Cybernetics and Systems: An International Journal
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ucbs20

REACTIVE AGENT DESIGN FOR INTELLIGENT TUTORING SYSTEMS

Ana Lilia Laureano Cruces, Fernando De Arriaga
Published online: 29 Oct 2010.

Cybernetics and Systems: An International Journal, 31:1–47, 2000

0196-9722/00 $12.00 + .00

REACTIVE AGENT DESIGN FOR INTELLIGENT TUTORING SYSTEMS

ANA LILIA LAUREANO CRUCES

Departamento de Sistemas,
Universidad Autónoma Metropolitana,
Azcapotzalco, México

FERNANDO DE ARRIAGA

Departamento de Matemática Aplicada a las Tecnologias de la Información,
Universidad Politécnica de Madrid, Spain

This paper presents a new approach to the analysis and design of intelligent tutoring systems (ITS), based on reactive principles and cognitive models, this way leading to multiagent architecture. In these kinds of models, the analysis problem is treated bottom-up, as opposed to that of traditional artificial intelligence (AI), i.e., top down. We present one ITS example called Makatsiná (meaning tutor in TOTONACA, a Mexican pre-Columbian language), constructed according to this approach, which teaches the skills necessary to solve the truss analysis problem by the method of joints. This learning domain is an integration skill. The classical ITS work is based on explicit goals and an internal representation of the environment. The new approach has reactive agents which have no representation of their environment and act using a stimulus/response behavior type. In this way they can respond to the present state of the environment in which they are embedded. With these elements, errors, and teaching plans, each agent behaves as an expert assistant that is able to handle different teaching

This work was supported in part by the Universidad Autónoma Metropolitana, Azcapotzalco and in part by grant 86325 of the Consejo Nacional de Ciencia y Tecnología (CONACYT), Mexico.

Address correspondence to Ana Lilia Laureano Cruces, Departamento de Sistemas, Universidad Autónoma Metropolitana-Azcapotzalco, Av. San Pablo 180, Col. Reynosa Tamps., 02200, Mexico, D.F., Mexico. E-mail: clc@hp9000a1.uam.mx

methods. Reactive agent programming is found to be simple because agents have simple behaviors. The difficulty lies in the interaction mechanism analysis and design between the environment and the intelligent reactive system.

Intelligent Tutoring Systems (ITSs) have paid a lot of attention to the implementation process (the hardware and software used by the application). However, different considerations concerning the learning process have been ignored or rarely taken into account. One of the results of this way of dealing with ITS has been the use of many engineering resources to produce intelligent machine behaviors without obtaining proportional results. As a result, cognitive sciences, which are interested in the internal aspects of the learning process, are becoming essential for the construction of ITS.

This paper describes a new approach to ITS based on cognitive principles. The approach introduces cognitive task analysis (CTA) and mental models as tools contributing to the analysis and design of ITS. The corresponding implementation is carried out by means of reactive agents. Reactive agents are software programs based on the adaptive behavior of living creatures (Beer, 1990), which were initially used basically for the control of mobile robots (Brooks, 1986). Later they were used for different applications such as problem-solving (Ferber, 1993) and ITS (Laureano & de Negrete, 1996; Laureano & de Arriaga, 1996; Laureano, 1997).

The paper includes a brief history of ITS, considering their patterns of development and their objectives. It also contains an analysis of the characteristics of the main constituents of ITS, according to what has been called the traditional ITS architecture, made up of four components. Afterwards the authors laid out the new approach with the integration of CTA, mental models, and the genetic graph. The reactive features of the agents are also introduced and studied. Later, they were fully incorporated into the approach. Finally, an operative ITS called Makatsiná, designed and implemented according to this approach, is described including its assessment and related work.

DEVELOPMENT OF ITS

The history and development of ITS may be summarized in two periods: the first corresponds to the foundation and crystallization of the technology. The second reflects advances and improvements in their constitutive elements.

Conventional ITS

The first period, 1972–1982, is based on a traditional or conventional architecture, formed by the modules:

1. domain or expert module
2. student module
3. tutorial module
4. user interface.

Artificial Intelligence (AI) techniques are usually applied in one or two of the following domains:

- The system's knowledge of the field,
- The tutorial process and its methods.

Systems representative of this era are SCHOLAR (Carbonell, 1970), SOPHIE I and II (Brown, Burton, & deKleer, 1982), and GUIDON (Clancy, 1982).

Innovative ITS

This period is characterized by different innovations. We can group the ITSs belonging to this era as follows:

Help and Advice Systems. Although the functionality of help and advice systems is similar to that of traditional ITS, they go further in the competence of these systems. They are first brought up as a friendly interface between the user and a system or product unknown to the user. The interface allows the connection or the communication with the system as well as the obtainment of different services, ranging from a simple consultation to getting written documents, to a guide of possible dialogues.

Typical examples are EMACS (Cameron & Rosenblatt, 1992), EUROHELP (Breuker, 1992), UnIX TUTOR (Wang & Kushniruk, 1992), and PANDORA (Faletti, 1982).

Specific Tutorial Strategies Modeling. The need for specializing ITS in different subjects has encouraged the authors to include not only general tutorial strategies but also additional strategies and tactics, more closely related to the learning domain.

