WELSA System

In order to validate the modeling and adaptation techniques proposed, we implemented them in an experimental educational system called WELSA (**W**eb-based **E**ducational system with **L**earning **S**tyle **A**daptation).

WELSA's functionalities are primarily addressed at the students, who can learn by browsing through the course and performing the instructional activities suggested (play simulations, solve exercises etc). They can also communicate and collaborate with their peers by means of the forum and chat. Students' actions are logged and analyzed by the system, in order to create accurate learner models. Based on the identified learning preferences and the built-in adaptation rules, the system offers students individualized courses.

WELSA provides also functionalities for the teachers, who can create courses by means of the dedicated authoring tool; they can also set certain parameters of the modeling process: behavioral pattern weights and threshold values.

Secondarily and for experimental purposes, WELSA is addressed also at researchers, who can use the learner data collected and processed by the system to evaluate the precision of the modeling method and the suitability of the chosen behavioral indicators and of the threshold values. They can also have access to aggregated information regarding the student actions and student preferences (e.g. total number of students with a particular learning preference, average reliability and confidence values).

Each of these functionalities will be presented in one of the following sections: first the overall system architecture is included in section 1, followed by the description of the intelligent way of organizing and indexing the learning material in section 2. Next, each of WELSA subcomponents is presented in turn, starting with the authoring tool (section 3), then the course player (section 4), the modeling component (section 5) and the adaptation component (section 6). The experimental validation of the system is done by creating and implementing a course module in the area of Artificial Intelligence (as described in section 7) and testing it with the students (as reported in section 8).

1. WELSA Architecture

The overall architecture of WELSA is illustrated in Figure 1.

As can be seen in Fig. 1, WELSA offers several functionalities:

- an authoring tool for the teachers, allowing them to create courses conforming to the internal WELSA format
- a course player (basic learning management system) for the students, enhanced with two special capabilities: i) learner tracking functionality (monitoring the student interaction with the system); ii) adaptation functionality (incorporating adaptation logic and offering individualized course pages)
- a data analysis tool, which is responsible for interpreting the behavior of the students and consequently building and updating the learner model, as well as providing various aggregated information about the learners.

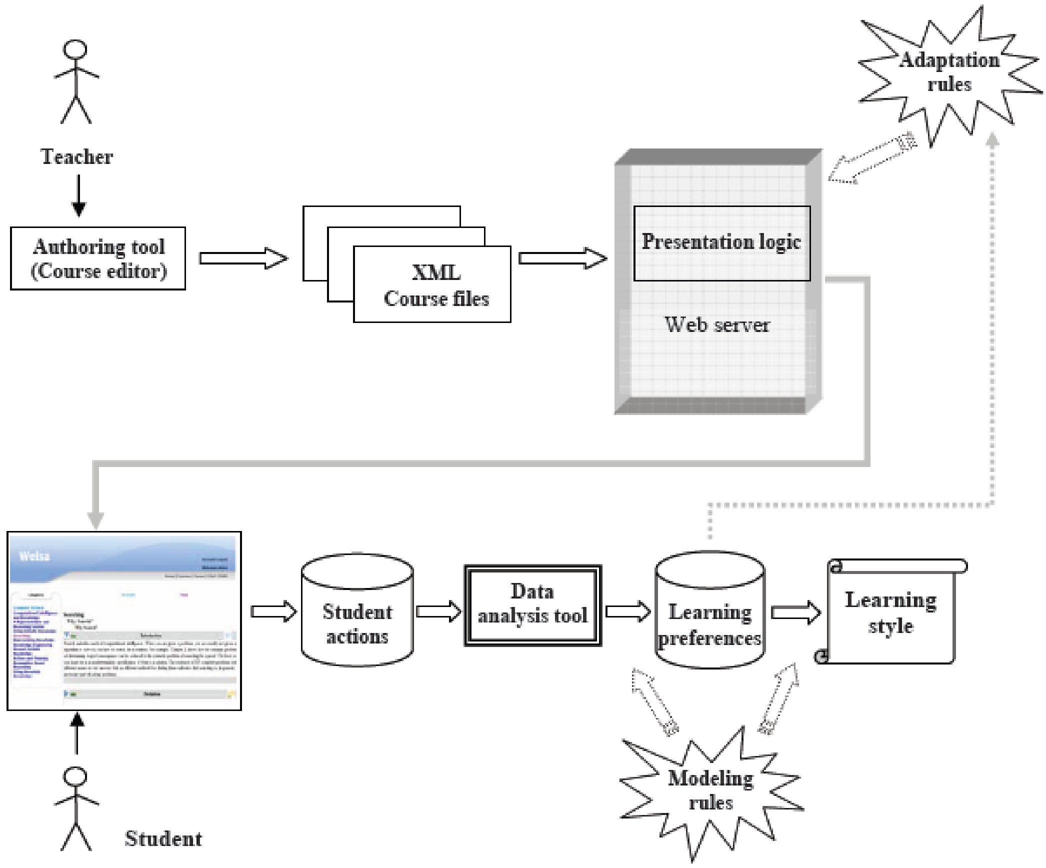


Figure 1. Overall WELSA architecture

We can thus identify two system views, corresponding to the two main actors involved in WELSA: student and teacher (author). Since it is an experimental system, there can be identified also a third role, the researcher, who mainly interacts with the Analysis tool, analyzing the modeling results and comparing them with those obtained by means of the questionnaires. Further details on the researcher role and the facilities offered to them by the system can be found in section 5.

As far as the implementation is concerned, Java-based and XML technologies are employed for all WELSA components. Apache Tomcat 6.0 (Tomcat, 2008) is used as HTTP web server and servlet container and MySQL 5.0 (MySQL, 2008) is used as DBMS. Further details will be presented in sections 2 – 6.

The first objective of WELSA system is to dynamically model the learner: identify the learning preferences by analyzing the behavioral indicators and then, based on them, infer the belonging to a particular learning style dimension. The second objective is to consequently adapt the navigation and the educational resources to match the student learning preferences. In order to achieve these two objectives, we need an intelligent way of organizing the learning material as well as a set of instructional metadata to support both the learner modeling and the adaptation, which are introduced in the next section.

2. Description and Organization of Instructional Resources in WELSA

In this section we need to answer one question: how do we categorize learning material so that we can identify the student preferences and also be able to select different content for different students? According to (Cristea, 2003), the existence of a static description of the learning content (metadata) is a necessary condition for introducing an adaptation model (dynamic description). We therefore address the problem of educational metadata in the following subsection, detailing the course organization in subsequent subsections. Our proposal for organizing and annotating the educational material was first introduced in (Popescu et al., 2008a; 2008b).

2.1. Educational Metadata

Educational metadata is a special kind of metadata that provides information about learning objects (i.e. any reproducible and addressable digital resource that can be reused to support learning (IMS MD, 2008)). Currently there are several initiatives for standardizing educational metadata, addressing the issues of reusability, interoperability, discoverability, sharing and personalization (Anido et al., 2002).

IEEE LOM (Learning Object Metadata) (IEEE LOM, 2008) is the most prominent standard, being elaborated by the IEEE Learning Technology Standards Committee. IMS Global Learning Consortium (IMS Global Learning Consortium, 2008) also contributed to the drafting of the IEEE LOM and consequently the current version of IMS Learning Resource Metadata specification (IMS LRM v. 1.3) is based on the IEEE LOM data model. LOM contains nine categories of metadata: General, Lifecycle, Meta-metadata, Technical, Educational, Rights, Relation, Annotation and Classification (see Fig. 2). The attributes that are relevant from the point of view of instruction and pedagogy are the educational ones, specifically *Learning Resource Type*. Its possible values are: *Exercise, Simulation, Questionnaire, Diagram, Figure, Graph, Index, Slide, Table, Narrative Text, Exam, Experiment, Problem Statement, Self Assessment, Lecture*.

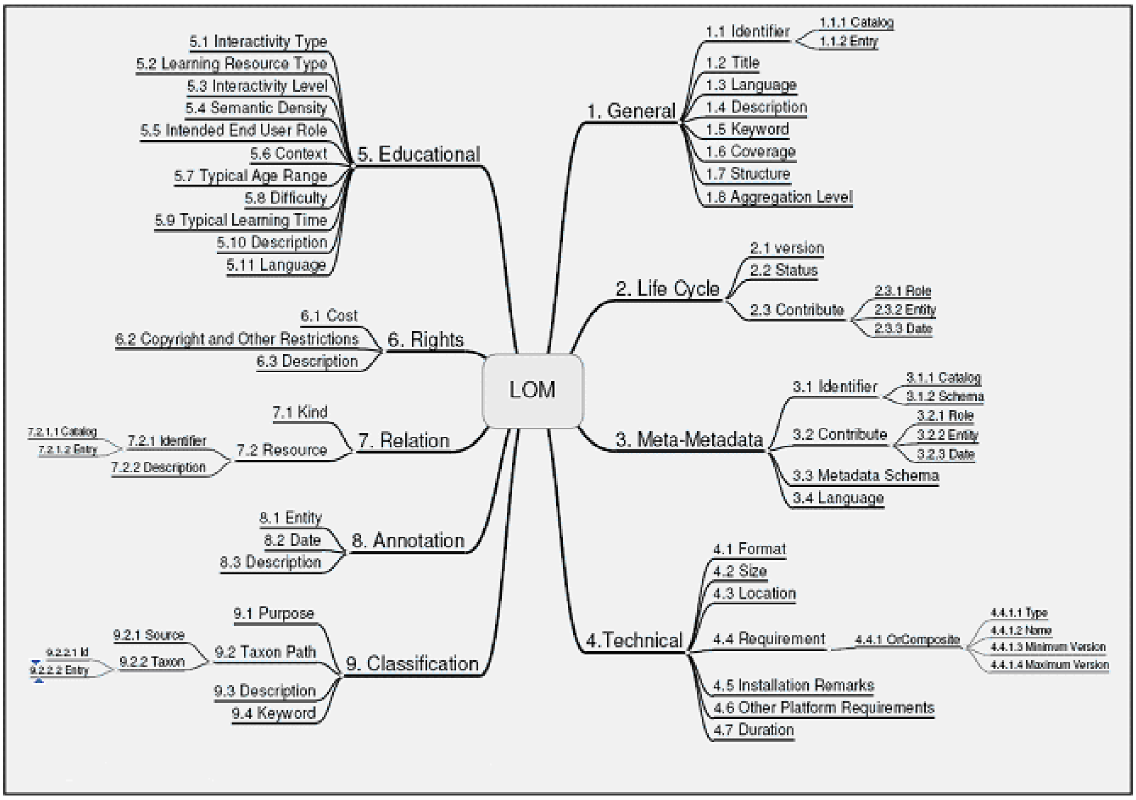


Figure 2. A schematic representation of the hierarchy of elements in the LOM data model (Barker, 2005)

Another widely known standard is *SCORM (Sharable Content Object Reference Model)* (ADL SCORM, 2008) which originates from e-learning requirements of the US Armed Forces, being produced by ADLNet (Advanced Distributed Learning Network) initiative. SCORM includes three types of learning content metadata: raw media metadata (that provide information about assets independently of learning content), content metadata (that provide information about learning content, independently of a particular content aggregation) and course metadata (that provide information about the content aggregation).

Dublin Core metadata standard (DCMI, 2008) is a simple yet effective general-purpose metadata scheme, for describing a wide range of networked resources. It was developed within the Dublin Core Metadata Initiative (DCMI). At present, there is a joint DCMI/IEEE LTSC Task Force activity, with the objective of developing a representation of the metadata elements of the IEEE LOM in the Dublin Core Abstract Model.

The main problem with these specifications is that they fail to include the instructional perspective (Ullrich, 2005). In case of LOM, the property *Learning Resource Type* attempts to address this issue, but mixes instructional and technical information. Thus some of the values describe the instructional role of the resource (*Exercise, Simulation, Experiment*), while others are concerned with their format (*Diagram, Figure, Graph, Slide, Table*). Moreover, some important instructional types are missing, such as *Definition, Example* or *Theorem*. In order to overcome this issue, Ullrich (2005) introduced an instructional ontology, which is domain independent and pedagogically sound. This is an ontology of instructional items, which is composed of two main classes: Concept (corresponding to Fundamental items, which describe central pieces of knowledge) and Satellite (corresponding to Auxiliary items, which provide additional information about the concepts); each of these two classes subsumes several other classes, as can be seen in Fig. 3. One of the most important advantages of this ontology is its pedagogical flexibility, being independent of a particular instructional theory. Moreover, as we will show further on, the ontology can also be enhanced to serve adaptivity purposes, from the point of view of various learning styles. Thus we will first describe the organization of the learning resources and afterwards we will introduce the educational metadata used.

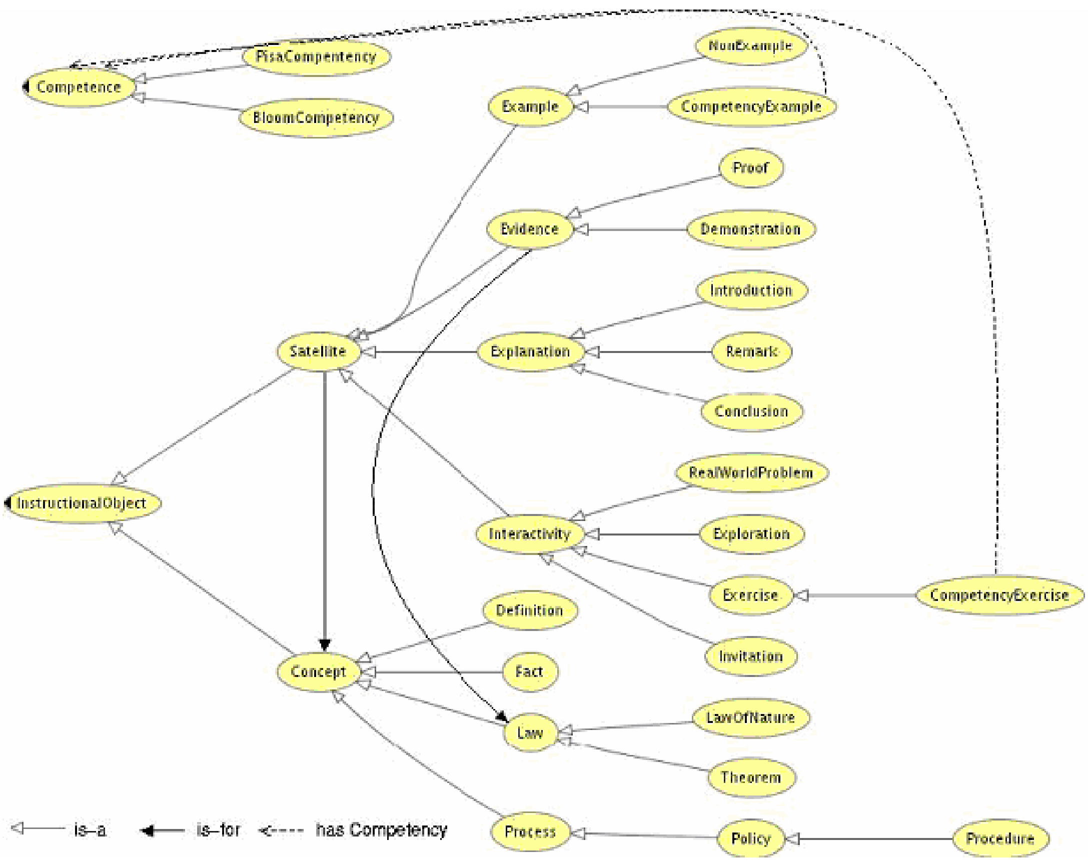


Figure 3. Ullrich's instructional ontology (Ullrich, 2005)

2.2. Organizing the Educational Material in WELSA

According to (IMS MD, 2008), learning objects represent any digital resources that can be reused to support learning. In our case, the most complex learning object (with the coarsest granularity) is the course, while the finest granularity learning object is the elementary educational resource. We have conceptualized the learning material using the hierarchical organization illustrated in Fig. 4: each course consists of several chapters, and each chapter can contain several sections and subsections. The lowest level subsection contains the actual educational resources. Each such elementary learning object corresponds to a physical file and has a metadata file associated to it. This fine grained representation of the learning content is needed to insure the adaptation and modeling requirements.

