning Design Best Practice and Implementation Guide*. Retrieved online 12 July 2006 at http://www.imsglobal.org/learningdesign/ldv1p0/imsld_bestv1p0.html

IMS (2004). *IMS Enterprise Services Specification*. Retrieved online 12 July 2006 at http://www.imsglobal.org/es/

ISO/IET (2002). *SC36/WG2: Collaborative Technology*. Retrieved online 12 July 2006 at http://collab-tech.jtc1sc36.org/

Kobbe, L., Weinberger A., Dillenbourg P., Harrer A., Hämäläinen, & Fischer F. (2005). *Framework for the specification of collaboration scripts* [Kaleidoscope Network of Excellence deliverable]. URL http://www.iwm-kmrc.de/cossicle/resources/D29-02-01-F.pdf

Koper, R., & Burgos, D. (2005). Developing advanced Units of Learning using IMS Learning Design level B. *Advanced Technology for Learning, 2*, 4, 252-259.

Koper, R., & Olivier, B. (2004). Representing the Learning Design of Units of Learning. *Educational Technology and Society, 7*, 3, 97-111.

LAMS (2006). *LAMS, Learning activity management system*. Retrieved online 12 July 2006 at http://www.lamsfoundation.org/

Miao, Y., Hoeksema, K., Hoppe, H.U., & Harrer, A. (2005). CSCL scripts: Modelling features and potential use. *Proceedings of Computer Supported Collaborative Learning 2005: The Next 10 Years!* (pp. 423-432) Mahwah, NJ: Lawrence Erlbaum Associates.

NISE (1997). *Doing CL: CL Structures*. Retrieved online 22 June 2006 at http://www.wcer.wisc.edu/archive/cl1/CL/

Tattersall, C., Vogten, H., Brouns, F., Koper, R., van Rosmalen, P., Sloep, P., & van Bruggen, J. (2005). How to create flexible runtime delivery of distance learning courses. *Educational Technology and Society, 8*, 3, 226-236.

Vignollet, L., David, J.P., Ferraris, C., Martel, C., & Lejeune, A. (2006). Workshop on "Comparing Educational Modelling Languages on a case study". *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies* (pp. 1149-1150) Kerkrade, the Netherlands.

Vogten, H., Martens, H., Nadolski, R., Tattersall, C., van Rosmalen, P., & Koper, R. (2006). CopperCore Service Integration - Integrating IMS Learning Design and IMS Question and Test Interoperability. *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies* (pp. 378-382) Kerkrade, the Netherlands.

Weinberger, A., Fischer, F., & Stegmann, K. (2005). Computer-Supported Collaborative Learning in higher education: scripts for argumentative knowledge construction in distributed groups. *Proceedings of Computer Supported Collaborative Learning 2005: The Next 10 Years!* (pp. 717-726) Mahwah, NJ: Lawrence Erlbaum Associates.

Zarraonandia, T., Dodero, J.M., & Fernández, C. (2006). Crosscutting runtime adaptations of LD execution. *Educational Technology and Society, 9*, 1, 123-137.

