Towards an Automatic Monitoring for Higher Education Learning Design

David Camacho
Dpto. de Informática. Universidad Autónoma de Madrid.
Ctra. de Colmenar Viejo, Km 15.
28049 Cantoblanco (Madrid), Spain.
email: david.camacho@uam.es

M. D. R-Moreno
Dpto. de Automática. Universidad de Alcalá.
Ctra. Madrid-Barcelona, Km. 33,6.
28805 Alcalá de Henares (Madrid), Spain.
email: mdolores@aut.uah.es

Abstract:
The development of new Information Technologies have originated new possibilities to develop pedagogical methodologies that provide the necessary knowledge and skills in the Higher Education.

These technologies are built around the use of Internet and other new technologies, such as the Virtual education, Distance Learning or Longlife Learning. This paper presents a metadata-based model representation that is used to represent, detect, and even automatically correct possible pitfalls in the schedule process of a learning design in e-Learning environments. This metadata-based model can be combined with other Artificial Intelligence (AI) techniques, such as, automated AI planning/scheduling to monitor how is evolving a particular Learning Design (LD), and to propose solutions in those modules of the design that learning problems among the students have been found.

Keywords: e-Learning, Learning Design, Higher Education, Metadata, Planning&Scheduling.

Reference to this paper should be made as follows: Camacho, D. and R-Moreno, M.D. (2006) 'Towards an Automatic Monitoring for Higher Education Learning Design', Int. J. Metadata, Semantics and Ontologies, Vol. 1, No. 2

Biographical notes: Dr. David Camacho received a PhD in Computer Science from Universidad Carlos III de Madrid in 2001. He has researched in several areas, including planning, inductive logic programming, and multiagent systems. Currently his research interests are related with multiagent systems, knowledge representation, e-Learning and Semantic Web Services.
Dr. Maria Dolores R-Moreno received a PhD in Computer Sciences in 2004 by the Universidad de Alcalá. Her research interests are in the area of Planning, Scheduling, Monitoring and Execution applied to real domains as satellites, workflow or e-Learning.

1 INTRODUCTION

In earlier sixties, [Baran (1964)] the engineers and researchers involved in the development of the first computer networks was thinking about the possibility of building a resources sharing domain for those people who needed to solve communication problems where the geographic situation was a constraint. These people possibly could not imagine the deep impact of their ideas in our actual information society. The evolution of these computer networks, specially once the Internet has been fully deployed, represents the most successful example, in the brief history of computer sciences, about the benefits that can be obtained when sustained investment and commitment to research and development in information infrastructure is applied in the society. The use of new information technologies in Internet, has changed traditional (human) concepts such as economy, business, commerce, or education. Moreover, from the evolution of information technologies and computer networks, new research areas like e-business [Jennings (2000)], e-commerce [Lee and Yang (2001); Mobasher et al. (2000)]) or e-learning [Klamma et al. (2003); Rowlands (2003); Klamma et al. (2003)], have been created and successfully deployed in this new environment.

The e-Learning [Kozma (1991)[research field has become a *hot* topic in recent years. On one hand, many educators have seen it as a way to give flexibility and re-use previous courses stored in a database, or in other electronic formats [Schmitz et al. (2002)]. On the other hand, the increasing computing power and the actual network infrastructure allows to share and distribute these courses between public institutions and private corporations. E-Learning techniques are changing the traditional image of having a classroom, desktops with students and a blackboard. These new educational approaches are evolving to use the new information technologies, and the Internet, like a virtual platform where all the involved people can implement new ways of communication. In this new educational environment, the interaction between students and educators can be modified in several ways:

- Virtual education can endow the *physical* courses of an atemporal mass media, not limited to a specific timetable and physical space (sometimes very constrained).
- The geographic problems (if they exist) can be smoothed or simply eliminated.
- It promotes a direct communication at any time or in any place between educators and students through the utilization of electronic mail or other technologies (i.e. IRC channels).
- It is possible to publish/adapt/update/change relevant documentation to make it accessible to anyone.
- Using the new Web-based applications and several related techniques, like machine learning or Artificial Intelligence planning & scheduling, the educators can manage the learning process for a particular group or student.

Any professional with experience in education knows that Internet, or any other kind of software (SW) application, cannot replace (completely) lecturers. The interaction in the classrooms between educators and students is necessary not only to provide some explanations about a particular topic or problem but this interaction allows to provide his/her own professional experiences to their students. This educator/students interaction constitutes an important part of the students' training.

But the integration of e-Learning and Information Technologies (IT) will be in the closed future a milestone in the universities that aim to go on, and will provide a way of assuring and evaluating the quality of education. The tendency will evolve from being experimental initiatives based on projects, to become an important element in the day-to-day university activities. We also believe and pursue as in [European Comission (2004)] this idea.

This paper presents a description of a new metadata-based model representation that can be used to implement Learning Designs for the Higher Education. This model is used to detect and solve problems that could appear in the learning process. Our proposed metadata model could be used in e-Learning environments and educational platforms such as Aula Global (http:// www.uc3m.es), First Class (http:// www.softarc.com), BlackBoard (http:// www.blackboard.net/), WebCT (http:// www.webct.com/), LMS (http://www.lotus.com/lotus/offering6.nsf/wdocs/homepage) or E-ducativa (http:// www.e-ducativa.com/). The main features of previous platforms can be summarized as follows: Internet is used as a repository were students and educators can store and retrieve documentation; the communication between students and educators is achieved using electronic mail, and/or IRC-channels; and finally, the student participation in the platform activities is used by educators to evaluate them.