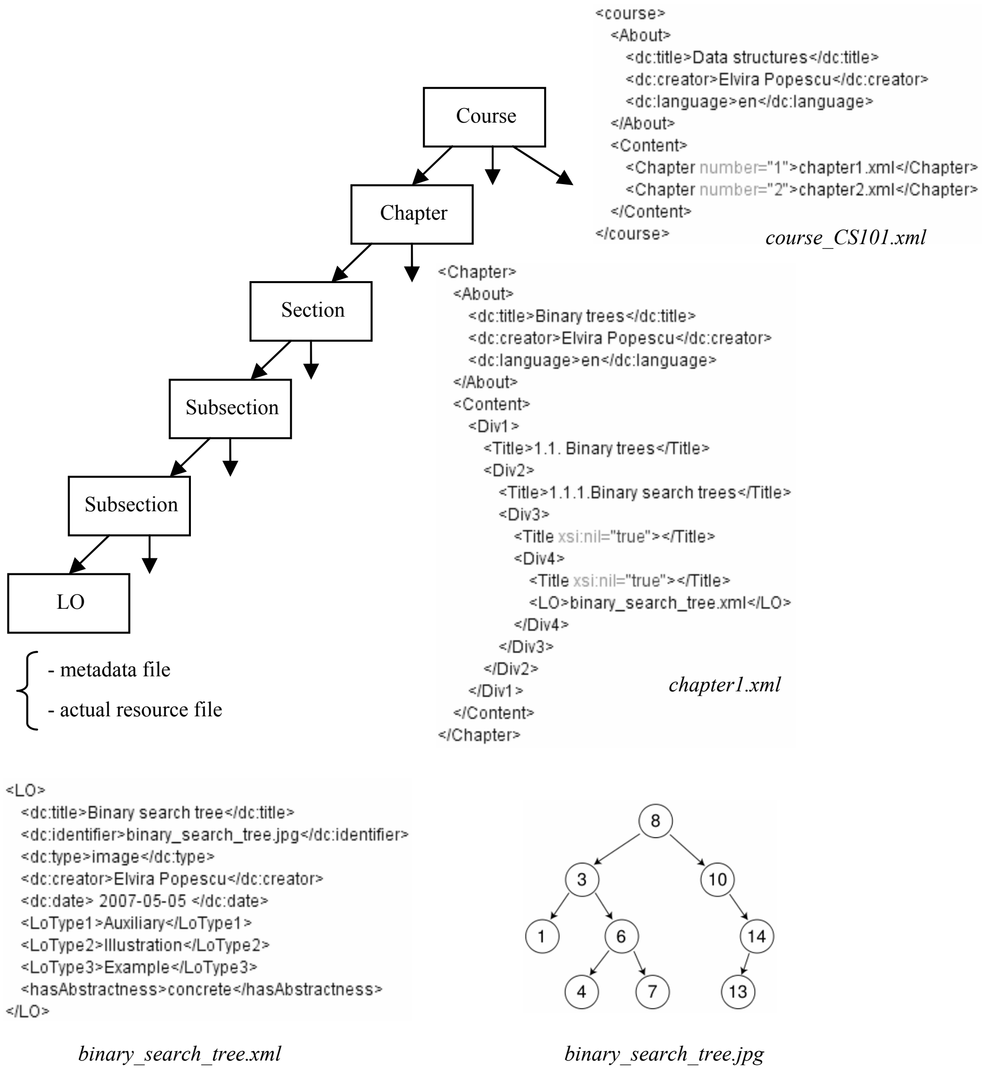


Figure 4. Organization of learning content in WELSA

Based on our teaching experience, this is the natural and most common way a teacher is usually organizing her/his teaching materials. Additionally, this hierarchical approach presents several advantages, facilitating:

- good reuse of the educational resources
- detailed learner tracking (since we know all the information about the learning resource that is accessed by the learner at a particular moment)

- fine granularity of adaptation actions.

In Fig. 5 and 6 we give a schematic representation of the XML schemas for the course and chapter files respectively, generated with Oxygen tool (Oxygen, 2008).

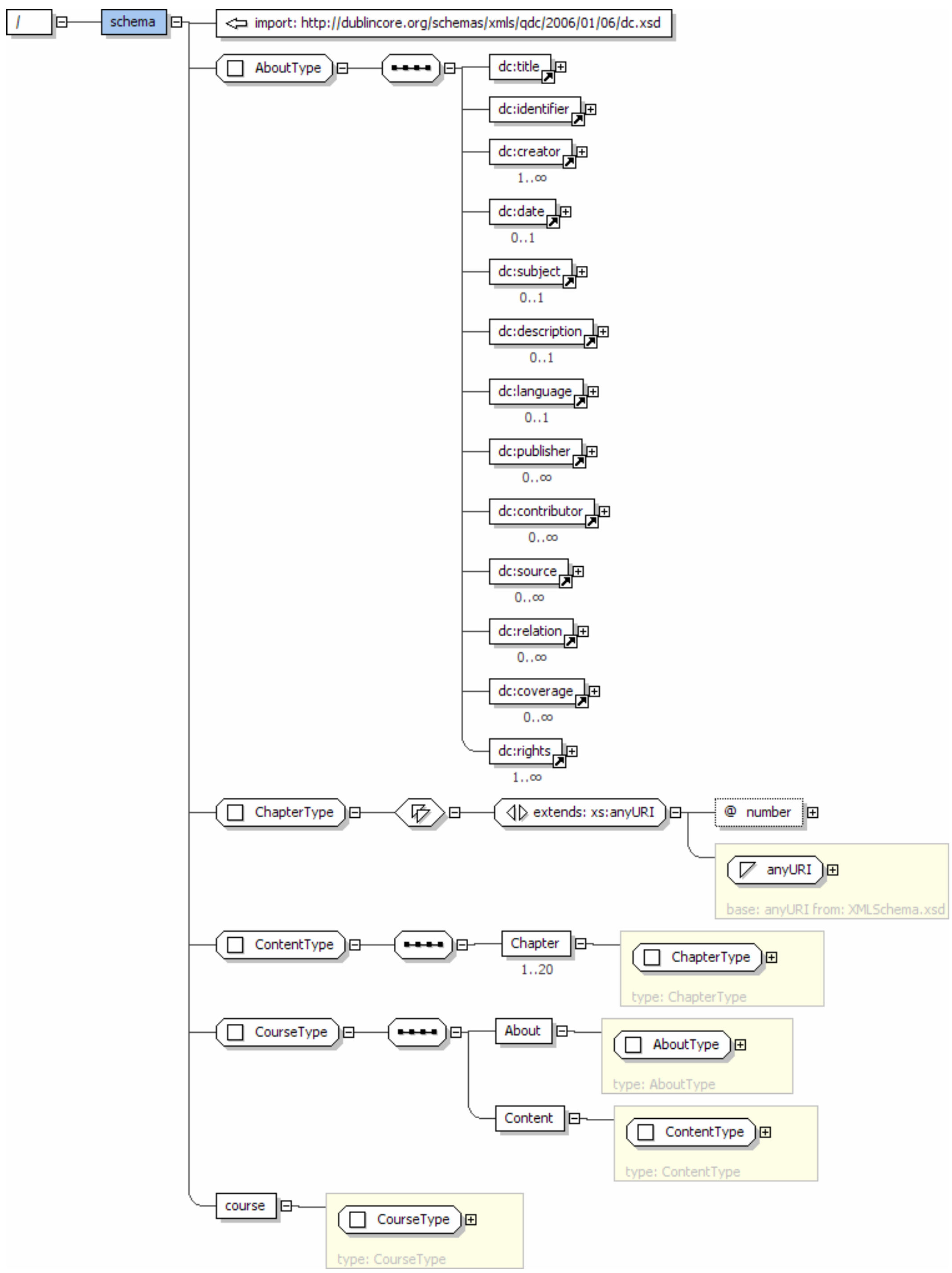


Figure 5. Graphical view of XML course file structure

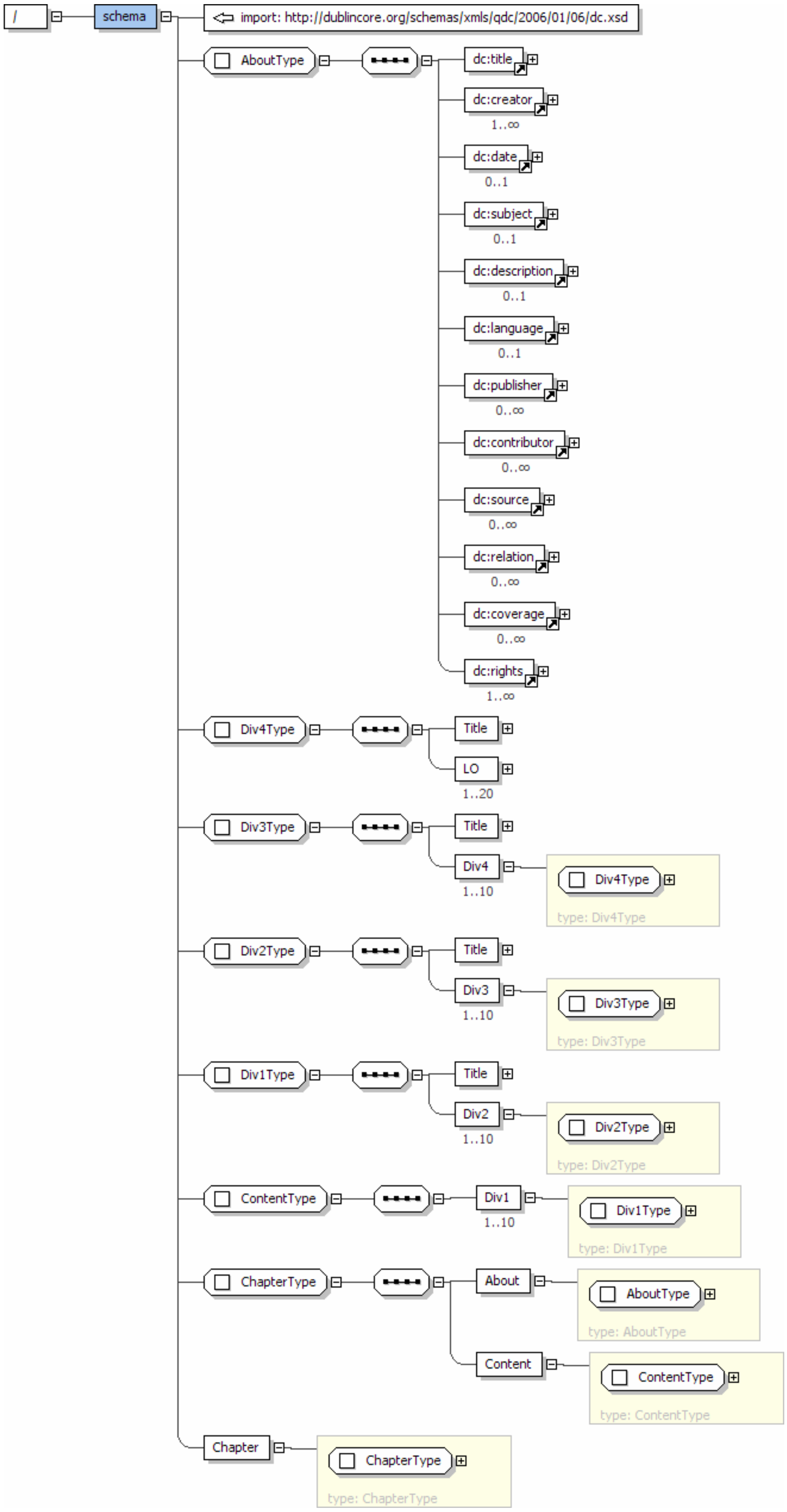


Figure 6. Graphical view of XML chapter file structure

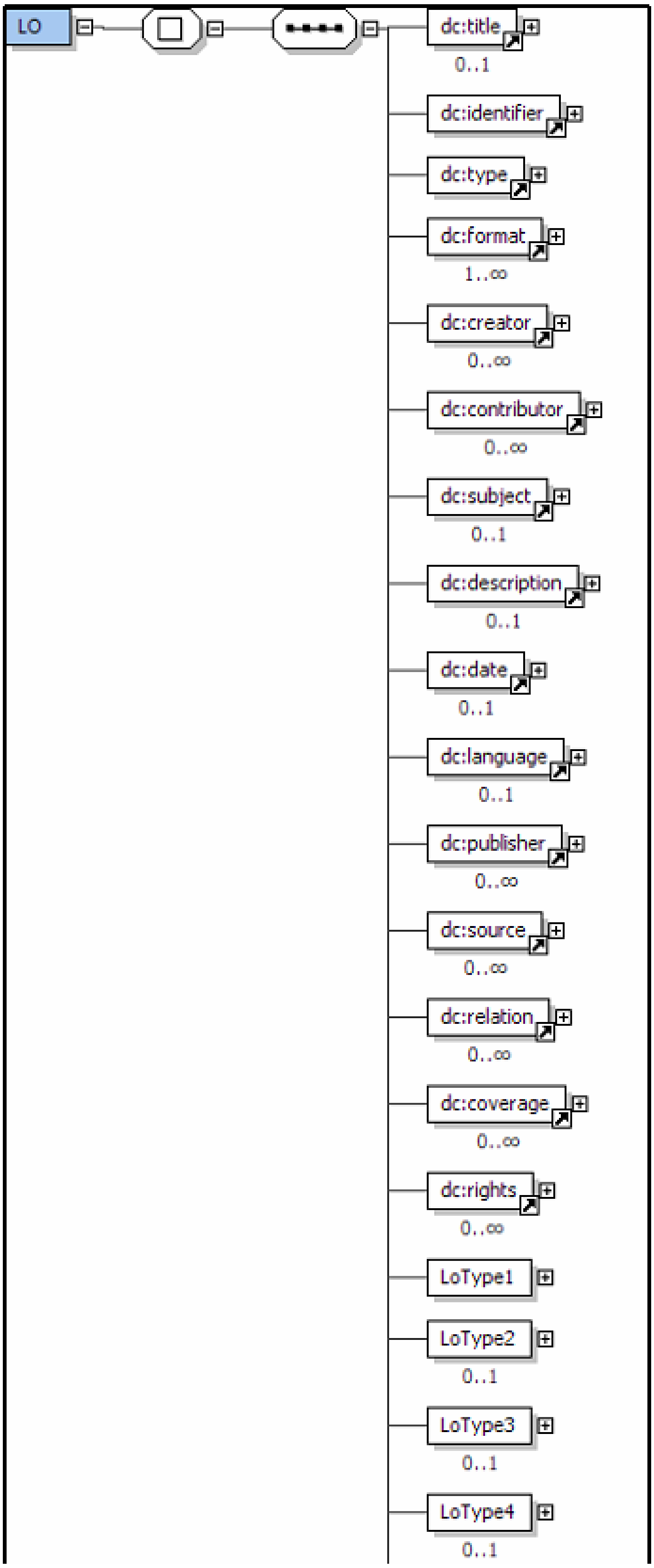
2.3. Indexing Learning Content in WELSA

As far as the educational metadata is concerned, one possible approach (which is used in (Gascuena et al., 2006)) would be to associate to each learning object the learning style that it is most suitable for. One of the disadvantages is that this approach is tied to a particular learning style. Moreover, the teacher must create different learning objects for each learning style dimension and label them as such. This implies an increase in the workload of the teacher, and also the necessity that she/he possesses knowledge in the learning style theory. Furthermore, this approach does not support dynamic learner modeling, since accessing a learning object does not offer sufficient information regarding the student (a learning object can be associated with several learning styles).

