Computer Applications in Engineering Education (2013). 21(3), 421-431.
DOI: http://dx.doi.org/10.1002/cae.20486

CourseEditor: A Course Planning Tool Compatible with IMS-LD

Patricia López Cuevas, Pedro J. Muñoz-Merino*, Carmen Fernandez-Panadero, Carlos Delgado Kloos

Department of Telematic Engineering, Universidad Carlos III de Madrid

Av. Universidad, 30. E-28911 Leganés (Madrid) Spain

pedmume@it.uc3m.es, Tel: (+34) 91-624-6233, Fax: (+34) 91-624-8749

Abstract

For the successful adoption of Computer Based Learning (CBL), it is necessary to provide teachers with user-friendly authoring tools that guide them in the process of planning a course. It is important that the documents generated by these tools are compliant with CBL standards to ensure that the instructional designs made by the teachers remain valid regardless of the Learning Management System (LMS) used to deliver the course. IMS-Learning Design (IMS-LD) is one of the most accepted specifications by the educational community for modelling of learning processes. There are some authoring-tools that allow export learning designs in IMS-LD but they are not course-planning oriented. In addition, a big challenge is how to improve the user interfaces of these tools in order to be easier to use for regular teachers.

This paper presents CourseEditor, a course planning authoring tool that allows a teacher to describe the complete planning of a course (objectives, contents, methodology and evaluation) and export the results in an IMS-LD compatible format. The paper provides a novel modelling of course planning that takes into account ideas from different instructional theories. CourseEditor combines power and flexibility, with the simplicity of an interface that can be used by teachers with no technical background. The paper illustrates the use of the tool creating a course on Telematic Services in the context of a Telecommunications Engineering degree. In addition, the relationships of course planning and IMS-LD are presented, showing which information of course planning can be represented in IMS-LD and which not.

Keywords: *authoring tool; course planning; software tool; educational standards*

* Corresponding author

1. Introduction

Course planning includes a complete course description. The traditional phases of course planning are:

1) *Objectives*. Descripcion of a set of skills, knowledge, or learning outcomes that is expected that the student has acquired at the end of the course.
2) C*ontents*. The description of the different didactic units with their related tasks.
3) *Methodology*. This includes the specific task sequencing, the collaborative techniques, the different kinds of tasks, content personalization, etc.
4) *Evaluation*. It is a measure of the students' level with respect to the course objectives. The course must introduce evaluation tasks for this purpose.

Course planning is a well-known issue in face to face learning and many books have been written on this topic. Many of the lessons learned in face to face learning remain valid with the introduction of computers to support learning but is necessary to review these methodologies and instructional theories to cope with the changes and new challenges presented by Computer Based Learning (CBL). It is necesary to consider extra aspects such as for example: the use of technological functionalities through Internet, the way to achieve a standard digital format of all the aspects of the course planning in order to enable course distribution through computers, interoperability between different systems or reusability between different teachers and course designers.

At present, there are different e-learning specifications that cover different issues of CBL. Some examples are IEEE-LOM [1] to describe metadata for learning resources, IMS-QTI [2] to describe question and test interoperability, IMS-CP [3] and SCORM [4] for content packaging, or IMS-LD [5] for learning design. Technological systems should be compliant with most of them. In the context of course planning, IMS-LD is the specification most widely accepted by the educational community. IMS-LD covers the description of important aspects of course planning, but not all of them.

For the successful adoption of CBL, it is necessary to provide user-friendly authoring tools that cover the edition of most of the course planning aspects, being compliant with e-learning specifications. There are several IMS-LD authoring tools that cover IMS-LD aspects but they are not course planning oriented so they do not include all the different phases of course planning. In addition, some of these tools cover a lot of potentiality of the specification but they are too difficult to use by teachers and course designers without technical knowledge.

In this paper a new authoring tool (*CourseEditor*) is presented to overcome some of the challenges of course planning in CBL. Its novelty lies in the combination of the following three aspects:

- Software solution that let the description of a complete course planning using different educational models. It provides a novel course planning modeling, using graphical and text representations.
- Compatible with the IMS-LD specification. This paper shows which aspects of course planning can be modelled using IMS-LD and which not. Course Editor generates XML (Extensible Markup Language) files according to the IMS-LD specification and it also generates other files about relevant issues in course planning not covered by IMS-LD. This extra- information is provided in XML format to facilitate the possible mapping to other educational specifications.
- Easy to use by teachers and course designers. In this way, a regular teacher has not complex barriers to introduce into the e-learning course creation. This is taken into account in the modelling phase.

In order to illustrate how to use this tool in Engineering Education, this paper shows the use of CourseEditor for the planning of a course in "Telematic Services" in the context of a Telecommunication Engineering degree.

The rest of the paper is organized as follows. Section 2 presents an overview of related work, pointing out the most innovative aspects of CourseEditor and the main differences between this tool and the previous

work done in this area. Section 3 describes the process of modelling a course (textual and graphically) using Course Editor. This process takes into account different theories from course planning. All the process is illustrated modelling a course in "Telematic Services" in the context of a Telecommunication Engineering degree using Course Editor. Section 4 is devoted to discuss the relationships of the IMS-LD specification with respect to course planning. Finally, Section 5 includes our conclusions and future work.

2. Related Work

This section presents an introduction to course planning, a brief description of the IMS-LD specification, and a detailed discussion about some existing IMS-LD authoring tools and their relationship with CourseEditor.

2.1 Course Planning

The concept about course planning has evolved during the time. There are different historical educational approaches that have influenced and contributed the present course planning theories. One of the first contributions is the proposed by Thorndike in [6] where there is a comparison between studying text books and traditional teachers in the classroom (normally as oral teaching), and a list of examples of teaching strategies or types of tasks is given such as illustrations, practice activities, questions, explanations, or directions. The objectives to achieve in a learning experience have been the focus of different studies such as the work of J.F. Bobbitt [7] or Tyler [8]. The Tyler work [8] included the objectives concept as base to elaborate a didactic planning composed by contents and the correspondent evaluation based on these objectives. The work authored by Skinner [9] is one of the key contributors of behaviourism to education and proposed to divide the material in small pieces of tasks and provide reinforcement or feedback depending on the student success or failure. Later, Skinner [10] developed a machine that provided contents

in a sequence to students but including reinforcing. Furthermore, educational cognitive approaches were introduced with student-focused theories. In this line, Gagne [11] established hierarchical relationships between tasks and activities, and a classification of objectives was provided. Mager [12] proposed the division of objectives in general, intermediate and specific. Pask [13] introduced some type of personalized learning where sequencing is not linear but depending on the student. The Hilda Taba model [14] states that the didactic planning must be based on a logical, chronological or methodological criterion. The different activities must be organized and sequenced properly, distinguishing individual from group activities and teacher from student ones. Furthermore, this model states the different phases that are accepted in most of today's course planning theories: objectives, contents, content organization (methodology), and evaluation. There are different reviews about course planning and more details can be obtained, such as for example in the work of Reigeluth [15].

More recent studies have focused on specific parts of course planning. The Monterrey didactic Techniques [16] provide several guidelines that among other things, enumerate a set of types of tasks in instruction. The Pedagogical Patterns Project [17] shows different pedagogical patterns. Different styles and strategies of learning can be extracted from Pask [18]. More recently, The Index of Learning Styles from Felder [19] provides studies about personalization, related to classify the different types of students in groups depending on their learning styles and shows the importance of providing students with strategies and techniques according to how they learn.

