Designing Intelligent Tutoring Systems: A Personalization Strategy using Case-Based Reasoning and Multi-Agent Systems

Carolina González[a], Juan Carlos Burguillo[a], Martín Llamas[a], Rosalía Laza[b]

[a] Departamento de Ingeniería Telemática, Universidad de Vigo

[b] Departamento de Informática, Universidad de Vigo

KEYWORD

Intelligent Tutoring Systems
Multi-agent Systems
Case-Based Reasoning
Health Education

ABSTRACT

Intelligent Tutoring Systems (ITSs) are educational systems that use artificial intelligence techniques for representing the knowledge. ITSs design is often criticized for being a complex and challenging process. In this article, we propose a framework for the ITSs design using Case Based Reasoning (CBR) and Multi-agent systems (MAS). The major advantage of using CBR is to allow the intelligent system to propose smart and quick solutions to problems, even in complex domains, avoiding the time necessary to derive those solutions from scratch. The use of intelligent agents and MAS architectures supports the retrieval of similar students models and the adaptation of teaching strategies according to the student profile. We describe deeply how the combination of both technologies helps to simplify the design of new ITSs and personalize the e-learning process for each student.

1 Introduction

Intelligent Tutoring Systems (ITSs) constitute a type of Intelligent Educational Systems (IESs). ITSs contain adequate knowledge domain and its purpose is to transmit that knowledge to the students by means of an individualized iterative process, trying to emulate the way a human tutor guides the student in his/her learning path. Developing and implementing an ITS is a difficult task, since the required technology often implies most of the areas of Artificial Intelligence (AI): knowledge representation, diagnosis, cognitive modeling, qualitative processing and causal modeling process. Besides, it is necessary to have a good knowledge on the domain or topic selected to be taught. The ITS intelligence is constituted by the diagnosis process and the tutoring process adaptation, according to the student profile. In this sense, a challenging research goal is the development of ITSs with adaptive characteristics. Adaptive ITSs can be obtained at several levels: (a) at the level in which the material or the help is presented, (b) considering the difficulty of the problems proposed, or (c) during the selection of the suitable instructional strategy according to its capacities, abilities and learning styles preferred.

In response to this challenge, in this article we propose a Case-Based Reasoning (CBR) approach to design Intelligent Tutoring Systems able to personalize the teaching process in different domains. This approach has three important advantages: (1) it provides a learning method, which uses knowledge adquired from past experiences, (2) it allows the retrieval of similar student models from multiorganizational distributed datasets and the adaptation of teaching strategies according to the student characteristics and (3) it preserves all the major pedagogical features associated with cognitive tutoring systems, a highly effective subtype of ITS. The reusable problem-solving method permits scalability, ease acquisition and maintenance of knowledge.

We also present a highly modular multi-agent architecture to create two interlacing components. One component produces the expert model as a dynamic and advancing representation of the solution and the other produces an instructional layer tailored to the specific student. The instructional layer is therefore independent of the expert model and it is able to provide feedback inspecting students progress across the entire solution.

The paper is organized as follows. Section 2 introduces the methods and technologies used in our approach. Section 3 explains the framework for designing ITS. Section 4 describes a case study of the implemented prototype. Finally, section 5 is devoted to present the conclusions.

2 Material and methods

Our approach incorporates aspects of cognitive tutoring and knowledge-based systems design within the framework of the INGENIAS methodology [18]. In the problem solving process, the Case-Based Reasoning paradigm is used. The system can effectively infer the students knowledge through the cases generated when the student solves a problem.

2.1 Intelligent Tutoring Systems (ITSs)

Intelligent tutoring systems started to be developed in the 80s, they were designed with the idea of providing knowledge based on some form of intelligence in order to guide the student in the process of learning [9]. An intelligent tutor is a software system that uses Artificial Intelligence techniques to represent the knowledge and interacts with the students in order to teach them [24]. In [8] the authors add the consideration of different cognitive styles of the students who use the system according to [2]. In the 90s, with the advances of cognitive psychology and the new programming paradigms, ITS have evolved from a mere instructional proposal to the design of environments of new knowledge discovery application [21].

2.2 Case-Based Reasoning

CBR is an approach to problem solving that emphasizes the role of prior experience (i.e. new problems are solved by reusing and, if necessary, adapting the solutions to similar problems that were solved in the past). Solving a problem by CBR involves obtaining a problem description, measuring the similarity of the current problem with previous problems stored in a case base (or memory) with their known solutions, retrieving one or more similar cases and attempting to reuse the solution of one of the retrieved cases, possibly after adapting it to account for differences in problem descriptions. The solution proposed by the system is then evaluated (e.g., by being applied to the initial problem or assessed by a domain expert). Following the revision of the proposed solution, the problem description and its new solution can then be retained as a new case. Thus the system has learned how to solve a new problem. Figure 1 shows the CBR cycle, adapted from (Aamodt & Plaza, 1994) [1]. It works as follows:

1) Retrieve previously experienced cases related to the current problem.
2) Reuse these cases in one way or another.
3) Revise the solution based on re-using previous cases.
4) Retain the new solution (as a new case) by adding it into the existing case-based database. Then, a CBR system will gradually grow larger and become a valuable resource.

The use of CBR has been considered in the past to enhance Intelligent Tutoring Systems with learning abilities. In [10] the authors propose the use of CBR as a technology for student modeling in ITSs. That approach follows the steps of the CBR cycle and it can build concrete student models by combining rule-based reasoning. But such approximation only supports the retrieval and reusing phases of the cycle. Other approaches recommend the use of CBR for instructional and route planning [16]. In [11] an Intelligent Tutoring System based on the CBR methodology was developed. This system is able to produce novel courseware arrangements for new students, based on a process of case adaptation. Elorriaga [6] proposes an approach for producing case-based in-