Instead, we propose a set of metadata that describe the learning object from the point of view of instructional role, media type, level of abstractness and formality, type of competence etc. These metadata were created by enhancing core parts of Dublin Core (DCMI, 2008) and Ullrich's instructional ontology (Ullrich, 2005) with some specific extensions to cover the requirements of a LSAES. Thus some of the descriptors of a learning object are:

- title (the name given to the resource) → *dc:title*
- identifier (a reference to the actual resource, such as its URL) → *dc:identifier*
- type (the nature of the content of the resource, such as text, image, animation, sound, video) → *dc:type*
- format (the physical or digital manifestation of the resource, such as the media type or dimensions of the resource) → *dc:format*
- instructional role, either i) fundamental: definition, fact, law (law of nature, theorem) and process (policy, procedure) or ii) auxiliary: evidence (demonstration, proof), explanation (introduction, conclusion, remark, synthesis, objectives, additional information), illustration (example, counter example, case study) and interactivity (exercise, exploration, invitation, real-world problem) → *LoType1, LoType2, LoType3, LoType4*.
- related learning objects: i) *isFor* / *inverseIsFor* (relating an auxiliary learning object to the fundamental learning object it completes); ii) *requires* / *isRequiredBy* (relating a learning object to its prerequisites); iii) *isA* / *inverseIsA* (relating a learning object to its parent concept); iv) *isAnalogous* (relating two learning objects with similar content, but differing in media type or level of formality).

A graphical representation of the metadata schema is included in Fig. 7.


LO
dc:title
0..1
dc:identifier
dc:type
dc:format
1..∞
dc:creator
0..∞
dc:contributor
0..∞
dc:subject
0..1
dc:description
0..1
dc:date
0..1
dc:language
0..1
dc:publisher
0..∞
dc:source
0..∞
dc:relation
0..∞
dc:coverage
0..∞
dc:rights
0..∞
LoType1
LoType2
0..1
LoType3
0..1
LoType4
0..1

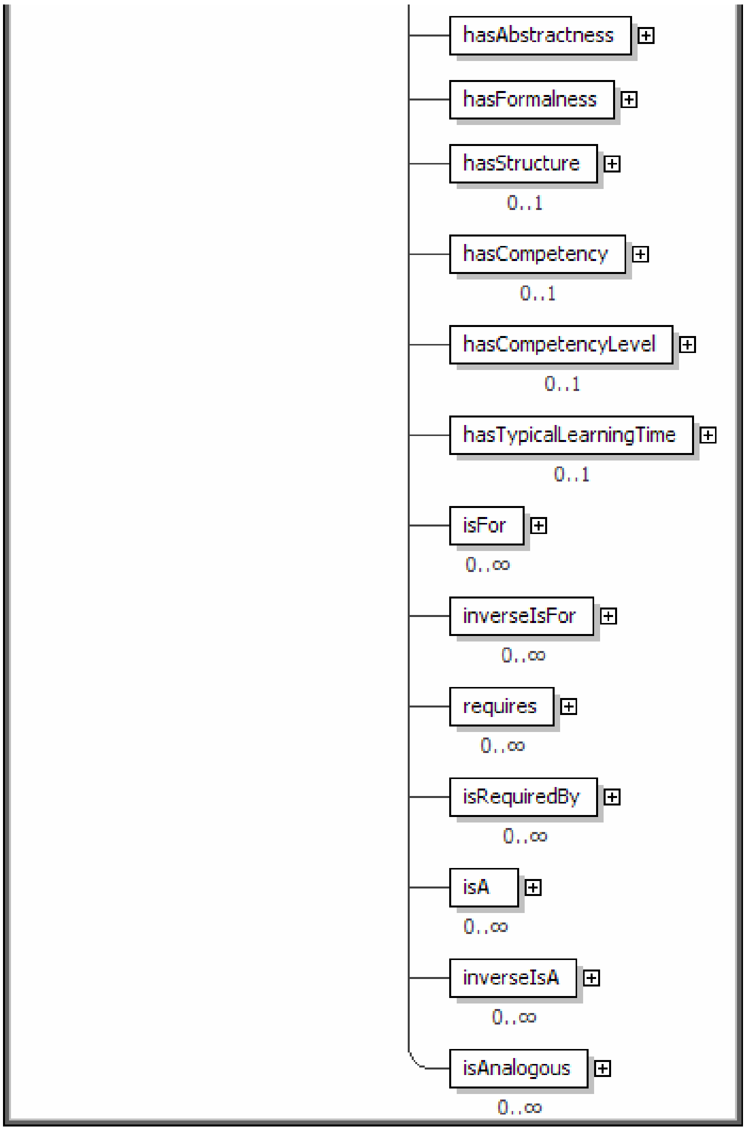


Figure 7. Graphical view of metadata schema

Obviously, these descriptors are independent of any learning style. However, by analyzing the interaction between the student and the learning objects described by these metadata (time spent on each learning object, order of access, frequency of accesses), the system can infer a particular learning preference of the student. Furthermore, the teacher has to supply only annotated learning content (the static description) while the adaptation logic (the dynamic description) is provided by the system. This means that the adaptation rules are independent of the learning content and that they can be supplied by specialists in educational psychology. Sections 5 and 6 will illustrate the use of these metadata for modeling the learner and providing adaptation respectively.

While the elements pointing to the instructional role of the learning objects (*LoType1, LoType2, LoType3, LoType4*) correspond to the pedagogical model, the domain model is represented by means of the *dc:subject* element. Furthermore, the different relationships between the concepts are represented by means of the *isFor* / *inverseIsFor, requires* / *isRequiredBy, isA* / *inverseIsA* and *isAnalogous* metadata elements.

2.4. Related Approaches

Currently there are several works that address aspects related to ontologies and metadata for personalized e-learning, such as: (Al-Khalifa and Davis, 2006; Brown et al., 2005; Devedzic, 2006; Dolog et al., 2004; Dolog and Nejdl, 2007; Gascuena et al., 2006; Geser, 2007; Shi et al., 2004). A few of them, that we will briefly discuss here, also take into consideration learning styles.

In case of (Gascuena et al., 2006) the ontology is tied to a particular learning style model, namely Felder-Silverman (FSLSM) (Felder and Silverman, 1988). There is a special class, *LearningStyle*, which represents the FSLSM dimension associated to a particular learning object (active-reflective, visual-verbal, sensing-intuitive, sequential-global). Thus all learning objects have to be classified according to FSLSM in order to allow for delivering of adapted content.

(Brown et al., 2005) proposes a learning style taxonomy, based on Curry's onion model (Curry, 1987). In the LAG adaptation model, each learning style can be associated with a specific instructional strategy, which can be broken down into adaptation language constructs, which in their turn can be represented by elementary adaptation techniques. It is the role of the author to specify not only the annotated learning content (the static description) but also the adaptation logic (the dynamic description).

Finally, (Shi et al., 2004) introduces the concept of Open Learning Objects, which represent distributed multimedia objects in SVG format. They incorporate inner metadata in XML format which is structured on several levels (content, adaptation, animation...). Each Open Learning Object is tied to a particular learning style dimension; however any learning style model can be employed, by configuring the adaptation markup.

In this section we sketched an intelligent way of organizing and indexing the learning resources in WELSA. The next step is to offer teachers an authoring tool to help them create courses conforming to this internal format, which is the subject of our next section.

3. Course Authoring in WELSA

Generally, the process of authoring adaptive hypermedia involves several steps (Stash, 2007):

- creating the actual content (which should include alternatives to correspond to various learner needs, in terms of media type, instructional role, difficulty level etc)
- creating the domain model (defining the concepts that are to be taught and the prerequisite relations between them)
- specifying the criteria to be used for adaptation (e.g. learner's knowledge level, goals, learning style)
- creating the adaptation model (defining the adaptation logic).

In case of WELSA, authors only have to create the actual content and annotate it with a predefined set of metadata (provide the static description). The hierarchical and prerequisite relations between concepts are implicitly specified by means of the *isFor, inverseIsFor, requires, isRequiredBy, isA, inverseIsA* metadata elements. The criteria to be used for adaptation are the learning preferences of the students, as defined in the ULSM. Finally, the adaptation model (the dynamic description) is supplied by the application, in the form of a predefined set of adaptation rules, as depicted in section 6.

In order to support the teacher in creating courses conforming to WELSA internal format, we designed an authoring tool, which assists the teacher in the process of assembling and annotating the learning resources; it automatically generates the appropriate file structure, as required by the specific way of organizing and indexing the educational content in WELSA. It should be noted that WELSA course editor does not deal with the creation of actual content (text, images, simulations etc) – a variety of existing dedicated tools can be used for this purpose (text editors, graphics editors, HTML editors etc). Instead, WELSA course editor provides a tool for adding metadata to existing learning resources and defining the course structure (specifying the order of resources, assembling learning objects in pages, subsections and sections).

The editor was implemented as a web-based tool, using JSP and XML technologies (JSP, 2008; XML DOM, 2008), Apache Tomcat 6.0 as application server (Tomcat, 2008) and MySQL 5.0 as DBMS (MySQL, 2008).

The Course Editor at Work

After logging into the system and selecting a course (Fig. 8), the teacher is offered the possibility to add, remove or modify existing chapters. Figure 9 shows the corresponding page for an Artificial Intelligence course, which currently contains 4 chapters. Next, the teacher can define the structure of a selected chapter, by creating sections and subsections and uploading the actual learning objects, as can be seen in Fig. 10. Finally, the metadata files need to be created, by using the supplied metadata editor (see Fig. 11). The corresponding XML files are subsequently generated by the application and stored on the server.

Figure 8. A snapshot of the course editor – selecting a course

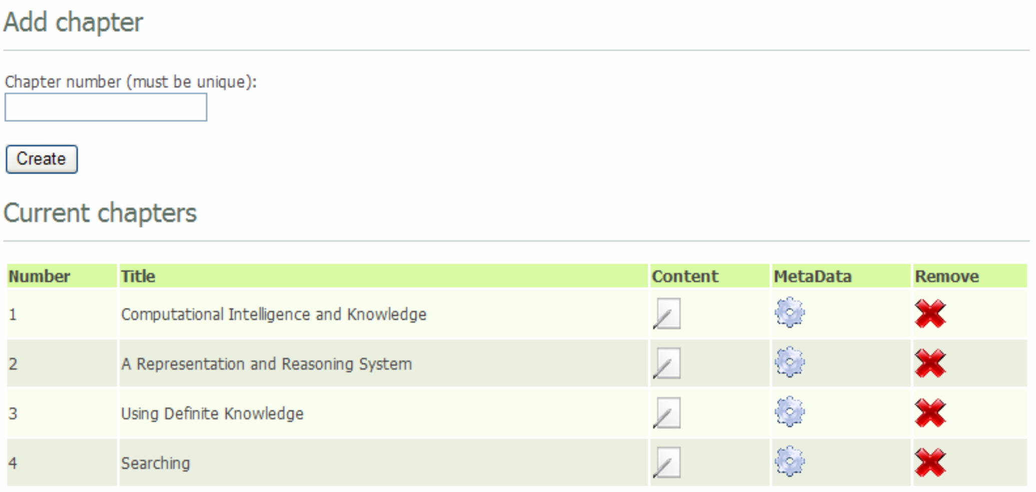
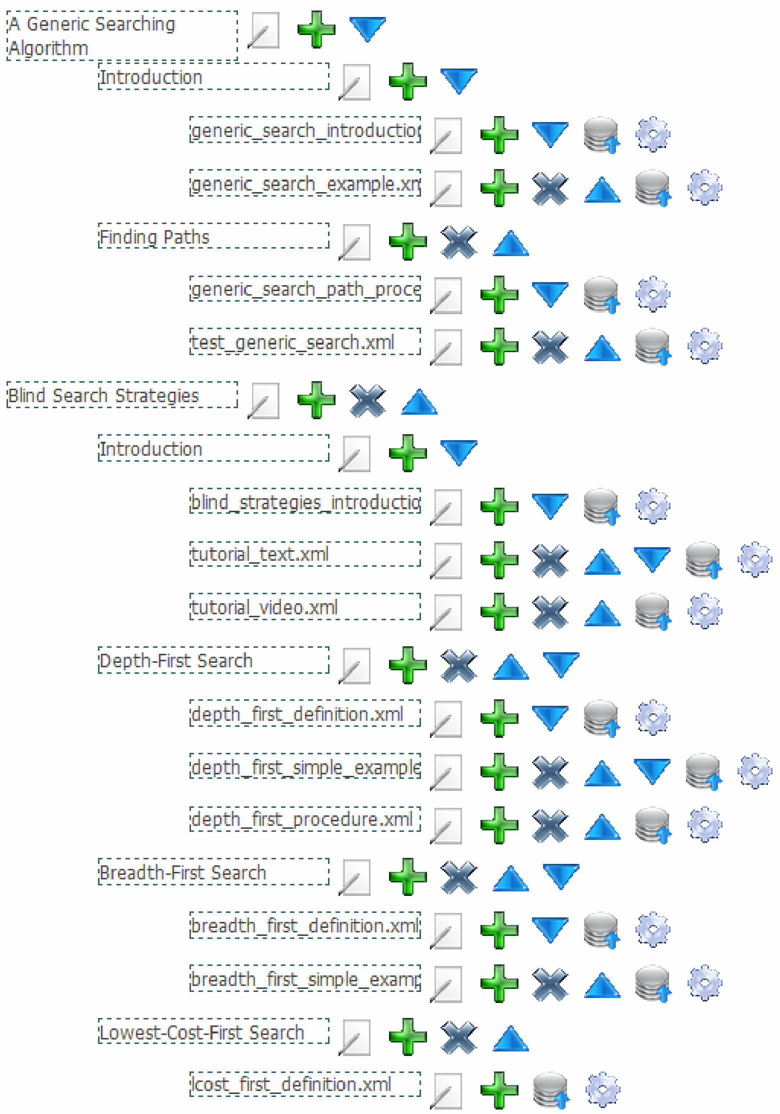


Figure 9. A snapshot of the course editor – adding chapters

a)



b)

Figure 10. A snapshot of the course editor: a) adding/removing learning objects; b) uploading learning objects

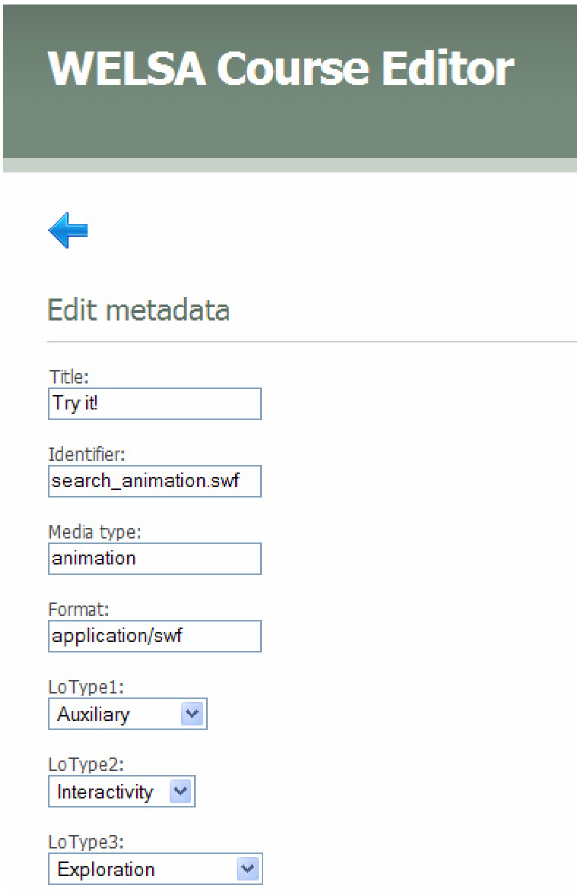


Figure 11. A snapshot of the metadata editor

Once the course files are created by the authoring tool, a player is needed in order to generate the HTML files that will be shown to the students. In the next section we describe this course player, which is enhanced with learner tracking capabilities, in order to monitor and record all student actions for further analysis.