2.2 IMS-LD

Learning Design (IMS-LD) [5] is one of the specification more related with course planning and it is also one of the most widely accepted by the educational community but its adoption by teachers and

educational institutions is still low. The expressiveness of IMS-LD for modeling courses and its wide acceptance by the education community have been the two key factors for choosing this specification as the format for storing and exchanging information for the CourseEditor tool. IMS-LD models different aspects of a Unit Of Learning of a course. Such aspects are for example the different roles that participate in a course, the activities or resources, the synchronization of different user actions or the activity or resource sequencing depending on conditions.

IMS-LD has a set of four documents: Conceptual model (it defines the basic elements), Information model (it defines all the elements and attributes in a formal way), XML binding (it represents the information using XML elements) and Exercises and Implementation Guide (it provides specific examples and guidelines to implement systems based on this specification). In addition, IMS-LD has three different implementation levels: Level A (defines the basic elements such as roles, activities, resources or tasks sequencing), Level B (includes properties and conditions) and Level C (adds notifications). The work [20] provides practical information about development of courses using IMS-LD, using tools, etc.

2.3 IMS-LD Authoring Tools and players

IMS-LD files can be run by players that interpret and execute these files. Some IMS-LD players already exist such as the RELOAD learning design player [21] or the one implemented in the .LRN platform [22]. These platforms are able to launch the complete design of a course for the different roles, and present the different activities according to the designed file description.

Moreover, there is a need of IMS-LD authoring tools for creating the correspondent units of learning that can be loaded within the players. There are several developed authoring tools at present. Griffiths and Blat [23] classified the different authoring tools between close or distant to the specification (high level tools). The close to the specification tools, such as RELOAD [24] [25], or COSMOS [26], permit most of the

possibilities of the specification but at the prize of not being easy to use by teachers without technical knowledge, so they are recommended for experts. Another close to the specification tool may be WebLD [27] that includes semantic technology based on a defined IMS-LD ontology.

On the other hand, the high level tools are easier to use by teachers without technical knowledge but they have usually more restricted functionalities. They can be divided into pattern based or graphical based. In the context of pattern based, Collage [28] is a collaborative learning design authoring tool based on patterns called Collaborative Learning Flow Patterns (CLFPs) [29] that include a set of best practices, while CHOCOLATO [30] is an authoring tool guided by different pedagogical models that uses an ontology. Some graphical representation tools are MOT [31], LAMS [32] [33], ASK-LDT [34], eLiveSuite [35], or UML based frameworks like the described in [36]. Each one includes a different part of the IMS-LD expressivity, and represents the activity sequences with a different graphical approach.

CourseEditor, the authoring tool presented in this paper, can be classified as a high level tool. The main difference with respect to previous work, is that it is course planning oriented. CourseEditor follows the recommendation of Koper in [37] about "concentrate on the educational problem, find solutions and implement the solutions in IMS Learning Design". This new authoring tool is focused on the way a teacher makes course planning, thinking about what a teacher does and hiding all technical details about IMS-LD but using this format to store and interchange course planning model. CourseEditor includes aspects related to course planning not covered as far as we know in other IMS-LD compatible tools, such as extra general information about the course, the classification of objectives and weight assignment, the definition of generic types of tasks with generic configurable parameters, the division in didactic units with correspondent tasks, the duration of the different elements, evaluation and submission parameters, a link between tasks and evaluation, etc. CourseEditor allows a wide spectrum of educational models.

In addition to it, another difference is that CourseEditor provides a specific graphical representation solution for the sequencing of tasks. The tool that is closer to our approach of sequencing is LAMS, but there are differences such as multiple arrows can have the same source element, support tasks can be visualized, or different way of setting the minimum number of tasks in an alternative branch.

3. Course Planning Modelling

In this section, we show the process of course planning modelling through the proposed software tool. The model takes as base the different theories about course planning that were cited in the related work. Different screens of CourseEditor, which represents the model, reflect the combination of different aspects of the course planning theories. The CourseEditor software tool is available at http://www.it.uc3m.es/pedmume/CourseEditor/ [38] where it can be downloaded and executed.

There is a trade-off between ussability for teachers without a high technological expertise and completeness. Usability is the key aspect that guided the design process and completeness has sometimes been sacrificed to achieve it. Therefore, the model can also be useful to understand and combines graphical and textual representations to support IMS-LD level A.

In order to illustrate the process of couse planning and to provide different views of the tool a study case is presented in which CourseEditor has been used to design a course in '*Telematics Services*' in the context of Engineering Education. Figure 1 shows that there are seven steps in the model for the creation of a course which corresponds to seven tabs.

3.1 Textual Modelling

The three initial steps to design a course are: provide general information, define the duration and select the objectives (learning outcomes) of the course. In the course general tab, a teacher fill in the name of the

course (‘*Telematics Services*’ in this case), the description (it includes why the course is useful and differences with respect to other related courses), the audience (the people to whom the course is intended for) and the prerequisites (the advertised previous required conditions to register in the course). The course duration tab sets the number of required hours (60 in our case), the start and end dates, and the ECTS (European Credit Transfer System) credits.

Figure 1 show the tab to introduce course objectives. The tool allows setting an unlimited number of course objectives. Each objective has a weight that represents its relative importance with respect to the course. The sum of all the course objective weights must be the unity. An objective can be also classified as generic to a course or specific of a didactic unit. In addition, an objective can belong to one or more of the following categories: conceptual (if it is related to student comprehension and transmission of knowledge), procedural (if it is related to skills acquirement) and attitudinal (if it is related to form habits and attitudes). For this specific course, there are several objectives from which we can see five in Figure 1.

