An autonomous component architecture to develop WWW-ITS

Trella, M., Conejo, R., Bueno, D., Guzmán, E.

Departamento de Lenguajes y Ciencias de la Computación
Escuela Técnica Superior de Ingenieros en Informática.
Universidad de Málaga
Campus de Teatinos
Boulevard Louis Pasteur, s/n. 29071
Málaga. SPAIN
Telephone: +34 952137152
Fax: +34 952131397
{trella, conejo, bueno, guzman}@lcc.uma.es

Abstract. In this paper we describe an on-going research work (MEDEA project) whose final goal is to develop a general framework to build open ITS. We understand "open system" as a set of autonomous educational modules that communicate between themselves following high-level pre-established protocols. Each of those modules can be an intelligent component with its own instruction strategies or a support tool that leaves all the adaptive capabilities to the ITS instructor core. The architecture opens up the possibility to include new components, provided that they are able to interact within the general framework.

1 Introduction

The Web has become a great source of information and resources that are available for tutoring. The advantages of adaptive web based tutoring are well known [1]. The intrinsic structure of the web is distributed, and in most cases it is used just as a huge repository of unorganized information. Teachers collect different URLs to give their students links where they can find lecture notes, exercises, simulation programs, etc. related to the matter that they are studying. Commonly, the systems that they use have been developed for different objectives, and have been designed by different people with different approaches. Some of them might be adaptive but generally non-adaptive materials are found. Some attempts have been made to integrate two preexisting web based adaptive systems, (see for instance [2]). This paper presents a proposal of a framework for systems integration.

One of the main task of a human teacher when using the web as a teaching resource is to redirect the students to those systems that are more appropriate

according to the student profile and supervise their progress. If the web pages are fixed, an intelligent behavior is still needed from the teacher, suggesting the students to skip some parts, conducting them through pages with exercises, simulators, etc, according to the specific goal of the course. On the other hand, if the web systems used have some kind of adaptive features or intelligent behavior, it should be advisable to take advantage of these features.

Following this idea, a web ITS might be constructed by gathering pages or systems, like *building blocks* in the sense of Chandrasekaran [3], some of them might be new, and some others reused from existing material. Most technologies used for Web ITS development are designed for small systems and they do not have the necessary resources to guarantee a high-level conceptual organization. This kind of problems has been traditionally treated with software engineering techniques, and more recently with knowledge engineering techniques, following the Newell's *knowledge level* paradigm [4]. Using this idea, high-level methodologies and tools for the intelligent systems development have been proposed as KADS [5], PROTÉGÉ [2], KSM [3], etc.

In this paper we describe the MEDEA system. MEDEA is a Spanish acronym of *Methodology and Tools for the Development of Intelligent Environments of Teaching and Learning*. MEDEA is not a new ITS authoring tool but a general framework for open systems development and integration [8] [9]. We understand "open system" as a set of autonomous educational components that communicate between themselves following high-level pre-established protocols. In the case of web-based open tutorial systems, this protocol is the well known HTTP. MEDEA also adds a new high level layer to communicate intelligent tutorial components, that is converted to HTTP and might be interpreted (or not) by the certain educational components.

The components used in MEDEA can be intelligent, with its own instruction strategies, or not, leaving all the adaptive capabilities to the ITS instructor core. The architecture opens up the possibility to include general-purpose components that are able to interact within the general framework. MEDEA and offers to the educators a generic environment to develop Web ITS without the limitations of a close set of utilities as in the most of the authoring systems [10].

2 MEDEA Architecture

The elements that compose MEDEA architecture can be classified in three main groups: those that contain knowledge (*knowledge modules*), those that use this knowledge for making adequate decisions along the instruction (*functional modules*) and those that serve to access and configure the system (*tools*). The base of MEDEA architecture is a core that plans the instruction based on a set of external tutorial components that are introduced in the system. The domain and pedagogical knowledge is distributed between the core of MEDEA that serves as a master index, and these components.