A significant number of tutors have been developed which concentrate mostly on subjects such as science and computer science, among them:

- The systems developed by Moore (1992) at Stanford University, to investigate student comprehension concerning variables and function concepts.
- ITSIE, a European project (ESPRIT p2615) developed by Sime and Leitch (1992), concerning intelligent coaching within industrial environments, allowing the use of multiple models for problem-solving. Another tutor of this type is INTZA (Gutiérrez, 1994), which deals with physical processes via simulations.
- ITS-DIITS, developed at Peking University (Yibin & Jianxiang, 1992), whose knowledge base is represented by an AND/OR graph, whose AND nodes include the execution of a series of techniques while the OR nodes involve the selection among a set of tools implemented as production rules.
- DIGITEF, built by the Aérospatiale Company (Choquet et al. 1992) for pilot training.
- The ITS built at the Centre National de la Recherché Scientifique in Marseille (Chouraqui & Inghilterra, 1992) for solution of geometric problems by means of analogies.
- GT-VITA, built at the Technological Institute of Georgia (Chu & Mitchell, 1992), for learning complex control systems.
- ADAPT, a Prolog tutor for the detection of programming errors, which has been built at Duke University (Gegg-Harrison, 1992). Another tutor of this type is CAPRA (Fernández, 1989), which integrates dynamic planning into tutorial process.

Machine Learning Systems. Machine Learning (ML) refers to the development of programs which modify the existing information or build new knowledge using environmental information. The main goals of these systems are:

- understanding the learning process
- providing learning capacity to the computer.

We consider the tutorial by Elorriaga, Fernández, and Gutiérrez to be related to this group (1995). Here an ML application to a student module is discussed. This paper also discusses other self-improving systems perhaps more respective of the type. We would like to add to the ITS of that tutorial:

- CIGOL (Michie & Bain, 1992) developed by Samuel Muggleton for inductive reasoning, working with first-order logic.
- GINESYS (Bratko, 1992), inspired by ID3 (Bratko, 1992; Michie & Bain, 1992) to generate production rules.

Student Modeling. Among the systems in this group we can find

- ACT, from Carnegie Mellon University (Anderson, 1982; Anderson al., 1985), regarding the learning probabilities of different programming rules.
- TACKLE (Dion & Lelouche, 1992), which defines by production rules, the way to acquire programming skills and the errors in the acquisition and use of those skills.
- One of the most interesting systems in this area is the one developed at Osaka University (Kono et al., 1992). Resting on the principle that the student possesses contradictory knowledge, it employs a process for tutorial error correction.

FEATURES OF INTELLIGENT TUTORING SYSTEM COMPONENTS

Domain or Expert Module

This module specifies knowledge that has been obtained from human experts devoted to the cognitive task (CT) for years and who are willing to teach. Research in this field concentrates on how knowledge can be encoded and how to represent the expertise.

There are different techniques for this purpose such as:

- black box
- glass box.

These techniques have advantages and disadvantages that have been documented in the ITS construction history.

Black Box. The name of this technique refers to the black box expert integrated by a systems engineer in a bigger or complete system. It only produces the correct input/output behavior over a range of tasks in the domain and so can be used as a judge of correctness, but it does not know anything about its constituents or structure. When this technique is used to encode the system knowledge, it knows the right answer, which is the correct output corresponding to the applied input. In other words, the system yields the correct answer but one cannot find out the details as to why the answer is the right one.

SOPHIE (Brown et al. 1975) is a typical example. It uses a general purpose electronic simulator to represent and analyze the circuits proposed to the students. The problem is to find the failure modes for each circuit suggested or designed by the system. The system acts as a tutor, using the simulator to determine the student measurements of the circuit functions (voltage, current intensity, etc.) at different points. However, the simulator works with a set of mathematical equations, different from the way humans reason about circuit failure. Therefore, the system is unable to explain the decision in human style detail. Nevertheless, this kind of knowledge representation can be used in combination with a reactive tutor, allowing it to tell the student whether his/her answer to the question is right or wrong, and even if it is necessary, providing him/her with the next right move or action to solve the problem. Later versions of SOPHIE (Brown et al., 1982) included a causal model of circuit failure so that the system could explain its decision in a "human style."

GUIDON (Clancy, 1982), an ITS devoted to teaching infectious disease diagnosis and therapy to medical students, was built in connection with MYCIN (Shortliffe, 1976), an expert system. GUIDON's lack of details and interconnections, all of them necessary for learning, demonstrated the relevance of a cognitive knowledge base and the importance of logical abstractions and relationships. The main operational objective for GUIDON was to examine the 450 MYCIN rules, find the appropriate case, and compare it with the student proposed solution. GUIDON's implementation was not very successful because medical therapy does not work with just a specific set of rules and facts; it also needs causal reasoning and cross references. A later version called NEOMYCIN (Clancy & Letsinger, 1984) included the reasoning and interconnections to MYCINs rules.

However, ITS should provide more useful information—the degree of correctness of a statement, critical points, the student situations, etc.

To accomplish these objectives, different knowledge representation techniques have been developed.

Glass Box. The usual procedure to implement this technique requires the cooperation of a knowledge engineer and at least one domain human expert. The latter has to identify the scope of the problem and also to formalize the key concepts of the domain, leading to a system implementation of the knowledge. Finally, the system is tested and refined iteratively. An ITS implemented with this technique acquires a great deal of articulated knowledge resembling human knowledge. As a result, learning is easier and more successful than in the case of the black box technique. But due to difficulties in eliciting and articulating the expert knowledge, only a few ITS use the glass box technique in the learning process.

One of the lessons learned from the GUIDON project was the importance of knowledge representation in the expert module and the way it is deployed. A good tutor must deploy its knowledge according to the same restrictions that a human does. Clancy's work was very important for this point of view. It illustrated that tutors were limited if they simply ported expert systems from artificial intelligence.

This last insight is the one which has moved us to the use of cognitive models, as will be stated in the description of the new approach.

Student Module

This module contains the ongoing student information for the effect diagnosis of the tutorial process. This information is used to choose the next topic to be taught and which tactics would be the most suitable to teach it, or, in the case of error, which would be the best remedy.

The student module can be split into two components: the database, representing student behavior during the tutorial process, and the diagnosis process for manipulating this database.