4. WELSA Course Player

WELSA doesn't store the course web pages but instead generates them on the fly, by means of the course player module. The schematic representation of this component's architecture is illustrated in Fig. 12.

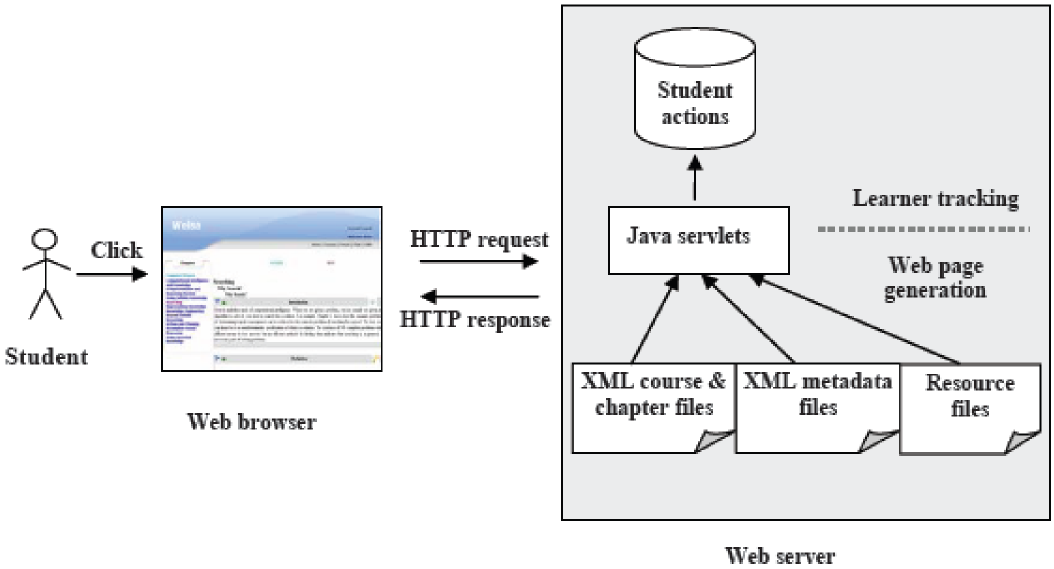


Figure 12. Course player schematic architecture

The main function of the course player is to generate the web pages so that they can be visualized by the students. These web pages are dynamically composed from the elementary learning objects, following the structure indicated in the XML course and chapter files (see Fig. 4). An example of such a web page resulted from composing several LOs is included in Fig. 13.

Another function of the course player is to track student actions (down to click level) and record them in a database for further processing. This is done with the help of JavaScript code added to the HTML page, coupled with Ajax technology (Ajax, 2008). Thus the application can communicate with the web server asynchronously in the background, without interfering with the display and behavior of the existing page. In traditional web applications, the server returns a new page each time the user submits input, so that the application may run more slowly and tend to be less user-friendly. With Ajax, the JavaScript code can communicate directly with the server (through the *XMLHttpRequest* object) and thus a web page can make a request to, and get a response from a web server without reloading the page.

Furthermore, using Ajax, a web application can request only the content that needs to be updated, which drastically reduces bandwidth usage. We therefore use it in WELSA when the student requires the expansion of a learning object, which means that only a small section of the page needs to be reloaded. By using Ajax, WELSA is more responsive, giving users the feeling that changes are happening instantaneously.

As far as the tracking data is concerned, for each student action its author, type, date and a short description are recorded. There are several such action types: *login, logout, home, jumpToCourse, jumpToChapter, jumpToPage, nextButton, prevButton, outline, accessLO, expandLO, collapseLO, lockLO, unlockLO.* The description differs with the action type, containing specific information, such as the LO identifier in case of an *expandLO* action, or the source and destination page in case of a *jumpToPage* action.

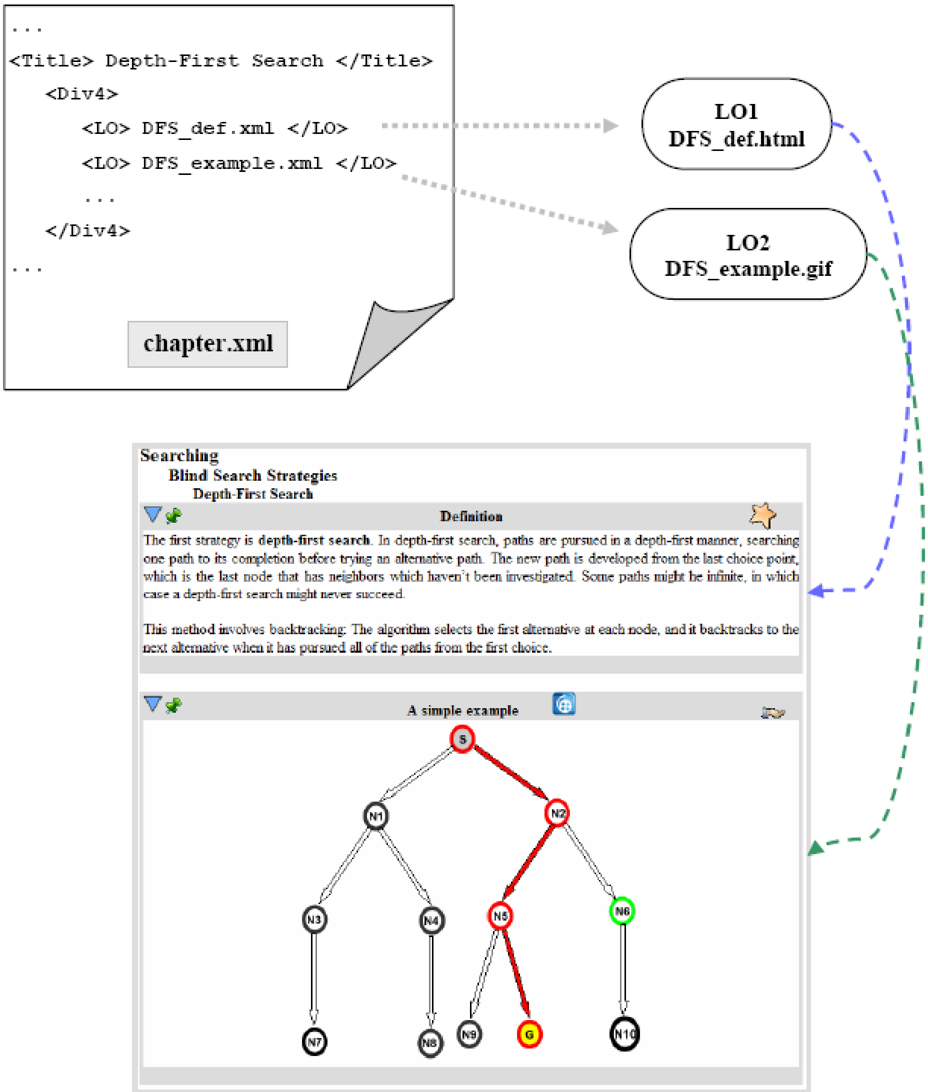


Figure 13. Composing a page from elementary learning objects

Using the Course Player

Apart from the two specific functionalities (web page generation and learner monitoring), WELSA course player also incorporates some basic LMS functions, such as: administrative support (registration and authentication) and communication and collaboration tools (discussion forum, chat).

When first accessing WELSA, the student is asked to provide login credentials, as in Fig. 14.

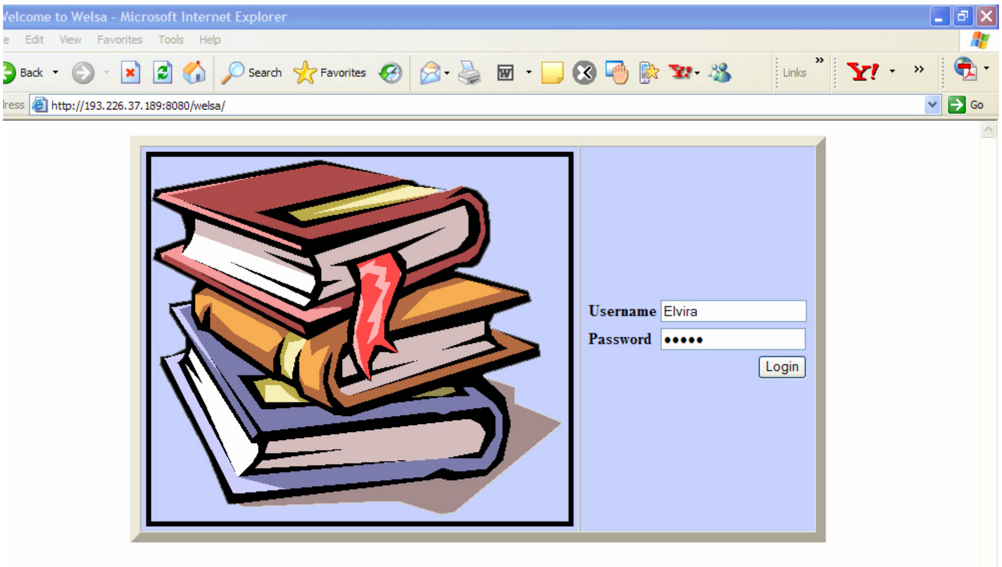


Figure 14. WELSA – login page

Next the student may choose between browsing through a course (Fig. 15), accessing the chat or visiting the forum (Fig. 16).

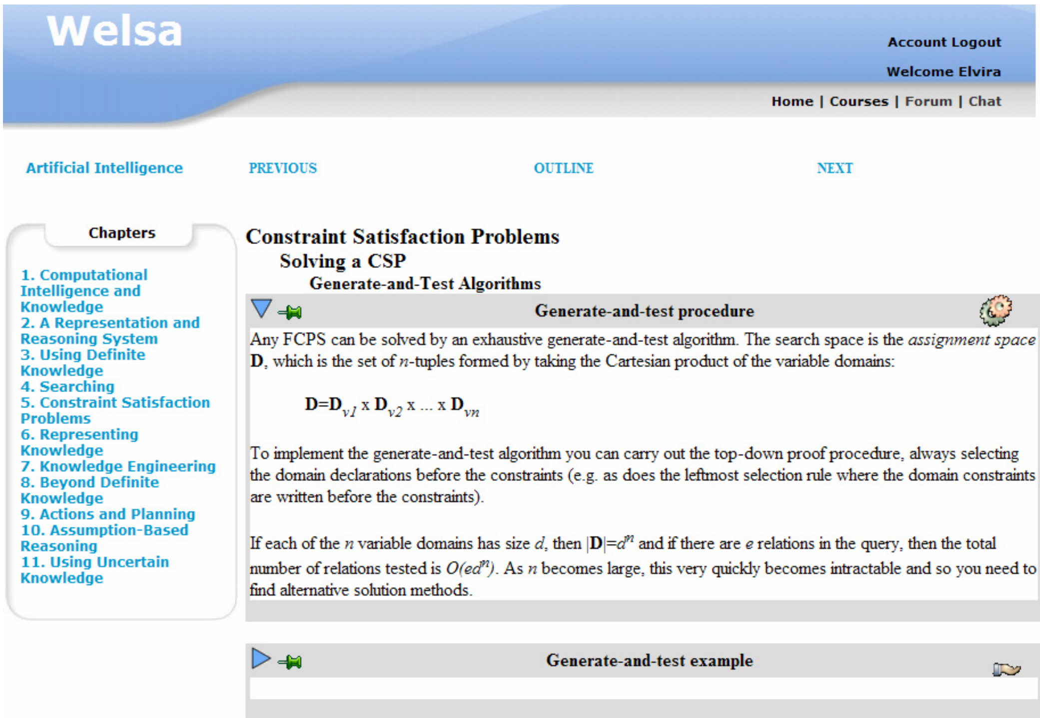


Figure 15. WELSA – a snapshot of the course player

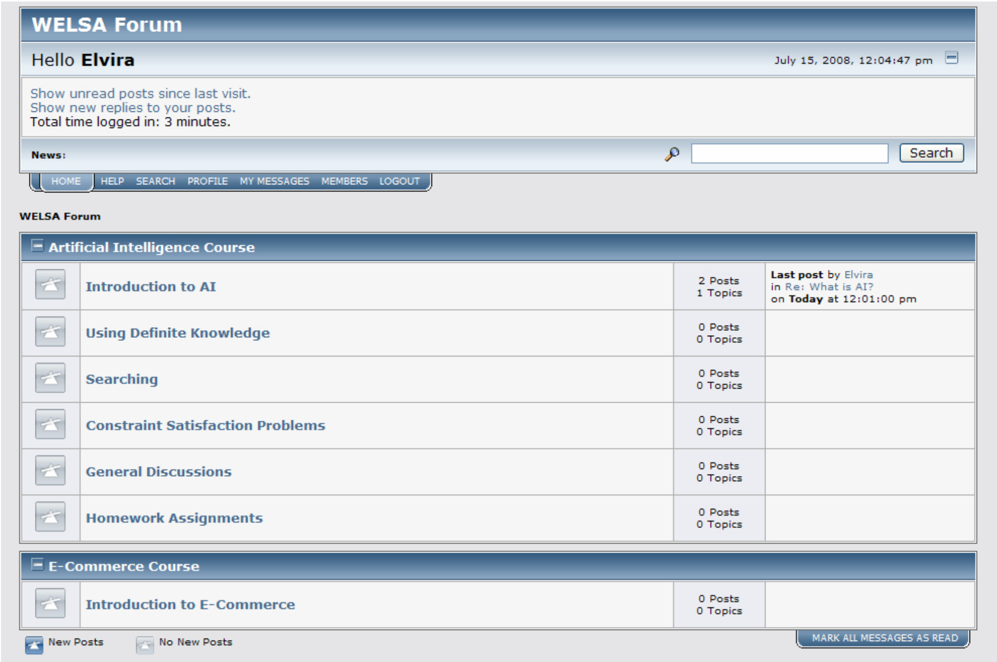


Figure 16. WELSA Forum

A few notes should be made regarding the generated web pages: the first resource (LO) on the page is entirely visible (expanded form), while for the rest of LOs only the title is shown (collapsed form). Of course, the student may choose to expand or collapse any resource, as well as locking them in an expanded state by clicking the corresponding icons (and , respectively). Also, there are specific icons associated to each LO, depending on its instructional role and its media type, in order to help the learner browse more effectively through the resources. Finally, navigation can be done by means of the Next and Previous buttons, the course outline or the left panel with the chapter list.

5. WELSA Analysis Tool

Once the learner actions are recorded by the course player, they have to be processed by the Analysis tool, in order to yield the learning preferences of the students. The modeling mechanism is depicted in Fig. 17.

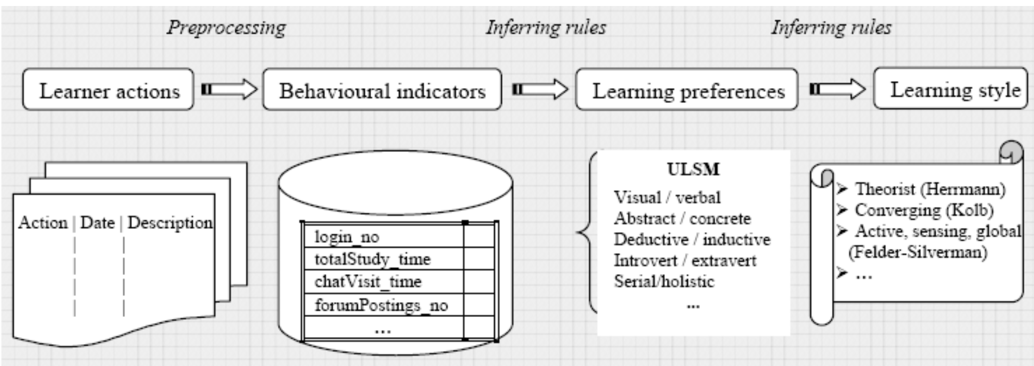


Figure 17. WELSA learner modeling mechanism

As we pointed out in section 1, the Analysis tool is mainly aimed at the teacher, who can modify the predefined pattern weights and thresholds. Since WELSA is an experimental system, the Analysis tool is also aimed at the researcher, who can visualize the data as well as use them for further analysis. The roles and interactions of the actors with the tool are illustrated in Fig. 18.

