ABITS: An Agent Based Intelligent Tutoring System for Distance Learning

Nicola Capuano, Marco Marsella, Saverio Salerno

CRMPA, Centro di Ricerca in Matematica Pura ed Applicata c/o DIIMA, University of Salerno, via Ponte Don Melillo, 84084, Fisciano (SA), Italy
niccap@crmpa.unisa.it

DIIMA, Dipartimento di Ingegneria dell'Informazione e Matematica Applicata, University of Salerno, Via Ponte Don Melillo, 84084, Fisciano (SA), Italy

Abstract. The purpose of this paper is to describe an Intelligent Tutoring Framework highly re-usable and suitable to several knowledge domains. In particular the system, named ABITS, has been realized in the context of the InTraSys ESPRIT project. It is able to support a Web-based Course Delivery Platform with a set of "intelligent" functions providing both student modeling and automatic curriculum generation. Such functions found their effectiveness on a set of rules for knowledge indexing based on Metadata and Conceptual Graphs following the IEEE Learning Object Metadata (LOM) standard. Moreover, in order to ensure the maximal flexibility, ABITS is organized as a Multi Agent System (MAS) composed by pools of three different kind of agents (evaluation, pedagogical and affective agents). Each agent is able to solve in autonomous way a specific task and they work together in order to improve the WBT learning effectiveness adapting the didactic materials to user skills and preferences.

1 Introduction

An efficient Training System should allow users to take a lesson without time and place constrains. In order to fulfil these requirements, at present days, the best solution has to be naturally Web based. It requires to end-users zero cost installation and provides them the maximum time/place flexibility.
Three kinds of Web-Based Tutoring (WBT) methodologies are available on the scene at this moment.

- **Static WBT**: teachers arrange learning material in order to cover one or more topics and convert them in interactive linked HTML pages (or different kinds of Web-deliverable objects). Material is then placed on-line in order to make it visible to everybody. Learners can exploit it only by following the path established by teachers.
- **Personalized WBT**: teachers, using a specific kind of software named Course Management System (i.e. Macromedia Attain) are able to perform manually a set of additional tasks. They can monitor student knowledge by testing them, assign

recovery material if necessary, define different paths through learning objects for different kind of learning goals, etc.

- **Adaptive WBT**: includes all features of a Personalized WBT but the teacher is supported/simulated in his activity by using Artificial Intelligence techniques.

In this paper we will present ABITS: an highly re-usable Intelligent Tutoring Framework able to extend a traditional Course Management System (CMS) with a set of "intelligent" functions allowing both student modeling and automatic curriculum generation. Adding ABITS as a module, any Personalized WBT will be able to become an Adaptive one. The requirement for the CMS is only one: it must be able to be extended with a scripting language supporting RMI invocation and able to access external data sources. Macromedia Attain, for example, fulfills these requirements.
ABITS (Agent Based Intelligent Tutoring System) has been thought and developed in the context of the InTraSys ESPRIT project. InTraSys (Intelligent Training System in Technical Assistance) is a very complex training and learning system geared towards high-tech organizations whose objectives are to improve the training and learning effectiveness, reduce the training costs, increase industrial intellectual capital retention and decrease the employee training time [5].
The already quoted ABITS intelligent functions are summarized in the use case diagram of figure 1 and will be described in the following chapters. In particular, the "Evaluate Curriculum" function is dealt with curriculum sequencing and will be detailed in chapter 4; "Evaluate Preferences" and "Evaluate Cognitive State" concern, instead, user modeling and will be fully described in chapter 3.

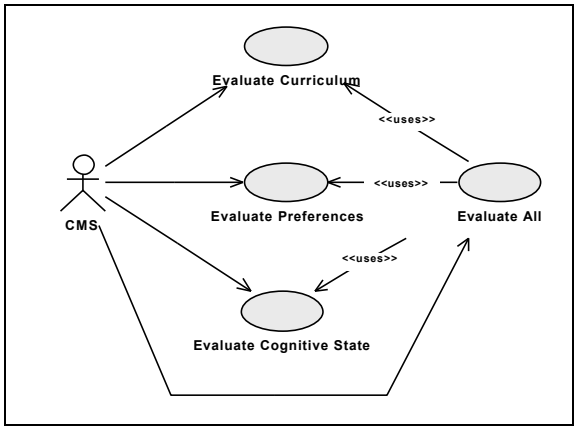

Fig. 1. ABITS Use Case Diagram

Such functions found their effectiveness on a set of rules for knowledge indexing based on Metadata and Conceptual Graphs. In the following chapter we will deepen this topic.