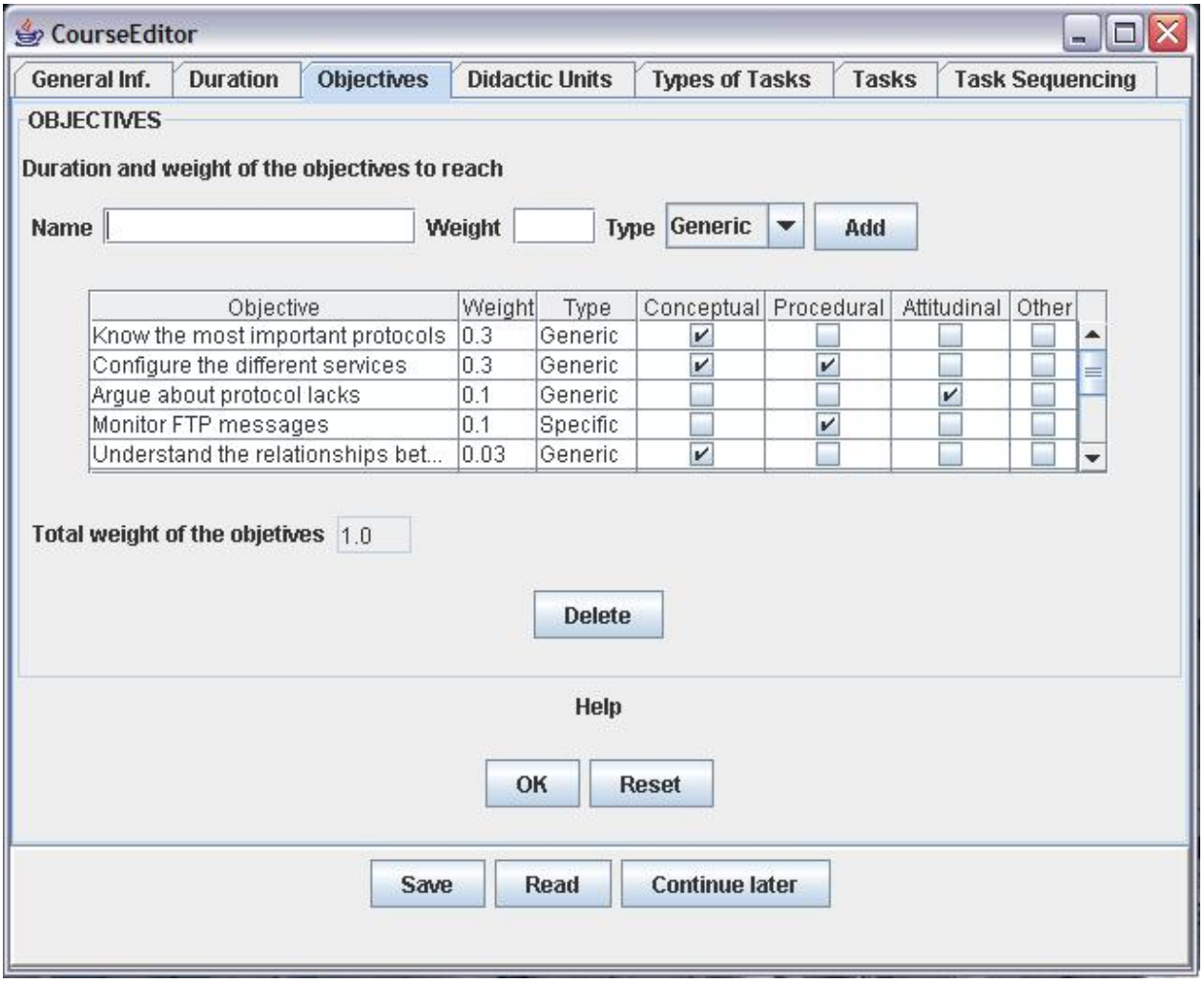

Figure 1: Course Objectives screen

For example "*Configure different services*" objective has a weight or importance of 0.3 out of 1, is generic, conceptual and procedural. New objectives can be defined with the "*Add*" button and the correspondent fields at the top, while existing objectives can be edited or removed.

Fourth step is to describe the different didactic units in which '*Telematics Services*' course is divided. An unlimited number of didactic units can be added with the tool. Each didactic unit is defined with a title, description and duration. In addition, each didactic unit is related to the defined objectives using a table, so if a didactic unit covers a specific objective, then the teacher must select it.

Figure 2 shows the different types of tasks that the teacher could select to deploy the course. Each type of activity belongs to one or more of these categories: conceptual, attitudinal, procedural, or others. By default, there is a predefined set of common used task types (chat, forum, test, practical exercise, study case, etc.) but the users can add new types if required by their courses. Each task type has an associated set of parameters that characterize it properly (name, description, type of service necessary to deploy at runtime, etc.).

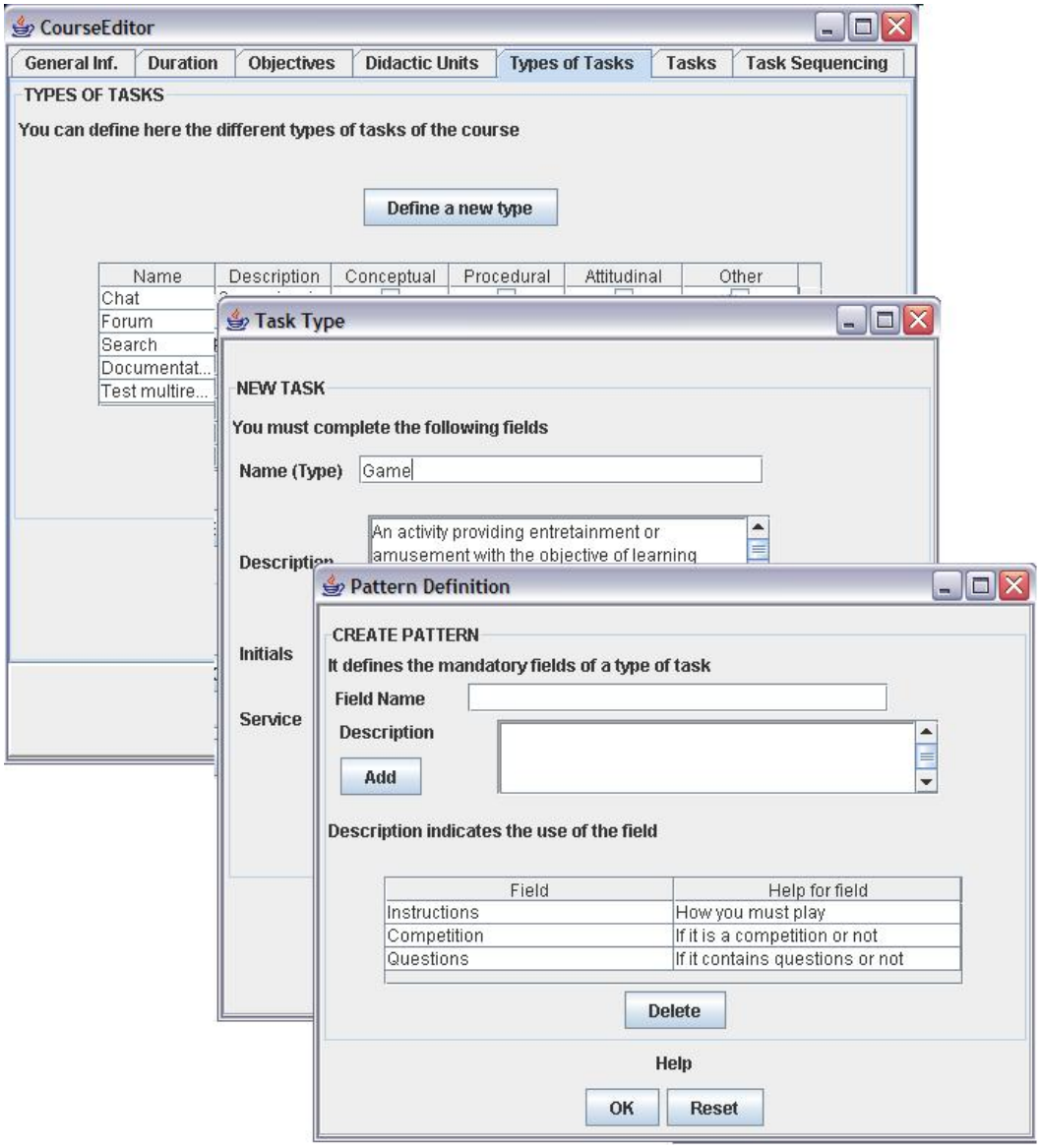

Figure 2: Screens for defining a new type of task

Figure 2, '*Define a new type*' tab, shows the two screens to define a new type of task. The first window is to write the value of mandatory parameters common to all types of tasks (name, description, initials, service), while the second window is to add specific parameters for the type of task. In this course, we have added a new type of task called '*Game*', with initial G. Its corresponding specific parameters are defined with a name and a text description. The specific game values for these parameters will be set when defining a particular task of this type. In this case the three parameters that are defined are: if it is a game competition or not, if it contains questions or not, and the game statement the teacher would write as introduction to students. Nevertheless, an undefined number of parameters can be defined to represent some type of activity. Furthermore, users can extend existing task types to create new ones, avoiding in this way starting from the scratch.