The proposed metada model for a LD representation develops new valuable possibilities that improve the quality of the e-learning process. These can be described as follows:

- It is possible to implement an *"annotated"* course program which incorporates new semantic information related with the contents of the course.
- The metadata-based model can be used by a planning module to automatically schedule the different educators and students activities taking into account time and resources constraints.
- Finally, using both, the Web (or the selected e-learning platform) as tests/exams repository (designed by educators), and a statistical module to automatically generate the results obtained by the students. It will be possible (using the metadata defined previously) to automatically modify the LD to improve its quality.

The paper is structured as follows. Section 2 describes how, through a cyclic Longlife Learning process, is possible to control and monitor the quality of the LD in Higher Education environments. Section 3 shows the proposed metadata model for a LD Representation that can be used in Virtual learning domains and educational platforms. Section 4 describes how several AI techniques, like Planning and Scheduling, can be used to help in the process of monitoring and quality assurance in the LD. Section 5 shows an example of a LD and how the planning/scheduling techniques can be used in this domain. Finally, Section 6 summarizes the conclusions of this work.

2 A CYCLIC LONGLIFE PROCESS

Our main goal is to define several processes that could be able to monitor the evolution of a specific course LD. The evolution of this program could be controlled by the educators involved in the deployment of the course. To ensure the quality and to monitor the LD [Koper (2004)], it is necessary to use a Longlife process to understand correctly how the course program is evolving. In this process, the interaction with the students play an important role. Figure 1 shows the five general processes of our approach. These processes can be summarized as follows:

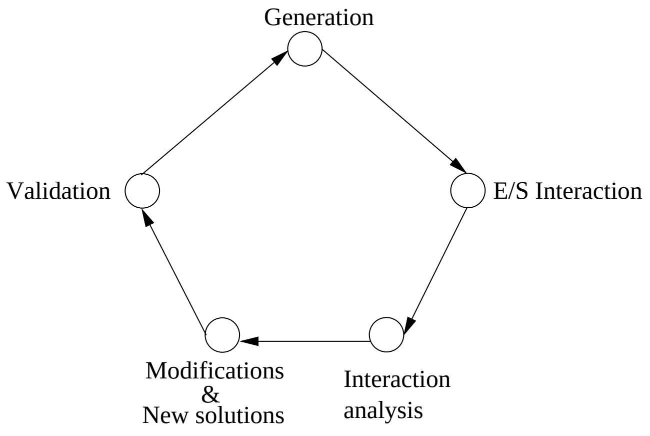

Figure 1: Longlife processes of a particular LD.

1. **Generation**. This process is related to the *definition* and *deployment* of the LD.

2. **E/S Interaction**. This is a set of subprocesses that allow the interaction among Educators (E) and Students (S).

3. **Interaction analysis**. From previous E/S interaction, the results obtained need to be analysed.

4. **Modifications & New solutions**. Using the observed behaviours of the learning objects (using previous analysis) this process generates new LDs that try to solve the detected problems.

5. **Validation**. Finally, the proposed solutions will be validated and approved by one or several educators.

Figure 2 shows how this cyclic process can be decomposed into more detailed subprocesses. These subprocesses are:

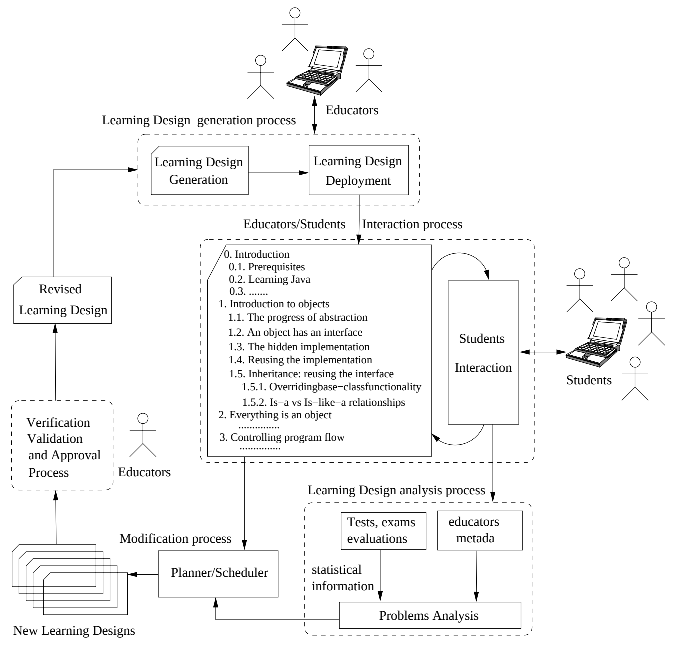
Educators
Learning Design generation process
Learning Design Generation
Learning Design Deployment
Educators/Students Interaction process
0. Introduction
0.1. Prerequisites
0.2. Learning Java
0.3. ....
1. Introduction to objects
1.1. The progress of abstraction
1.2. An object has an interface
1.3. The hidden implementation
1.4. Reusing the implementation
1.5. Inheritance: reusing the interface
1.5.1. Overridingbase−classfunctionality
1.5.2. Is−a vs Is−like−a relationships
2. Everything is an object
3. Controlling program flow
Students Interaction
Students
Revised Learning Design
Verification Validation and Approval Process
Educators
Learning Design analysis process
Tests, exams evaluations
educators metada
statistical information
Problems Analysis
Modification process
Planner/Scheduler
New Learning Designs

Figure 2: Detailed Longlife cyclic processes.