2 Knowledge Indexing

Inside ABITS, all didactic material is organized in several Learning Objects and stored in a Course Material File System. A Learning Object is defined [1] as any entity which can be used, re-used or referenced during technology-supported learning. In our case, a Learning Object is a logical container that represents an atomic Web-deliverable resource such as a Lesson (an HTML page), a Simulation (a Java applet), a Virtual World (a VRML file), a Test (an HTML page with an evaluating form) and each kind of Web-deliverable object.
Learning Objects must be indexed in order to let ABITS know what each one of them is about and how it can be used during the learning process. Some kind of information about Learning Objects is so required. We call this kind of information Metadata.

2.1 Metadata

"Metadata is information about an object, be it physical or digital and its main goal is to locate in efficient and effective way resources over a system or a computer network" [2].
In the field of learning materials, several organizations such as IEEE, EDUCOM etc. have focused their attention on the creation of metadata standards specifying the syntax and the semantics of the so-called **Learning Object Metadata**. A Learning Object Metadata standard defines the minimal set of properties needed to allow these objects to be managed, located, and evaluated. It accommodates moreover the ability for locally extending the basic properties.
Many advantages come from referring to a Learning Object Metadata standard:

- To take advantage of a complete syntax and semantic already created by experts of the Learning Technology.
- To enable the automatic importation of extern learning objects that adopt the same Metadata description standard.
- To enable the exportation/sale of learning objects created for ABITS to extern systems/clients that adopt the same Metadata description standard.

We chose to adopt for ABITS the IEEE LTSC Learning Object Metadata (LOM) standard [1]. We seen, in fact, that other organization are slowly converging to this one: for the future it can be the best choice.
LOM Metadata is structured in a hierarchical way: **schemes** consist of data elements. The latter are defined through: "sub-schemes" if they are a collection of data elements themselves; "data types" if their values are strings, decimals, etc; "vocabularies" if their values come from an enumerated list. Data elements moreover can be "mandatory" (must be present) or "optional" (may be present).
IEEE metadata definition implies that descriptors of a learning resources are grouped in meaningful categories. Our schema proposes six categories: a subset of the eight defined by IEEE standard:

- **General** : groups all context independent features plus the semantic descriptors for the resource.

- **Life Cycle**: groups the features linked to the lifecycle of the resource.
- **Meta Metadata**: groups the features of the description itself (rather than those of the resource being described).
- **Technical** : groups the technical features of the resource.
- **Educational** : groups the educational and pedagogic features of the resource.
- **Rights Management**: groups features that depend on the kind of use envisaged for the resource.

Table 1 resumes all categories and data elements that constitute ABITS Metadata Scheme. This is an IEEE LTSC LOM formally derived scheme.

General	MetaMetaData	Educational	RightsManagmnt
Identifier	**Create**	**PedagogicalType**	**Role**
Title	**MetadataScheme**	**CoursewareGenre**	**Description**
Language	**Technical**	Format	**Conditions**
Description	**Format**	**Approach**	Reciprocity
Domain	**Size**	**InteractivityLevel**	Attribution
Idea	**LocSpec**	**SemanticDensity**	Prize
Structure	**Requirements**	**EducationalUse**	MonetaryUnit
LifeCycle	Type	Role	Amount
Version	Name	Difficulty	UnitOfPricing
Create	MinimumVersion	Level	
	MaximumVersion	Duration	

Table 1. ABITS Learning Object Metadata Scheme

2.2 Conceptual Graphs

Metadata schemas not only have to provide information about a single Learning Object but they must also provide information about object relations and interdependency. For this purpose LOM standard provides an important data element called "**Idea**" (under the "General" category) that allows the so-called Domain Conceptualization.

A Conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose. A Conceptual Graph is an explicit specification of a Conceptualization. They can be viewed as structures of Concepts and conceptual relations where every arc links some conceptual relation ***r*** to some concept ***c*** [3].

With the term Concept we intend an abstract notion that refers to a particular Conceptual Graph. Within the ABITS context, Conceptual Graphs are used to link concepts underlying the knowledge domain with three kinds of relations: prerequisite, sub-concept and general relation (see table 2 for more details).