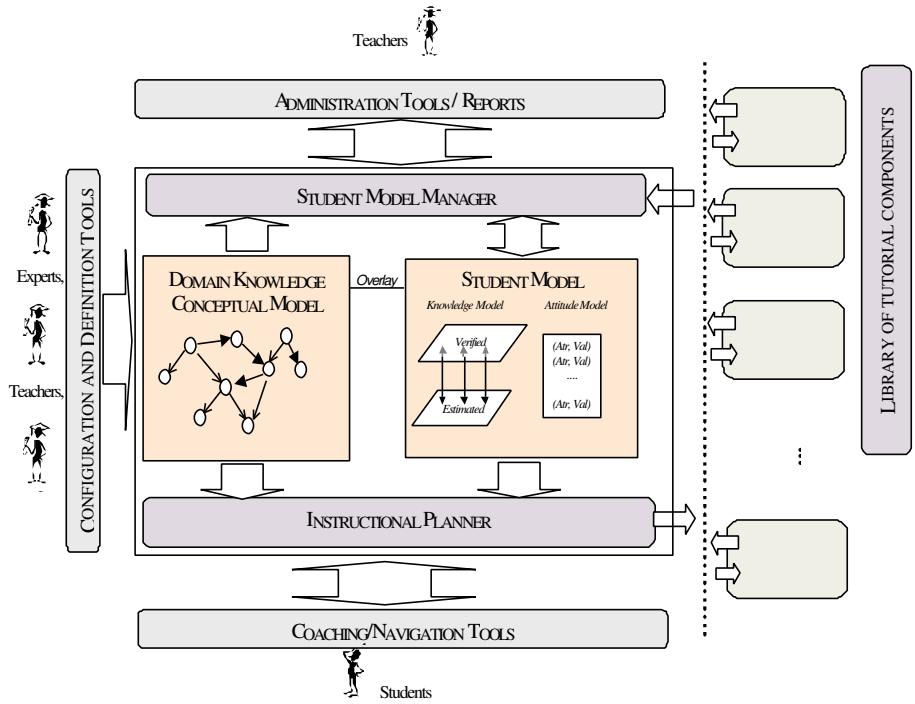

Fig. 1. MEDEA architecture

We could say, comparing the MEDEA architecture and the traditional ITS architecture, that *what* (knowledge about the learning subject) is explicitly represented by the domain model, *who* (knowledge about the student) is given by the student models (knowledge and attitudes) and *how* (teaching knowledge) is divided between the instructional planner, tutorial components and domain model. All knowledge models are represented in XML for computational use. Figure 1 shows the structure of MEDEA modules:

Knowledge modules:

? **Conceptual Knowledge Domain Model**. This module contains knowledge about the subject to be taught. Both domain concepts and relationships among them are represented. The domain model will be explained better in the next section.

? **Student model**. This module is decomposed in the *Knowledge Student Model*, and the *Attitudes Student Model*. The former contains information about the student's state of knowledge during the learning process. Not all the evidence generated during student's interaction with the system will be processed in the same way. If some kind of evaluation is performed, the results will be used to update the *verified student knowledge model*, and it will replace the *estimated*

student knowledge model. If there is no evaluation the system estimates the students knowledge increase and actualized the *estimated student knowledge model*. On the other hand, the *Attitude Student Knowledge* contains those features that describe the student profile but are not related to his current state of knowledge, like motivation, style of learning, speed of web connection, etc. The student model is described in section 4.

Functional modules:

? **Instructional planner**. This module will provide students with the necessary guidance during the learning process. It will design and compose the tutorial sessions, that is, it will decide in each moment the adequate task to be performed by the student. To this end, the information that will be used is a) conceptual domain knowledge, b) student state of knowledge, c) student profile and d) tutorial components definitions.

  The tutoring systems generated with our tool will allow students to freely navigate. The system only recommends but not imposes the next student action. The planner will also have the capabilities to justify the recommended actions, so the student has the necessary information before deciding to accept or not the suggestions made by the system.

? **Student Model Manager**. The function of this module is to create and update the student model. Once a tutorial component is executed, the student model is updated. Some tutorial components (for example a component that poses a test or an exercise) can evaluate the student knowledge about some domain concepts or subjects. The information provided by these components goes to the *Verified Student Knowledge Model.* MEDEA assumes the existence of other components that are not be able to determine exactly the variations of the student knowledge level, for example a component that display a text to be read or presents a simulation. There might be also components that do not even return any feedback to MEDEA, for example a static web page. MEDEA is just informed that the student has visited those components. This information is treated differently using the *Estimated Student Knowledge Model.*

? **The library of tutorial components**. They are external educational tools, each of which makes a concrete task (electronic books, simulation tools, exercises about the subject, making tests, etc.). From MEDEA point of view, the architecture of a component (Fig.2.) is composed by a *partial domain model*, a *development interface* to introduce contents, a *student interface*, a *temporal student model*, a *component functional description* (where the tasks of the teaching component and the way of communicating with MEDEA planner are defined) and a *control* that is the execution engine of the component.