Figure 3 shows two windows: the course tasks list and the specific information for each task. The course task list includes all the tasks for all the didactic units, there is a scroll bar to select one of them. Each task is showed with information of its type, duration, didactic unit, number of support activities, and related objectives. The user can add new tasks and configure all of these details for each one of them.

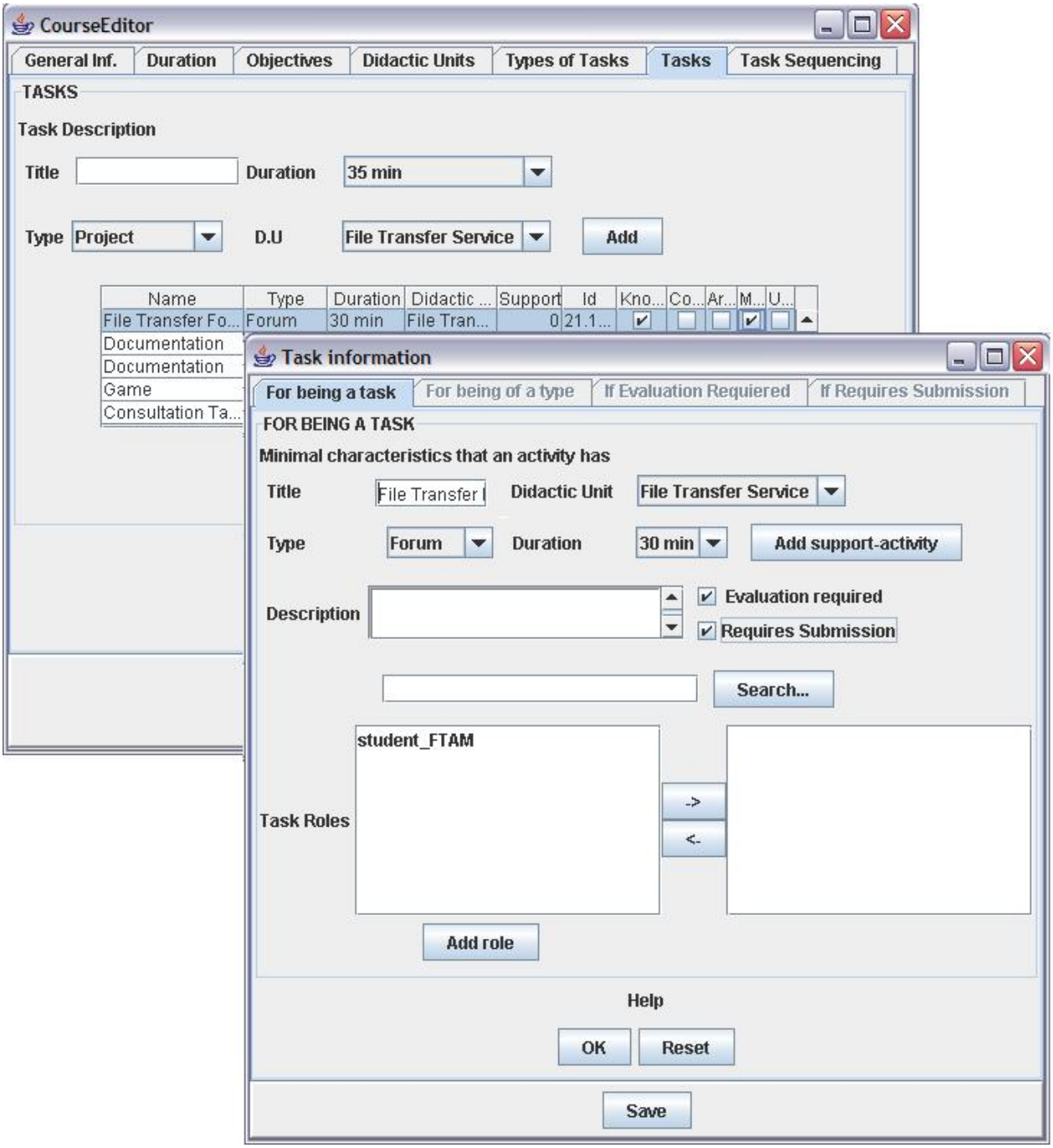

Figure 3: Screen for task configuration

In Figure 3, a new task called "*File Transfer Forum*" is added in the "*File Transfer Service*" didactic unit. In this task the student has to participate in a discussion forum and then send a file to the teacher with the conclusions of the debate. As shown in this figure, this task has been estimated in 35 minutes, it is related

to two objectives and it is an instance of the forum task type. Moreover, Figure 3 shows the configuration window for a task. In this window the "*File Transfer Forum*" task is marked as *evaluable* (because the student will receive a score for it) and *submission required* (because it requires that the student send a file with the conclusions to teacher). In the *Description file* field the teacher writes the task statement directly or it can be uploaded from a file text using the Search option. Every task must have at least one role to perform it, but it can have more. The roles that must perform the task are selected from a list. New roles can be added with the '*Add Role*' option. If this option is selected a new window is opened in which two fields must be fulfilled: *role name* and *description*, and one parameter must be selected: if it is a *student role* or a *teacher role*. This last parameter will assign the inheritance of the defined role. Task roles can be added or removed to a task with the correspondent left and right arrows.

Moreover, support activities can be added related to the task with the "*Add support-Activity*" option. A support activity is a task to support the initial task. For example, in this case a support activity for the teacher role could be to answer students' questions on the forum. CourseEditor provides a window for defining *support activities* that request the following information: *task name*, *description*, the *role* who makes the support, and the *role* to whom the support is made.

The "*For being of a type*" tab shows the information to fill by the fact of being this task from a type. In this case, it would show all the parameters of a forum type and it let introduce the correspondent text value for each parameter. The "*Requires evaluation*" and "*Submission Required*" tabs are only activated in case these two features are selected in the initial window. If the activity requires submission, then a set of fields need to be added with the criterions, scoring, solution and feedback to the student. In case the activity was submission required, then a new window (Figure 4) would need to be filled with the available date, submission deadline, what to submit, how to submit, and if submissions out of time are allowed.