Tutorial Module

This module absorbs all the knowledge related to curriculum development problems and the way to teach it. The curriculum means the selection and sequencing of the teaching material. The tutorial process involves the current presentation of this material to the student, according to Halff (1988). Intelligent tutoring systems could use different teaching techniques but, in general, the tutorial interventions should include at least some of the following functions:

1. To control the curriculum and its sequence
2. To answer the user questions
3. To realize when the user needs help and which type of help.

The three main aspects most closely related to the design of this module are the learning style, the teaching style, and the type of cognitive task domain.

User Interface

This is a very important element within the ITS architecture due to its fulfillment of the following basic activities for global system functioning:

- It is the communication bridge between the student and the system.
- It is the only way to perceive and understand the student learning development.

The user interface also fulfills the following didactic activities:

- It represents the interventional means used by the ITS.
- In accordance with the teaching domain, the interface potential has to be fully exploited, using the means necessary to best facilitate the comprehension of concepts and procedures.

THE NEW APPROACH

The new approach will be described with a special reference to differences to the traditional one. We introduce the notion of cognitive models and discuss the process of cognitive task analysis, which leads to the identification of knowledge and skills in a given task.

Cognitive Models

The main objective of cognitive models is to develop an effective simulation of problem-solving within a certain field, from the human point of view. According to this technique, knowledge is divided into components directly related to human classification and use. The merit of this approach is that it provides an expert module in a form that can be easily, and in detail, communicated to the student. Although it is more expensive to develop cognitive models than expert systems, due to the amount of time required for their development and the quantity of details that have to be incorporated, there have been dramatic improvements in the ability of cognitive science to develop such models (Redding, 1989; Ryder & Redding, 1993; Redding, 1992; Wild, 1996), which makes it an attractive option today.

There are three basic questions to consider related to this technique:

- What are the important ingredients for the tutorial process provided by cognitive analysis?
- To what level of detail should components be represented?
- And finally, how shall the different types of knowledge (procedural, declarative, and qualitative) be treated?

The CTA has a basic role, because from this analysis we will get the environmental features to be monitored by different agents configuring the only level we have. This is a very important point to obtain the emergent functionality.

Cognitive Task Analysis (CTA)

According to Castañeda (1993), CTA is represented by an evaluation based on clear descriptions of the semantic, procedural, and strategic expert knowledge or student knowledge. In our case we refer to the expert, the one who possesses the skill. The CTA is a recursive analysis, carried out with the purpose of illuminating the psychological process employed in the cognitive constructions of the skill. The task is recursively divided into more specific subtasks. In order to identify the components of the global task, in each task or subtask it is necessary to add required mental processes. Castañeda (1993, 1994) and Redding (1989) give a more detailed explanation of the CTA.

We assert that there is an important difference between strategy and tactics. With strategy we refer to decisions concerning long-term problem solutions, while with tactics we refer to decisions in short-term problem solutions. Therefore, tactics are always subordinated to strategies.

In a cognitive model, cognition is emphasized, expertise is analyzed, knowledge is evaluated in correlation with all the work, and expert models are considered.

According to Ryder and Redding (1993), CTA is recommended for the following types of work:

1. Tasks with a very high degree of cognitive complexity in problem-solving and decision making;
2. Tasks requiring a great deal of workload or attention-switching;
3. Tasks that involve high performance skills (i.e., require massive amounts of practice to become expert);
4. Tasks that require large amounts of information to be assimilated during training;
5. Tasks that experts have considerable difficulty verbalizing or demonstrating through other actions;
6. Tasks that lead to considerable variability among individuals due to the number of cognitive performance strategies available.

CTA considers the following points of interest for the development of ITS:

- The complex interdependencies among cognitive process, structures, and strategies, forming a certain task and consequently the teaching of this CT.
- Identification of concepts and/or necessary skills for the execution of the task and also effective ways for their teaching.
- Identification of implicit concepts, enabling successful teaching by reducing the effects of the existence of several skills that have to be combined and where the absence of any of them can lead to the inability of learning this CT.
- Teaching is delicate due to the fact that there are no rules for building situations leading to the magical learning of a cognitive task.
- Even with an appropriate and well-constructed CTA, it is difficult to recognize the components necessary to create teaching tactics as this is done using insight and expertise as well.
- Another important issue is the difference between the states of novices and experts. It is frequently stated that these differences are

represented by relationships created during the task realization where these relationships have been learned over time and reaffirmed with practice. One way of dealing with these differences is to establish hierarchical difficulty levels for the realization of the task.
- It is also necessary to consider the changes of the different states within the intellectual development of the "students" during task learning, as well as the history of the changes of these states and the level of knowledge before starting the learning process.

Obtainment of CTA

Having explained the importance of the CTA, we will now proceed to give a short explanation of the basic points of its obtainment, according to Castañeda (1993). It will be developed within a table with columns as follows:

- development steps
- content of steps
- representation type
- evaluation forms
- complexity of the underlying processes of the CT development.

The information contained in these columns will be analyzed at each stage of the current cognitive task analyzed—in our case the method of joints.

Development Steps: we will begin by identifying the steps required for task development; these steps will guide us towards the discovery of the different knowledge sets that by mutual interaction will constitute the tutorial modules. To do so we will use a *mental model.*

Content of Steps: this column will include the type of knowledge for each step, classified as 1) factual, conceptual, or procedural knowledge, or 2) strategic or tactical knowledge. This column plays an important role due to the fact that its obtainment can be based on declarative or procedural knowledge according to a knowledge tutor or skill tutor. This column will help us to find the best knowledge representation within the system.

Representation Type: this column follows the one devoted to step contents. Its function will let us find out the type of instructional plans as well as the related instructional tactics necessary for the student to learn a specific knowledge or skill.

Evaluation Forms: competence for the different abstraction levels is required to evaluate the specific activity within the tutorial. This part is represented by communication among the different knowledge sets, if communication exists.