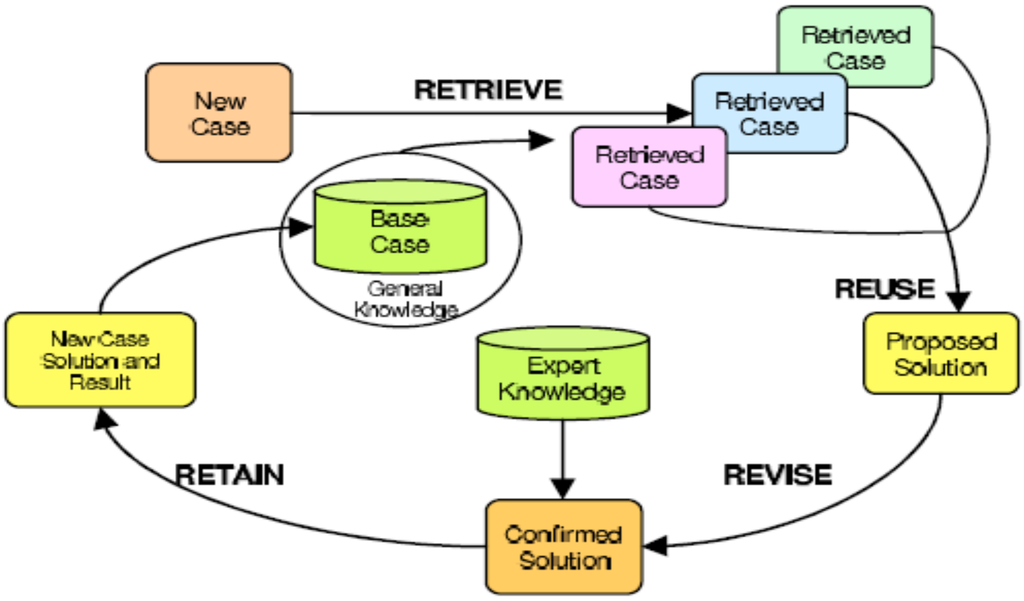


Fig. 1. The CBR cycle. Adapted from (Aamodt & Plaza, 1994)

structional planners that are integrated in ITS to enhance the pedagogical model.

The works mentioned above only use CBR as a technology for building isolated ITS modules but they do not consider CBR as a methodology that integrates all the components of the ITS architecture.

The use of CBR presents the following advantages in our approach:

- It provides a better prediction accuracy to model the student than other techniques (p.e. Bayesian Networks) [9], [23].
- It reflects the same method as a human tutor uses when making students estimations by applying analogical reasoning.
- It can handle both quantitative and qualitative data (i.e. prescore/motivation).
- It can use an existing solution to adapt it to the new students.
- It allows fast prototyping.
- It simplifies the acquisition and knowledge management.
- It can effectively support all the steps in the ITS design by storing past cases, retrieving similar cases and adapting them to new problem.
- It takes advantages of expert prior knowledge.

2.3 Multi-agent Systems (MAS)

Agents can be defined as autonomous, problem-solving computational entities capable of effectively performing operations in dynamic unpredictable environments. Such environments are known as multi-agent systems [25]. Agents interact and maybe cooperate with other agents. They are capable of exercising control over their actions and interactions.

The integration of agent technology and CBR has been proposed in mobile [13], adaptive agents [17] and active CBR [14]. These approaches are focused on the retrieval mechanisms and the associated case representation and indexing. However, a major problem for these systems is the difficulty to adapt and evaluate the proposed solution.

The main benefits of using intelligent agents within CBR environment are:

- Autonomy: the ability of agents to make an independent decision.
- Ability to learn from experience autonomously.
- Goal-driven: the provision of detailed knowledge so that goals can be achieved.
- Reactivity: capability to react to changes in the environment.

- Ability to cooperate: a group of agents work together to achieve a common goal.
- Ability to communicate: the agents must be able to communicate with other agents and/or users.

Our ITS-CBR framework is composed of intelligent agents working to find the most similar cases. Agents access local case bases to retrieve the best matching cases, which, when assembled, may not result in the best overall case in terms of global measures. But cooperation among them may lead to the achievement of the overall goal. Which means that the teaching strategy selected does not just rely on a few cases stored locally, instead of this it is affected by larger and distributed datasets).

2.4 The INGENIAS Methodology

INGENIAS [18] is an agent based methodology which has evolved from an object oriented approach [19]. The role of agent oriented methodologies is to assist in all the phases of the agent life cycle and its management.

The elements that an agent oriented methodology must provide could be grouped into four main categories [10]: (1) concepts and properties are basic notions about the domain area where the methodology will be applied; for example, notions of agent and its characteristics, (2) notations and modelling techniques are related to the specific symbols used in the methodology for representing the concepts and properties (the modelling language), (3) the process indicates which stages of the software development cycle are covered by the methodology and finally, (4) pragmatics considers aspects related to the management and the use of the methodology for example, facility and costs of adopting it, expertise required, support tools for the application of the methodology, etc.

INGENIAS covers these four basic categories, but it also provides a process to guide the software development; a language based on the main concepts of agent theory (for example the notions of agent, role, mental state, goals, believes, tasks, etc.); different models for describing different views of the system at different abstraction levels and a modelling tool.

The ITS Multi-agent system presented in this article has been designed by using INGENIAS.

3 Adaptation of CBR and MAS for designing ITSs: The framework

The relationship between CBR systems and ITSs is established by representing student models as cases. The advantage of this approach is that a problem can be easily conceptualized in terms of agents and be implemented as a CBR system afterwards. ITS-CBR updates its base of cases continually and consequently it adapts itself to changes in the environment. Moreover, each stage of the CBR cycle is automated by the system.
The framework proposed consists of an integrated set of components which are distributed and divided into smaller parts called agents. The complementary properties of CBR and agents technology can be advantageously combined to solve the ITS design, where any single technique fails to provide a satisfactory solution. Within this approach, the ITS-CBR functional architecture consists of the following components: (1) the student model generation layer, (2) the multi-agent case base reasoning layer and (3) the knowledge module and the delivery layer, which can all be seen in Figure 2.

3.1 Student Model Generation Layer

The student module models the knowledge that the student has about the domain he/she is trying to learn and how it evolves. The student module is composed of the student model and the diagnostic process. On the one hand, the student model describes the knowledge that the student has adquired in the domain to be learnt. Different types of techniques can be used: vectors, semantic networks, Bayesian networks, affirmation repositories, etc. On the other hand, the diagnostic process is in charge of updating the student model based on the current student model and the student performance during the learning process, according to diverse variables previously defined (problem evaluation, answers to questions, time spent in studying each explanation, etc.).