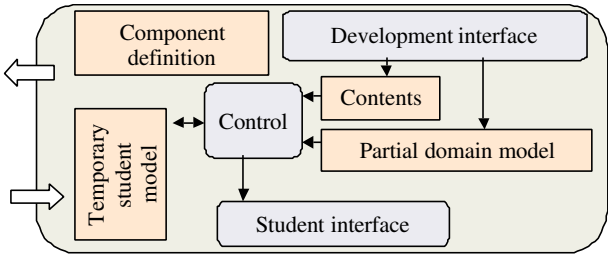

Fig. 2. Tutorial component architecture

Tools (Interfaces)

? **Configuration and definition tools**. This module will be used by domain experts, teachers and designers. They will be able to introduce the contents, define and configure the data and knowledge modules using specific interfaces.

? **Administration tools**. This module will be used by teachers to monitor the evolution of their students. It will show the progress of each student, the statistics about the course use, and average student performances, and other secretariat and administrative tasks.

? **Navigation tools**. This module is used by student to support their navigation and interaction with the whole system. It can be conceptualized as an advisor during the learning and designed as an additional bar in the browser. It is currently implemented as an additional frame of the web browser.

To describe the behavior of MEDEA we can use the simile of a school personal tutor. Suppose that a student asks help to his personal tutor (*instructional planner*) to study a subject (*domain model*). The tutor examines his academic expedient (*student model*) and taking into account other student's features (*student attitude model*), selects the topic and the style of teaching that is better for the student. After that, he consults the school staff (*tutorial components library*) and chooses the better teacher for that specific topic according to the current student's profile. The tutor must know the teaching methods of all the teachers available (i.e.: this usually does a lot of exercises, that likes better to explain theory, this wants the students makes exercises on the blackboard, etc.). The tutor send the student to the teacher selected (*tutorial component*) with a message (*communication protocol*) indicating the topic that the student must study, and may be other notices (i.e: this is the third time this student tries this topic, this student has low learning rate). This teacher will apply his effort and expertise to improve the student knowledge in the topic proposed (*intelligent component*), or he may just gives the student a good text to read (*non-intelligent component*) without asking what he is doing. When the teacher ends the class he sends back a message to the tutor explaining the students behavior (*evaluation feedback*), or the student simply returns to the tutor with no message. With this new information, the tutor (now *the student model manager*) actualizes the student expedient. This process will be repeated until the student completes the subject.

3 Domain model

The domain model represents the knowledge about the subject to be learned. Different approaches exist [11] depending of the nature of the domain to be represented. In MEDEA we have focused on declarative domains representation. The most extended models for representing this kind of domains are the semantic networks of knowledge units, used in systems as DGC [12], Eon [13] and IRIS [14].

From a conceptual point of view, the domain is defined by a) a semantic network of concepts and relations among them and b) pedagogical knowledge needed for the instruction.

From the implementation point of view, the basic elements of MEDEA domain model are a) the *concepts*, b) the *relations* between concepts, and c) the *evaluation types* used for representing the degree of knowledge.

a) *Concepts* are the basic pieces in which the subject is divided for teaching purposes. During the task of modeling a knowledge area for an ITS the pedagogical purpose of the system should be taken into account. First of all, in order to choose the net granularity, that is, it has to be decided when a knowledge unit must be discomposed in simpler units. As Anderson [11] says, sometimes to model accurately a domain requires a computational charge that it is not necessary from the tutorial point of view. Second, it is necessary to include pedagogical knowledge in the domain model to guide to the student through the knowledge units [13]. This pedagogical information is included as attributes associated to the concepts, for instance threshold marks that represents the minimum mark needed for considering the concept learned.