Complexity of the underlying processes of the CT development: complexity will be classified in this column as simple or multiple discrimination, knowledge integration in micro or macro structures, generalization, and low or high complexity in problem-solving. This step will allow us to find essential elements to structure the instructional didactics, which leads the student to learn the knowledge or the required ability for the pursuit of the specific task.

Supplementing the CTA

According to Ryder and Redding (1993), the CTA should be supplemented by a recursive analysis of the CT. This recursive analysis is represented by three stages. The initial analysis stage (orientation) is devoted largely to developing an overall understanding of the job and components of expertise that comprise the job, and to determining the methods for analysis of each component of expertise in subsequent stages. The intermediate stage (basic analysis) is devoted to the analysis of component performance. The final stage (skill acquisition and refinement analysis) is devoted to progression analysis of skill acquisition from novice to expert. The three progressive analyses are only carried to the depth necessary to support instructional design.

The progressive analysis of these states will be carried out according to three components representing the expertise (knowledge, skills, and mental models). This analysis will allow us to obtain new significant data as well as restrictions, such as finding out whether a data flow exists among tasks (i.e., the case of multitasks). In the following paragraphs each component will be explained.

Knowledge. We have already discussed this concept in the CTA where the column content of steps has represented it. Here we will further its definition. Besides domain concepts and relationships, this component includes rules and procedures to complement the work in verbal form.

Skills. This component includes every kind of procedural knowledge, as well as the skills that are required for the work to be taught. It also includes the associated type of learning and development strategies.

Ryder and Redding (1993) put some emphasis on doing this analysis by considering the work as a whole, where these skills may or may not correspond to individual tasks or be part of one single task.

The skill component is analyzed by dividing the task into a sequence of steps. In the case of a cognitive procedure, that division will be based on mental processes. This is the same as the column of development steps from Castañeda.

Ryder and Redding (1993) summarize their skill taxonomy as:

- Perceptual skills or pattern recognition skills, referring to the identification and classification of information coming from the senses (e.g., visual).
- Decision-making skills or strategy skills, referring to decision-making in problem-solving; they include central processing and deal primarily with verbal data and/or stimuli, which are not predictable.
- Gross motor skills, referring to motions of standard components where the development is guided by kinesthetic cues.
- Perceptual motor skills, referring to continuous motions or motions whose control depends on dynamic perceptual inputs, i.e., they include perceptual and motor components.
- Procedural skills, referring to a sequence of motor or cognitive actions in predictable situations and having low cognitive demand in the case of motor actions.
- Interactive skills, including interpersonal skills such as communication, persuasion, and supervision.
- Skill integration and time-sharing, referring to the integration of several skills in one single task and also to the attention-switching strategies in complex multitask environments.

All of the above are based on the study of conditions necessary to transfer from a single to a double task. This type of study is important in work with a very large number of conditions and where there is also the need for coordinating tasks and skills.

We also consider an additional type of skill, important in the development of cognitive skills dealing with problem-solving:

- Tactical skills that are allocated to a level subordinated to that of strategy, and represent decision-making for the immediate resolution of problems, once the strategy subordinating these tactics has been chosen.

Following this classification it is necessary to take into account

1. the conditions for improving their acquisition (e.g., coach),
2. testing methods (e.g., to conceal the explicit characteristics of the problem),
3. automatic teaching techniques (e.g., reactive teaching),
4. analysis techniques (e.g., according to the different types of CTA).

Mental Models. Mental models are defined as functional abstractions with the work or works that can set a didactic framework for problem-solving (Ryder & Redding, 1993). A mental model is distinctive from other forms of knowledge representation, such as semantic networks or formal rules, in terms of its structure. Johnson-Laird referenced in Wild (1996) says that a mental model makes those objects or entities explicit, whose properties and relations are relevant to potential action. Thus each entity is represented by a corresponding token in a mental model. The properties of entities are represented by the properties of their tokens; the relationships among entities are represented by the relationships among their tokens.

Within the specific domain of a particular task a mental model contains

- conceptual knowledge, with respect to the situation of the system,
- procedural knowledge, about how to use the system or act in a specific situation,
- decision-making skills for analyzing the system or situation,
- strategic knowledge, about when and why different procedures and decision-making skills should be used and how the tasks components interact or are related. Here the tactical knowledge is also considered.

Mental models are important because they keep aware of the global work situation and allow inferences to be found in the task domain.

Therefore, mental models are particularly important in complex cognitive tasks.

While no exact mental model of an expert can be obtained by conducting a CTA, the features set derived may be used as a framework for the teaching domain.

A mental model is represented as instructions and may include clear and suitable restrictions and features of the task. Often a mental model is represented at different abstraction levels making important principles and conceptual relationships explicit, which would otherwise be difficult to understand.

The model layout could be a visual schematic such as maps, Venn diagrams, flow diagrams, structure diagrams, pseudocode, etc.

In summary, we may conclude that the psychological notion of information represents a very useful tool for ITS development. It allows for the discovery of the main building blocks, such as different instructional plans joined to their instructional tactics to reach the needed abstractions according to the difficulty of the task's hierarchy, or to find out the different critical points in the general strategy and the competence of the different skills identified to be required for the realization of this task.

Although it is difficult and time-consuming to build good cognitive models, carrying out a good CTA is useful in order to identify the knowledge and implied skills in complex cognitive tasks.

The Genetic Graph (GG)

The GG is a tool to represent knowledge. It was introduced by Ira Goldstein (1976, 1977), based on the genetic epistemology of Jean Piaget (Ginsburg & Opper, 1986). From a general point of view, the GG covers knowledge of any kind, in clusters or islands, and links to relate them. These relationships could be order or inclusion relationships, as in the case of Gagné's nested hierarchies (Gagné, 1985). The GG can also register the history of a student's learning style, by taking into account the visited islands and the frequency of some of the links used.