The student model has as many instances as students using the ITS. Each of these instances tracks the student during his/her use of the system. The student model can spread over several courses and curricula. It is initialized when the student takes his/her first course within the ITS. The most important attributes to be considered in the student model are:

- Knowledge Level: tutorial, topic and concept.
- Capacities: problems solved with right answers.
- Limitations: exercises where the student had problems.
- Attitudes: exercises solved using some kind of help.
- Learning path: The route through topics and concepts that the student follows in the learning process.

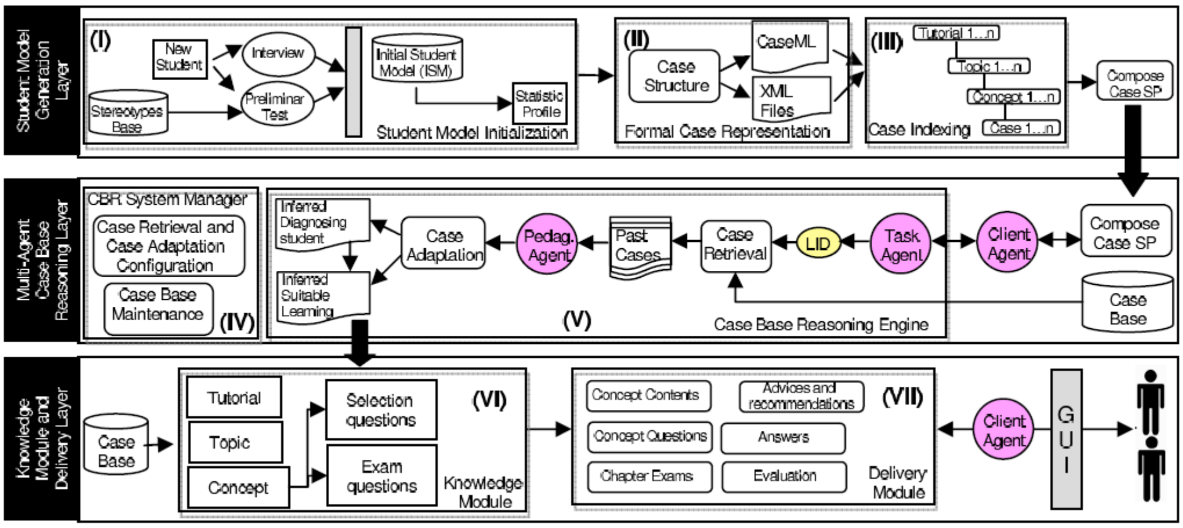


Fig. 2. Multi-agent CBR architecture

In ***Student Model Initialization Process*** the information about a new student is acquired by means of an interview and preliminary test (Stage I in Figure 2). At first, the student is interviewed about some personal data required to set an initial student model. The interview takes place the first time that a student interacts with the system. It contains questions related to personal and domain independent data, such as the student's name, age, etc. as well as several indirectly domain dependent characteristics. In order to assess the prior knowledge level of the student concerning the domain being taught and/or certain important prerequisite topics, the system uses a preliminary test. This test contains representative questions that cover the whole domain previously taught. In addition, important topics about the domain of interest that should be previously known are included. According to the students' performance on the preliminary test, the system assigns the student to a stereotype category concerning her/his knowledge level. At the end of the process, an initial student model (ISM) is obtained. In this model, each new student is regarded as a case and the students knowledge level is inferred taking into account his/her performance on the preliminary test.

In the ***Formal Case Representation Process*** the representation scheme is dependent on the case size and the complexity of the attributes which describe the case (Stage II in Figure 2). These attributes are used as a basis for finding similar past teaching strategies of known cases. A case of the ITSCBR platform consists mainly of three parts: (1) the problem description, (2) the solution and (3) the relationship. The description part contains the values of the attribute describing the behaviors of the case, while the solution part contains the solutions. The relationship part describes the links among cases. Multiple cases can be use to represent a single problem.

Traditionally, there were several types of methods for representing cases: (a) textual approach, (b) attribute-value and, (c) structured representation. However, the textual approach needs a human interpreter. The attribute-value representation has no structural or relational information and fails to describe complex objects. The structured representation as an objected oriented case requires approaches for similarity assessment that allow to compare two differently structured objects, which is quite difficult. Thus, we decided to use the Case Markup Language (CaseML) [3] a standard vocabulary for case description, which improves the issues above described and ensure the success of case interchange and distributed case-based reasoning. CaseML is conceptually built around an existing activity description framework: IMS Learning Design (which was in turn adapted from Educational Modeling Language developed by the Open University of the Netherlands) [12], [5]. The main elements of IMS Learning Design are essentially the same for CaseML and they appear in Figure 3. The related concepts are:

- Objectives: The intended outcomes of the case.
- Prerequisites: The starting conditions required to start the case.
- Triggers: The events or conditions that start and stop the activity.
- Actors: The individuals involved in the case (roles in IMSLD).
- Primary activities: The activities directly part of the case activity (such as diagnosis, or teaching strategy selection).
- Support activities: The activities that support the case activity.
- Environment/scenario: The context in which the case is conducted.
- Services: The tools required to conduct the case.

The classes in CaseML are: CaseBase, Case, Problem, Feature, Solution and SimilarityAssessment. The properties in CaseML are: hasProblem, hasSolution, hasDescription, has-SimilarityAssesment and hasAdaptation. Figure 4. depicts the classes and the properties mentioned. Details about them can be found in [3].

In *Hierarchical Case Indexing Process* the cases are divided into groups. In the highest level, there is a tutorial. In the second level, there are different topics that compose the course. In the next level, there are concepts, which are knowledge units of each topic. Finally, we find the cases themselves, grouped according to the concepts. Other elements included are: selection and exams questions. These elements are used to obtain information about level of knowledge acquired by the student in some parts of the tutorial. The hierarchical organization reduces the space of cases to be analyzed, as a result the system can focus on potential cases to be reused (Stage III in Figure 2).

Finally, the Compose Case Student Profile (SP) is obtained. It corresponds to the set of cases structured adequately for being matched in the next phase.