Table1: Requirements of CSCL macro-scripts. Universanté and ArgueGraph Scripts

Requirements	*Universanté Script*	*ArgueGraph Script*
Hierarchy of groups	There are country groups and thematic groups. Each thematic group is composed of case groups.	An even number of at least 4 participants (works best with 20-30 participants) and a tutor. Participants will be distributed among groups of two (see group formation policies at runtime).
Group size	(See amount of groups and group formation policies. Since there are at least two countries, each case-group has (at least) 2 persons of different countries. Since there are 2 cases per theme, each thematic-group has (at least) 4 persons. Since there are four cases, each country-group has (at least) 4 persons.)	
Amount of groups	At least two different thematic groups, two case groups (per thematic) and two country groups	
Group formation policies	Each case group is formed of at least 1 participant per country.	In one of the phases, all participants are distributed (by the tutor) evenly among groups of two, composed of participants with maximal difference in their responses to the questionnaire.
Dynamic group formation	(Not applicable)	
Role distribution	Each person belongs to three different types of groups, which implies playing specific roles depending on their "case", "theme" and "country".	(Pairs can be considered as different roles to be distributed, see group formation at runtime.)
Resource distribution	All case descriptions are distributed evenly among all case groups.	One copy of a questionnaire (including argument sheets) for each participant and another copy for each small group.
Flow of CL activities	• Within each case group, all participants discuss a clinical case using a discussion forum; regularly the case groups with the same thematic gather in the same discussion forum and identify common points and differences between the cases; • The tutor stimulates and guides both discussions; • Within each country group, the members of each thematic group in turn present (face to face) a synthesis of their case experience; • Within each thematic group, the members of each country group create a fact sheet concerning the thematic status in their country; • Within each thematic group, all participants discuss the similarity and difference between the fact sheets of different countries using a forum; • Within each country group, the members of each thematic group in turn present their fact sheets; • The tutor prompts some methodological issues; • Within each country group, the member of each thematic group modify the fact sheet according to the methodological comments; • Within each case group, all participants propose a health strategy to cope with the case problem; • The tutor has access to the final fact sheets and health strategies.	Survey phase: • Students individually fill in a questionnaire about design principles in courserware; • This process is monitored and ended by the tutor; • All learners then see all responses and jointly discuss them informally; • The tutor forms pairs of students; Conflict phase: • Within each small group, the participants negotiate on a single choice to the same questionnaire and generate a shared argument. They can read their previous answer; • The tutor reviews all arguments produced by individuals and relates them to the various theoretical approaches in the domain; Elaboration phase: • The tutor reviews all arguments produced by the pairs and relates them to the theories; • Students read tutor's comments on how to relate their arguments to theories and concepts; Reflection phase: • Each student individually writes a synthesis of all arguments, taking into account the advise of the teacher; • This process is monitored and ended by the tutor.
Floor control	Fact sheets and health strategies are shared artefacts that require floor control mechanism to ensure data consistency.	Achieving a consensus when answering the questionnaires should consider a floor control mechanism to ensure the consistency of the shared answer.
Flow of artefacts	Since the fact sheets are created until they are finally made available to the teacher, they are used in discussions within theme groups, presented within country groups, commented by the teacher, and modified by their authors.	The answers to the questionnaires (choices and arguments) of individuals and small groups are displayed in different activities.
Flexible group composition	What happens if we do not have the precise desired number of participants or if it varies at runtime?	

Table 2: Computationally representing the requirements using LD, Universanté Script