The links used in this graph could be extended according to the needs of the domain to model. Bretch and Jones (1988), Fernández (1989), Laureano (1997), and Bermond (1993) present different uses according to their needs. Next we are going to present the meaning of the links used during the development of Makatsiná.

- PreCond- this link involves a precedence order, before-that.
- PostCond- this link involves a subsequent order, knowledge to the access after-by studying the knowledge to which it has been linked.
- Comp component- this relationship implies that one knowledge or skill is composed of another component.
- Class- involves the existence of a conceptual hierarchy or skill hierarchy.
- SubClass- involves the existence of granularity levels in the definition of conceptual or skill abstractions.
- ItIs- this link represents the definition of a specific component according to the domain.

Besides determining the order of execution of the task, the links among islands imply the relationships as well as input and output data, which will exist among islands and different abstraction levels, if they exist. These abstraction levels represent the execution of the ITS. This part is also related to the obtainment of critical points of the general strategy and the competence at different levels, in terms of teaching.

THE IMPLEMENTATION INSTRUMENT: REACTIVE AGENTS

This paper starts out with the assertion that ITSs are systems operating in an unpredictable and very complex environment; that is, ITSs are very difficult to model because they contain the instruction for an intellectual or cognitive domain. Consequently, we argue in the following that they are systems which should have reactive features.

During the last 10 years, distributed artificial intelligence (DAI) has undergone significant improvements; multiagent architectures have been produced and among multiagents, we can find reactive agents (Ferber, 1993; Ferber & Drogoul, 1992). Our work has been inspired by one particular field of AI called reactive robotics (RR), which aims at the creation of autonomous robot models that can work without human guidance. This paradigm attempts to achieve complex objectives by means of very simple actions represented by agents, where an agent is thought of as an expert assistant (Hayes-Roth, 1997) monitoring the student's development in a CT.

Reactive agent architectures were introduced in the 1980s (Beer, 1990; Brooks, 1991a, 1991b; Maes, 1993a). Their main characteristics are

- dynamic interaction with the environment;
- internal mechanisms allowing work to proceed with limited resources and incomplete information (Sombé, 1993).

The main objective of these architectures is the emergence of functionality; in other words, the combined behavior of some agents cause the system behavior to emerge, depending on the intensive interaction of the system with the environment. According to Brooks (1991b), the world is the best model for an intelligent system, and this idea forms the suitability criterion in the new conceptual frame of the behavior-based system design.

The agent behavior does not explain the full working functionality of the system, meaning that functionality depends a great deal on the dynamic environment properties and the interaction between agents.

Consequently, we are interested in modeling the dynamic environment characteristics.

This kind of design philosophy leads us to the following consideration. With this type of architecture, it is not possible to tell agents in a simple form how to reach an objective; instead, in the design phase we should be able to find which environment features have to be taken into account to design an interaction loop between agents and environment, allowing the convergence toward the objective we are aiming for.

This way of considering system execution entails that agent action toward an objective not be predetermined, but will depend on the sequence in which environment characteristics are going to be detected, as well as on the final objective. This new form of control requires distributed implementations, due to local parallel interaction among agents themselves and between agents and the environment.

The selection-action dynamics for this type of system will emerge in response to two basic aspects—the environmental conditions and the internal objective of each agent. As a consequence, we can conclude that the selection-action of agents has to be carried out in real time.

However, it is worthwhile to say that sophisticated behaviors depend much more on system-world relationships than on particular sensor readings. According to this, Kaelbling (1987) proposes that agents could have a common environment representation, and García-Alegre and Recio (1998) suggest that agents on a particular level should share a world representation with the degree of abstraction required to perform the embedded tasks. Such a representation entails a specific spatial-temporal window to the world that has an ever finer granularity as it proceeds to the level, wherein agents deal with physical devices (sensors and actuators).

The authors' work sets up a new multiagent architecture (Laureano, 1997; Laureano & Arriaga, 1997; Laureano & Arriaga, 1998), related to those of Brooks (1986, 1991a, 1991b), Kaelbling (1987), and García-Alegre et al. (1995).

Considerations with Respect to the Cognitive Environment

The environment of a robot is a physical one; in contrast, the environment of an ITS is cognitive. Therefore, there is no precise one-to-one mapping between their respective elements. We will try to adapt the RR philosophy by importing those ideas which can help to model an ITS multiagent architecture.

Behavior Levels

Each level encapsulates an agent set with shared perception language and abstraction. For robots, the behavior levels fulfill the function of determining complex behaviors from simple ones by abstraction granularity, but in the concrete case of ITS we do not usually have these levels. Intelligent tutoring systems frequently has a simple level related to a complex behavior, which is to teach, and according to Kaelbling (1987) and García-Alegre et al. (1995), we have a stronger dependence on the environment than on particular sensor data. The problem is how to integrate sensor data into the agents to produce coherent behavior.

Reactivity

But where? In traditional ITS, the tutorial module is in charge of what will be taught and how. Interconnecting the tutorial module with the other ITS modules usually accomplishes this.

But according to what has been stated, to get the emergent functionality of the agent we have to provide an intensive environment system interaction. The robot sensors are the elements that register environmental changes, but in an ITS, the task is more difficult because the cognitive state of the user representing our environment cannot be sensed. The way to monitor the user's cognitive state is by observing its development in the task to be taught. We do that via the user interface and the cognitive analysis of the task we wish to teach.

MAKATSINÁ: AN EXAMPLE OF AN INTELLIGENT TUTORING SYSTEM

Makatsiná is an ITS developed according to the proposed cognitive approach, implemented by means of a multiagent architecture. Makatsiná's objective is to reinforce the learning of the strategies and tactics for the solution of truss problems, previously explained to the student in the classroom. Makatsiná acts like a coach, informing the user, among other things, about where errors are made and how many there are, their classification and level, and their corrections and remedies. One of the main problems of its design has been the obtainment of the necessary subskills (often designed as skills) for developing the complete skill (cognitive integration task).