3.2 Multi-agent Case-Based Reasoning Layer

The MAS-CBR component provides the featured CBR techniques employed within ITS. The CBR System Manager (Stage IV in Figure 2) allows users to

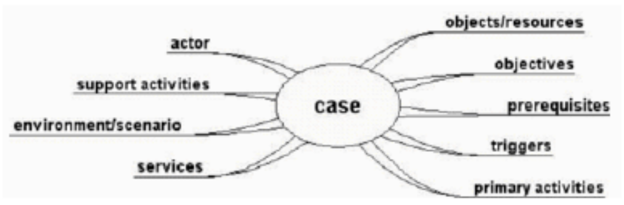


Fig. 3. CaseML Root Elements

configure the various case-retrieval and case-adaptation parameters incorporated in the *CBR engine* (Stage V in Figure 2): field-level weights, case-retrieval and thresholds.

The MAS-CBR has the steps of the case-based reasoning methodology. This is to say that the system goes in 4 cycles that are: Retrieval, Reuse, Revise and Retain.

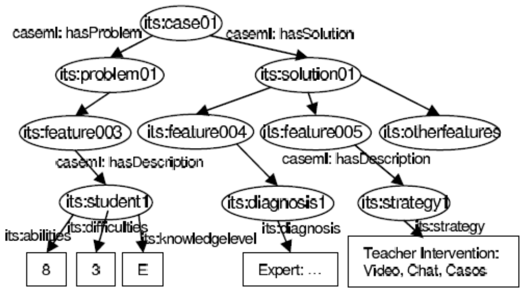


Fig. 4. CaseML Scheme

- *Retrieval Phase*: In this phase, the system searches for a similar solved case by comparing new cases with the existing case base. Once the Initial Student Model is obtained, the first task is the specification of the characteristics that will formulate the input space of the retrieval algorithm. The involvement of domain experts and human teachers is very important in this process, since they are the most appropiate source for providing such information. In this sense, we had selected attributes such capacities and students limitations. During the assessment, the capacities attribute corresponds to a list of exercises that the student has solved correctly and the limitations attribute corresponds to a list of exercises that the student was not able to solve correctly. Figure 5 illustrates how an assessment is proposed.

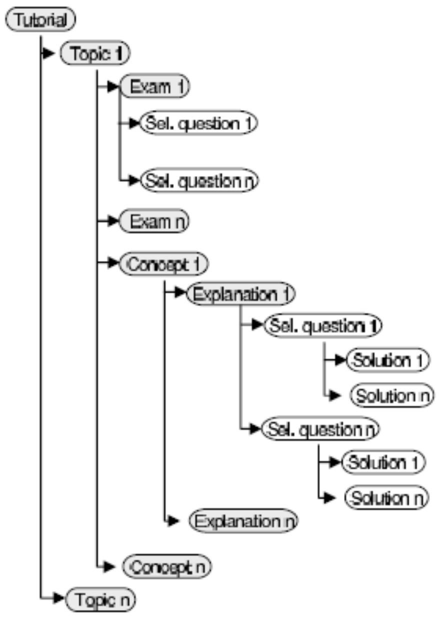
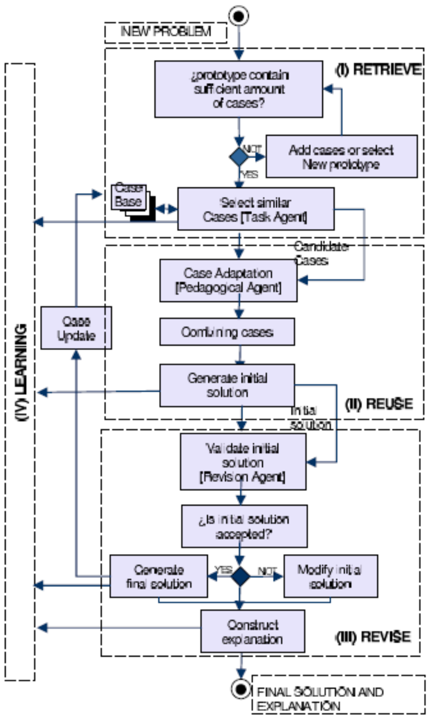


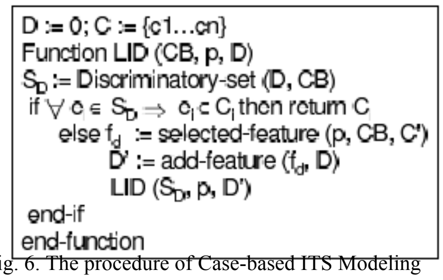
Fig. 5. Assesment Process

An assessment is composed of exam questions and selection questions. The exam questions are the questions contained within the topic exams. They consist of ”written” exercises that the student must solve and they will be evaluated and corrected by a teacher or an expert outside the system. The selection questions are contained among the textual contents of each concept. Their objective is to evaluate the knowledge obtained by a student on the concept as s/he examines the explanations. This type of question consists of a heading and a series of answers to choose from, of which one/s could be correct. The task agent performs the case retrieval process (Stage I in Figure 6.).
This agent executes the following tasks: (a) to receive candidate cases from case base, (b) to merge candidate cases and (c) to choose the best case(s). In the students’ classification based on the initial student model, the ”Lazy Induction Descriptions (LID)” algorithm is used.

The main aim of LID is to determine which the more relevant features of the problem are and to search for cases sharing these relevant features


NEW PROBLEM
¿prototype contain sufficient amount of cases?
(I) RETRIEVE
NOT
Add cases or select New prototype
YES
Case Base
Select similar Cases [Task Agent]
Candidate Cases
Case Adaptation [Pedagogical Agent]
Case Update
Combining cases
Generate initial solution
Initial solution
(II) REUSE
(IV) LEARNING
Validate initial solution [Revision Agent]
¿Is initial solution accepted?
Generate final solution
YES
NOT
Modify initial solution
Construct explanation
(III) REVISE
FINAL SOLUTION AND EXPLANATION

in the case base. The problem is classified when LID finds a set of relevant features shared by a subset of cases belonging all of them to the same solution class. Then, the problem is classified into that solution class. LID follows a top-down strategy to build a description D containing the most relevant features of the problem p so that all features in D are satisfied by a subset of cases in the case base CB. In general, cases in this subset belong to different solution classes. LID adds relevant features to D until the subset of cases satisfying D belong to one unique solution class. LID takes this class as the solution for the current student. In this phase the main attributes considered are the Learning Path and the Knowledge Level. The LID algorithm is described in Figure 7.



```
D := 0; C := {c1...cn}
Function LID (CB, p, D)
$S_D$ := Discriminatory-set (D, CB)
 if $\forall c_i \in S_D \Rightarrow c_i \subset C_i$ then return C
      else $f_d$ := selected-feature (p, CB, C')
             D' := add-feature ($f_d$, D)
             LID ($S_D$, p, D')
 end-if
end-function
```