b) MEDEA uses a set of binary *relations* between concepts that can be used to describe the domain. These relations are: *prerequisite*, *part of*, *is a*, *belongs to*, *is useful to understand*, *is similar to* and *is opposite to*. Each relation should define an acyclic graph of concepts that is used by the functional modules to guide the instruction (*the instructional planner)* and make inferences of the student knowledge (*the student model manager*). The actual semantic of the relation is so fixed at the functional modules. However, MEDEA includes an informal description of the semantic of the relations to guide authors while creating these graphs. Currently, the domain model support all these relations, but only *prerequisite*, and part *of*, are used by the *instructional planer*. Other relations can be defined by course authors and stored for future use. The *student model manager* is still under development.

c) As we will see in next section, the knowledge student model in MEDEA is based on the overlay technique. Each concept will have a magnitude associated that represent the student degree of knowledge, the *evaluation types* in MEDEA are the types of those magnitudes. They are not fixed. but defined when the course is created among a set of *internal types* supported by the architecture. The internal types are: *enumerated* (i.e. the knowledge level about a concept can be A, B, C or D), *real* (a real number) and *distribution* (i.e. the knowledge level can be {A/0.2, B/0.3, C/0.4, D/0.1}, indicating that the probability that the student knowledge level in that concept be A is 0.2 and so on). MEDEA is a general framework that can use different components for its instructional purposes. Each of these components, can have its own internal representation of the student knowledge. Implicit conversion between internal types has been defined to support compatibility between different components.

Figure 3 shows a fragment of an XML file that contains the domain model of a course of Logic. At the beginning, the evaluation types used to evaluate each domain concept are defined. The second part is a list of concepts that includes some pedagogical information needed for the instruction as the difficulty level of each concept. At the end there is a list with the relations among concepts.

```
<!DOCTYPE DOMAIN_MODEL SYSTEM "http://sirius.lcc.uma.es/medea/dtd/DOMAIN_MODEL.dtd">
<DOMAIN_MODEL id="domain_model01" name="Logic of proposals">
   <EVALUATION_TYPES>
        <EVALENUM id="EvalEnum" default_minimum_mark="Passed">
            <ENUMERATED id="Passed"/>
            <ENUMERATED id="Failed"/>
        </EVALENUM>
        <EVALREAL id="EvalReal" lower_boundary="0" upper_boundary="10"default_minimum_mark="5"/>
   </EVALUATION_TYPES>

   <CONCEPTS>
        <CONCEPT id="t1" idref_evaluation="EvalEnum" name="Introduction"  difficulty="Low"/>
        <CONCEPT id="t2" idref_evaluation="EvalEnum" name="Formal sintax" difficulty="Low"/>
        <CONCEPT id="t3" idref_evaluation="EvalEnum" name="Semantic" difficulty="Low"/>
        […]
        <CONCEPT id="c32" idref_evaluation="EvalReal" name="CDN" difficulty="High"/>
   </CONCEPTS>

   <RELATIONS>
        <RELATION id="r1" id_origin_concept="c21" id_destiny_concept="c1" type="prerequisite"/>
        <RELATION id="r4" id_origin_concept="c4" id_destiny_concept="c2" type="is_a "/>
        […]
        <RELATION id="r73" id_origin_concept="c31" id_destiny_concept="t6" type="belongs_to"/>
        <RELATION id="r74" id_origin_concept="c32" id_destiny_concept="t6" type="belongs_to "/>
   </RELATIONS>
</DOMAIN_MODEL>
```

Fig. 3. Representation of the domain model

4 Student model

The *Student Model* in MEDEA is divided in two main subcomponent. The *Student Knowledge Model* and the *Student Attitude Model*. The first represent what the student knows about the subject, the second represent other features of the student.

The *Student Knowledge Model* is an overlay model divided in two levels: *estimated model* and *verified model*. Each level is a list of *concept/mark* pairs. As we explain in section 2, in the *verified model* the concept marks have been obtained using some evaluation method. On the other hand the *estimated model* collects the indirect information and inferences of the student knowledge degree, calculated based on the student behavior during the instruction. This multilayer approach has already been used by other authors like Brusilovsky, in the last versions of ELM-ART-II [15], that uses different layer for concepts visited, evaluated, inferred, etc. At these moment we have only proposed two layer, grouping in the estimated layer all the uncertain information and inferences.