Makatsiná Domain: Mechanics of Nonrigid Bodies

Mechanics can be defined as the science describing and predicting the conditions of a body in motion or at rest under the action of forces. It has been divided into three parts: rigid body mechanics, fluid mechanics, and deformable body mechanics. The first one is also divided into static and dynamic (Beer & Jhonston, 1980). Mechanics constitutes the general background for most engineering technologies; therefore, its study is a compulsory requisite for engineering students. Although it is an empirical science and, more than that, an applied science, it has an important theoretical core allowing fruitful deductive reasoning.

The design of a structure consists of two parts. The first part deals with the determination of forces at any point or member of the given structure and the second part deals with the selection and design of the suitable selection to resist these forces. The first part is termed structural analysis, the second part structural design.

Makatsiná's domain belongs to static structural analysis. The selected structure is a plane truss, and the classical structural analysis method to achieve it is the method of joints.

Trusses

Trusses are framed structures whose members are straight and connected by frictionless pins. The axes of their members that frame at a joint intersect at a point. Trusses are loaded by concentrated forces acting upon their joints. They are subjected only to axial forces, inducing a uniform state of axial tension or compression.

The basic form of a truss is a triangle formed by three members joined together at their common ends forming three joints. Another two members connected to two of the joints with their far ends connected to form another joint form a stable system of two triangles. If the whole structure is built up in this way it will be internally rigid (Figure 1).

Analysis by the Method of Joints

A member of a truss is subjected only to an axial tensile or compressive force. In general, the internal forces exerted by the members of a planar truss on a joint and the external forces acting on this joint must satisfy two equations of equilibrium. For a joint of a planar truss these

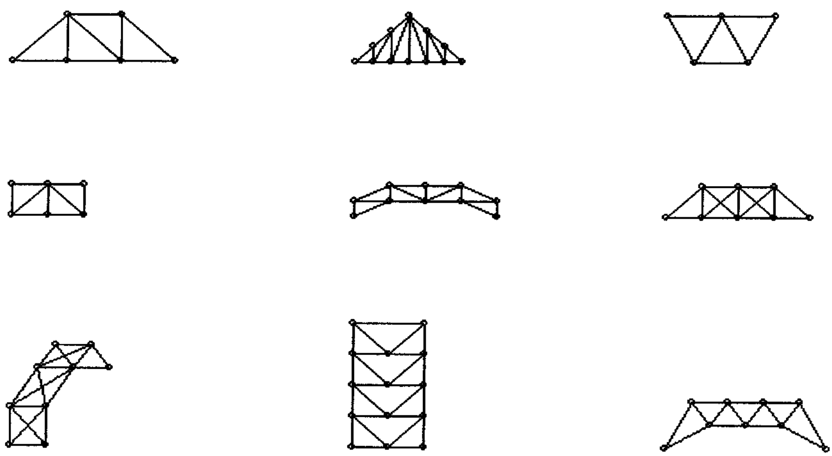

Figure 1. Common types of trusses.

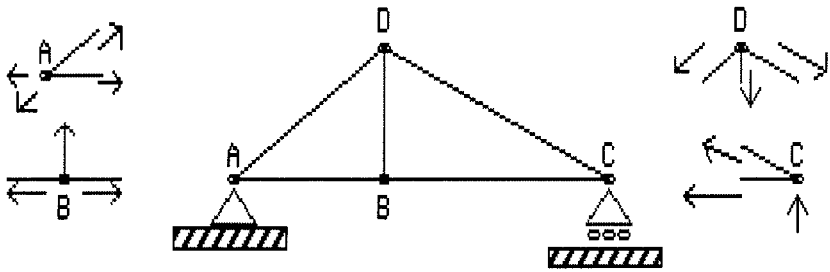

Figure 2. Free body diagram of joints.

equations are

$$\sum Fx = 0 \quad \text{and} \quad \sum Fy = 0.$$

That is, for a planar truss with NJ joints (where NJ represents the number of joints), we have a system of 2NJ equations of equilibrium. For a statically determinate truss, the number of independent equations of equilibrium is equal to the number of unknown reactions plus the number of unknown internal axial forces in their members. Consequently, we can establish the reactions of statically determinate trusses and the internal forces in their members by drawing the free-body diagrams of their joints, writing the equations of equilibrium for each joint, and solving the resulting system of linear, algebraic equations (Figure 2).

Once the reactions of a simple truss are established, a joint can be found which is subjected to, at most, two unknown internal forces. Thus, these forces can be evaluated by writing and solving the equations of equilibrium for the joint. The remaining unknown forces in the members of the truss can be established by considering the equilibrium of its joints one after the other. Care must be taken to select joints for which the forces of all, but at most two, of the members framing into them have previously been computed. This approach is referred to as the method of joints. The analysis of a truss can be simplified if one is able to establish by inspection which of its members are not subjected to equal internal forces and which of its members are subjected to equal internal forces.

The chosen domain is a skill integration domain discussed in Laureano (1992) and Beer and Jhonston (1980). To find the complete solution of the domain problems requires a high level of expertise which is in contrast to the basic knowledge that is needed—Cartesian plane, static vectors, and basic physics. This was the reason for the authors' interest.

Skills

Skill acquisition has been postulated as the assimilation of three states, according to Anderson et al. (1981) and Rassmusen (1986):

1. The verbal learning of knowledge in the form of elementary rules;
2. Task components are combined and gradually automated during skill learning;
3. Task automation is developed at the highest expertise level; thereafter, conscious attention is no longer required for execution of routine tasks.

Several experienced consultants and lecturers were cognitively interviewed; different problem-solution sessions and discussions were held, and protocols were analyzed to fully characterize the learning environment including several expert mental models. Figure 3 shows the most important one.