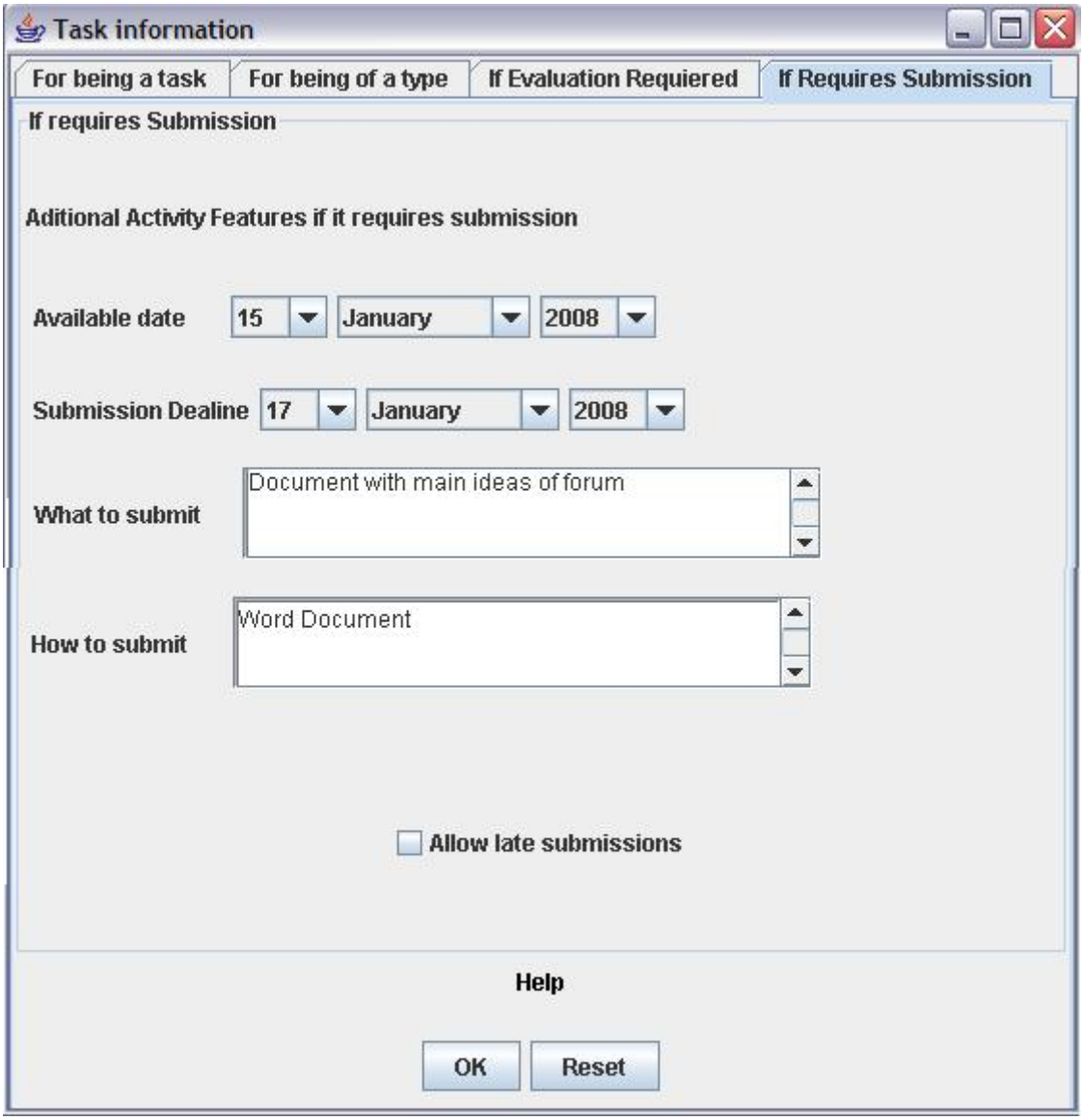

Figure 4: Screen for activities in which submission is required

Finally, the last step is to sequence the different defined tasks for each didactic unit and the own didactic units between them. In the "Task Sequencing" tab, the different didactic units of the course are presented and they can be moved in whichever order. This is the order in which the didactic units will be sequenced within the course.

Moreover, the synchronization level between roles can be set in this window. The tool let three types of role synchronization:

1) *Task synchronization*. A user that belongs to a role and he/she has finished a task, can only continue the following task if and only if all the users of the different roles have finished their correspondent task at this point.
2) *Didactic unit synchronization*. A user that belongs to a role and he/she has finished a didactic unit, can only continue the following didactic unit if and only if all the users of the different roles have finished the same didactic unit at this point.
3) *Course synchronization*. A user that belongs to a role can advance through the course with independence of the rest of the course roles.

3.2 Graphical Modelling

Regarding the graphical modelling, in the same window where the synchronization level is selected, we must select every didactic unit to sequence their tasks. Pressing the correspondent button of "*Task Sequencing*", a graphical interface appears.

Figures 5 and 6 show two graphs illustrating the process of task sequencing using the CourseEditor's graphical interface. On the left, there are a set of buttons with different types of tasks. If one of them is pressed, then an instance task of this task-type would appear at the graphical editor on the right. The teacher can select among previous defined tasks for each specific didactic unit. The graphical interface shows all the previous defined tasks-types for each didactic unit.

Figure 5 corresponds to the "*File Transfer service*" didactic unit. There are two different roles of students, called: student_FTAM that studies the FTAM (File Transfer Access and Management) protocol, and student_FTP that studies the FTP (File Transfer Protocol) protocol. Both students perform the "*File*

Transfer Forum" task (this task was commented previously). Next, we can see four tasks between two union elements. Two of these tasks are related to student_FTAM (the reading tasks of Documentation type) while the other two tasks (the Game and Search tasks) to student_FTP role. Through the union element properties we can set that student_FTAM role must perform at least 1 out of the 2 reading activities, and student_FTP role must perform the two activities to advance to the following task. Finally, both roles must perform the "*Chat Transfer service*" task which finishes the course. In the graphical modelling, when a union is found, it denotes that a role has different tasks to select for which a minimum must perform or that different roles must perform different tasks or a combination of both.