- *Learning Design Generation.* Initially the educator must define the units of learning. However, to allow automatic reasoning about the evolution and results of this program, it is necessary to add other *metadata* information. Therefore, several new features for units of learning will be added to the general information of the learning design. Section shows how to define a *metadata-based* LD.

- *Learning Design Deployment.* Once the program has been defined by the educators, this will be deployed through the use of a particular e-learning platform. This platform should be able to provide at least two main features. On the one hand, to provide the course documentation and to allow the necessary interaction between studens and educators. On the other hand, to provide some evaluation tools that allow the educators obtaining quantitative data about the current state of the knowledge acquired by the students. Previous features are frequently implemented in most of the e-Learning platforms.

- *Educators/Students Interaction.* For this paper, the more important interaction process between the course design and the students are implemented through the implementation of several specific tests (designed by educators) that will be completed by the students.

- *Learning Design analysis process.* This process is carried out using the statistical results obtained from previous process. This process allows obtaining quantitative measures about how the knowledge (learned in several units) has being acquired by the students.

- *Planner/Scheduler.* Dynamically the contents of the course can be modified using the current evaluations obtained from the student interaction, and from the metadata included in the course program. This process will be carried out by an intelligent module whose reasoning process is achieved using automated planning/scheduling techniques. The automatic LD process is analysed in Section 5.

- *Validation Process.* Finally, the proposed changes by the intelligent module will be analysed and approved by the team of involved educators. This process will be divided into three subprocesses: *Verification*, if the proposed changes in the LD are pedagogical coherent, *Validation* if the real deployment of these modifications is possible, and then the *Approval* of the selected modifications. Finally, the modifications approved will be provided to the e-Learning platform as the new course design for the next (academic) year.

Previous processes will be repeated until the quality parameters defined by the educators have been achieved (i.e. number of students that are able to finish the course, students dedication time to the course, etc...). These processes need to be carried out during several (academic) years, so the use of several AI techniques, such as planning and scheduling, allow to modify dynamically these designs and provide *advice* about what elements in a LD could (or should) be modified.

3 A METADATA-BASED MODEL REPRESENTATION

This section describes how to define and implement a metadata-based model of a LD. This new extended representation of a classical course program will be used by an intelligent module (planner/scheduler) to reason about the detected problems (if exists) and will be able to propose new solutions to solve them. To define and monitor a LD, it is necessary to add the following information:

- *Learning Design structure.* This information represents all the information related with the contents of the course [Sicilia (2005)]. Usually, any course can be divided in several *topics* that can be subdivided into other *subtopics*. This topics represent the learning contents to be given to the students. We consider that any course can be structured into chapters, sections and subsections (the bibliography refers to these elements as *Unit of Learning*, or *Unit of Study*).
- *Annotations.* They are comments in natural language provided by educators to describe any important characteristic of a particular *Unit of Learning*.
- *Dependencies.* This information is used to define logical relationships between the different *Unit of Learning*. These dependencies are defined by the educators and represent the logical order between different units of learning in the course. There exist two possible kind of dependencies:
  - *Strong dependencies.* These kind of dependencies represent both, an order relation and several semantic relationships between several units in the LD. When these dependencies are used means that these units belong to a *super-knowledge item*, i.e. if a strong dependency is defined between two units A (*"Introduction to objects"*) and B (*"Everything is an object"*) means that it is necessary to give both units (in the order $A \rightarrow B$) to understand correctly the concept of *"object"*.
  - *Weak dependencies.* A weak dependency only represents an order relation between two units of the learning design. When one of these dependencies are defined between A and B units, means that it is necessary to give the unit A before to access unit B. However, it should be possible to avoid B and jump to other units. For instance, if we have a unit A (*"Controlling program flow"*) where the syntax, and other basic concepts, of a programming languague are explained, we could access later to a unit B (*"Initialization and cleanup"*) where is explained how is cleaned or initializated an object, or to a unit C (*"The Java IO system"*) that describes the Input/Output mechanisms implemented in this languague.
- *Priority.* Any unit in the LD will have asigned a priority between a maximum (represents any essential unit of learning for the students) and a minimum value (represents a unit of learning that can be relaxed from the course). These values will be assigned by educators. Currently, we have defined a numerical value from 10 (maximum priority) to 0 (minimum priority).
- *Time duration.* This is a tuple: (min, med, max), that represents the minimum, medium and maximum time required to acquire the unit of learning from a module of the LD. Each unit of learning is divided into time units (we consider as a time unit = one hour), this tuple represents the effort from the educators to give the different units.
- *Roles.* These metadata allow to define the roles that the people on the LD can play. There are two kinds of roles used to represent people: learner or staff (educators), Although individuals are not generalisable components, roles are. Each role defines the potential competencies that exist in the design and they are defined independently of the persons to whom the roles will be assigned.
- *Resources.* There are several types of resources that can be considered: web content, imsld content, person, services (i.e. chats or discussions forums) and available tools. In the LD, we need to have in mind the resources available to achieve the learning objectives.

- *Pedagogical complexity.* This semantic parameter is defined by educators to represent the knowledge difficulty of a particular unit of the course. Currently it is possible to use the following values: (trivial, very low, low, medium, high, very high, extreme).