Fig. 6. The procedure of Case-based ITS Modeling

Fig. 7. LID Algorithm

The set of cases SD that are subsumed by the description D is called discriminatory set. A case C is subsumed by a description D when all the information contained in D is also contained in C. Initially, D is an empty description so it subsumes all the cases in CB (SD= CB). Consequently D has to be specialized.
The specialization of a description D is achieved by adding features to it. In particular, LID adds a feature f with the value v that this feature has in the current problem p. After that, the new description D' = D + (f=v) has a smaller discriminatory set SD0 formed by those cases subsumed by D'. Thus, specialization reduces the discriminatory set SD at each step. LID uses a heuristic measure based on the López de Mántaras distance (RLM) [15] to determine the feature to be added. LID specializes D by selecting one feature f from all the features used in p in the following way: each feature fi in p induces a partition Pi = Si1... Sing in the set SD so each Sik that belongs to Pi contains those students in SD having the same value vk in the feature fi. Intuitively, the RLM distance assesses how similar a partition is with regard to a referent partition in the sense that the fewer the distance the more similar they are.

- *Reuse of Adaptation Phase*: Once the similar cases are identified through the case retrieval phase; their corresponding solutions need to be adapted so that, a fine grained personalized solution is derived and expose to the active student. Generally speaking, the retrieved solutions require adaptations in order to be applied to the new problem. The adaptation process may be either as simple as the substitution of a component from the retrieved solution or as complex as a complete change of the solution structure. In MAS-CBR pedagogical agents use compositional adaptation [20] to reuse the solutions of the retrieved case(s) and to propose suitable solutions to the active student (Stage II in Figure 6.). The procedure for the adaptation is described as follows:
  1) Compute the similarity between a retrieved case and a new problem (np - new student). The similarity value was obtained using the LID algorithm described in the Retrieval Phase.
  2) For each similar case *Ci*, compute the normalized similarity (NS) between a retrieved case *Ci* and the new student over the set of retrieved cases (RC) as follows: For every student *np*:

$$Temp = \sum_{i=1}^{RC} 1/Sim(np, C_i) \qquad (1)$$

$$NS(np, C_i) = 1 - (1/(Sim(np, C_i) * Temp)) \qquad (2)$$

  3) Determine the appropriateness degree of available solution components. Let *SolCi* be a component of a solution from a past case *Ci* and

$$AD_{np}^{SolC_i} \qquad (3)$$

  be the appropriateness degree for *SolCi*, then, for every student np:
  For i=1 to RC
  If *SolCi* exists in the solution of the similar case *Ci* then:

$$AD_{np}^{SolC_i} = AD_{np}^{SolC_i} + NS(np, C_i) \qquad (4)$$

  The appropriateness degree is calculated at a component level. If *SolCi* is greater that some predefined threshold value, then the component would appear in the final solution.
  4) After combining the components from multiple cases to form the final solution, the resulting new case is added to the case base.

With this adaptation method, global and attribute-level similarity are taking into account. This means that the new solution obtained is specific according to the student profile.

- *Revision Phase:* This phase has traditionally been one of the most difficult to automate in a CBR system [7]. In our work a ***revision agent*** (Stage III in Figure 6.) uses an evaluation system to perform this task. When a case solution generated by the adaptation phase is wrong, the revision agent is responsible of modifying the solution taken into account the available knowledge about the problem. This agent performs two tasks:
  1) It revises each step that the student follows in the learning process: this task is supported by a set of ***concept agents***. These agents evaluate the degree of knowledge attained by the student at the time of learning a concept. Besides, (a) it provides the student with the necessary explanations to learn the concepts, (b) it monitors the time devoted by the student to study the textual explanations of the concept; (c) it informs the student if the chosen answer or answers to a question are correct or not and supplies the correct answers if needed and (d) it finally provides the pedagogical agent with the value of the degree of knowledge ob-

tained by the student in the concept when it is requested.

2) *Repair the case solution using domain-specific knowledge:* This task involves detecting mistakes in the initial solution and retrieving or generating explanations for them. The agent uses the failure explanations to modify the solution in such a way that these faults do not occur again.

- *Retain Phase:* Once the ***revision agent*** ensures the correctness of the solution, the new case can be retained (Stage IV in Figure 6). Otherwise, if the agent detects irregular conditions, the case is not stored and a report is generated.

Finally, the ***Client Agent*** is responsible for the interaction with the user. It is able to understand the students requests and translate them to the other agents. This agent is the unique communication interface of the student and it has different tasks that are crucial for the correct operation of the whole system: (a) assisting the student on performing requests and compiling his profile, (b) deducing the students information needs by both communicating with him and observing his behavior, (c) translating the users request and selecting the agent(s) able to solve his problem(s) and (d) presenting and storing the retrieved data.

3.3 Knowledge Module and Delivery Layer

In the Intelligent Tutoring Systems field, the elements that represent the knowledge of the domain are included in this part, known as the expert module (Stage VI and VII in Figure 2). These elements are organized in a structure called ***curriculum structure.*** The elements that represent all the knowledge to be adquired by a student in a determined tutorial are stored in the system knowledge database. These elements are introduced into the system by an expert in the tutorial domain. Based on concepts of the domain knowledge for each tutorial, a division has been chosen to represent the curriculum structure, as efficiently as in systems like BITS [4] or SMODEL [26]. Thus, the student adquires knowledge of the domain at the same time as s/he adquires knowledge in each of the tutorial concepts. Hence, in the curriculum structure of the system the following elements can be distinguished:

- **Tutorial:** The tutorial element includes all the knowledge about a specific tutorial that can be found on a superior level.
- **Topics:** On the upper level, each tutorial is divided into a number of topics. Each topic contains several concepts associated with the information that is going to be taught and an exam that must be passed by the student.
- **Concepts:** The concepts correspond to knowledge units that must be learnt by the student throughout the tutorial. Each concept is made up of a series of textual contents that explain the information, which corresponds to the concept.
- **Selection questions:** Among the textual contents of each concept, a series of selection questions are inserted. Their objective is to evaluate the knowledge adquired by a student on the concept. This type of question consists of a heading and a series of answers to choose from. From them, one or more questions could be correct.
- **Exam questions:** These are the questions contained within the exam topics. They are exercises that the student must solve, which will be evaluated and corrected by a teacher or expert.