Kind of Relation	Relation Name	Abbreviation
Prerequisite	… "IsRequiredBy" …	**IRB**
	… "Requires" …	**R**
Sub-Concept	… "IsPartOf" …	**IPO**
	… "HasPart" …	**HP**
General Relation	… "IsRelatedTo" …	**IRT**

Table 2. ABITS Concept Relations

As an example of Conceptualization consider the concepts of addition, subtraction, multiplication, division and "basic operations" inside the domain of mathematical operations. Naturally in the Conceptual Graph of the Domain of Arithmetic, concepts must be represented as in figure 2.

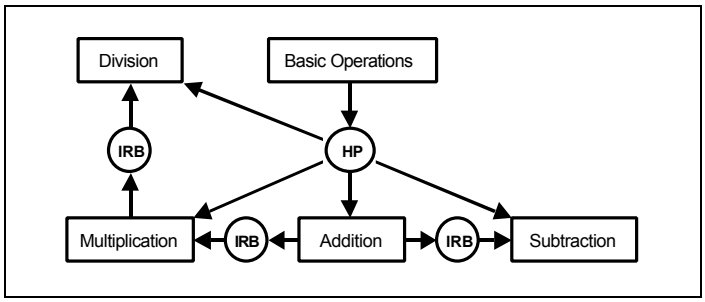

Fig. 2. An example of Conceptual Graph

A standard to allow communication between systems that require a structured representation for logic is going to be defined. It is the **Conceptual Graph Interchange Form** (CGIF) [4] and has been developed as a conceptual schema language, as specified by ISO/IEC 14481 on Conceptual Schema Modeling Facilities (CSMF). We chose to represent ABITS Conceptual Graphs using this format.
Such kind of information about the domain is massively used by ABITS functions in conjunction with Metadata fields for Cognitive State modeling and automatic Curriculum generation. While Metadata fields give information about Learning Object including explained Concepts (the already quoted "Idea" field), Conceptual Graphs give information about how Concept explained in these Learning Objects are related between themselves. In the following chapters we will see how such information is exploited by ABITS.

3 Student Modeling

In every instant ABITS must be able to determine both **Cognitive State** and **Learning Preferences** for each student basing on his developed activities. Such set of information constitutes the so-called **Student Model**. Such structures, as we will see

in the following two paragraphs, are composed by many "fuzzy" fields so it is useful now to provide a small Fuzzy Numbers overview.
A **Fuzzy Number** is a concept related to the fuzzy set theory, an extension of the conventional set theory born in 1965 by the work of Zadeh [6]. The fuzzy set theory is dealt with subsets of an universe where the transition between the full membership and the non-membership is gradual rather than sudden. If X is an objects space, a fuzzy subset A of X is a set characterized by a membership function of the type:

$$\mu(x|A): X \rightarrow [0,1]. \quad (1)$$

where x is a generic element of X and $\mu(x|A)$ is told *membership degree* of x in A. A subset in its classical meaning (crisp) can be then seen as a particular case of fuzzy subset with membership function with values in $\{0,1\}$ where 1 indicate the full membership and 0 the non membership.
A Fuzzy Number is a fuzzy subset of the set of real numbers with membership function $\mu(x|A)$ continuous, normal and convex able to satisfy the following requirements:

$$\exists x \in \Re \text{ such that } \mu(x|A)=1 \text{ (normality);} \quad (2)$$

$$\mu(x|A) \geq \min\{\mu(x_1|A), \mu(x_2|A)\} \quad \forall x \in [x_1, x_2] \text{ (convexity).} \quad (3)$$

Exploiting fuzzy numbers it is possible to mathematically represent quantity of the kind: "approximately 5" or "few less than 3" and so on, in way to model uncertainty in intuitive manner and without resorting to the artifice of probability distributions.
A fuzzy number can be graphically represented through the so-called *belief graphs* that map membership function for all support values (it is said support of a fuzzy set the set of all elements of the universe that have membership degree greater then 0). Figure 3 shows a belief graphs for a triangular fuzzy number where only a value has maximum membership degree.

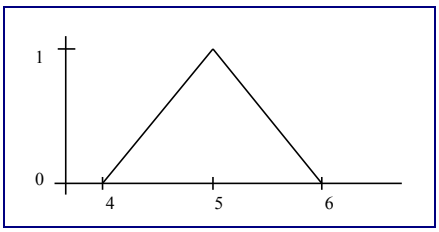

Fig. 3. A Triangular Fuzzy Number

3.1 Cognitive State

For **Cognitive State** it is intended the knowledge degree, reached by a particular student, of every Domain Concept (for each provided Domain). This information is logically represented by a string of fuzzy numbers (one for each concept) [8].