Using the previous data (specific information about the course) and metadata (information about other features of the course), is possible to implement a new metadata-based model representation of a LD. Figure 3 shows an schematic representation of this metadata-based model.

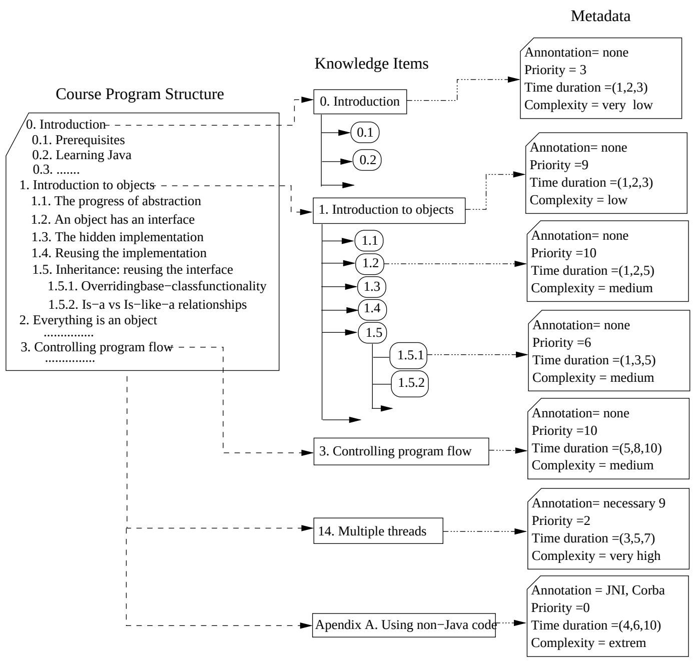

Figure 3: A metadata-based model for a course program.

As Figure 3 shows, the metadata can be defined for each unit of learning (chapter, module, section) o sub-units. For example, in the first unit (introduction) the metadata has only being declared once, therefore all their sub-units inherit this metadata. However, in the second unit (Introduction to objects), the general unit has its own metadata, but it has also being defined some information in other sub-units, such as 1.2 and 1.5.1. This metadata definition level will affect directly to other important processes like planning and scheduling because some of this information (time duration and priority) will be used in the process. Therefore, only with those knowledge items that have defined its metadata will be suitable to be used, and modified, by the intelligent module.

Figure 4 shows a possible Java LD that has been implemented using a well known Java Book used by engineering students in different Universities, the Bruce Eckel (2002) book. From this simple index it is possible to define a new semantic for a Java Learning Design by adding the information described previously.

Figure 5 shows the definition of the (learning concepts) *dependencies* between the different units of learning. These dependencies perform a learning graph that describes how

```
Foreword
0:Introduction
      0.1. Prerequisites
      0.2. Learning Java
      .....
1: Introduction to objects
      1.1. The progress of abstraction
      1.2. An object has an interface
      1.3. The hidden implementation
      1.4. Reusing the implementation
      1.5. Inheritance: reusing the interface
            1.5.1. Overriding base-class functionality
            1.5.2 Is-a vs. is-like-a relationships
      .....
2: Everything is an object
3: Controlling program flow
4: Initialization and cleanup
5: Hiding the implementation
6: Reusing classes
7: Polymorphism
8: Holding your objects
9: Error handling with exceptions
10: The Java IO system
11: Run-time type identification
12: Passing and returning objects
13: Creating windows and applets
14: Multiple threads
15: Network programming
......
A: Using non-Java code
......
```

Figure 4: A LD from the Bruce Eckel's Java book.

the course can be developed. If the *time duration* data is added to this graph, a new graph with the related time restrictions could be generated.

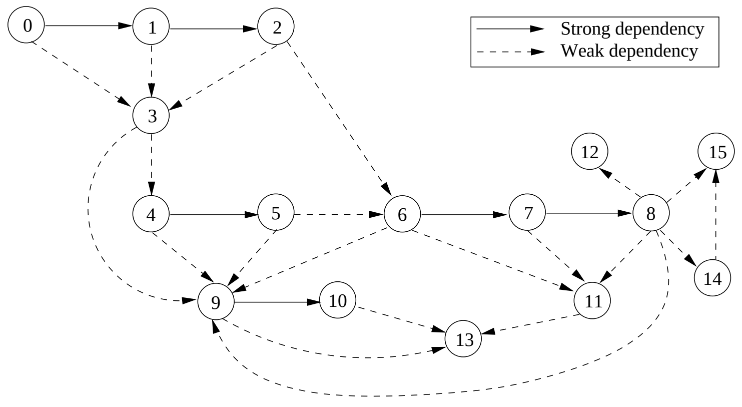

Figure 5: Graph dependencies representation for a Java LD

The strong dependencies showed in previous figures mean that the educators have defined four blocks of units of learning. Therefore, when any of their units is given to the students it should be mandatory to finish with the other related units to achieve the desired learning goals. Table 1 shows the four learning blocks, the contents extracted from the LD (see Figure 4), and the semantic meaning for each block.

With the metadata-based representation showed in this section, and using the previous Java index [1] to perform the *Learning Design structure*, the educators could be able

[1]The complete table of contents is available at http://www.codeguru.com/java/tij/tij_c.shtml

Table 1: Knowledge blocks defined using strong dependencies.