4 Designing the CBR Multi-agent System

The system application domain is Health Education, which includes medical training for different participants: medical doctors, nurses and the community in general. The system has been designed to provide rich learning environments to help in the improvement of the decision making process in Health Education. In order to do this, we have identified the main roles played by the system users, the sources of information they consult and the kinds of knowledge they have or require to perform their tasks.

4.1 Using INGENIAS to design the Multi-agent System

As was mention above, INGENIAS proposes a process in which the analysis starts focusing on the main structure of the system, the main agents, the roles they play and how they are organized. Then, the main goals and tasks of the agents are defined. The process continues with the identification of workflows and agents interactions and it goes on until the use cases of special situations are modelled. In this section, we have focused on the system architecture, the agents goals and tasks and how they interact with other agents in their environment (such as the user, resources or other applications).

1) Modeling System Architecture: The architecture of the system is based on the main elements involved in the learning process. From the analysis of these elements, five main agents were defined: client, task,

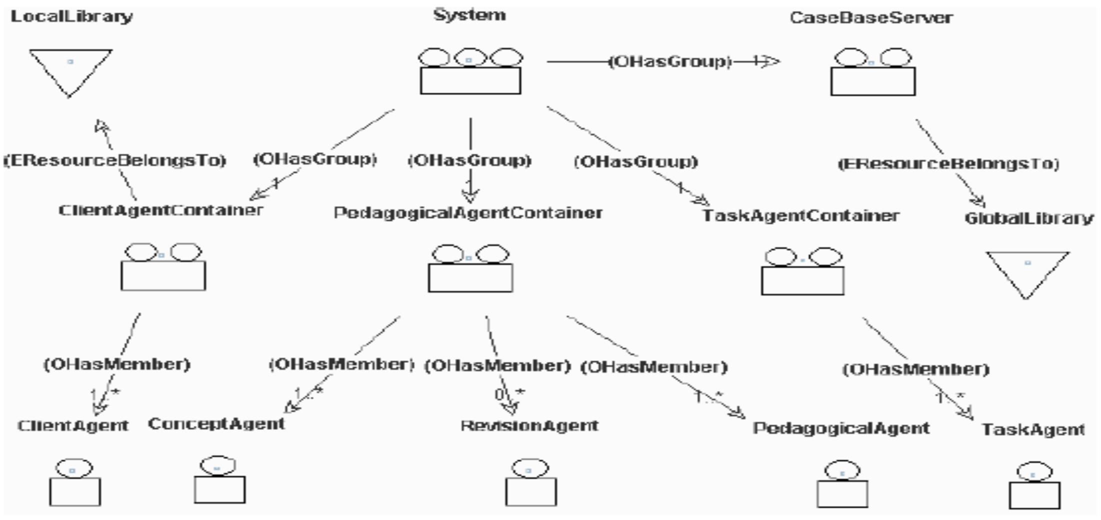


Fig. 8. Organizational Model that illustrates the system architecture

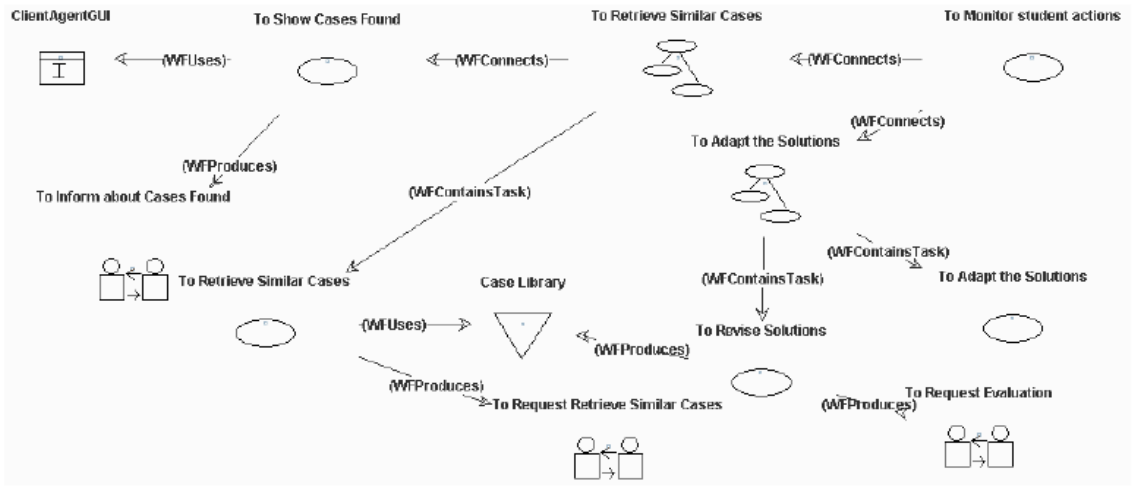


Fig. 9. Workflow diagram that illustrates the main activities and interactions in the system

pedagogical, concept and revision agents. Figure 8 illustrates the organizational model that describes the system architecture. As can be seen, the agents are organized in three containers, client, pedagogical and task agent. This happens because the containers can be each one in a different machine. In the architecture there is also a case base server with a global library, where the knowledge base will reside and a local case library for each client agent container. The system uses this local library to store information about a student (profile and learning path) in its local machine that can be afterwards relevant to update its student model.

2) Modeling Workflows and Agents Interactions: If a student starts a new session, the information in the local case library is extracted by the *client agent*. If the student selects a tutorial, an event is triggered and captured by the *client agent* and a new case is sent to the *task agent*. This agent obtains the students information as cases and starts the retrieval process in order to find cases similar to the new one. When the search has finished, the *task agent* sends a message to the *pedagogical agent* informing it about the cases found. The *pedagogical agent* receives the information about similar cases and adapts the solution for the new case. That is, the *pedagogical agent* infers the state of knowledge of the student and selects among a set of teaching strategies, the most adequate according to the student profile. Figure 9 shows a workflow model of the systems interactions.