Graphically such string can be view as in figure 4. This string is updated dynamically by ABITS during training and testing activities.

Fig. 4. A Graphical Representation of a Cognitive State

The decision to use fuzzy numbers in Cognitive States arises from the necessity to manage uncertainty in the student evaluation. In this way, in fact, we can admit different kind of evaluations with different degree of reliability. For example, when a student reads an expositive Learning Object with a particular set of Concepts involved, ABITS can infer that there is a little increase in the student Cognitive State for these concepts but with a very large degree of uncertainty. When a student answer to a test in a correct way there is also an increase in his cognitive state in relation to involved concepts but with a more and more low degree of uncertainty. To represent this kind of information we use more and more "narrow" fuzzy numbers.
An other important aspect to take into consideration is that a student could forget what he has learned some times before. In order to model the attitude that have humans to forget what they learn, ABITS applies a **Forgetting Function** to Cognitive States. Cause that not all humans forget in the same manner, this algorithm can't decrease knowledge degrees but only can wide the amplitude of representing fuzzy number signifying that evaluations are more and more unreliable.

3.2 Learning Preferences

Within **Learning Preferences** we enclose all information about the student perceptive capabilities i.e. to which kind of resources a specified student is shown to be more receptive. To evaluate student preferences we exploit data elements contained into the "**Educational**" IEEE metadata category such as: format (kind of media), pedagogical approach, interactivity level, semantic density and difficulty [8]. Table 3 resumes all Learning Preferences fields with all possible values.

Fields	Possible Values
Format	Text, Image, Slide, Hypertext, Video Clip, Simulation, Virtual Reality
Approach	Inductive, Deductive, Explorative
Interactivity Level	Very Low, Low, Medium, High, Very High
Semantic Density	Very Low, Low, Medium, High, Very High
Difficulty	Very Low, Low, Medium, High, Very High

Table 3. ABITS Learning Preferences Possible Values

For the **Format** data element, ABITS maintains a list of fuzzy number where each one of them evaluates the receptiveness of a particular student to involved media

(text, hypertext, video clip, simulation, virtual world, slide, etc.). The same idea is adopted in relation to pedagogical **Approach** field to evaluate if student is more interested in inductive, deductive or exploratory approach to learning. In the case of **Interactivity Level**, **Semantic Density** and **Difficulty**, represented by a single level value (very low, low, medium, high, very high), ABITS maintains a fuzzy number (for each parameter) representing the best receptive level for a particular student.
In order to evaluate student Preferences, ABITS exploits this idea: during the learning process there are some Milestones (established by tutors) when the Cognitive State is updated with respect to activities performed by students. This means that for each Domain Concept involved in the student performed activities, a new evaluation is given.
Exploiting the *conceptual variation* for each Concept and Metadata information on Learning Objects visited concerning that Concept, ABITS can evaluate the pedagogical effectiveness of Learning Object typologies. For example, if the knowledge of a particular Concept is sensibly raised between two milestones and visited Learning Objects concerning this particular Concept have been for a great part simulations, ABITS infers that the student is receptive to simulations. The system increases consequently the "format" Preference that refers to simulations.
The information calculated by ABITS about Student Models can be exploited directly by tutors or can be re-used by ABITS in the automatic Curriculum generation procedure. In the following chapter we will explain how ABITS can do that.

4 Automatic Curriculum Generation

Each student can be assigned to one or more different Courses. An ABITS **Course** is composed essentially by an set of Learning Goals and by a Curriculum [8].
With **Learning Goals** (that are strongly different from Learning Objects) we intend a set of key Concepts necessary to be learned to successful complete a specific **Course**. Such Concepts (as all other Concepts) are part of a Domain and are represented inside the Conceptual Graph of such Domain.
With **Curriculum** we intend, instead, an ordered list of Learning Objects that can be used to provide to a specific student all necessary knowledge to complete a specific **Course**. While Learning Goals indicate what (which Concepts) a student has to learn, Curriculum specify how these Concepts has to be learnt.
Different students can require different Curriculums to learn about the same Learning Goals depending on their Cognitive States and Learning Preferences. For this reason a Curriculum Generation procedure is also provided by ABITS.
Starting from the list of Learning Goals that must be covered, ABITS **Curriculum Generation** is done by a three-step procedure [9].