Block	Item	Description	Learning contents
1	0	Introduction	Basic definition concepts
	1	Introduction to objects	in Object Oriented
	2	Everything is an object	Programming (OOP)
2	4	Initialization	Management and
		and cleanup	initialization
	5	Hiding the	of objects, interface
		implementation	definition
3	6	Reusing classes	Use of basic concepts
	7	Polymorphism	in OOP such us inter-
	8	Holding your objects	faces or inheritance
4	9	Error handling	Error treatment
		with exceptions	in Java
	10	The Java IO system	and IO mechanisms

to define and implement a semantic design that can be used later for other kind of systems to reason about the behaviour of the proposed course (through the interaction with the students).

4 AI TECHNIQUES FOR AUTOMATIC LD

In the last few years an increasing development of workflow tools for e-learning (activity management within learning) have emerged. Usually, the task of defining the LD for a course is performed with the aid of a set of tools that allow the graphical representation, together with the relations among the activities that occur within the processes. This task is just performed by drag and drop activities into the workspace of a Learning Activity Management System (LAMS) and the use of connecting arrows to organise the activities into a sequential workflow. Learning activities may be sequenced or otherwise structured carefully and deliberately in a learning workflow to promote more effective learning.

Workflow systems for companies hold the promise of facilitating the everyday operation of many companies and work environments. Despite the popularity of these products, there is still a lack of maturity in some respect, i.e. a lack of a semantic associated to the models or an easy way to reason about that semantic. We identify the same problems in these learning tools and those problems as shown in R-Moreno and Kearney (2002), could be overcome using techniques coming from other fields such as Artificial Intelligence (AI).

The AI community and in particular the planning and scheduling field, has been applying successful techniques in different and complex domains like robotics [Estlin et al. (1999)], satellites [R-Moreno et al. (2004)], workflow [R-Moreno and Kearney (2002)] or military logistics [Tate and Whiter (1984)]. In these domains, there are activities that must be performed in a temporal horizon that consume or produce resources. During execution, completion of activities, and delays and other problems are detected to take the appropriate measures (rectify the situation, or in more drastic cases, a new plan) to satisfy the goals. In order to represent this information, rich representation models are needed, the majority of them based on predicate logic as is the case of the planning standard language, PDDL2.2 [Edelkamp and Hoffmann (2004)].

AI P& S consists of a set of techniques that enable efficient searching for solutions of problems with time and resources. Traditionally, there is a clear subdivision of techniques and roles that belong to planning and scheduling. Planning generates a plan (sequence or parallelization of activities) such that it achieves a set of goals given an initial state and satisfying a set of domain constraints represented in operators schemas. In scheduling systems, activities are organised along the time line having in mind the resources available. These systems can perfectly handle temporal reasoning and resource consumption, together with some quality criteria (usually focused around time or resource consumption) but they cannot produce the needed activities and their precedence relations given that they lack an expressive language to represent the activities. Traditionally, planning was first performed and the solution was given as an input to the scheduling systems. So in these systems, the user should first supply a domain description that is composed of a set of operators that allow the planner to go from a defined initial state to a state in which a set of goals is fulfilled in a given deadline. Nowadays it is being an increasing interest to integrate these two fields because of real domains needs. From this perspective, by combining scheduling and planning systems synergistically the weaknesses of both areas can be solved. Then, systems as IPSS [R-Moreno et al. (2004)], SAPA [Do and Kambhampati (2003)], IxTeT [Ghallab and Laruelle (1994)] or JSHOP2 [Ilghami and Nau (2003)] are suitable candidate to solve this type of problems. The last planner belongs to what is known as HTN planners. The rest to STRIPS-style planners [Fikes and Nilsson (1971)].

An instance of a LD in this new Higher Education Area is analogous to a plan in AI. A Higher Education course includes allocation of resources (learning objects store in different web sites, teachers from several universities collaborating in the same master course, etc) and target start and end times, while in AI terminology this task is usually performed by scheduling techniques.

Using a high level description, the inputs of a planner are:

- Domain theory: the STRIPS representation originally proposed by Fikes and Nilsson is one of the most widely used alternatives [Fikes and Nilsson (1971)]. It was introduced to overcome what were seen as computational difficulties in using states to construct plans. In the STRIPS representation, a world state is represented by a set of logical formulae, the conjunction of which is intended to describe the given state. Actions are represented by so-called operators. An operator consists of pre-conditions (conditions that must be true to allow the action execution), and post-conditions or effects (usually constituted of an add list and a delete list). The add list specifies the set of formulae that are true in the resulting state while the delete list specifies the set of formulae that are no longer true and must be deleted from the description of the state. Each course can be defined in terms of a set of learning activities that are performed by students. Therefore, there is a strong relation between operators in planning and learning activities to perform in order a student to successfully complete a course in learning environments.

In HTN planners, actions are usually called tasks, and correspond to state transitions. A task network is a collection of tasks that need to be carried out, together with constraints on the order in which tasks can be performed.

The primary difference between HTN planners and STRIPS-style planners is in what they plan for, and how they plan for it. In the second one, the objective is to find an ordered set of actions that will bring the world to a state that satisfies certain conditions or attainments goals. Planning proceeds by finding operators that have the desired effects and by making the preconditions of those operators into subgoals. In contrast the first one searches for plans that accomplish task networks, which can include things other than just attainment goals. But methods must contain all possible ways tasks can be achieved, so this is usually a tedious tasks.