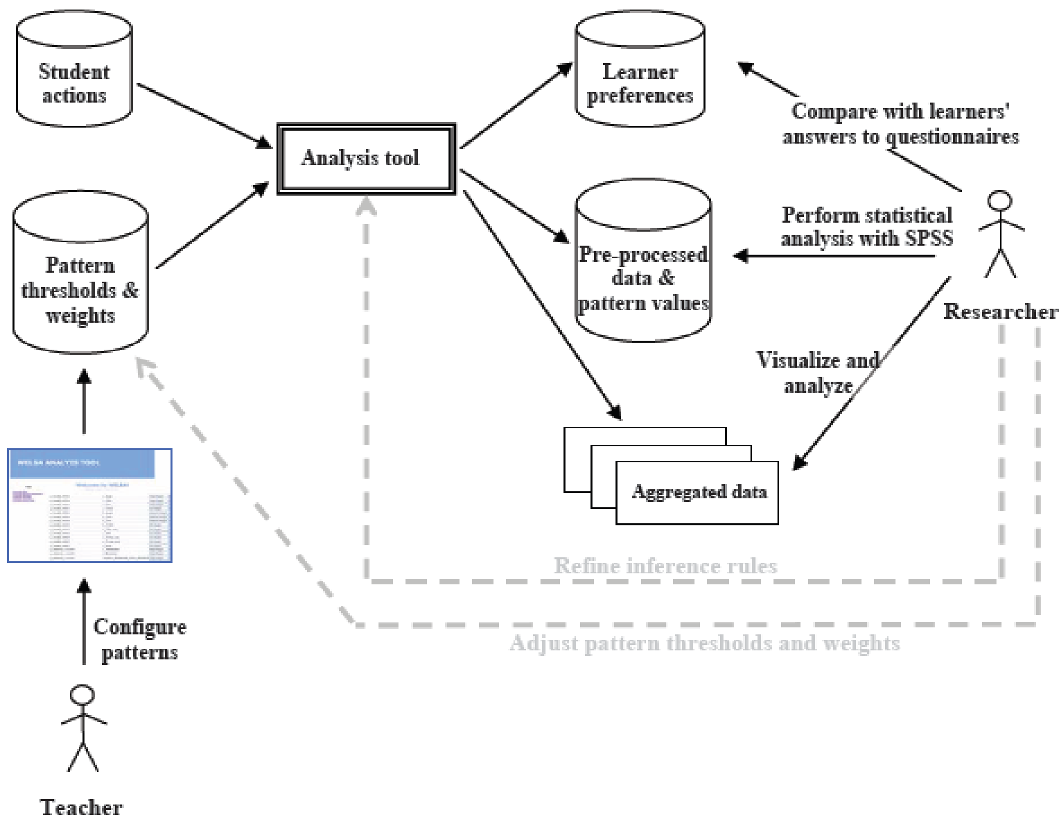


Figure 18. Users' interaction with the Analysis tool

The Analysis tool implements the automatic learner modeling method. Besides the function of diagnosing the student learning preferences, the Analysis tool also offers various aggregated data that can be used by the researcher for comparisons and statistical purposes. Furthermore, all the intermediate data (duration of learner actions, pattern values, pattern thresholds, reliability and confidence values) can be visualized by the researcher.

In order to compute the pattern values, a pre-processing phase of the raw data (i.e. the student actions and the associated timestamps) is necessary. The first step is to compute the duration of each action for each student, eliminating the erroneous values (for example, accessing the outline for more than 3 minutes means that the student actually did something else during this time). Next the access time for each LO is computed, again filtering the spurious values (for example, an LO access time of less than 3 seconds was considered as random or a step on the way to another LO and therefore not taken into account). The data were then aggregated to obtain the pattern values for each student (e.g. total time spent on the course, total number of actions performed while logged in, time spent on each type of LO, number of hits on each category of LOs, the order of accessing the LOs, the number of navigation actions of a specific type, the number of messages in chat / forum etc). The reliability levels of these patterns are calculated as well.

Next the Analysis tool computes the ULSM preferences values, based on the pattern values, their reliability levels and their weights. The confidence values are also computed, based on the availability of data for each student, and consequently on the reliability levels. Finally, the learner model is updated with the newly identified ULSM preferences.

At teacher's request, the analysis tool also computes and displays aggregated information, such as the total number of students with each ULSM preference, the total and average number of student actions, the average reliability and confidence values etc.

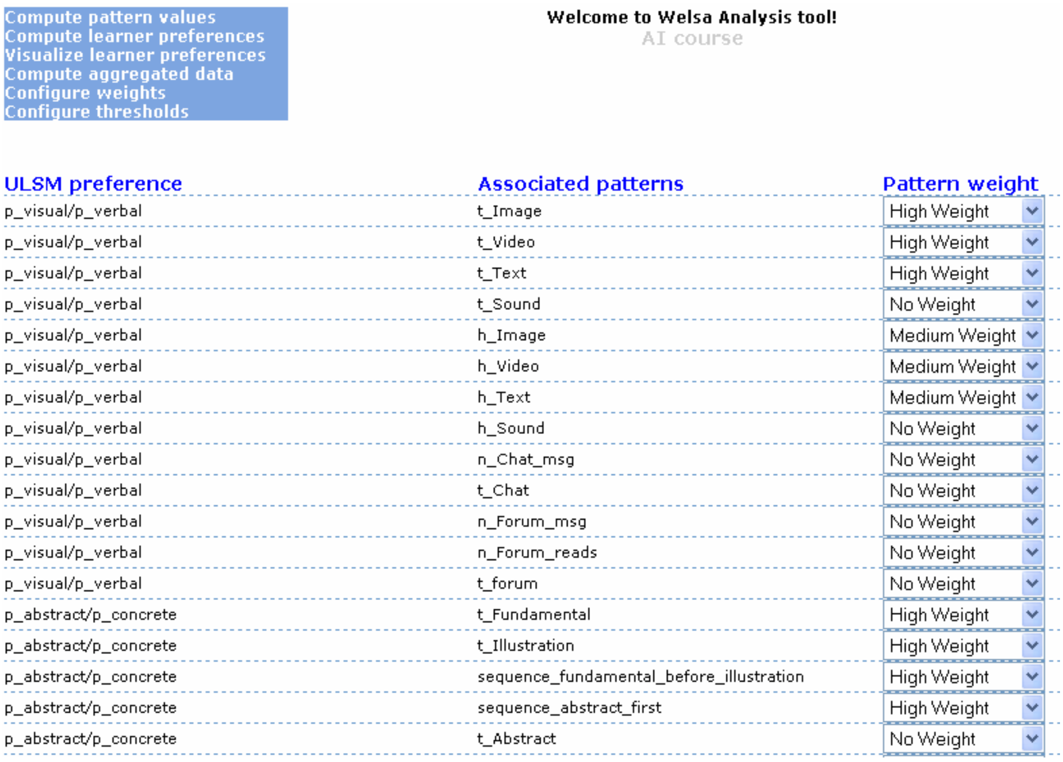
Using the Analysis Tool

After logging into the system, the teacher/researcher can choose between configuring the pattern weights / thresholds, visualizing the learner preferences or various aggregated data, as seen in Fig. 19.

Figure 19. The main page of the Analysis tool

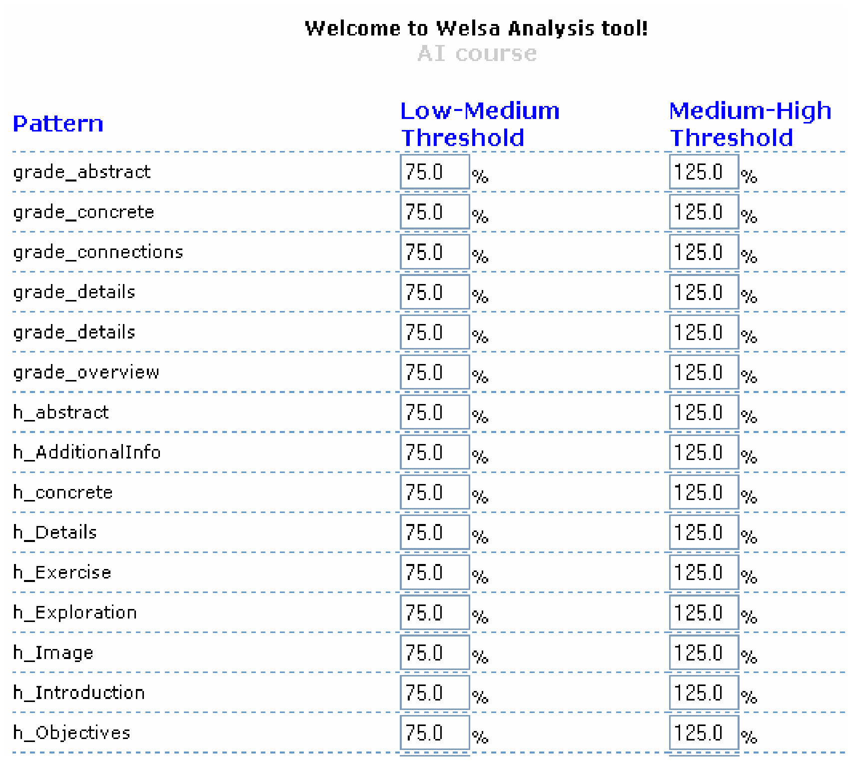
The pattern weights and thresholds depend to a certain extent on the structure and the subject of the course, so the teacher should have the possibility to adjust the predefined values to correspond to the particularities of her/his course or even to eliminate some of the patterns, which are not relevant for that course. This is why the Analysis tool has a configuration option, which allows the teacher to modify the weight and threshold values, as seen in Fig. 20.

a)



ULSM preference	Associated patterns	Pattern weight
p_visual/p_verbal	t_Image	High Weight
p_visual/p_verbal	t_Video	High Weight
p_visual/p_verbal	t_Text	High Weight
p_visual/p_verbal	t_Sound	No Weight
p_visual/p_verbal	h_Image	Medium Weight
p_visual/p_verbal	h_Video	Medium Weight
p_visual/p_verbal	h_Text	Medium Weight
p_visual/p_verbal	h_Sound	No Weight
p_visual/p_verbal	n_Chat_msg	No Weight
p_visual/p_verbal	t_Chat	No Weight
p_visual/p_verbal	n_Forum_msg	No Weight
p_visual/p_verbal	n_Forum_reads	No Weight
p_visual/p_verbal	t_forum	No Weight
p_abstract/p_concrete	t_Fundamental	High Weight
p_abstract/p_concrete	t_Illustration	High Weight
p_abstract/p_concrete	sequence_fundamental_before_illustration	High Weight
p_abstract/p_concrete	sequence_abstract_first	High Weight
p_abstract/p_concrete	t_Abstract	No Weight

b)



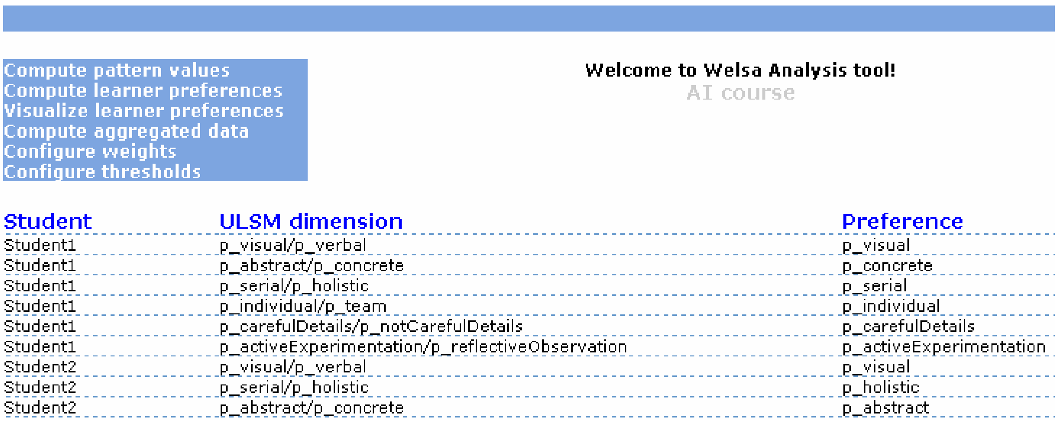
Welcome to Welsa Analysis tool!

AI course

Pattern	Low-Medium Threshold	Medium-High Threshold
grade_abstract	75.0 %	125.0 %
grade_concrete	75.0 %	125.0 %
grade_connections	75.0 %	125.0 %
grade_details	75.0 %	125.0 %
grade_details	75.0 %	125.0 %
grade_overview	75.0 %	125.0 %
h_abstract	75.0 %	125.0 %
h_AdditionalInfo	75.0 %	125.0 %
h_concrete	75.0 %	125.0 %
h_Details	75.0 %	125.0 %
h_Exercise	75.0 %	125.0 %
h_Exploration	75.0 %	125.0 %
h_Image	75.0 %	125.0 %
h_Introduction	75.0 %	125.0 %
h_Objectives	75.0 %	125.0 %

Figure 20. Analysis tool – configuration options:
a) Modify weights; b) Modify thresholds

Finally, when the teacher/researcher selects the "Compute learner preferences" option, the rules for computing ULSM preferences are applied on the currently available student data. The results are displayed in Fig. 21.



Compute pattern values
Compute learner preferences
Visualize learner preferences
Compute aggregated data
Configure weights
Configure thresholds

Welcome to Welsa Analysis tool!

AI course

Student	ULSM dimension	Preference
Student1	p_visual/p_verbal	p_visual
Student1	p_abstract/p_concrete	p_concrete
Student1	p_serial/p_holistic	p_serial
Student1	p_individual/p_team	p_individual
Student1	p_carefulDetails/p_notCarefulDetails	p_carefulDetails
Student1	p_activeExperimentation/p_reflectiveObservation	p_activeExperimentation
Student2	p_visual/p_verbal	p_visual
Student2	p_serial/p_holistic	p_holistic
Student2	p_abstract/p_concrete	p_abstract

Figure 21. A snapshot of the Analysis tool – visualizing the students' ULSM preferences

6. WELSA Adaptation Component

The adaptation component consists of a Java servlet which automatically generates the individualized web page, each time an HTTP request is received by the server, as illustrated in Fig. 22.

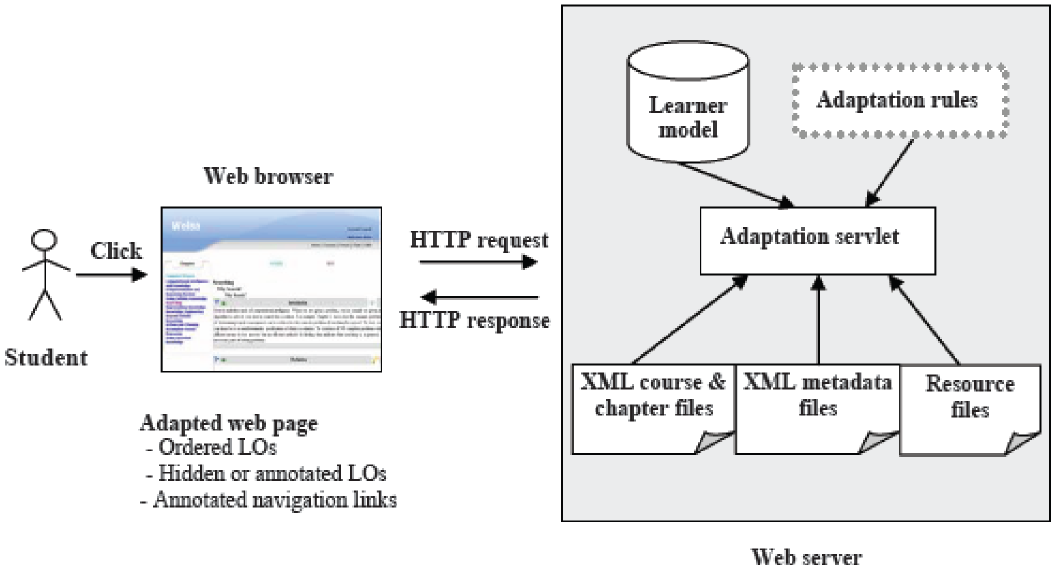


Figure 22. Adaptation component schematic architecture

The adaptation servlet queries the learner model database, in order to find the ULSM preferences of the current student. Based on these preferences, the servlet applies the corresponding adaptation rules and generates the new HTML page. These adaptation rules involve the use of LO metadata, which as already stated in section 2, are independent of any learning style. However, they convey enough information to allow for the adaptation decision making (i.e. they include essential information related to the media type, the level of abstractness, the instructional role etc). Next the web page is composed from the selected and ordered LOs, each with its own status (highlighted, dimmed or standard). This dynamic generation process is illustrated in Fig. 23 and Fig. 24, for two learners with different ULSM preferences. The relationship between various ULSM dimensions and the adaptive features of the system is thus highlighted.