1. The list ***LG*** of Learning Goals is exploded in order to obtain the list ***C*** of all Concepts that a student must learn to reach such Goals. This is obtained using a recursive function: for each concept ***c*** contained in ***LG*** a test is done: if ***c*** is already known by the student (looking his Cognitive State) then ***c*** is discarded, otherwise the ***c*** is added to ***C*** and all ***c*** requisites are kept (looking the Conceptual Graph of the ***c*** Domain) and added to ***LG***. The procedure is repeated until ***LG*** is empty.

1. The list of concept ***C*** is transformed in a list of Learning Objects ***LO*** finding the best ordered sequence of Learning Object from the Course Material Database matching student Learning Preferences (comparing Metadata "Educational" Fields with student Preferences fields) and covering ***C***. ***LO*** must be ordered: if concept c_1 requires concept c_2 then any Learning Object explaining c_1 will be placed in the Curriculum after any Learning Object explaining c_2.
2. The list of Learning Objects ***LO*** is transformed in a Curriculum ***CURR*** by adding Testing Material and Milestones. Testing material are simply Testing Learning Objects (i.e. HTML pages with evaluation forms) while Milestones are embedded ABITS calls. Milestones are placed after each testing block and at the end of the whole curriculum in order to advice the Course Management System that ABITS must be called in this place to update the model and/or the Curriculum sequence for a specified student.

It is important to note that all ABITS functions could be used by the Course Management System in a systematic or occasional manner according to tutors and administrators policies.
Information generated in the student modeling phase in-fact can be used directly by ABITS in the Curriculum generation process or simply by tutors to take decisions about lessons and recovery activities to assign. In the same manner, the Curriculum generation facility itself can be used both to help tutor course management activities or directly by ABITS to change in run-time student Curriculums basing on performed activities.

5 ABITS Architecture

While in the preceding chapters we looked ABITS from the functional point of view (what functions are provided by ABITS and how they work), this chapter will instead show ABITS from the architectural point of view (how ABITS is composed). As we already said in the introduction, ABITS is born as a module of a greater system named InTraSys but, thanks to its auto-consistency, ABITS can be extrapolated from this context.
Figure 5a depicts relations and interdependencies between ABITS and a generic external Course Management System. As you can see, modules are strongly separated. Communication happens only through CMS – ABITS directed RMI invocations. Both modules, moreover, must have the ability to access to a shared courses database. From this preliminary discussion it appears already clear that any Course Management System allowing both RMI calls and access to external data sources through a scripting language can be supported by ABITS.
Many data sources can be found in figure 5a, let's describe them.

- **Course Material Database** contains all Learning Object in the form of web-deliverable files (HTML, VRML, CLASS, etc.);
- **Metadata Base** contains all Metadata schemas indexing Course Material in XML/RDF format [7] (such database can be accessed and modified using the Metadata Authoring Tool);

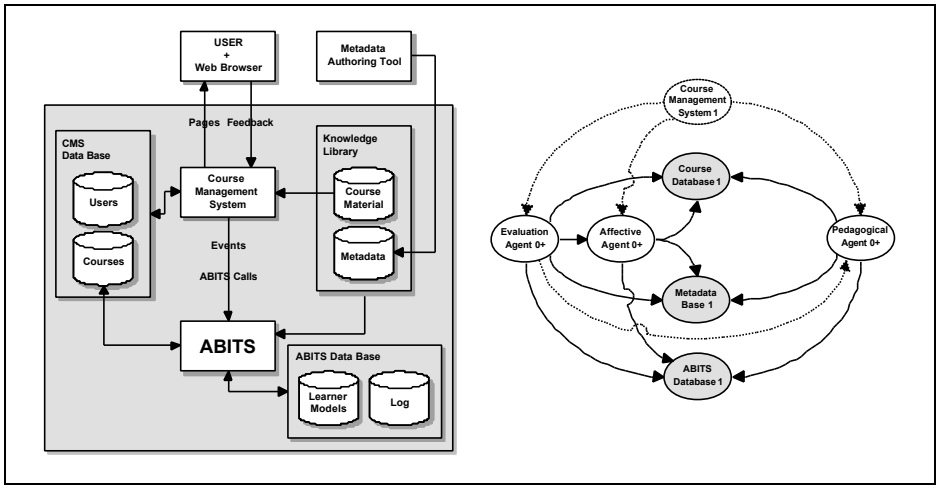

Fig. 5. ABITS External (a) and Internal (b) System Architecture

- **Log Database** contains all student activities performed during the learning experience (visited pages, permanency times, test results etc.);
- **Learner Models Database** contains ABITS-calculated Cognitive States and Learning Preferences for each student;
- **Users Database** contains all information about system users (log-in names, roles, course assigned, etc.);
- **Courses Database** contains all information about Learning Goals and Curriculum of all student courses.