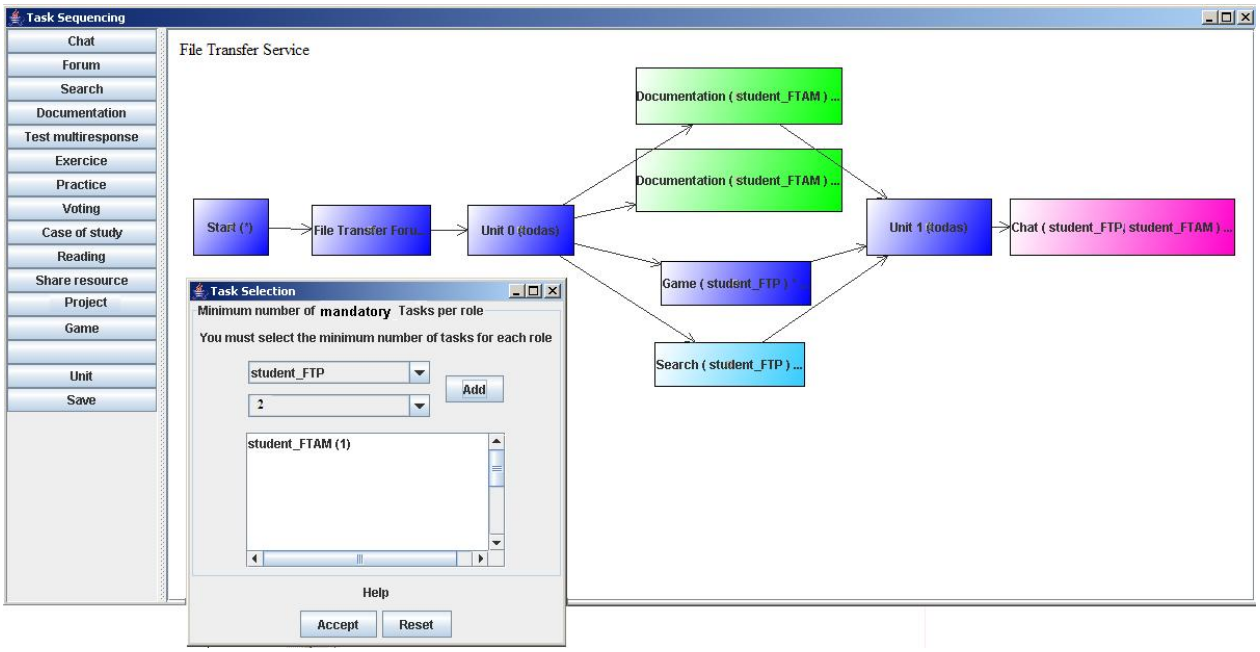

Figure 5: "File Transfer service" didactic unit

Figure 6 shows the sequence of the "*electronic mail service*" didactic unit. There are two roles of students for this didactic unit: student_SMTP and student_POP. The student_SMTP role studies concepts related to the SMTP (Simple Mail Transfer Protocol) protocol while student_POP studies concepts related to the POP (Post Office Protocol) protocol. The first task is common for both roles and it is a reading with general aspects about the electronic mail service. This task also contains a support activity in which teachers (that belong to the teacher role) help students with these initial general concepts. Next, there is a union that indicates that student_SMTP role will be delivered with different tasks with respect to the

student_POP role. The student_SMTP role receives a reading documentation task about SMTP protocol. When a student finishes this task, then he/she can select to do 1 out of these 2 things:

1) A study case and a project,
2) A multi-response test.

When at least one of these two things is performed, then the student can advance to the following task. This selection chance is represented by another union.

The student_POP role has a total analogy with respect to the student_SMTP as it can be seen in its correspondent branch.

Finally, from Figure 6, we can infer that both student roles perform together a set of tasks. First, there is a chat task with the purpose that a student group tells the other group about POP and the other group tells the initial one about SMTP, interchanging the learned information. There are a study case and a project activities where both student roles must work collaboratively in order to combine the SMTP and POP concepts to solve it.

When the seven steps are completed, the course is perfectly defined and it can be saved and the correspondent IMS-LD file and other XML files with different information are generated, describing the created course.

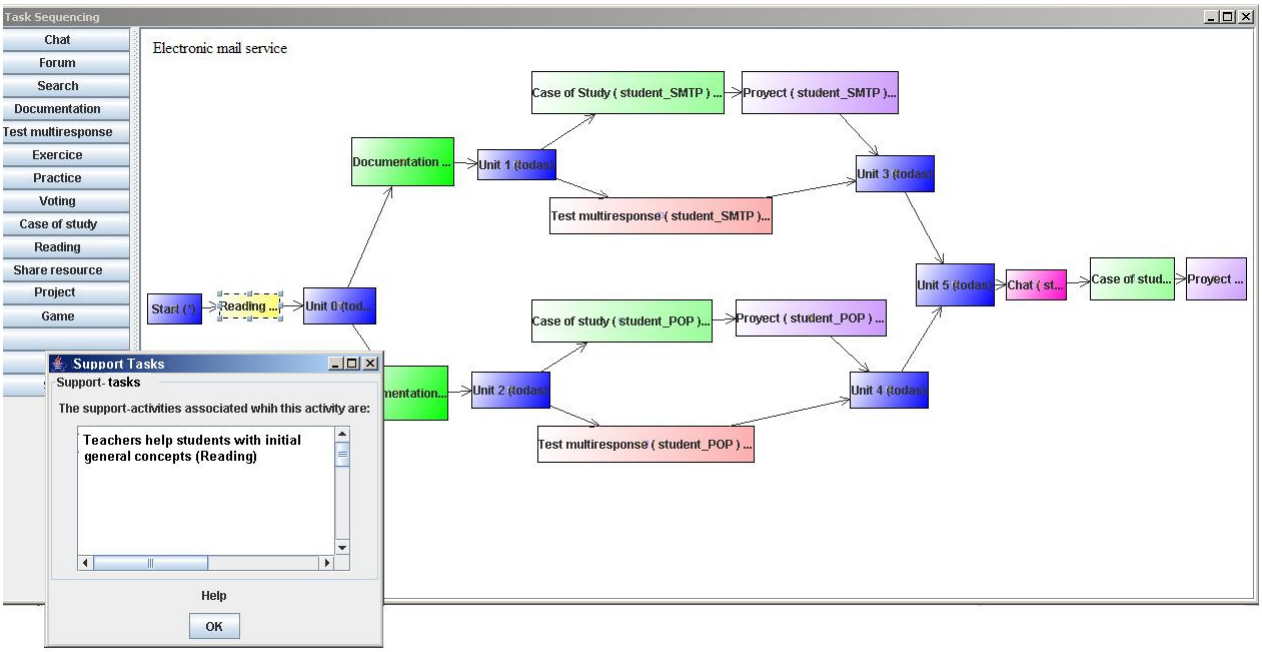

Figure 6: "Electronic mail service" didactic unit

4. Integration of the IMS-LD specification in CourseEditor

IMS-LD is oriented to the description of units of learning and the learning process among them, but not for the complete course description and modelling process. For this reason, all the information generated by CourseEditor cannot be translated into IMS-LD. On the other hand, although all the issues covered in IMS-LD can be considered as a part of course planning, CourseEditor tool can only represent a subset of IMS-LD. This design decision has been taken to ensure the usability of the tool for teachers with little technical skills. This section describes which aspects of course planning cannot be mapped into the IMS-LD specification and which aspects of IMS-LD are not covered in the CourseEditor tool.

Figure 7 shows an outline of the system architecture including the CourseEditor authoring tool.

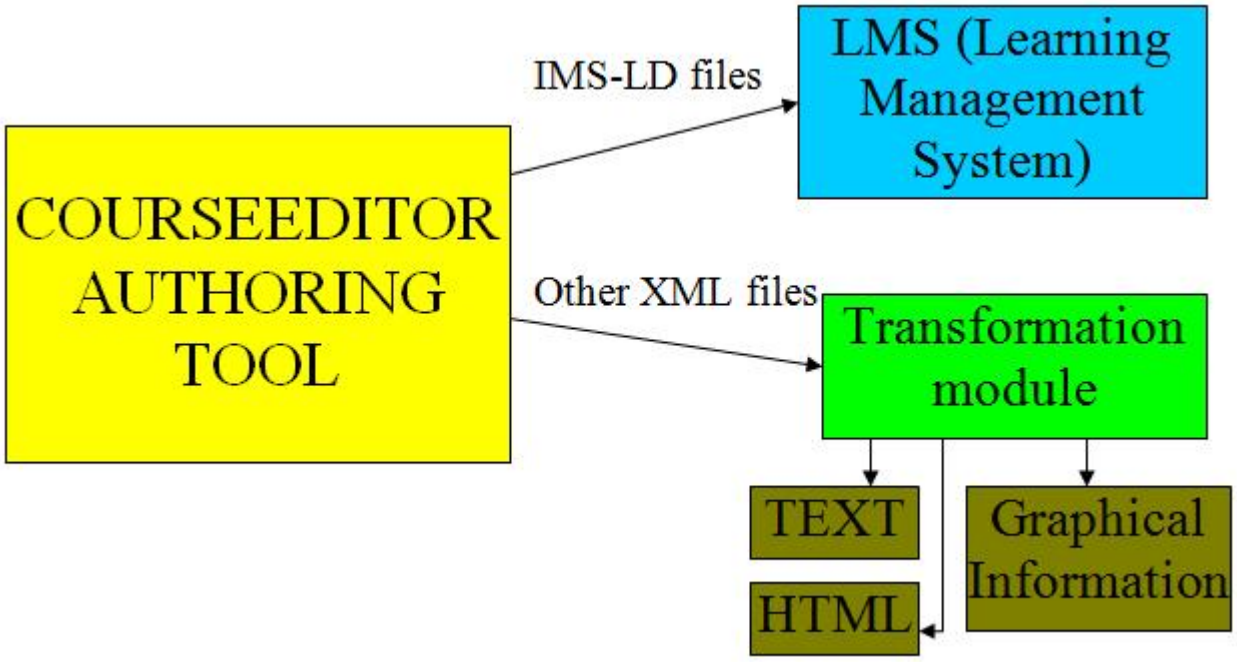

Figure 7: Outline of the System Architecture including CourseEditor

CourseEditor generates two kinds of XML files with different information:

1. IMS-LD files. This information can be interpreted by learning design players.
2. Extra information not convertible into IMS-LD. This information is part of course planning but it is not part of IMS-LD. This information has been stored using XML to facilitate future management. This information could be mapped to other educational standards (IMS-LOM, SCORM, etc.), being managed by the LMS or be presented to the user in different formats: HTML (to generate, for example, a course program to distribute between possible clients) or graphics (to observe the different relationships between tasks and objective classifications)

In order to obtain IMS-LD files, it is necessary a transformation between the information that the teacher introduces in CourseEditor GUI and the IMS-LD language. One of the challenges during the implementation was the decision about how to perform this transformation because sometimes, the same course planning information of the tool can be transformed into different XML IMS-LD instances and the

decision taken should be consistent with the prescriptions of the specification and reasonable for the teacher's intuition

4.1 Aspects of Course Planning that cannot be Mapped into IMS-LD

The following aspects of course planning, which are represented in CourseEditor, cannot be translated into the IMS-LD specification (so they are part of the other XML files generated by CourseEditor):

- General information of the course such as the title, the number of ECTS credits, etc.
- Objectives. IMS-LD allows defining learning objectives but only partially, because it does not support information about the type of objectives or their weight with respect to the global course.
- Hierarchy of contents. Course planning allows having certain level of content hierarchy, such as tasks are part of different didactic units. However, the IMS-LD specification does not allow representing this hierarchy of contents. Therefore, the different didactic units with their related information (description, duration, objectives etc.) cannot be represented using IMS-LD.
- Types of tasks. IMS-LD defines three types of services (send-mail, conference, and index-search), but it does not define a taxonomy of types of tasks. CourseEditor planning defines a set of standard types of tasks (chats, forums, etc.) and it allows defining new tasks with additional parameters. Therefore, IMS-LD cannot represent the different types of tasks with their related parameters. Examples of some other possible types of tasks or services with their defined parameters can be found in [32].
- Specific tasks. The tasks of the course planning modelling are instances of the types of tasks, and these tasks have more assigned parameters than the ones defined in IMS-LD. Therefore, this cannot be completely modelled in IMS-LD.
- Submission. IMS-LD does not support any information related to the submission of a task.
- Evaluation. Course planning takes care of the evaluation of the different objectives of a course for the students, and there are tasks that can be evaluated and different parameters are related to such evaluation definitions. IMS-LD does not support the representation of evaluation issues

4.2 Aspects of IMS-LD not Covered in CourseEditor

When a course is completely configured, CourseEditor can generate XML files that are compliant with the IMS-LD specification according to the defined tool parameters. The tool cannot generate all the IMS-LD specification possibilities, but only a subset of it. This is because the tool through its different windows allows only configuring some specific aspects that we considered more important in course planning and that are not difficult to set by teachers and course designers without technical knowledge. The inclusion of other aspects considered in IMS-LD (levels B and C) would increase the complexity of the tool and one of the key points of the design was the usability by teachers and course designers with low technical skills. Therefore, there is a trade-off between the set of functionalities available and usability. CourseEditor can create XML files compatible with a subset of IMS-LD level A specification. The only restrictions of the Course Editor tool to generate files compatible with IMS-LD level A are:

- CourseEditor allows introducing some metadata such as the title of the course, objectives, etc. but it is not possible to generate a complete set of IMS-LOM meta-data using the CourseEditor GUI (Graphical User Interface) because it might be very hard for a teacher to define all the possible parameters. This information can be used for example for searching purposes.
- CourseEditor allows defining roles but cannot have an unlimited hierarchy. There are a predefined student and teacher roles. In addition the tool let define whichever number of roles that inherit from student and teacher, but it does not allow a second level of inheritance. This model covers the most frequent course situations and introduction of hierarchy would increase complexity for teachers.
- The '*environment-ref*' tag is only used in the '*learning-activity*' tag, but not in the '*support-activity*' and '*activity-structure*".
- The '*unit-of-learning*' element is not possible within the '*activity-structure*'
- The search services ('*index-search*') are not used.
- The tool allows three possibilities of synchronization between roles: course, didactic unit and activity. Instead, IMS-LD allows whichever possibility of synchronization, but not only three types. The simplification is in order to make easier this configuration to a teacher, which can be complex.

- The elements '*complete-play*', '*complete-act*', and '*on-completion*' are not used.

5. Conclusions and Future Work

In this paper, a novel software tool for creating courses has been presented. The software tool is available at http://www.it.uc3m.es/pedmume/CourseEditor/ [38]. This tool is compliant with the three initial project objectives: 1) it includes the main aspects of course planning, building a new modelling of course planning taking into account different theories. 2) it is easy to use by teachers and course designers without high technological expertise. 3) it is compatible with the IMS-LD e-learning specification. In many cases, the achievement of usability for teachers was opposite to the achievement of the two other objectives, and in such cases there was a trade-off.

CourseEditor allows the teacher to model a complete course planning (general information, objectives, requirements, duration, didactic units, types of tasks, tasks, roles, etc.). It also allows synchronizing roles with activities, assigning roles to the different activities, choosing a maximum number of alternative activities in a specific moment or sequencing the different activities of every didactic unit graphically. A study case has been presented to illustrate how to use this tool by modeling a course on Telematic Services in the context of Telecommunication Engineering degree.

The CourseEditor tool stores all the information introduced by the teacher in XML files. Part of these files is transformed into IMS-LD. This IMS-LD files grant interoperability and reusability of this part of course planning among different learning systems and course designers. Moreover, CourseEditor creates proprietary XML files with the course planning information that is not covered in IMS-LD but that it is prepared to be reused in other contexts (to be mapped to other specifications or to be showed to the user).

As future work, the CourseEditor tool can be extended in the following ways:

1) Including **new aspects of course planning not covered in IMS-LD**, for example, new predefined types of tasks (wikis, social networks, etc.) or more levels in the hierarchy of the course. Currently CourseEditor only supports three levels (course, didactic unit and tasks), but it would be interesting to extend this model to see more cases (for example: course, chapters, topics, sections and subsections).
2) Including **new aspects of course planning covered in IMS-LD**. CourseEditor includes most of the functionality for IMS-LD level A but not for levels B and C.The inclusion of such aspects can be in conflict with usability for non-technical users. For example, level B of IMS-LD includes the definition of properties and conditions about such properties can be set, letting different personalization paths for every student. The inclusion of this functionality into the tool would complicate the graphical user interface by introducing new arrows, new elements, the definition of loops, etc. making CourseEditor very complex to use by teachers.
3) **Mapping XML files not compatible with IMS-LD with other specifications**. The evolution of educational specifications could allow in a near future to introduce some of the XML files generated by Course Editor not compliant with IMS-LD using other specifiation This effort would improve interoperability and reusability of course planning information generated by Course Editor.