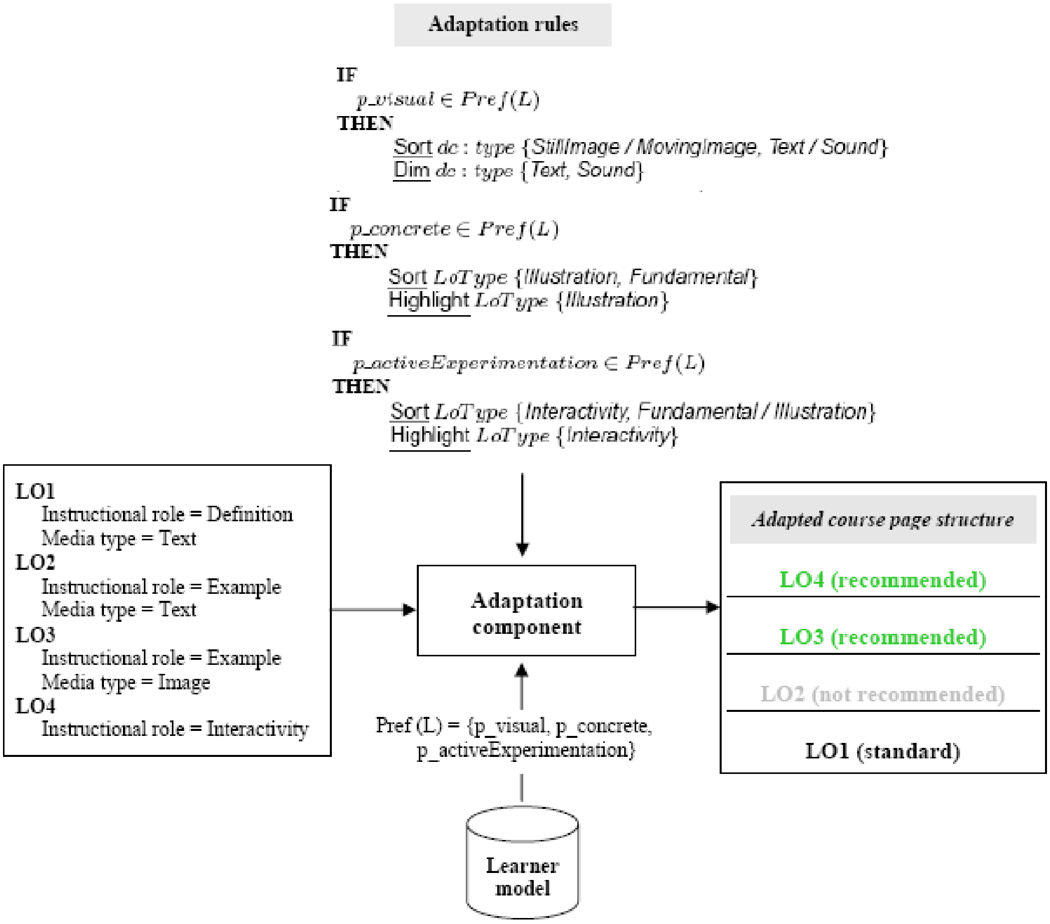


Figure 23. Automatic generation of an adapted course page for a student with preferences towards visual perception modality, concrete examples and active experimentation

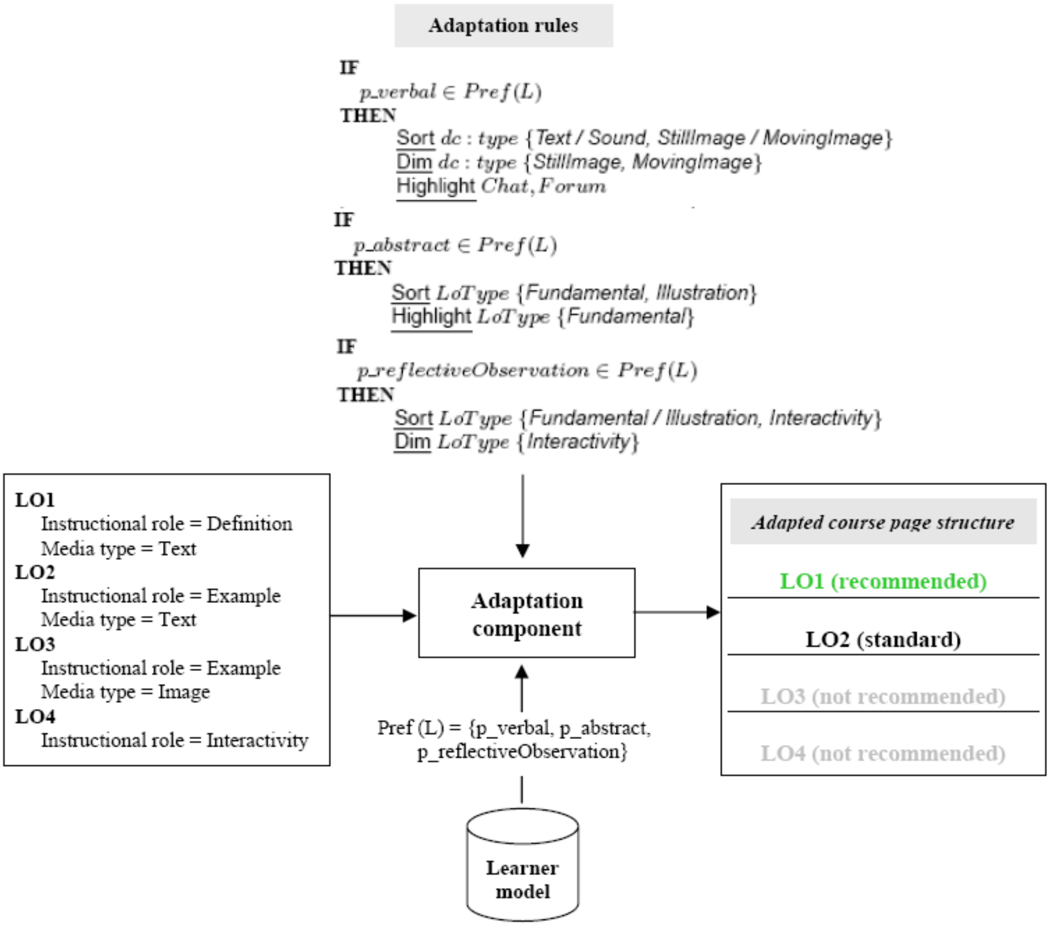


Figure 24. Automatic generation of an adapted course page for a student with preferences towards verbal perception modality, abstract examples and reflective observation

Next we add a concrete example from WELSA to illustrate the adaptation mechanism (see Fig. 25).

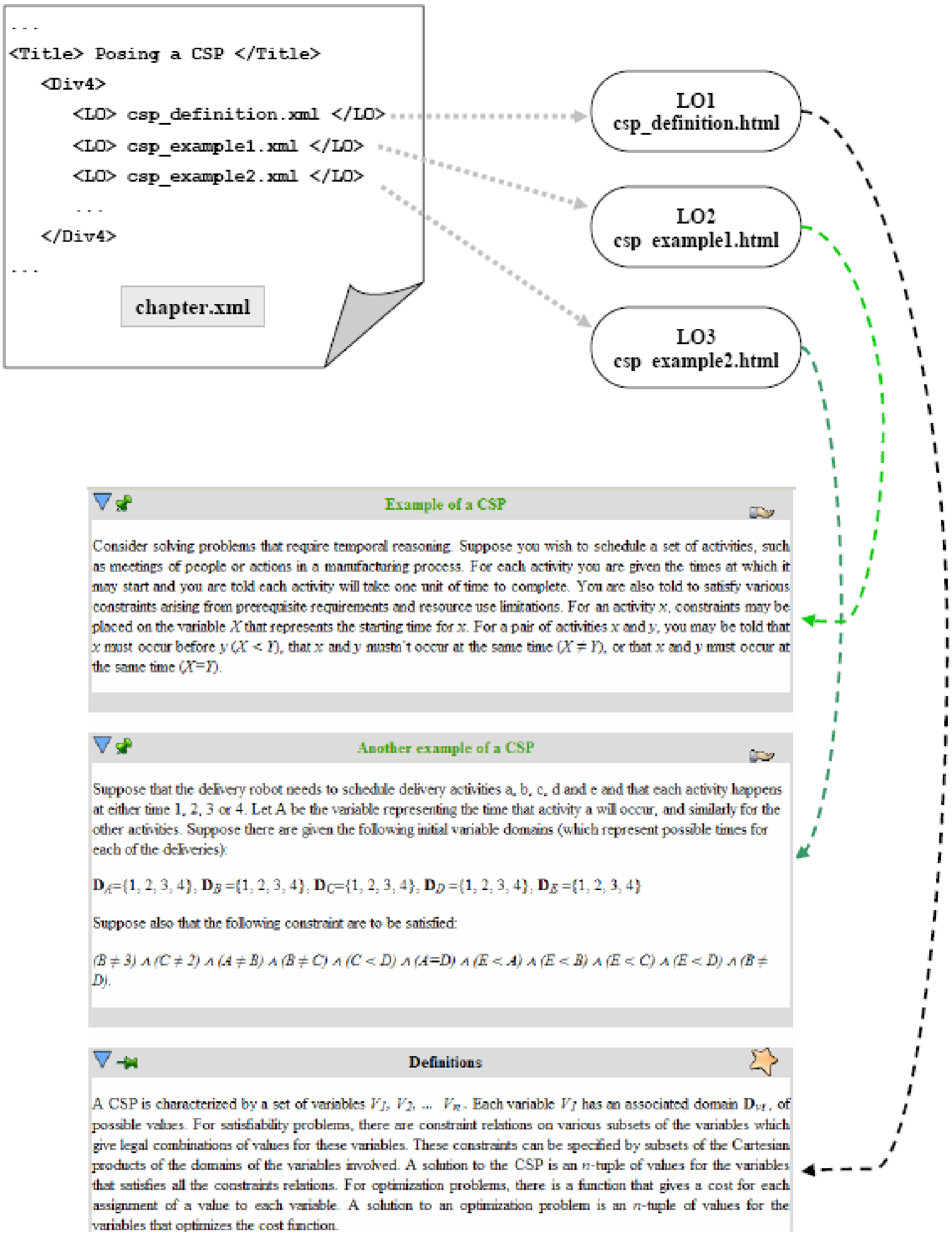


Figure 25. Output of WELSA adaptation component for a student with *Concrete* preference

7. An Artificial Intelligence Course in WELSA

In order to validate our approach, we implemented a course module in the domain of Artificial Intelligence, based on the chapter dedicated to search strategies and solving problems by search, from the classic textbook of Poole, Mackworth and Goebel (1998). The module consists of 4 sections and 9 subsections, including a total of 46 LOs. The distribution of LOs from the point of view of media type and instructional role is summarized in Table 1.

LoType1="Fundamental"	12	LoType1="Auxiliary"	34	dc:type="Text"	35
LoType2="Definition"	5	LoType3="AdditionalInfo"	4	dc:type="StillImage"	1
LoType3="Policy"	7	LoType3="Demonstration"	1	dc:type="MovingImage"	7
		LoType3="Example"	14	dc:type="InteractiveResource"	3
		LoType3="Exercise"	5		
		LoType3="Exploration"	3		
		LoType3="Introduction"	5		
		LoType3="Objectives"	1		
		LoType3="Remark"	1		

Table 1. Number of LOs composing the "Searching" chapter

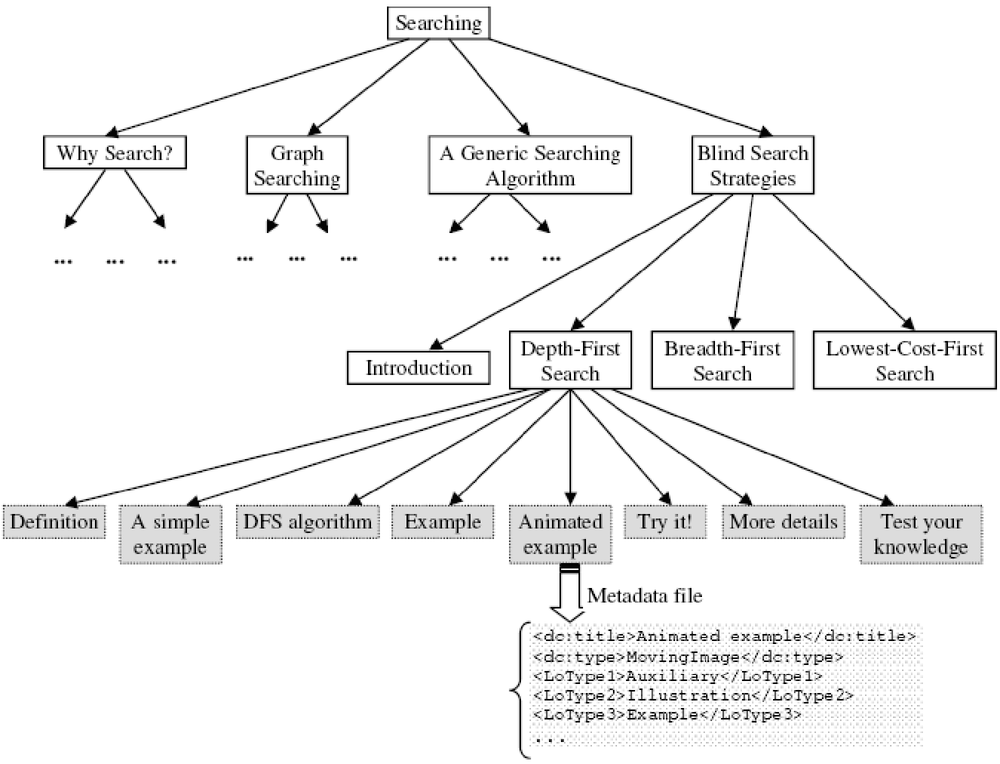


Figure 26. AI chapter hierarchical organization (white boxes designate sections and subsections, while grey boxes designate LOs)

The structure of the course chapter is illustrated in Fig. 26, with a focus on the "Depth-First Search Strategy" subsection.