Makatsiná Approach

Makatsiná belongs to the group devoted to specific tutorial strategies, and is a skill tutor, where AI has been fundamentally integrated into the tutorial process principles. Because the field knowledge has been included in a compiled expert, this method is called compiling the expert out (Anderson, 1988). This method consists of performing in advance all possible computations of the expert for a particular problem and to store it in some efficiently indexed scheme. Unfortunately, such a system can only tutor a specific set of problems. This method is successful in some applications, such as in Makatsiná where the tutor has reactive characteristics, so we need to know the next step before the user can do it.

Makatsiná uses the cognitive model to design the expert module, according to the research carried out by Castañeda (1993) and Ryder and Redding (1993), and according to the latter authors, our ITS has two stages (Table 1, Table 2, and Figure 4).

If a triangular structure is isostatic and stable ***then*** **step 1**

    Repeat
        Use simplification rules (are internal forces equal?) **step 2**
    Until none of the simplification rules can be applied

    Repeat
        Without evaluating reactions obtain the greatest number
        of results **step 3**
    Until the reactions are needed

    If reactions are needed ***then*** **step 4**
        Evaluate them
    end-if

    Repeat
        Evaluate the remainder axial forces
    Until knowing all the forces in the bars **step 5**

else
    There is no solution by this method **step 1**
end-if

Figure 3. Main mental model for the truss solution by method of joints.

Table 1. Specification of the CTA for the Truss Analysis Problem Taking into Account its Skills

Development steps	Steps content	Representation type	Evaluation forms	Complexity of (c) the underlying processes of the CT development
S.1 To know if a triangular structure is isostatic and stable	Strategic and procedural	Rules and structures	Problems and MiniTests	Simple and multiple discrimination
S.2 Use simplication rules	Tactical and procedural	Rules and structures	Problems and MiniTests	Generalization
S.3 Obtain the greatest axial forces without evaluating reactions	Tactical and procedural	Rules and structures	Problems and MiniTests	Generalization
S.4 Evaluate the reactions	Tactical and procedural	Rules and structures	Problems and MiniTests	Simple and multiple discrimination
S.5 Evaluate the remainder of the axial forces	Tactical and procedural	Rules and structures	Problems and MiniTests	Generalization

Taking into account Urretavizcaya's work and notation (1991), the student module built in Makatsiná is implicitly individualized for each student by means of the on-going information acquired during the user/machine interaction. The model is basically qualitative because the student is described in terms of relationships to different kinds of knowledge. Its precision depends on the evaluation models it uses by means of order relationships (qualitative grades). The student model is

Table 2. Skill Taxonomy, Knowledge Type and Relationship to the Main Mental Model (Ryder et al., 1993)

Skills (c)	Conceptual knowledge	Mental model
Strategic	Static	Step 1
Perceptual, tactical and procedural	Basic physics	Step 2
Tactical and procedural	Static	Step 3
Tactical and procedural	Static	Step 4
Tactical and procedural	Static	Step 5
Strategic	Static	Step 1

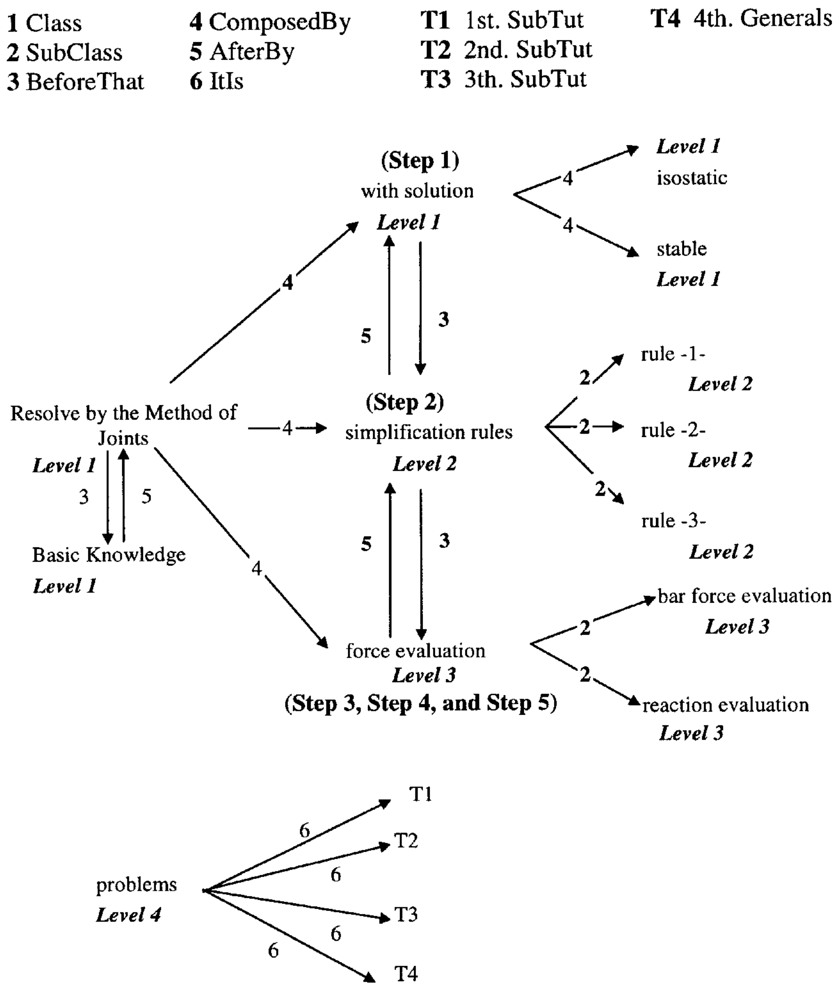

Figure 4. Conceptual domain diagram based on the GG.

an ad-hoc model because its knowledge base is built with specific aspects of the cognitive task to be learned; also, the student model is differential, due to the contained differences between the expert and the student which are implemented as errors.