- Problem: is described in terms of an initial state and goals. Those states are represented by a logical formula that specifies a situation for which one is looking for a solution. When a student starts a course, s/he has a previous knowledge and the university that offers the course knows the resources available. As a result, the student will learn at least the minimum concepts required in the course for a given time.

- Initial state: in planning, one has to specify the starting situation of the posed problem. Examples of initial states would be: the previous knowledge of the students, the resources that the course uses and when they are available, the maximum number of student for each teacher, the priority of each module, etc.

- Goals: are often viewed as specifications for a plan. This concept is equivalent to the *learning goals* in learning environments. They describe the successful behaviours that execution of the plan should produce: specify what one would like to be true at the end of the solution of the problem for a given time. In our case, that the student is able to apply critical thinking about a specific subject.

For scheduling systems, many techniques used in this area come from the Operational Research (OR) area (i.e., branch and bound, simulated annealing, lagrangian relaxation). Lately, Constraint Programming (CP) has been applied to the different scheduling problems with very good results. Two common problems in scheduling are the Job-Shop Scheduling and the RCPSP$_{max}$ problem [Kolisch and Hartmann (1999)].

The Job-Shop Scheduling problem consists of a finite set of n jobs, each job consists of a chain of operations; and a finite set of m machines, each machine can handle at most one operation at a time. Each operation needs to be processed during an uninterrupted period of a given length on a given machine. The purpose is to find an allocation of the operations to time intervals to machines, that has minimal length.

RCPSP$_{max}$ consists of a set of activities where two kinds of constraints can be interrelated:

- Precedence constraints that force that an activity cannot start before its predecessor activities.

- Resource constraints among activities that consume the same resource due to the limited capacity.

The objective is to find precedence and resource assignments for all the activities in the horizon imposed.

These two problems that the Schedule area perfectly solves, can be seen as a generalization of the LD Scheduling Problem. In this case, instead of having machines and jobs, we have students and teachers, and units of learning in courses.

Each unit (operation) needs to be processed during a period of time for a given student (machines), and the unit will be supervised by a learner. The course will also have a limited duration (deadline). Each learner will also have a maximum number of students (we consider a learner as a resource with a total resource capacity given by the number of students). We need to know the initial and end time of each unit of learning considering precedences constraints among them. The variable values are imposed by the problem conditions: learning activity durations, course duration, number of learners, etc.

As a result of AI integrated planner & scheduler systems, they generate as an output a plan or set of plans if a solution exists for the given deadline. A plan can be seen as a sequence of operator applications (learning activities) with a specific duration that can lead from the initial state to a state in which the goals are reached with the resources available.

5 AN AUTOMATIC LD: A CASE STUDY

This section shows how the Java LD example defined using the metadata-based representation in Section , and the information generated from the Educators/Students Interaction, can be applied into a particular Planner/Scheduler to generate new LDs courses that solve the detected problems.

Let us suppose that two students (Mary and Tony) with different programming skills, have signed on a Java programming course in a particular European University. After the registration, those students have an entrance test to evaluate their knowledge and their psychological model. This test that can be performed through the Web, will allow us to define and know the student profile. Actually, when a student starts a course, the student previous knowledge is uncertain and the teacher does not know what can be the main difficulties that he/she has to face with. Also, if this were feasible, in most of the European Universities, there are too many students to perform a personalised monitoring.

Thanks to the new Information Technologies and well made tests, this information can be known almost immediately and it will be automatically translated into the initial state of a planning problem. In our example, Mary has basic programming knowledge and she is a quite active student, and Tony has more programming knowledge but he has a passive learning aptitude. Of course, we are considering a small problem subset of the obtained results.

At the beginning both students will start the course with the first unit of learning: "Introduction". The scheduler will assign the same duration (2 hours, that is, the minimum time duration) to both of them due to the low priority and complexity values. Until know, there are not many options for the scheduler to plan for different solutions.

After one or several units, let us suppose that an exam is planned. The students are now in the "Controlling program flow" unit of learning, and thanks to the tests, we have a personalised knowledge of the weak points of the already learnt subunits. From the results we can know that the in the "Inheritance: reusing the interface" subunit both students had bad results so a failure in the LD has been detected. This information is saved for the future LD revisions. In this situation, the pedagogical responsible can decide to add more examples and learning objects to this subunit, what implies the increase in the minimum, medium and maximum duration time. This increase of time in one of the modules will produce a reduction in other modules in order to keep consistency with the global course duration (deadline). That decision will be made automatically by the scheduler, but it is the responsibility of the pedagogue to check the consistency from the pedagogical point of view.

It would be recommended for our two students to revise again those concepts learnt on the sub-unit 1.5 for the unit of learning that they are going to start. In this situation, the scheduler helps again the tutor in re-planning the rest of the course, adding new learning objects that the students can consult, increasing the number of hours in this unit of learning and adjusting the rest of units.

In this process there are several learning activities to perform that are common to all the LD (for pedagogical reasons we are just considering three general activities): LD_General_Descomposition, LD_Selection or LD_Composition. These learning activities can be performed by any of the resources implied in the LD, having in mind the availability and the roles that they can play.