The corresponding course, chapter and metadata files are included in Fig. 27, 28 and 29 respectively. The XML files follow the structure described in section 2.

course.xml


```xml
<?xml version="1.0" encoding="UTF-8"?>
<course xmlns:p="http://purl.org/dc/elements/1.1/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="course.xsd"
        xsi:schemaLocation="http://purl.org/dc/elements/1.1/
        http://dublincore.org/schemas/xmls/qdc/2006/01/06/dc.xsd ">
  <About>
    <title>Artificial Intelligence</title>
    <identifier>CS104</identifier>
    <creator>Elvira Popescu</creator>
    <date>12-01-2008</date>
    <subject>Artificial Intelligence</subject>
    <description>This is an introductory course on AI</description>
    <language>en</language>
    <source>Computational Intelligence - D. Poole, A. Mackworth, R. Goebel</source>
    ...
  </About>
  <Content>
    <Chapter number="1">Computational Intelligence and Knowledge</Chapter>
    <Chapter number="2">A Representation and Reasoning System</Chapter>
    <Chapter number="3">Using Definite Knowledge</Chapter>
    <Chapter number="4">Searching</Chapter>
    <Chapter number="5">Representing Knowledge</Chapter>
    <Chapter number="6">Knowledge Engineering</Chapter>
    <Chapter number="7">Beyond Definite Knowledge</Chapter>
    <Chapter number="8">Actions and Planning</Chapter>
    <Chapter number="9">Assumption-Based Reasoning</Chapter>
    <Chapter number="10">Using Uncertain Knowledge</Chapter>
  </Content>
</course>
```


Figure 27. The XML course file for the Artificial Intelligence course, conforming to the WELSA internal format

chapter.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Chapter xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="chapter.xsd"
    xsi:schemaLocation="http://purl.org/dc/elements/1.1/
    http://dublincore.org/schemas/xmls/qdc/2006/01/06/dc.xsd ">
    <About>
        <title>Searching</title>
        <creator>Elvira Popescu</creator>
        ...
    </About>
    <Content>
        <Div1>
            <Title>Why Search?</Title>
            <Div2>
                <Title></Title>
                <Div3>
                    <Title>Why Search?</Title>
                    <Div4>
                        <Title></Title>
                        <LO>search_introduction.xml </LO>
                        <LO>search_def.xml </LO>
                        <LO>search_objectives.xml </LO>
                    </Div4>
                </Div3>
            </Div2>
        </Div1>
        <Div1>
            <Title>Graph Searching</Title>
            <Div2>
                <Title></Title>
                <Div3>
                    <Title>Graph Searching</Title>
                    <Div4>
                        <Title></Title>
                        <LO>graph_search_introduction.xml</LO>
                        <LO>graph_search_example.xml</LO>
                    </Div4>
                </Div3>
                <Div3>
                    <Title>Formalizing Graph Searching</Title>
                    <Div4>
                        <Title></Title>
                        <LO>fgraph_search_definition.xml</LO>
                        <LO>fgraph_search_example.xml</LO>
                        <LO>test_graph_searching.xml</LO>
                    </Div4>
                </Div3>
            </Div2>
        </Div1>
        <Div1>
            <Title>A Generic Searching Algorithm</Title>
            <Div2>
                <Title></Title>
                <Div3>
                    <Title>A Generic Searching Algorithm</Title>
                    <Div4>
                        <Title></Title>
                        <LO>generic_search_introduction.xml</LO>
                        <LO>generic_search_example1.xml</LO>
                        <LO>generic_search_procedure1.xml</LO>
                        <LO>generic_search_example2.xml</LO>
                        <LO>generic_search_demonstration.xml</LO>
                        <LO>generic_search_addinf.xml</LO>
                        <LO>generic_search_costs.xml</LO>
                    </Div4>
                </Div3>
                <Div3>
                    <Title>Finding Paths</Title>
```

```
                    <Div4>
                        <LO>generic_search_paths_introd.xml</LO>
                        <LO>generic_search_paths_procedure.xml</LO>
                        <LO>generic_search_paths_example.xml</LO>
                        <LO>test_generic_search.xml</LO>
                    </Div4>
                </Div3>
            </Div2>
        </Div1>
        <Div1>
            <Title>Blind Search Strategies</Title>
            <Div2>
                <Title></Title>
                <Div3>
                    <Title>Blind Search Strategies</Title>
                    <Div4>
                        <LO> blind_strategies_introduction.xml</LO>
                        <LO> tutorial_text.xml</LO>
                        <LO> tutorial_video.xml</LO>
                    </Div4>
                </Div3>
                <Div3>
                    <Title>Depth-First Search</Title>
                    <Div4>
                        <LO> depth_first_definition.xml</LO>
                        <LO> depth_first_simple_example.xml</LO>
                        <LO> depth_first_procedure.xml</LO>
                        <LO> depth_first_example_txt.xml</LO>
                        <LO> depth_first_example_anim.xml</LO>
                        <LO> depth_first_example_sim.xml</LO>
                        <LO> depth_first_addinf.xml</LO>
                        <LO> test_DFS.xml</LO>
                    </Div4>
                </Div3>
                <Div3>
                    <Title>Breadth-First Search</Title>
                    <Div4>
                        <LO> breadth_first_definition.xml</LO>
                        <LO> breadth_first_simple_example.xml</LO>
                        <LO> breadth_first_procedure.xml</LO>
                        <LO> breadth_first_example_txt.xml</LO>
                        <LO> breadth_first_example_anim.xml</LO>
                        <LO> breadth_first_example_sim.xml</LO>
                        <LO> breadth_first_addinf.xml</LO>
                        <LO> test_BFS.xml</LO>
                    </Div4>
                </Div3>
                <Div3>
                    <Title>Lowest-Cost-First Search</Title>
                    <Div4>
                        <LO> lcost_first_definition.xml</LO>
                        <LO> lcost_first_simple_example.xml</LO>
                        <LO> lcost_first_procedure.xml</LO>
                        <LO> lcost_first_example_txt.xml</LO>
                        <LO> lcost_first_example_anim.xml</LO>
                        <LO> lcost_first_example_sim.xml</LO>
                        <LO> lcost_first_addinf.xml</LO>
                        <LO> test_LCFS.xml</LO>
                    </Div4>
                </Div3>
            </Div2>
        </Div1>
    </Content>
</Chapter>
```

Figure 28. The XML chapter file for the "Searching" chapter

depth_first_definition.xml


```
<?xml version="1.0" encoding="UTF-8"?>
<LO xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="metadata.xsd"
    xsi:schemaLocation="http://purl.org/dc/elements/1.1/
    http://dublincore.org/schemas/xmls/qdc/2006/01/06/dc.xsd ">
    <title>Definition</title>
    <identifier>depth_first_definition.html</identifier>
    <type>text</type>
    <format>text/html</format>
    ...
    <LoType1>Fundamental</LoType1>
    <LoType2>Definition</LoType2>
    <hasAbstractness>neutral</hasAbstractness>
    <hasFormalness>informal</hasFormalness>
</LO>
```


Figure 29. The XML metadata file for the "Depth-First Definition" LO

Initially, only the first LO on each page is expanded, the rest being shown in a strechtext format, including only the resource title and some visual cues such as icons for the instructional role and the media type. However, the student has the possibility to expand any LOs on the page and "lock" them in the expanded format. She/he can thus choose between having several LOs available at the same time or concentrating on only one LO at a time. The course also includes access to two communication tools, one synchronous (chat) and one asynchronous (forum) and offers two navigation choices - either by means of the *Next* and *Previous* buttons, or by means of the *Outline*.

Figure 30 shows a part of the "Depth-First Search Strategy" subsection, as it is visualized by the end-user (the student), including one LO with *LOType*2 = *"Definition"* and *dc:type = "Text"* and one LO with *LOType3 = "Example"* and *dc:type = "MovingImage"*, both in an expanded state.

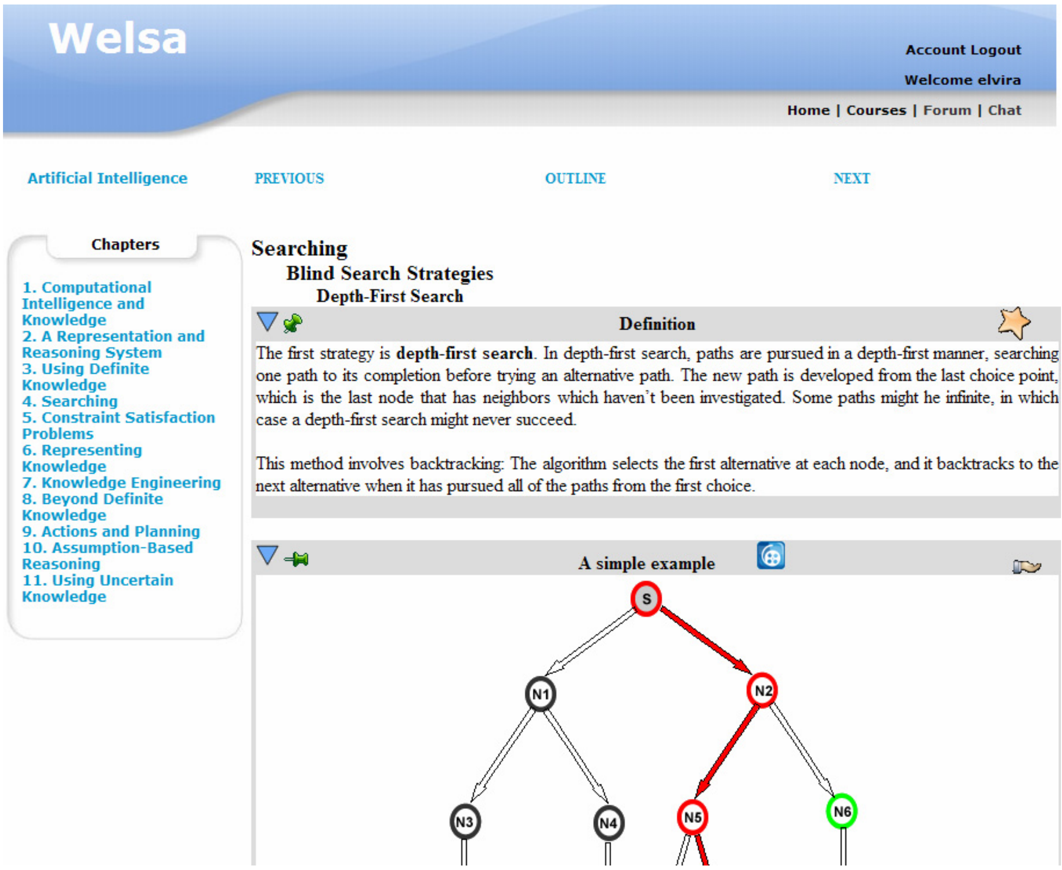


Figure 30. "Depth-First Search Strategy" subsection

The course was conceived so that a concept is illustrated by various multimedia objects, in order to accommodate different modality perception preferences. For example, the Breadth First Search strategy is explained in textual form (see Fig. 31a), as an animated image (see Fig. 31b) and by means of an interactive simulation (Fig. 31c).

a)

Breadth first search algorithm

To implement breadth-first search you can represent the queue as a list such that the first element of the list is the earliest element of the list. The neighbors are added to the end of the list instead of the beginning as in depth-first search:

```
select(Node, [Node|Frontier], Frontier).
add_to_frontier(Neighbors, Frontier1, Frontier2)←
    append (Frontier1, Neighbors, Frontier2).
```

Notice how we have merely changed the order of arguments to *append* to transform depth-first search into breadth-first search.

Example

Consider breadth-first search on the graph for the delivery robot domain. Initially the frontier is *[o103]*. This is replaced by its neighbors, making the frontier *[ts, l2d3, o109]*. These are the nodes one step away from *o013*. The next three elements of the frontier chosen are *ts*, *12d3* and *o109*, at which stage the frontier contains the neighbors of *ts* followed by the neighbors of *12d3* followed by the neighbors of *o109*, namely

[mail, l2d1, 12d4, o111, o119].

These are the nodes that are two steps away from *o103*. These five nodes are the next elements of the frontier chosen, at which stage the frontier contains the elements three steps away from *o103*, namely

[13d2, 12d2, o109, storage, o123].

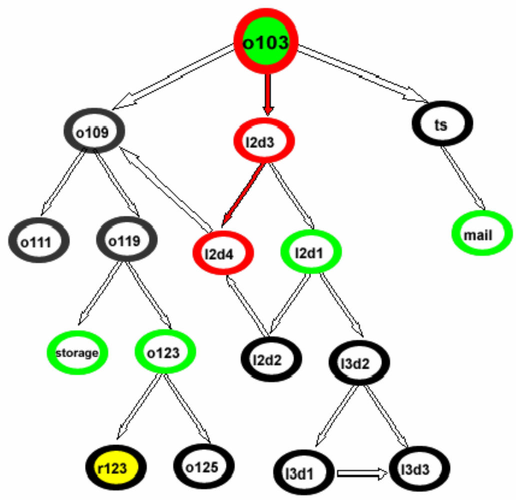
The same frontier for the path-finding search algorithm is

[node(13d2, [12d1, 12d3, o103], 11),
node(12d2, [12d1, 12d3, o103], 14),
node(o109, [12d4, 12d3, o103], 18),
node(storage, [ol19, o109, o103], 35),
node(ol23, [o119, o109, o103], 37)].

Note how each of the paths on the frontier has the same number of steps. For breadth-first search the number of steps in the paths on the frontier always differ by at most one.

b)

Animated example



c)

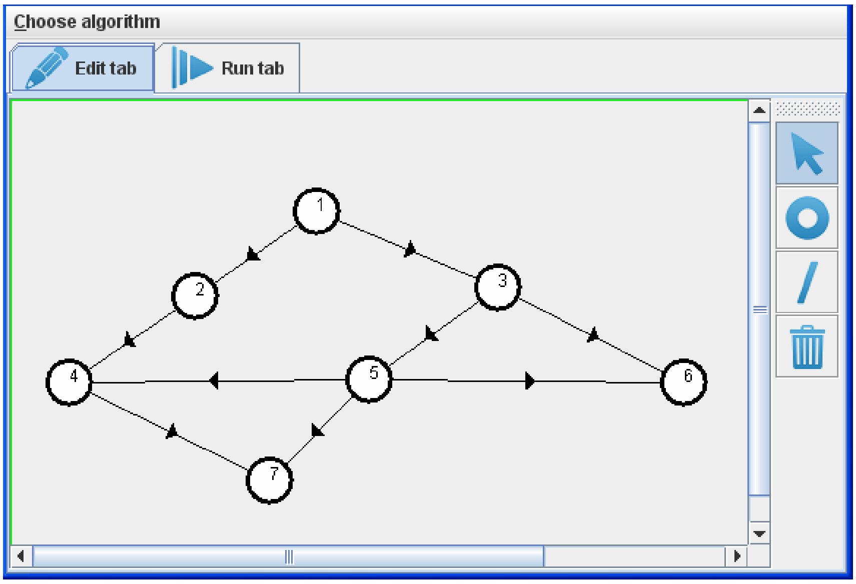


Figure 31. Learning objects illustrating Breadth First Search strategy:
a) *dc:type = Text*, b) *dc:type = MovingImage* c) *dc:type = InteractiveResource*

A similar course module was implemented for the adaptive session. The module deals with Constraint Satisfaction Problems and is based on the same textbook of Poole, Mackworth and Goebel (1998). Figure 32 shows a part of the "Consistency Algorithms" subsection, as it is visualized by the student, including one LO with *LOType*3 = *"Introduction"* and one with *LOType2* = *"Definition"*, both having *dc : type* = *"Text"*.

Figure 32. "Consistency algorithms" subsection

8. System Validation

The final step of our research was the global evaluation of WELSA system. In order to assess the validity and effectiveness of our system, we used the empirical evaluation approach, involving two experiments with undergraduate students. After interacting with WELSA for a course session, the students were asked to fill in some questionnaires, stating their opinions about the system. In what follows we will discuss and analyze the students' answers to those questionnaire items that deal with the WELSA system as a whole and its value as an educational platform.

After the first course session (non-adaptive version), the 71 students who actively participated in the experiment were asked to evaluate various aspects of their learning experience with WELSA system, on a 1 to 10 scale. Thus they had to assess the course content, the presentation, the platform interface, the navigation options, the expand/collapse functionality for the resources, the communication tools and the course as a whole. The results are presented in Fig. 33.

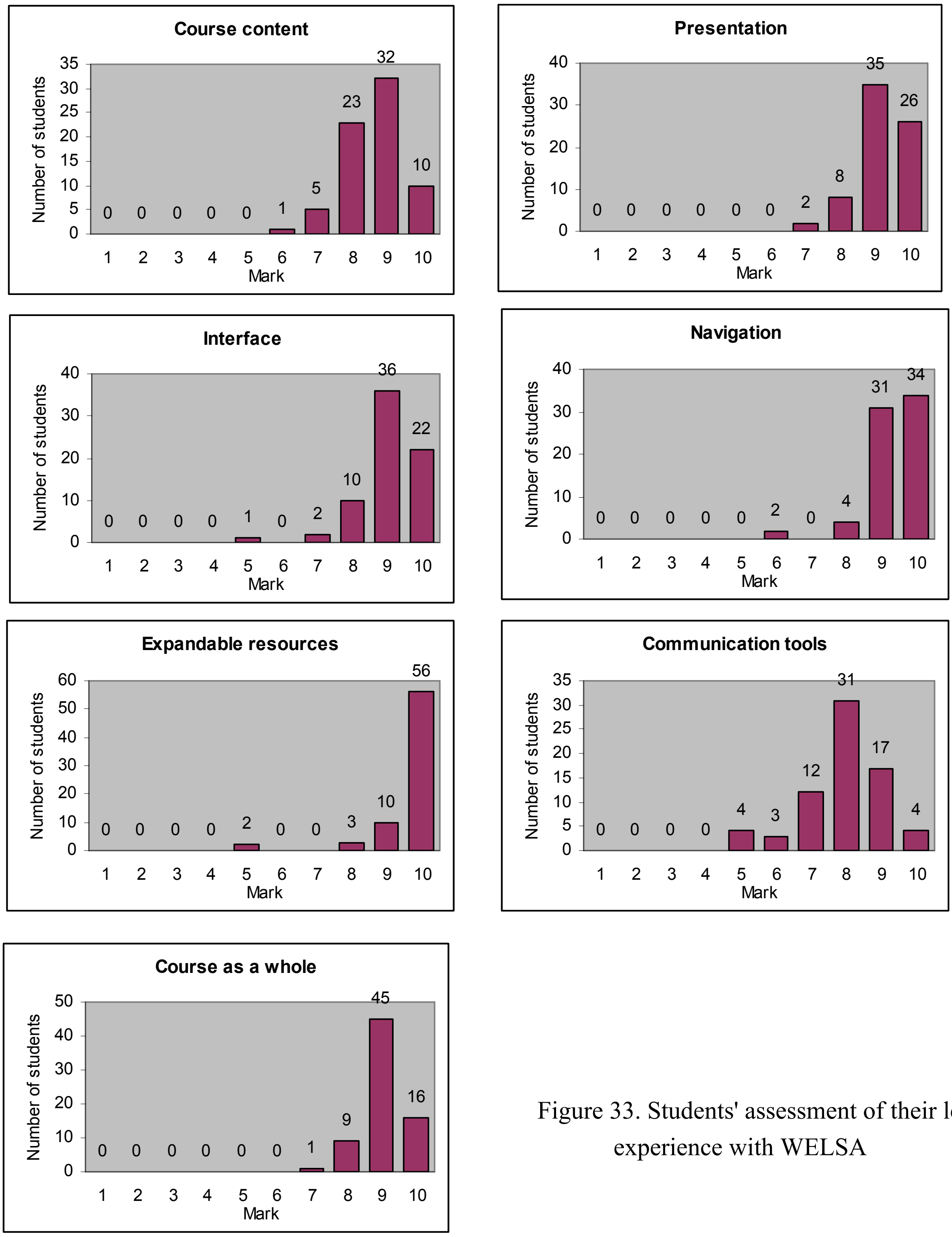


Figure 33. Students' assessment of their learning experience with WELSA

As we can see from Fig. 33, the students' evaluation of the AI course and WELSA platform is very positive. 59.15% of the students assessed the course content as very good (marks 9-10), 39.44% as good (marks 7-8) and only one student as average. The criticism points were contradictory, some students claiming that the course contained too much theory, while others argued that it should include more details and theoretical aspects. Hence the need for providing individualized courses, in order to respond to the various students' preferences. Furthermore, some students pointed out that there was redundant information ("many examples, in different forms, but illustrating the same thing – it was a bit annoying"). As explained in the previous section, this redundancy was introduced on purpose, in order to offer students the possibility to choose the preferred representation modality. However, the fact that learners considered this as distracting shows once again the necessity of filtering out unnecessary information. Thus providing a variety of resources is not necessarily beneficial, increasing the cognitive overload of the students.

As far as the presentation is concerned, the majority of students (85.92%) found it very enjoyable, ("it was very attractive due to the multitude of animations, images and simulations"), while the rest of 14.08% were also quite pleased with it. Students declared themselves equally satisfied with the course interface, 81.69% of them assigning it marks 9 and 10, 16.90% marks 7 and 8, and only one student describing it as "boring".

Students also appreciated positively the navigation features offered by the system, 91.55% of them giving very high marks (9-10). They were mainly attracted by the course outline, which they considered "a very good idea". The highest marks were obtained by the expandable resource feature (with an average of 9.63) which was appreciated as "original and very useful", "interesting because it allows having both a global view of the course and concentrating on only one fragment". The lowest marks were obtained by the communication tools: the chat was described as "too basic" and the need for more advanced communication tools (audio / video conference, whiteboard) was outlined. Consequently the average mark was only 7.93.

The course as a whole received marks 9-10 from 85.92% of the students, the rest evaluating it as good (marks 7-8).

All in all, very good marks were assigned to most of the features, with only one feature (the communication tools) receiving lower (but still satisfactory) marks. We can therefore conclude that students had a very positive learning experience with WELSA.

We should also mention here WELSA's support for self-regulated learning (SRL). SRL is an important concept in education, being introduced by Zimmerman and Schunk (1989) and subsequently expanded in (Schunk and Zimmerman, 1998; Boekaerts et al., 2000; Perry et al., 2006; Steffens, 2006).

According to (TELEPEERS, 2008), SRL refers to "a set of cross-curricular skills which allow learners to make the most of their learning by being aware of and monitoring the cognitive, motivational, emotional, volitional and social aspects of their learning activities". Indeed, by using WELSA, students become more cognizant of their learning styles and preferences, which helps them more appropriately tackle learning tasks. 73.44% of the 64 students who participated in both experiment sessions reported a substantial increase in their awareness regarding their own strengths and weaknesses in the learning process, as compared to only 10.94% of students who reportedly possessed this self-knowledge before the experiment.

Another important aspect that was evaluated through the questionnaires was the privacy issue: identifying student learning preferences implies the collection of usage data from the students. Learners' willingness to accept the monitoring of their interaction with the system on an everyday basis in exchange for a personalized learning experience was predominant, as can be seen in Fig. 34. Thus, 32.39% of the students agreed with the collection of their data in any conditions, 63.38% agreed as long as the data were analyzed in an anonymous fashion and only 4.23% didn't like the idea of their actions being recorded. This is a further proof of the students' need for individualized learning.

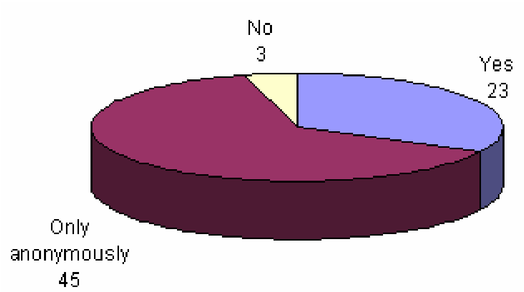


Figure 34. Students' opinion on privacy issue: "Would you agree with having your interaction with the system monitored and analyzed?"

The main goal of WELSA system is the provisioning of an adaptive learning experience. Therefore evaluating the adaptive version of the system is of a particular interest. Comparisons between the adaptive and non-adaptive versions as well as between matched and mismatched learners are outside the scope of this report. Here we are interested in the overall student satisfaction and the desire to use the WELSA system on an everyday basis. We will therefore take into account for our analysis only the 32 students who took part in the matched adaptive course session. The results are summarized in Fig. 35 and 36.

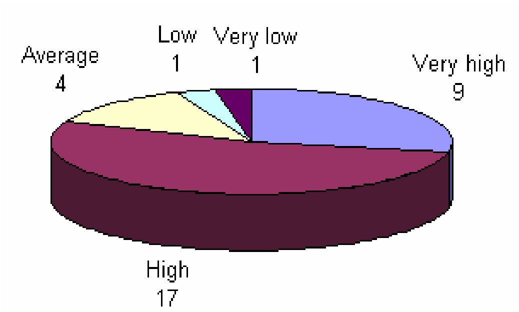


Figure 35. Students' overall satisfaction with the adaptive version of the WELSA system

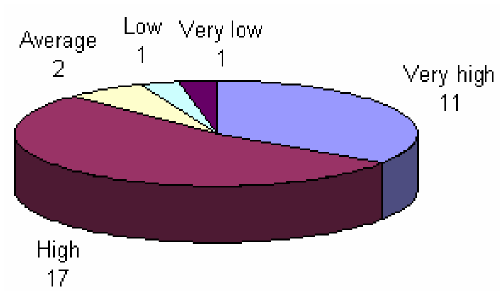


Figure 36. Students' willingness to adopt WELSA system for everyday use

As can be seen from the figures, the large majority of the students (81.25%) reported a high or very high degree of satisfaction with WELSA and only 6.25% a low or very low degree. These findings are reflected also in the readiness of the students to adopt WELSA system for large scale use with 87.50% willing to do so and only 6.25% reluctant.

The level of satisfaction offered by the adapted system should be corroborated with the level of importance students attribute to learning style adaptation. Indeed, an educational platform is effective only when the features it offers are both valuable and satisfactory for the learners (Levy, 2006). We therefore asked the students to assess the importance they grant to having the courses adapted to their learning styles. The results are summarized in Fig. 37, showing a large majority of the students (90.63%) who perceive learning style adaptation as highly important.

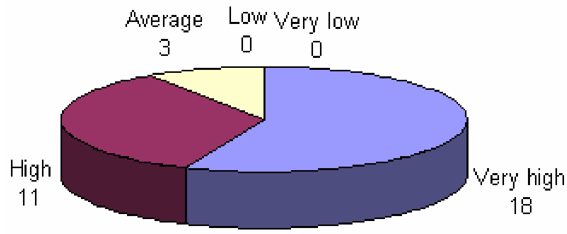


Figure 37. Students' perceived importance of learning style adaptation

We can conclude that the overall results of the two experimental studies proved the validity and effectiveness of WELSA system. The analysis of students' answers to the survey instruments supports this claim, revealing the high degree of learner satisfaction with the system.

References

H. S. Al-Khalifa, H. C. Davis (2006). The Evolution of Metadata from Standards to Semantics in E-Learning Applications. *Proceedings of Hypertext'06*, Odense, Denmark.

L. Anido, M. Fernandez, M. Caeiro, J. Santos, J. Rodriguez, M. Llamas (2002). Educational Metadata and Brokerage for Learning Resources. *Computers and Education*, Elsevier, 38, pp. 351-374.

P. Barker (2005). What is IEEE Learning Object Metadata (the LOM) / IMS Learning Resource Meta-data?. CETIS, http://metadata.cetis.ac.uk/guides/WhatIsLOM.pdf

M. Boekaerts, P. R. Pintrich, M. Zeidner (2000). *Handbook of Self-regulation*. San Diego, CA: Academic Press.

E. Brown, A. Cristea, C. Stewart, T. Brailsford (2005). Patterns in Authoring of Adaptive Educational Hypermedia: A Taxonomy of Learning Styles. *Educational Technology and Society*, 8, pp. 77–90.

A. Cristea (2003). Adaptive Patterns in Authoring of Educational Adaptive Hypermedia. *Educational Technology & Society*, 6 (4), pp. 1-5.

L. Curry (1987). *Integrating Concepts of Cognitive or Learning Style: A Review with Attention to Psychometric Standards*. Ottawa, ON: Canadian College of Health Service Executives.

V. Devedzic (2006). *Semantic Web and Education*, Springer.

P. Dolog, N. Henze, W. Nejdl, M. Sintek (2004). Personalization in Distributed eLearning Environments. *Proc. Intl. Conf. WWW2004.*

P. Dolog, W. Nejdl (2007). Semantic Web Technologies for the Adaptive Web. In P. Brusilovsky, A. Kobsa and W. Neidl (eds.): *The Adaptive Web: Methods and Strategies of Web Personalization*. Lecture Notes in Computer Science, Vol. 4321, Springer, pp. 697-719.

R. M. Felder, L. K. Silverman (1988). Learning and Teaching Styles in Engineering Education, *Engineering Education*, 78 (7). Preceded by a preface in 2002: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/LS-1988.pdf

J. Gascuena, A. Fernandez-Caballero, P. Gonzalez (2006). Domain Ontology for Personalized E-Learning in Educational Systems. *Proc. 6th IEEE Intl. Conf. on Advanced Learning Technologies - ICALT 2006*, IEEE Computer Society Press.

G. Geser (2007). *OLCOS Roadmap 2012 - Open Educational Practices and Resources*, Austria.

Y. Levy (2006). *Assessing the Value of E-Learning Systems*. Information Science Publishing.

N.E. Perry, L. Phillips, L. R. Hutchinson (2006). Preparing Student Teachers to Support for Self-Regulated Learning. *Elementary School Journal, 106*, pp. 237-254.

D. Poole, A. Mackworth, R. Goebel (1998). *Computational Intelligence: A Logical Approach*, Oxford University Press.

E. Popescu, C. Badica, P. Trigano (2008a). Learning Objects' Architecture and Indexing in WELSA Adaptive Educational System. *Scalable Computing: Practice and Experience*, Vol. 9(1), pp. 11-20.

E. Popescu, C. Badica, P. Trigano (2008f). Description and Organization of Instructional Resources in an Adaptive Educational System Focused on Learning Styles. *Advances in Intelligent and Distributed Computing, Studies in Computational Intelligence Series*, Vol. 78, Springer, pp. 177-186.

H. Shi, O. Rodriguez, S. Chen, Y. Shang (2004). Open Learning Objects as an Intelligent Way of Organizing Educational Material. *International Journal on E-Learning*, 3(2), pp. 51–63.
D. H. Schunk, B. J. Zimmerman (1998). *Self-Regulated Learning: From Teaching to Self-Reflective Practice*. New York: Guilford.

N. Stash (2007). *Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System*. PhD Thesis, Eindhoven University of Technology, Netherlands.

K. Steffens (2006). Self-Regulated Learning in Technology Enhanced Learning Environments: Lessons of a European Peer Review. *European Journal of Education*, vol. 41, no. 3-4, pp. 353-379.

C. Ullrich (2005). The Learning-Resource-Type is Dead, Long Live the Learning-Resource-Type!. *Learning Objects and Learning Designs*, 1, pp. 7–15.

B. J. Zimmerman, D. H. Schunk (1989). *Self-Regulated Learning and Academic Achievement: Theory, Research, and Practice*. New York: Springer.

Web references

ADL SCORM (2008). http://www.adlnet.gov/scorm/index.aspx.

Ajax (2008). Asynchronous JavaScript and XML.
http://java.sun.com/developer/technicalArticles/J2EE/AJAX

DCMI (2008). Dublin Core Metadata Initiative. http://dublincore.org

IEEE LOM (2008). http://ltsc.ieee.org/wg1.

IMS Global Learning Consortium (2008). http://imsglobal.org.

IMS MD (2008). http://www.imsglobal.org/metadata/index.html.

JSP (2008). JavaServer Pages Technology. http://java.sun.com/products/jsp.

MySQL (2008). http://www.mysql.com.

Oxygen (2008). Oxygen XML Editor. http://www.oxygenxml.com.

TELEPEERS (2008). http://www.hds.utc.fr/telepeers.

Tomcat (2008). Apache Tomcat. http://tomcat.apache.org.

XML DOM (2008). http://www.w3.org/DOM