Requirements	*Involved LD elements and attributes*	*Illustrative excerpts, supposing that there will be 2 countries and 4 participants per country, ie 2 thematic groups comprising 2 case groups*
Hierarchy of groups	Groups are modelled using roles, which can be bound to several persons. Roles can be nested, indicating that a role is divided in sub roles.	Each thematic group comprises several case groups. <roles> <learner identifier="R-thematic-group-cancer" min-persons="4"> […] <learner identifier="R-case-group-breast_cancer" min-persons="2"> […] </learner> <learner identifier="R-case-group-lung_cancer" min-persons="2"> […] </learner> </learner> <learner identifier="R-thematic-group-aids" min-persons="4"> <learner identifier="R-case-group-pregnant" min-persons="2"> […] </learner> <learner identifier="R-case-group-drug_addict" min-persons="2"> […] </learner> </learner> <learner identifier="R-country-group-switzerland" min-persons="4"> […] </learner> <learner identifier="R-country-group-cameroon" min-persons="4"> […] </learner> […] </roles>
Group size	Role attributes min-persons and max-persons specify the required minimum and maximum numbers of persons bound to the role.	
Amount of groups	Each group can be modelled as a role. An alternative is using the role attribute create-new, which indicates that multiple instances of the role (and their sub roles) can be created during runtime (Hernández-Leo et al., 2005).	
Group formation policies	This requirement cannot be formally specified but it can be added as information of the role in the referenced resource.	<learner identifier="R-case-group-breast_cancer" min-persons="2">[…] <information> <tem identifier="I-relation-case-country-groups" identifierref="R- relation-case-country-groups " /> </information> </learner> […] The resource "R- relation-case-country-groups" can be a text file indicating "Each case group is composed of at least 1 participant per country"
Role distribution	Persons can be bound to one or several roles in the same run of the UoL. In this example each person should be bound to one country group and one case group (and thus to one thematic group). The moment in which they are playing each role is specified in the learning flow using the role-part element. To explicitly indicate that persons can be matched exclusively to one of the sub roles (eg case groups within a thematic group) LD provides the role attribute match-persons.	
Resource distribution	The resources can be associated to activity-descriptions or to environments, referenced in turn by other LD elements depending on the distribution needs.	In this example, an environment per "case description" (a learning-object) is defined. An activity-structure per case is also defined. Each activity-structure references one of the environments and a common learning-activity explaining the task. Each activity-structure is bound to a role in different role-parts of the same act.
Flow of CL activities	The flow of activities is expressed in the method. A method contains one or more plays, which are modelled according to a theatrical play with acts and role-parts. The plays run in parallel. Acts together with conditions (and also notifications) determine whether, when, and for what roles activities and resources need to be available.	This example requires a method with five acts. Each act contains a role-part per role of the "type" of role that corresponds to each phase. In the cases that the activities are performed by persons belonging at the same time to two groups (Eg "within each thematic group, the members of each country group create a fact sheet"), it is necessary to add conditions with two expressions of type is-member-of-role (see Figure 1) <if> <and> <is-member-of-role ref="R-thematic-group-cancer" /> <is-member-of-role ref="R-country-group-switzerland" /> </and> </if> <then> <show> <class class="C-fact-sheet-cancer-switzerland" /> </show> <hide> <class class="C-fact-sheet-cancer-cameroon" /> <class class="C-fact-sheet-aids-switzerland" /> <class class="C-fact-sheet-aids-cameroon" /> </hide> </then>
Flow of artefacts	Properties can be used to model individual and shared artefacts. Global-elements and monitor services are used to set and view the value of their own or that of others properties. These elements are referenced by the different activities that require the artefacts.	The value of the properties modelling the facts sheets can be set and viewed by the participants by means of global-elements in the several activities. In this excerpt users can modify their fact sheet. <html […]> Please modify the fact sheet (Cancer status in Switzerland) according to the methodological comments: <ld:set-property ref="L-fact-sheet-cancer-switzerland"/> […] </html>

Table 3: Computationally representing the requirements using LD, ArgueGraph Script