Figure 4 shows an example of the student knowledge model represented in XML. Each concept has a value associated according to the evaluation type defined in the domain model. The last concept that has been taught is also stored.

```
<!DOCTYPE STUDENT_MODEL SYSTEM "http://sirius.lcc.uma.es/medea/dtd/STUDENT_MODEL.dtd">
<STUDENT_MODEL courseid="d03" lastconcept="c5" studentid="s432">
 <ESTIMATED_STUDENT_MODEL>
      <CONCEPT id="c1" value="VERY WELL"/>
      <CONCEPT id="c2" value="VERY BAD"/>
      [...]
      <CONCEPT id="c5" value="WELL"/>
      <CONCEPT id="c6" value="VERY BAD"/>
 </ESTIMATED_STUDENT_MODEL>
 <CHECKED_STUDENT_MODEL>
      <CONCEPT id="c1" value="VERY BAD"/>
      <CONCEPT id="c3" value="REGULAR"/>
      <CONCEPT id="c5" value="REGULAR"/>
      <CONCEPT id="c6" value="VERY BAD"/>
 </CHECKED_STUDENT_MODEL>
</STUDENT_MODEL>
```

Fig. 4. Representation of the student knowledge model

There are also some relevant student's features that are important for the learning process. In MEDEA we have identified some of them that have been included in the *Student Attitude Model*.

Feature	Values
Cognitive development (formalization and abstract concepts understanding skills)	**`High, Medium, Low`**
Motivation	**`High, Medium, Low`**
Learning style	**`Theory, Exercises`**
Time dedicated to the subject study. It represents the student effort degree to pass the subject.	**`Time in minutes`**
Progress. It represents the student learning speed.	**`Bad, Regular, Good`**
Experience with computers.	**`Very much, Some, Little, None`**
Internet connection speed.	**`High, Medium, Low`**

This model is used by teachers or course designers to establish relations between a concrete student profile and some instruction parameters. For example, a teacher can specify in the course definition that when a student with *low motivation* level does a test it is better that he sees the right answer each time he makes a question than he sees all the right answers at the end of the test.

5 Instructional planner

The instructional planner is the core of MEDEA architecture. It is the module in charge of sequencing the domain and adapting the instruction process to each student. In order to do this task we observed human teacher behavior. Usually a teacher takes

decisions in several levels. First he decides the instruction goals (concept to be taught) and then he takes the other decisions as the topics content, material to be used, which pedagogical strategy is going to be used, etc.).

There are several examples in the ITS literature in which the planner task is divided in subtasks: some oriented to concept selection and others to decide how to teach the concept selected [14] [16] [17].

The MEDEA planner takes three main decisions: 1) Does the student need to be evaluated?, 2) if YES: about which topic will he be evaluated?, if NO: which concept has the student to learn now? and 3) How will he be evaluated or taught better?. The knowledge needed to answer these questions is in the domain model, the student models and the pedagogical elements of the system (domain model pedagogical contents and tutorial components).

The modular structure of the MEDEA architecture and the separation between knowledge representation and knowledge use allows that this module could be also easily changed. The final goal of this project is to have a planner library in the system. For the first prototype a heuristic planner has been implemented. In this planner, the task of selecting a concept to be evaluated or learned will be carried on in two phases: first the planner selects an ordered set of candidate concepts. This selection is done taking into account the relations *prerequisite* and *part of* and the current student model state. Second, the planner selects the first concept of the candidates set.

The criterions to decide if a concept is a candidate or not are:

1. The student has not reached the minimum required by the teacher to pass the concept.
2. The concept is *prerequisite* of any other concept that cannot be learned because the student has not passed this one.
3. The concept is *part of* any other concept that cannot be learned because the student has not passed this one.

A weight is assigned to each candidate. The most weighted are those that are selected by the criterion 2.

As we have said before, the other important task of the planner is to decide how the student will be evaluated or taught better. Our implemented planner takes this decision using the teacher's criterion. That is, when a teacher designs a course, he links each tutorial component registered in the system to a student profile. The planner only has to consult the attitude student model and assigns him the more adequate tutorial component according to the teacher.

6 Tutorial components

A tutorial component is an external educational tool that is able to complete a tutorial task as posing tests, presenting theory contents in hypertext, posing a game, etc.