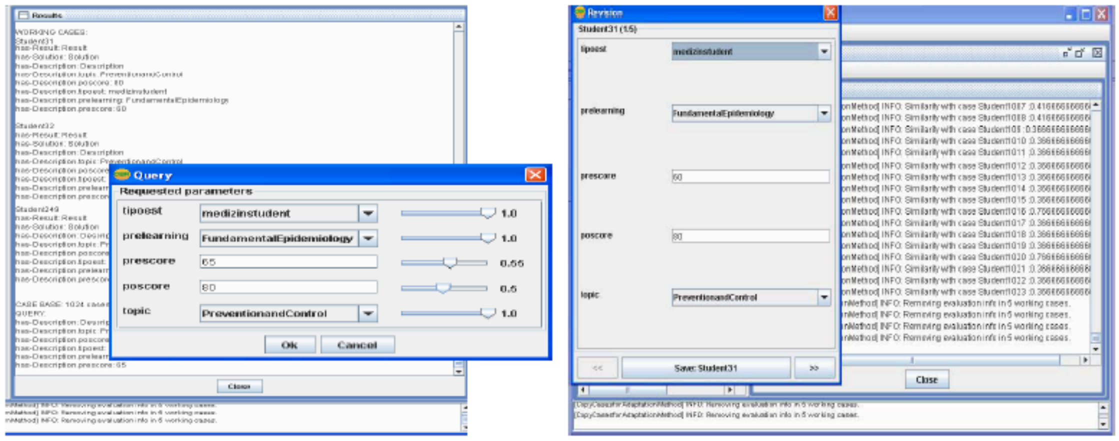
3) Modeling Agents Task and Goals: A very important step in multi-agent systems design is the identification of the goals that agents will follow and the tasks they must perform in order to complete those goals. In our system the main goal is to help the student in the accomplishment of his/her learning process. This goal is composed of other two; the first is to identify the students need and profile and the second is to provide the student with a personalized learning that can help to solve those needs. In order to fulfill these goals, four main tasks are defined: (a) to generate the student model, (b) to monitor the student activities, (c) to retrieve similar cases and (d) to adapt the solutions.

4.2 Implementation of the System

In order to evaluate the feasibility to implement the STIMTutor architecture, we have developed a prototype for training in Tuberculosis Infectious Diseases. The JColibri CBR framework [22] has been used to create our own CBR components. Figure 10(a) shows how the prototype presents information about similar cases found by the task agent during the retrieval process. In this example, the student decides to carry out a learning process in ”Prevention and Control” of Tuberculosis like Communicable Disease. When the student selects the tutorial, an event is triggered and captured by the client agent. This agent obtains information about the student. Then, the client agent communicates with the task agent in order to retrieve the most similar cases to the new student. Finally, the results obtained are revised and adapted in order to obtain new case suited to the student. Figure 10(b) shows revision and retain processes. The results obtained using CBR and MAS were important in STIM-Tutor in the following ways: (a) the evaluation of the student performance helped to decide when to give hints or answers if the student could not answer a question, (b) the student reply history allows the tutor to finish a dialogue and return to the original plan when the student could not continue along a causal link, (c) the category student answer, a part of the student reply history, is effective in helping to decide on different retry strategies. It recognizes near misses and other categories of answers that could be previously treated as totally incorrect, (d) the tutoring history prevented the tutor from giving the same hint repeatedly and (e) the teaching strategy was adequate according to the performance and the students profile. In this sense, an ITS developed under this approach incorporates a better understanding of the learning process and provides richer and more effective instructional experience to the students.

5 Conclusions

Intelligent tutoring systems have significant advantages over existing training methods. ITSs offer constant feedback and help aimed at efficiently bringing students to mastery. By constantly monitoring and maintaining a representation of how the student is progressing, the system can adapt to provide personalized training. Considering that the ITS design is


```
WORKING CASES:
Student31
has-Result: Result
has-Solution: Solution
has-Description: Description
has-Description.topic: PreventionandControl
has-Description.poscore: 80
has-Description.tipoest: medizinstudent
has-Description.prelearning: FundamentalEpidemiology
has-Description.prescore: 60

Student32
has-Result: Result
has-Solution: Solution
has-Description: Description
has-Description.topic: PreventionandControl
has-Description.poscore
has-Description.tipoest:
has-Description.prelearn
has-Description.prescor

Student249
has-Result: Result
has-Solution: Solution
has-Description: Descrip
has-Description.topic: Pr
has-Description.poscore
has-Description.tipoest:
has-Description.prelearn
has-Description.prescor

CASE BASE: 1024 cases
QUERY:
has-Description: Descrip
has-Description.topic: Pr
has-Description.poscore
has-Description.tipoest:
has-Description.prelearn
has-Description.prescore: 65
```

Requested parameters

tipoest	medizinstudent	1.0
prelearning	FundamentalEpidemiology	1.0
prescore	65	0.66
poscore	80	0.5
topic	PreventionandControl	1.0

```
nMethod] INFO: Removing evaluation info in 5 working cases.
nMethod] INFO: Removing evaluation info in 5 working cases.
nMethod] INFO: Removing evaluation info in 5 working cases.
```

Student31 (1/5)

tipoest	medizinstudent
prelearning	FundamentalEpidemiology
prescore	60
poscore	80
topic	PreventionandControl

```
onMethod] INFO: Similarity with case Student1007 :0.41666666666
onMethod] INFO: Similarity with case Student1008 :0.41666666666
onMethod] INFO: Similarity with case Student1009 :0.36666666666
onMethod] INFO: Similarity with case Student1010 :0.36666666666
onMethod] INFO: Similarity with case Student1011 :0.36666666666
onMethod] INFO: Similarity with case Student1012 :0.36666666666
onMethod] INFO: Similarity with case Student1013 :0.36666666666
onMethod] INFO: Similarity with case Student1014 :0.36666666666
onMethod] INFO: Similarity with case Student1015 :0.36666666666
onMethod] INFO: Similarity with case Student1016 :0.70666666666
onMethod] INFO: Similarity with case Student1017 :0.36666666666
onMethod] INFO: Similarity with case Student1018 :0.36666666666
onMethod] INFO: Similarity with case Student1019 :0.36666666666
onMethod] INFO: Similarity with case Student1020 :0.76666666666
onMethod] INFO: Similarity with case Student1021 :0.36666666666
onMethod] INFO: Similarity with case Student1022 :0.36666666666
onMethod] INFO: Similarity with case Student1023 :0.36666666666
nMethod] INFO: Removing evaluation info in 5 working cases.
nMethod] INFO: Removing evaluation info in 5 working cases.
nMethod] INFO: Removing evaluation info in 5 working cases.
nMethod] INFO: Removing evaluation info in 5 working cases.
[CopyCasesforAdaptationMethod] INFO: Removing evaluation info in 5 working cases.
[CopyCasesforAdaptationMethod] INFO: Removing evaluation info in 5 working cases.
```