Interactions between databases and modules (which module can access to which data source) are still depicted in figure 5a where arrows represent data flows.
As we already said, ABITS, internally, is conceived as a Multi Agent System (MAS). The motivation of this choice rely on the high degree of scalability allowed by this kind of architecture. ABITS in fact is composed by pools of three different kinds of **Agents**. The three typologies are the followings:

- **Evaluation Agents**: are interested of evaluating and updating Cognitive States and whole Student Models (remember the "Update All" use case); to do this, they interact with the *Metadata Base*, the *ABITS Database*, the *Affective Agent* and the *Pedagogical Agent*.
- **Affective Agents**: are interested of evaluating and updating Learning Preferences; to do this, they interact with the *Metadata Base* and the *ABITS Database*.
- **Pedagogical Agents**: are interested of evaluating and updating Curriculums; to do this, they interact with the *Metadata Base*, the *ABITS Database* and the *Courses Database*.

Figure 5b shows how ABITS agents interact between themselves and between external modules and data sources. It is important to note the multiplicity number near each object. While CMS and data sources have multiplicity 1 (only one component of this kind must be present), each agent has multiplicity 0+ (zero or more agents of this

kind may be present). These are the advantages in terms of scalability of this kind of architecture.
Thanks to that, in fact, many scenarios are feasible using subsets of ABITS agent kinds that can add to a Course Management System only needed subsets of ABITS functions (according to tutors and administrator policies). Moreover, for each kind, any number of agents can be invoked and placed on different machines in order to fulfill any work load request.

6 Conclusions

In this paper we described ABITS: an highly re-usable Intelligent Tutoring Framework suitable to several knowledge domains. Such system, as we shown, is able to support any Web-based Course Delivery Platform (RMI compliant and able to access external data sources) with a set of "intelligent" functions allowing student modeling and automatic curriculum generation.
We described ABITS intelligent functions pointing on how they are implemented. As we seen, such functions found their effectiveness on a set of rules for knowledge indexing based on Metadata and Conceptual Graphs following, respectively LOM/RDF and CG/CGIF standards.
We discussed finally about ABITS internal architecture and we shown that, in order to maximize flexibility and scalability, it is organized as a Multi Agent System (MAS) composed by pools of different kind of agents where each agent has its own task to solve.

References

1. **Wayne Hodgins et al**. "Learning Object Metadata, Working Draft Document 2.5" *IEEE Learning Technology Standards Committee (LTSC)*, 1999. (http://ltsc.ieee.org/wg12).
2. **InTraSys project** "InTraSys Metadata Specification and Design v1.0" *InTraSys ESPRIT project official deliverable*, 1999.
3. **Sowa, John F**. "Conceptual Structures: Information Processing in Mind and Machine" *Addison-Wesley*, 1984.
4. "Conceptual Graph Standard, draft proposed American National Standard (dpANS)" *Information Technology (IT) – Conceptual Graphs*, 1999.
5. **InTraSys project** "InTraSys: Intelligent Training System in Technical Assistance – Description of the RTD Project " *InTraSys ESPRIT project official deliverable*, 1999.
6. **D. Dubois, H, Prade** "Fuzzy Sets and Systems – Theory and Applications" *Academic Press*, 1980.
7. **World Wide Web Consortium** "Resource Description Framework (RDF) Model and Syntax Specification" *W3C Proposed Recommendation*, 1999. (http://www.w3.org/RDF/).
8. **InTraSys project** "ABITS: Agent Based Intelligent Tutoring System Specifications v1.3" *InTraSys ESPRIT project official deliverable*, 1999.
9. **InTraSys project** "ABITS: Agent Based Intelligent Tutoring System Design v1.0" *InTraSys ESPRIT project official deliverable*, 1999.