Requirements	*Involved LD elements and attributes*	*Illustrative excerpts, supposing that in ArgueGraph there will be only 4 participants, ie 2 pairs*
Dynamic group formation	Local personal properties can be used to model groups so that their value can be determined at runtime using global-elements. The persons with the same value of the property are in the same group. Conditions are in charge of coordinating the activities and artefacts according to this value of each participant's property (ie according to their group).	The tutor dynamically determines at runtime to which group each student is bound (see Figure 2). Depending on the group, access to an activity containing a different instance of the shared questionnaire is provided. <locpers-property identifier="LP-pair"> [...] <datatype datatype="string"/> <restriction restriction-type="enumeration">PairA</restriction> <restriction restriction-type="enumeration">PairB</restriction> </locpers-property> [...] <if> <!-- If the student is in Pair A --> <is> <property-ref ref="LP-pair"/> <property-value>PairA</property-value> </is> </if> <then> <show> <learning-activity-ref ref="LA-fill-in-pairs-questionnaire-PairA"/> </show> <hide> <learning-activity-ref ref="LA-fill-in-pairs-questionnaire-PairB"/> </hide> </then>
Flow of CL activities	(See Table 2)	This example requires a method with four acts. Each act contains two role-parts one for the activities of the students and one for the activities of the tutor. The exception is the act corresponding to the "conflict" phase. This act comprises three role-parts one for each Pair and one for the tutor. Note that although the role referenced in the learners' activities are not explicitly the Pair but the Students, the conditions manage to show the correct activity to the students (according to their associated Pair). <method> <play identifier="P-1" isvisible="true">[…] <act identifier="A-2"> <title>Conflict phase</title> <role-part identifier="RP-Student-2-PairA"> <role-ref ref="Student"/> <learning-activity-ref ref="LA-fill-in-pairs-questionnaire-PairA"/> </role-part> <role-part identifier="RP-Student-2-PairB"> <role-ref ref="Student"/> <learning-activity-ref ref="LA-fill-in-pairs-questionnaire-PairB"/> </role-part> […] </act> […] </play> […] </method>
Floor control	Properties (local or global properties or local role properties) can model "shared" artefacts whose consistency is managed as following. All participants with access via global-elements to the property can view and modify its value. The final value is the latest set.	The persons in the same group (Pair) share local properties for their joint answers to the questionnaire and related arguments. <-- Answer to the question: In a courseware, when a student makes an error is better to --> <loc-property identifier="L-pairA-answer1"> <datatype datatype="text"/> <initial-value>Select</initial-value> <restriction restriction-type="enumeration"> 1. Tell the student he made a mistake and give him the correct answer. </restriction> [...] <restriction restriction-type="enumeration"> 4. Give the student some time to find out the mistake by himself. </restriction> </loc-property> <loc-property identifier="L-pairA-argument1"> <datatype datatype="text"/> </loc-property>
Flexible group composition	The possibility of detailing the maximum and minimum number of persons within each group (see Table 2) expresses flexibility regarding group composition variability. In this example, more than 4 persons can be involved and the small groups can be of more than two persons depending on the students' participation in a certain moment. (Eg if there are five participants, the teacher can form a group of two and a group of three). In addition, since the tutor forms the pairs at runtime, the particular needs of the moment will be considered. Later the tutor can also change the association of a student to a group coming back to the pair formation activity.	

Figure 1: Screenshot of a run corresponding to the UoL that represents the Universanté Script. In this activity the members of the same thematic group (cancer) and the same country group (Switzerland) create a shared fact sheet concerning the thematic status in their country

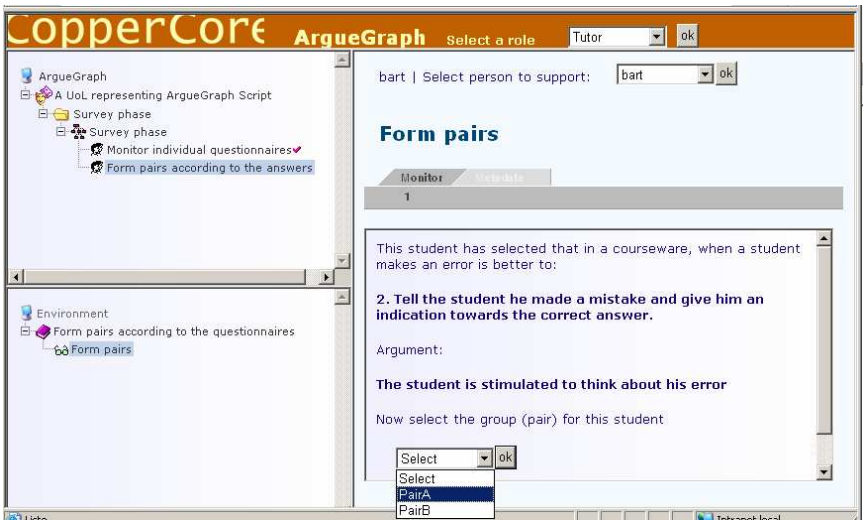

Figure 2: Screenshot of a run corresponding to the UoL that represents the ArgueGraph Script. In this activity the tutor forms pairs of students according to their answers so that each student will argue with somebody having opposite opinions