6. Acknowledgments

This research has been partially funded by the following projects: project "Learn3: Towards Learning of the Third Kind" funded by the Spanish Ministry of Science and Innovation under grant No. TIN2008-05163/TSI and project E_MADRID: "Investigación y Desarrollo de Tecnologías para el e-Learning en la Comunidad de Madrid" funded by the Madrid Regional Government under grant No. S2009/TIC-1650.

References

[1] IEEE, *IEEE Learning Object Metadata Standard*, 2002, Available at: http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf

[2] IMS Global Learning Consortium, *Question & Test Interoperability v 2.1*, 2008, Available at: http://www.imsglobal.org/question/

[3] IMS Global Learning Consortium, *Content Packaging*, 2004, Available at: http://www.imsglobal.org/content/packaging/

[4] Advanced Distributed Learning, *SCORM*, 2004. Available at: http://www.adlnet.gov/scorm/index.cfm

[5] IMS Global Learning Consortium, *Learning Design Specification*, 2003. Available at: *http://www.imsglobal.org/learningdesign/index.html*

[6] E. Thorndike, *Education: A First Book*. New York, 1923.

[7] J.F. Bobbitt, *The Curriculum*. Boston, MA: Houghton, Mifflin, 1918.

[8] R. Tyler, *Basic Principle of Curriculum and Instruction*. University of Chicago Press, Chicago, 1949.

[9] B.F. Skinner, "The science of learning and the art of teaching," *Harvard Educational Review*, Vol. 24, N. 2, 1954, pp. 86-97.

[10] B.F. Skinner, "Teaching machines," *Science*, 128 (3330), 1958, pp. 969-977.

[11] R.M. Gagne, *The conditions of learning*. New York: Holt, Rinehart and Winston, 1965.

[12] R. Mager, *Preparing objectives for programmed instruction*. Fearon, 1962.

[13] G. Pask, *Artificial Organisms*. General Systems Yearbook 4, 151, 1959.

[14] H. Taba, *Curriculum Development: Theory and practic.* New York: Harcourt Brace and World, 1962.

[15] C.M. Reigeluth, *Instructional-Design Theories and Models. A New Paradigm of Instructional Theory*. Lawrence Erlbaum, 1999.

[16] Monterrey Institute. *Las Técnicas Didácticas en el Modelo Educativo del TEC de Monterrey*, 2000. Available at *www.sistema.itesm.mx/va/dide/documentos/inf-doc/tecnicas-modelo.PDF*

[17] The Pedagogical Patterns Project, 2002, available at www.pedagogicalpatterns.org

[18] G. Pask, "Styles and strategies of learning," *British Journal of Educational Psychology*, Vol. 42, 1976, pp. 128-148.

[19] R. Felder, *ILS: Index of Learning Styles*, 1999, available at www2.ncsu.edu:8010/unity/lockers/users/f/felder/public/ILSpage.html

[20] R. Koper, and C. Tattersall, *Learning Design - A Handbook on Modelling and Delivering Networked Education and Training*. Heidelberg: Springer, 2005.

[21] RELOAD project, *Learning Design Player*. Available at http://www.reload.ac.uk/new/ldplayer.htmlhttp://www.reload.ac.uk/new/ldplayer.html

[22] J.P. Escobedo, L. Fuente, S. Gutierrez, A. Pardo, and C. Delgado Kloos, "Implementation of a Learning Design Run-Time Environment for the .LRN Learning Management System," *Journal of Interactive Media in Education* Vol. 13, 2007.

[23] D. Griffiths, and J. Blat, "The Role of Teachers in Editing and Authoring Units of Learning Using IMS Learning Design," *Advanced Technology for Learning*, Vol. 2 N. 4, 2005.

[24] C.D. Milligan, P. Beauvoir, and P. Sharples, "The Reload Learning Design Tools," *Journal of Interactive Media in Education*, 2005.

[25] RELOAD project, *Learning Design Editor,* 2005. Available at http://www.reload.ac.uk/ldeditor.html

[26] Y. Miao, "CoSMoS: Facilitating learning designers to author units of learning using IMS LD," *13th International Conference on Computers in Education*, pp. 275-282, 2005.

[27] E. Sanchez, M. Lama, R.R. Amorim, A. Negrete, "WebLD: A web portal to design IMS LD units of learning," *7th IEEE International Conference on Advanced Learning Technologies*, Niigata, 2007, pp. 880-881.

[28] D. Hernández-Leo, E.D. Villasclaras-Fernández, J.I. Asensio-Pérez, Y. Dimitriadis, I.M. Jorrín-Abellán, I. Ruiz-Requies, and B. Rubia-Avi, "COLLAGE: A collaborative Learning Design editor based on patterns," *Educational Technology & Society*, Vol. 9 N. 1, 2006, pp. 58-71.

[29] 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*, Vol. 2 N. 3, 2005, pp. 223-232.

[30] S. Isotani, R. Mizoguchi, S. Isotani, O.M. Capeli, N. Isotani, and A. R. P. L. de Albuquerque, "An Authoring Tool to Support the Design and Use of Theory-Based Collaborative Learning Activities," *10th Conference on Intelligent Tutoring Systems*. Pittsburgh, 2010.

[31] I. de la Teja, K. Lundgren-Cayro, G. Paquette, "Transposing MISA Learning Scenarios into IMS Units of Learning," *Journal of Interactive Media in Education* Vol. 13, 2005.

[32] J.R. Dalziel, "Implementing Learning Design: The Learning Activity Management System (LAMS)," *20th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education*. Adelaide, 2003.

[33] *LAMS: Learning Activity Management System*, 2004, available at http://www.lamsinternational.com/

[34] D. Sampson, P. Karampiperis, P. Zervas P, "ASK-LDT: a Web-based Learning Scenarios Authoring Environment based on IMS Learning Design," *Advanced Learning Technologies Journal*, Vol. 2 N. 5, 2005.

[35] elive Learning Design, available at http://www.elive-ld.com/content/index_ger.html

[36] I. Martínez-Ortiz, P. Moreno-Ger, J.L. Sierra-Rodríguez, and B. Fernández-Manjón, "Supporting the Authoring and Operationalization of Educational Modelling Languages," Journal of Universal Computer Science, Vol. 13 N. 7, 2007, pp. 938-947.

[37] R. Koper, R., "Current Research in Learning Design," *Educational Technology & Society*, Vol. 9 N. 1, 2006, pp. 13-22.

[38] CourseEditor authoring tool, available at http://www.it.uc3m.es/pedmume/CourseEditor/

[39] P.J. Muñoz-Merino, C. Delgado Kloos, J. Fernandez Naranjo, "Enabling Interoperability for LMS Educational Services," Computer Standards & Interfaces Vol. 31 N. 2, pp. 484-498, 2009.

Figure legends

Figure 1: Course Objectives screen

Figure 2: Screens for defining a new type of task

Figure 3: Screen for task configuration

Figure 4: Screen for activities in which submission is required

Figure 5: "File Transfer service" didactic unit

Figure 6: "Electronic mail service" didactic unit

Figure 7: Outline of the System Architecture including CourseEditor