The pedagogical knowledge in MEDEA system is distributed among several modules: the domain model, the instructional planner and the tutorial components although it is not strictly necessary for the instructional process that a component provide any knowledge. Most of the instruction tasks fall on the pedagogical system core composed by the domain model and the planner. A tutorial component can

complement this task providing its tutorial strategies and a control more exhaustive over student's actions.

MEDEA classifies tutorial components as *evaluation components* or *information components*. The difference is that components of the former type are able to evaluate the student knowledge level about a concept.

The problem of the components integration in the system is very important in the development of MEDEA project. This problem can be approached as the communication between two software systems. MEDEA and tutorial components are Web systems so the communication can be established through URLs. In order to do its work, the planner needs from the component low-level knowledge about its performance (call format, parameters, etc.) and high-level knowledge about its pedagogical offer (tutorial strategies, options, user options, etc.).

Communication with a component can be established through two different interfaces: from teacher interface for course creation and from student interface for instruction session execution.

Now we are going to describe the different types of tutorial component's actions that MEDEA users (teachers and students) can request:

1. *AUTHOR*. Contents creation (tests, HTML pages, exercises, etc.). In order to do that the teacher would access the component author tool.
2. *NEWCOURSE*. If a course about a subject is already defined in the component, the teacher only has to establish the correspondence between component domain concepts and MEDEA domain concepts.
3. *NEWUSER*. If a course about a subject is already defined in the component and there are some students registered, the teacher must communicate to MEDEA system the existence of these students.
4. *EXECUTE*. Execution of a session with a component.

The component creator must specify, for each of these actions, the call and return URL formats. Not all components used by MEDEA must have all of these actions. The only compulsory action is the last one that should exist to call this component for a tutorial session. The component interface specifications are also defined in an XML file.

A library of general-purpose web based web-based authoring tools has been implemented. These authoring tools are fully compliant with the actions defined in MEDEA, but can also be used as stand-alone applications. A web based tutorial system can be constructing by developing specific components using these tools, that are also called *general-purpose components*. Currently, the general-purpose components predefined in MEDEA are:

- HERMES, which is an authoring tool to create web based electronic books. It is mainly an HTML editor that makes easier for non-programmers the creation of a set of structured web pages. The development tool is based on page templates. The result from HERMES are a set of static web pages structured hierarchically.

- SIGUE, which is a systems that allows the construction of a web course by collecting the references to existing web pages. SIGUE adds some adaptive capabilities, like adaptive links hidden, and/or reinforcement. SIGUE has his own student model that is actualized as the student goes through the course content. [18]

? SIETTE, which is an adaptive test generation system. It has a large set of item types and different ways of evaluation. [19]

The general idea is that predefined general-purpose components will works as plugs-in. Once defined they can be invoked from the MEDEA core, either for authoring or for course taking. The authoring tools of these general components is linked to the development tools of MEDEA, (action *AUTHOR*), the administration tools, are linked to the administration tools of MEDEA, (actions *NEWUSER* and *NEWCOURSE*) and the course presentation is linked to the student interface (action *EXECUTE*).

7 Conclusions

As a consequence of the increasing importance of the distance education and the advance of this field due to the new information technologies, many researchers have seen the needed of applying intelligent techniques of existing educational systems to the Web.

MEDEA is a proposal of an open architecture for Web ITS development. MEDEA is an open system that contains the traditional modules of an ITS architecture that has been designed to allow the integration and reutilization of educational tools and teaching material already developed. The idea of MEDEA is that any teacher could develop a course reusing his own material and software.

MEDEA is still under development, this presentation is just a first glance of the system. No testing has been carried out yet because it is not fully operational.

References

1. Eklund, J.; Brusilovsky, P.: The Value of Adaptivity in Hypermedia Learning Environments: A Short Review of Empirical Evidence , In *Proceedings of the Second Workshop on Adaptive Hypertext and Hypermedia at Ninth ACM International Hypertext Conference, Hypertext'98*. (1998). Available on line at http://wwwis.win.tue.nl/ah98/Eklund.html
2. Brusilovsky, P.; Ritter, S.; Schwarz, E.: Distributed intelligent tutoring on the Web. In In Brusilovsky, P., Nakabayashi, K., Ritter, S. (eds.) *Proceedings of the Workshop 'Intelligent Educational Systems on the World Wide Web' at AI-ED'97*, 8th World Conference on Artificial Intelligence in Education (1997). Available on line at http://www.contrib.andrew.cmu.edu/~plb/AIED97_workshop/Brusilovsky/Brusilov sky.html
3. Chandrasekaran, B.: Generic Task in Knowledge Based Reasoning: High level building blocks for expert systems design. *IEEE Expert*, 1, (1986) 23-29. Reimpreso en Buchanan,B.G. y Wilkins,D.C. (eds.) *Reading in Knowledge acquisition and learning*. Morgan Kaufmann. San Mateo. CA. (1993) 170-177
4. Newell, A. The Knowledge level. Artificial Intelligence. (1982); 18.