(a) Case Retrieval in STIM-Tutor

(b) Case Revision and Retain in STIM-Tutor

Fig. 10. CBR Stages in STIM-Tutor

a complex process, we have developed a framework which uses CBR and MAS technologies. The framework presented here was used to create STIM-Tutor, an ITS for training in Communicable Diseases in the Health Education domain.

The approach presented reconstructs the essential characteristics of cognitive tutoring systems and utilizes CBR as an approach to diagnosis the students knowledge, the adaptive generation of problems and the acquisition of new knowledge.

With CBR the designer can construct the desired solution through solved cases and previous experience with students.

The use of agents to automate the CBR stages is also considered significant. In our approach, the multi-agent system distributes the case base and the CBR cycle among several agents. These agents support the formulation and the breakdown of problems and they help in the identification and retrieval of reusable cases. The modular architecture enables to reuse components and the efficient management of the case base.

Therefore, the main contributions of this article are: first a framework which use CBR for the ITS design, adapting learning contents and teaching methodologies to the student profile. Second, the use of multi-agent systems to automate the CBR cycle enabling modularity and component reuse. We also have adapted the algorithms used in the phases of the CBR cycle to be used by agents.

6 References

[1] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.
[2] S. Cern. Intelligent tutoring systems. *Springer Verlag Pub.*, 2002.
[3] H. Chen and Z. Wu. On case-based knowledge sharing in semantic web. *15th International Conference on Tools with Artificial Intelligence*, pages 200– 206, 2003.
[4] V. Devedzic, L. Jerinic, and D. Radovic. The get-bits model of intelligent tutoring systems. *Journal of Interactive Learning Research*, 11(3):411–434, 2000.
[5] R. Ellaway. Modeling virtual patients and virtual cases. *Medbiquitous Annual Conference*, 2005. 10
[6] J. Elorriaga and I. Fernandez-Castro. Using case-based reasoning in instructional planning. towards a hybrid self-improving instructional planner. *International Journal of Artificial Intelligence in Education*, pages 416–449, 2000.
[7] F. Fernandez-Riverola and J. M. Corchado. Employing tsk fuzzy models to automate the revision stage of a cbr system. *10th Conference of the Spanish Association for Artificial Intelligence, CAEPIA*, pages 302–311, 2004.
[8] L. Giraffa and et. al. Multi-ecological: an learning environment using multi-agent architecture. *Multia-Agent System: Theory and Application Proceedings*, 1997.
[9] C. Gonzalez, J. Burguillo, and M. Llamas. A qualitative comparison of techniques for student modeling in intelligent tutoring systems. *ASEE/FIE Frontiers in Education Conference*, 2006.
[10] S.-G. Han, S.-G. Lee, and G.-S. Jo. Case-based tutoring system for procedural problem solving on the www. *Expert Systems with applications*, pages 573–582, 2005.
[11] M. Kharrat, N. Reyhani, and K. Badie. A case-based reasoning approach to intelligent tutoring system by considering learner style model. *Proceedings of the 2003 Ssystems and Information Engineering Design Symposium*, 2003.
[12] R. Koper, B. Olivier, and T. Anderson. Ims learning design best practice and implementation guide. 2003.
[13] O. Kwon and N. Sadeh. Applying case-based reasoning and multi-agent intelligent system to context-aware comparative shopping. *Decision Support Systems*, 37(2):199–213, 2004.
[14] S. Li and Q. Yang. Active cbr: An agent system that integrates casebased reasoning and active databases. *Knowledge and Information Systems*, 3(2):225–251, 2004.
[15] R. Lopez de Mantaras. A distance-based attribute selection measure for decision tree induction. *Machine Learning*, pages 81–92, 2005.
[16] L. McGinty and B. Smyth. Collaborative case-based reasoning: Applications in personalised route planning. *4th International Conference on Case Based Reasoning, ICCBR. Lecture Notes in Computer Science*, 2080, 2001. 362.
[17] A. Montazemi and K. Gupta. An adaptive agent for case description in diagnostic cbr systems. *Computers in Industry*, 29(3):209–224, 2004.
[18] J. Pavn and J. Gmez-Sanz. Agent oriented software engineering with ingenias. *International Central and Eastern European Conference on Multi-Agent Systems*, 2691:394–403, 2003.

[19] O. Rodriguez, A. Martnez, A. Vizcaino, J. Favela, and M. Piattini. Developing a multi-agent knowledge management system with ingenias. 2005.
[20] S. Sibte and A. Raza. A case base reasoning framework to author personalized health maintenance information. *In 15th Symposium on Computer Based Medical Systems*, 2002.
[21] E. Sierra, R. Martnez, Z. Cataldi, and et.al. Towards a methodology for the design of intelligent tutoring systems. *Research in Computing Science Journal*, pages 181–189, 2006.
[22] J. Tomas, P. Calero, and B. Diaz. Jcolibri: An object-oriented framework for building cbr systems. *Lecture Notes in Artificial Intelligence*, pages 32–46, 2004.
[23] M. Urretavizcaya. Sistemas inteligentes en el ambito de la educacion. *Revista Iberoamericana de Inteligencia Artificial*, ISSN 1137-3601(12):5–12, 2001.
[24] K. VanLehn. *Student Modelling*. M. Polson. Foundations of Intelligent Tutoring Systems, 1988.
[25] M. Wooldridge. *Introduction to MultiAgent Systems*. John Wiley and Sons, 2002.
[26] J. Zapata-Rivera and J. Greer. Smodel server: Student modelling in distributed multi-agent tutoring systems. *International Conference on Artificial Intelligence in Education AIED 2001*, 2001.