Figure 6 shows a Selection learning activity using the syntax of the integrated planner and scheduler system IPSS [R-Moreno et al. (2004)]. For a HTN approach instead of a STRIP-based one, refer to [Ullrich (2000)].

```
(OPERATOR LD_Selection
 (params <lo> <st>)
  (preconds
   ((<lo> LEARNING_OBJECT )
    (<st> STUDENT)
    (<t0> TEACHER)
    (<c0> CONCEPT)
    (<c1> (and CONCEPT (diff <c0> <c1>)))
    (<il> (and INTERACTIVITY_LEVEL
             (gen-from-pred (interactivity-level
                                      <lo> <il>))
                                      (> <il> 3))))
    (<p0> (and PRIORITY (gen-from-pred
                             (priority <c1> <p0>))
                                (>= <p0> 6))))
   (and (part-of <c0> <c1>)
        (knows <st> <c0>)
        (adquired-concept <lo> <c1>)))
  (effects
   ()
   ((add (knows <st> <c1>)))
  (resources
   ((<req-from-teach> (and CAPACITY
                        (resource-from-pred
                         (availability
                            <t0> <req-from-teach>)))))
  (constraints
   ((<duration> (and DURATION
                    (constraint-from-pred
                      (duration  <c1> <duration>)))))
   ((DURATION <duration>)))
  (costs
   ((<r0> ROLE (cost-from-pred
                    (Expert_Level <t0> <r0>))))
   ((EXPERT_LEVEL <role>))))
```

Figure 6: IPSS operator corresponding to a LD_Selection activity.

The precondition of the LD_Selection activity checks if there is a concept (<c0>) that is part of another concept (<c1>). The student knows the first concept and the concept we want the student to acquire is available in a learning object with a level of interactivity greater than 3. The concept to acquire must have a priority equal or higher than 6. As a post-condition, the execution of the activity will cause that the student acquires the concept. The field `param` visualises the values of the variables associated with the activity when IPSS (or any other planner) instantiates them in the plan.

The variable <c0> is used to represent the element instantiated by IPSS among the possible concepts that the problem might have (i.e. oo-programming).

The *resource-from-pred* function binds the capacity of the resource required to perform the learning activity. The function checks that the value is not higher than the global capacity of the resource. In the example of Figure 7, Dr. Russel has assigned a student monitoring of 30, surpassed that capacity, the system has to find another learner available to supervise the student.

The *cost-from-pred* function generates a list of values using the information of the current state to compute the effective cost of performing the activity under a defined metric. IPSS allows defining more than one metric and when looking for solution it tries to find a solution which minimise/maximise a define metric. In our case, we want the activities to be performed by high qualified experts. Then, this variable is specified by the role that the learner belongs to (i.e. senior_teacher, junior_teacher, etc) and IPSS will try to assign to the activities, teachers that belongs to the higher levels of the hierarchy.

The duration value is specified in the *constraints* field by the *constraint-from-pred* function. This function allows specifying the minimum, medium and maximum value of the duration for learning a concept and let the schedule module of IPSS reason about it. Also, through this function we can specify temporal windows in the operator start time. For example, that one activity must start 30 hours after the beginning of the course.

With respect to the problem definition, Figure 7 shows the initial conditions, some of them generated automatically from the test solved by the students, and other, from the staff structure involved in the problem. This includes issues such as which concept is part of another `(part-of basic-programming oo-programming)`, a concept that a student knows `(knows Mary basic-programming)`, the priority of a concept `(priority Introduction 3)`, etc

The Figure also shows the goals that IPSS needs to accomplish. In this case, two only goals: that Mary and Tony acquire the knowledge in OOP using Java.

With these conditions IPSS generates a plan (or learning design) with resources, roles and times assigned to each learning activity.

```
(part-of basic-programming procedural-programming)
(part-of basic-programming oo-programming)
(requires-of oo-programming procedural-programming)
(duration Introduction (1 2 3))
(complexity Introduction very_low)
(priority Introduction 3)
(annontation Introduction none)
(availability Dr. Russel 30)
(adquired-concept learning_object1
                  procedural-programming)
(interactivity-level learning_object1 3)
(knows Mary basic-programming)
(learning-style Mary active)
(knows Tony procedural-programming)
(learning-style Tony passive)
(is-lenguage Java oo-programming)
  ...
(goal (and (knows Mary Java) (knows Tony Java) ))))
```

Figure 7: Problem representation of the model of Figure 3.

6 CONCLUSIONS AND FUTURE WORK

This paper has presented two main contributions. On the one hand, a metadata-based model to represent any course program in Higher (or other) Education has been defined. This model has been applied in a particular LD example to show how the metadata can be used to add semantic content. On the other hand, this paper has presented an Artificial Intelligence approach, that allows through the use of automatic planning and scheduling, reasoning about the problems detected in the evolution of a LD, and how this design can be automatically modified to propose new solutions. This technique can be integrated in any Learning Activity Management System (LAMS) for the automatic learning design generation.

In the next future the authors wish to deploy both, the metadata-based representation and the intelligent Planner/Scheduler module into an e-Learning platform to allow the real interaction with the students. Currently, all the experimental evaluation of this technique has been carried out through a simulation process, where the possible "students problems" were simulated.

ACKNOWLEDGMENT

This work was funded by the Universidad de Alcalá project UAH PI2005/084 and the CICYT project TSI2006-12085.

REFERENCES

Baran, P. (1964) 'On Distributed Communications Network', *IEEE Trans. Comm. Systems.*

Jennings, N. R., Faratin, P., Norman, T. J., O'Brien, P. and Odgers, B. (2000). 'Autonomous Agents for Business Process Management', *Int. Journal of Applied Artificial Intelligence*, Vol. 14, No. 2, pp.145–189.