5. Wielinga, B.J., Schreiber, A.T., y Breuker, J.A. : KADS: a modeling approach to knowledge engineering. *Knowledge Acquisition* 4, (1992). Reimpreso en Buchanan, B.G. y Wilkins, D.C. (eds.) *Readings in Knowledge Acquisition and Learning*. Morgan Kauffman. San Mateo. CA (1993) 92-116
6. Musen, M. A. Automated Support for Building and Extending Expert Models. Machine Learning. (1989); 4:347-375.
7. Cuena, J. and Molina, M. KSM: An Environment for Knowledge Oriented Design of Applications Using Structured Knowledge Architectures. Applications and impacts. IFIP'94; (1994).
8. Self, J. (1999), Open Sesame?: Fifteen Variations on the Theme of Openness in Learning Environments. Keynote speaker at AI-ED'99. Abstract published in Lajoie, S., Vivet, M. (eds.) *Artificial Intelligent in Education*: IOS Press (1999).
9. Murray, Tom: A Model for Distributes Curriculum on the World Wide Web. In *Journal of Interactive Media in Education*. (1998),5
10. Murray, Tom , Authoring Intelligent Tutoring Systems: an analysis of the state of the art. In *Intl. J. of Artificial Intelligence in Education*. (1999), 10, 98-129
11. Anderson, J. R. The Expert Module. In Polson, M.C., Richardson, J.J. (eds.) *Foundations of Intelligent Tutoring Sytems*. Lawrence Elrbaum; (1988).
12. Vassileva, J. (1997), Dynamic Course Generation on the WWW. In Brusilovsky, P., Nakabayashi, K., Ritter, S. (eds.) *Proceedings of the Workshop 'Intelligent Educational Systems on the World Wide Web' at AI-ED'97*, 8th World Conference on Artificial Intelligence in Education (1997). Available on line at: http://www.contrib.andrew.cmu.edu/~plb/AIED97_workshop/Vassileva/Vassileva.html
13. Murray, Tom (1998), Authoring Knowledge-Based Tutors: Tools for Content, Instructional Strategy, Student Model, and Interface Design. In The Journal of Learning Sciences. V. 7, N.1, pp. 5-64
14. Arruarte, A., Fernández-de-Castro, I., Ferrero, B., and Greer, J. (1997), The IRIS shell: How to build ITSs from pedagogical and design requisites. In International Journal of Artificial Intelligence in Education. V. 8, N.3-4, pp. 341-381
15. Weber, G., & Specht, M.: "User modeling and adaptive navigation support in WWW-based tutoring systems"; In Jameson, A., Paris,C., & Tasso, C. (Eds.), User Modeling, Springer-Verlag, Wien (1997) 289-300. http://citeseer.nj.nec.com/weber97user.html
16. Woo, Chong Woo (1991), Instructional Planning in an Intelligent Tutoring System: combining global lesson plans with local discourse control. PhD. Thesis.
17. Woolf, B. and McDonald (1987), D.Context-Dependent Transition in Tutoring Discourse. In Proceedings of AAAI.
18. Carmona, C., Bueno, D., Guzman, E. Conejo, R.: SIGUE: Making Web Courses Adaptive In De Bra, P, Brusilovsky P., Conejo, R*.: Proceedings of the AH2002 Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, Springer-Verlag LNCS 2347 (2002).
19. Ríos, A., Millán, E., Trella, M., Pérez-de-la-Cruz, J. L., & Conejo, R. (1999). *Internet Based Evaluation System.* In Proceedings of the 9th World Conference of Artificial Intelligence and Education AIED'99. Amsterdam: IOS Press.