Lee, C. H. and Yang, H. C. (2001) 'Developing and adpative search engine for e-commerce using a web mining approach', *Procs. Int. Conf. Inform. Technol.: Coding and Computing*, pp.604–608.

Mobasher, B. and Dai, H. and Luo, T. and Nakagawa, Y. and Sun, Y. and Wiltshire, J. (2000) 'Discovery of aggregate usage profiles for web personalization', *Procs. KDD-2000 Workshop Web Mining E-Commerce.*

Klamma, R., Spaniol, M. and Jarke, M. (2003) 'Collaborative capturing and use of multimedia semantic in MPEG-7 based knowledge management systems', *IEEE Multimedia*.

Spaniol, M., Klamma, R. and Jarke, M. (2003) 'ATLAS: A web-based software architecture for multimedia e-Learning environments in virtual communities', *Procs. of the 2nd International Conference on Web-based Learning (ICWL 2003)*.

Rowlands, J. (2003) 'A Field Guide to E-Learning', *Multimedia Information and Technology*, Vol. 29, No. 4, pp.125–126.

Kozma, R. (1991) 'Learning with media', *Review of Educational Research*, Vol. 61, No. 2, pp. 179-212.

Schmitz, C., Staab, S., Studer, R., Stumme, G. and Tane, J. (2002) 'Accessing Distributed Learning Repositories through a Courseware Watchdog', *Procs. of the E-Learn 2002 - World Conference on E-Learning in Corporate, Government, Healthcare for Higher Education*.

European Comission, 'eLearning: Designing tomorrows education', *eLearning: Designing tomorrows education*, http://www.europa.eu.int/comm/elearning.

Koper, R. (2004) 'Use of the Semantic Web to Solve Some Basic Problems in Education: Increase Flexible, Distributed Lifelong Learning, Decrease Teacher's Workload', *Journal of Interactive Media in Education*, Vol. 6.

Sicilia, M. A. and Lytras, M. (2005) 'On the representation of change according to different ontologies of learning', *International Journal of Learning and Change*, Vol. 1, No. 1.

Bruce Eckel (2002) 'Thinking in Java, 2nd Edition', *Prentice Hall*.

R-Moreno, M. D. and Kearney, P. (2002) 'Integrating AI Planning with Workflow Management System', *Int. J. Knowledge-Based Systems*, Vol. 15, pp. 285-291.

Estlin, T., Rabideau, G., Mutz, D., and Chien, S. (1999) 'Using Continuous Planning Techniques to Coordinate Multiple Rovers', *Procs. of the IJCAI99 Workshop on Scheduling and Planning meet Real-time Monitoring in a Dynamic and Uncertain World*.

R-Moreno, M. D. and Borrajo, D. and Meziat, D. (2004) 'An AI Planning-based Tool for Scheduling Satellite Nominal Operations', *AI Magazine*, Vol. 24, No. 2, pp.9-28.

Tate, A. and Whiter, A. M. (1984) 'Planning with Multiple Resource Constraints and an Application to a Naval Planning Problem', *Procs. of the first Conference on the Applications of AI*.

Edelkamp, S. and Hoffmann. J. (2004) 'PDDL2.2: The Language for the Classical Part of the 4th International Planning Competition.', *Technical Report No. 195*.

R-Moreno, M. D., Oddi, A., Borrajo, D., Cesta, A. and Meziat, D. (2004) 'IPSS: Integrating Hybrid Reasoners for Planning and Scheduling', *Procs. of the 16th European Conference on Artificial Intelligence, ECAI04*.

Do, M. B. and Kambhampati, S. (2003) 'Sapa: A Scalable Multi-Objective Metric Temporal Planner', *Jounal of Artificial Intelligence Research*, Vol. 20, pp.155-194.

Ullrich, C. (2000) 'Course Generation Based on HTN Planning' *Proceedings of 13th Annual Workshop of the SIG Adaptivity and User Modeling in Interactive Systems*.

Ilghami, O. and Nau, D.S. (2003) 'A general approach to synthesize problem-specific planners.', *Technical Report CS-TR-4597, Department of Computer Science, University of Maryland*.

M. Ghallab and H. Laruelle (1994) 'Representation and Control in IxTeT, a Temporal Planner', *Procs. of the Second International Conference on AI Planning Systems (AIPS-94)*.

Fikes, R. and Nilsson, N. (1971) 'STRIPS: A new Approach to the Application of Theorem Proving to Problem Solving', *Artificial Intelligence*, Vol. 2, pp.189-208.

Kolisch, R. and Hartmann, S. (1999) 'Heuristic Algorithms for Solving the Resource-Constrained Project Scheduling Problem: Classification and Computational Analysis', *Project scheduling: Recent models, Algorithms and Applications*, pp.147-178.

MacMillan, S.A and Sleeman, D.H. (1987) 'An architecture for a self-improving instructional planner for intelligent tutoring systems', *Computational Intelligence*, Vol. 3, pp.1727.

Vassileva, J.and Deters, R.(1998) 'Dynamic Courseware Generation on the WWW. British Journal of Educational Technology', Vol. 29, No. 1, pp.5-14.

Elorriaga, J. and Fernndez-Castro, I (2000) 'Using case-based reasoning in instructional planning. towards a hybrid self-improving instructional planner' *International Journal of Artificial Intelligence in Education*, Vol. 11, pp.416449.