As far as the diagnosis techniques are concerned, due to its tutorial purpose (to act as a coach) (Burton & Brown, 1982), and to the fact that we only have a single strategy for teaching the solution of truss analysis problems, model-tracing (VanLehn, 1988) is followed. Issue-tracing

(VanLehn, 1988) is also used after considering the features of the ordered procedural knowledge for the solution of problems, divided into phases and related to subtutors. Each subtutor surveys the student use of different skills within its related microworld.

Makatsiná has emphasized the development of the control interface by ideograms and tutorial interventions, as well as student interactions, by means of graphics. As for the use of multimedia, we have considered the domain to be completely suitable for association learning as discussed in Laureano (1993b). Animated visual effects, colors, and sound to indicate errors, or to congratulate in case of success, have been fully utilized in Makatsiná.

Let us proceed in determining how many skills are necessary to achieve skill competence in this CT.

MAKATSINA IMPLEMENTATION

In the following sections we will describe the way to get the skills by integrating this cognitive task, as well as the implicit complementary information needed to solve the problem.

Why a Single Competence Level?

According to Garcia-Alegre and Recio (1998), a level is a set of agents that shares a common language made up of a set of terms related to perception, knowledge representation, and actions. A level gravitates around a specific representation of the system environment. In the case of the ITS, we only have one environment representation (the interface). For cognitive processes it is the only way to perceive and understand student development. In other words, we can't see the interior of the mind. Therefore, we have only one conduct, to teach. The conduct complexity will be treated within this level.

The multiagent design and analysis for the ITS is implemented in the tutorial module. One of the main contributions of Kaelbling's architecture (1987) is the hierarchically mediated behaviors. The mediator decides what to do on the basis of a weighted average of the outputs of the subbehaviors and their respective degree of urgency. In a recursive form, a level belongs to a complex conduct. The outputs of subbehaviors, as well as the world model and other perceptual data, are handled by a mediator that combines them to produce the final conduct.

In Makatsiná the teaching conduct was broken down into subbehaviors, but these do not need the inclusion of a mediator. They all represent the teaching of subskills which will be monitored separately due to the fact that the CT is hierarchical. These subbehaviors are represented by subtutors.

For each subtutor we have used Brooks's (1986) basic architecture, perception action, but in our case the dependence of actions on behavior levels no longer exists, due to the implicit hierarchy of the task to be taught, so communication among modules is not necessary.

Kaelbling (1987) also mentions that complex conducts depend on global conditions. She proposes a general mechanism to synthesize the information of different sensors to get information about the world. This point of view is also shared by the architecture of García-Alegre and Recio (1998) for complex conducts. In Makatsiná the student model represents this requirement.

Number of Necessary Agents

First of all, the corresponding CTA has been obtained. More details can be found in Laureano and Arriaga (1996), Laureano (1997). The expert mental model and the domain conceptual graph (represented by means of the GG) represented in Figure 3, Table 1, Table 2, and Figure 4, have been used to obtain the number of skills (subtutors) used to divide the problem, according to the following procedure.

First. We have obtained the representation of the CT mental model shown in Figure 3. Afterwards, we have found the steps for the development of the CT necessary to find the other CTA components whose data are synthesized in Table 1. We would like to point out that in the column showing the content of steps, every step includes procedures and strategies, suggesting the idea of skill competence.

Second. The CTA has been supplemented with the suggestions of Ryder & Redding (1993). In this analysis we aim to identify the number and type of skills, based on the mental model (Figure 3), the skill taxonomy of Ryder & Redding (1993), and also the type of conceptual knowledge related to each step in the mental model. The skill groups of different types are synthesized in Table 2.

Detected Skills

From Tables 1 and 2 we obtain the first approximation to the skill number. There are four associated steps of the mental model. They are:

1. To detect the physical characteristics, to find out whether it is possible to use the method (strategic skill) (Step 1).
2. To know and have the ability to apply the bar simplification rules (perceptual skills, tactical skills, and procedural skills) (Step 2).
3. To know when and how to obtain the reactions (tactical skill and procedural skill) (Step 3 and Step 4).
4. To have enough knowledge in mechanics to evaluate the remaining forces in the bars (tactic skill and procedural skill) (Step 5).

This CTA is represented in Table 2.

Third. From an analysis of the domain conceptual diagram (Figure 4) and by observing that rules 1, 2, and 3 are subclasses of the skill simplification rules, while the bar force evaluation and the reaction evaluation are a subclass of the skill force evaluation represented by the link number two, we deduce that a subtutor will monitor each class representing a skill.

Two concepts (isostatic and stable) immersed in the skill to detect physical characteristics are needed to find out whether the method could be used. Therefore, they make up the subtutor teaching how to detect these characteristics.

The chain of three subtutors can be deduced from the previous reasoning to monitor the skills for the development of CT.

From the Previous Reasoning We Can Deduce the Necessity of Three Subtutors to Monitor the Skills for CT Learning. T1, T2, and

T3 represent the problem domain, which each subtutor will use to test its skill. T4 represents the complete set of problems that the system accesses when a teaching session starts (represented by the link number 6). The basic components are a prerequisite to learn the global skill and knowledge—Cartesian plane, static vectors, and basic physics—which are not dealt with by this tutor.

There are two links, numbers 3 and 5, implying the access order to the skills. In the case of BeforeThat, it implies a skill that has to be mastered before entering another island. However, it does not imply an order but rather represents a prerequisite. On the other hand, the link AfterBy forces an access order. In the particular case of learning the solution skill of trusses by the joint method, it is necessary to integrate skills in that order. If the user has the skill competence of one, or even all of them, it would not mean she/he could have the whole task competence skill.

In this particular case, levels represent the necessary capacity of one of those skills to reach the following level. In other words, learning a different skill does not mean learning a higher problem complexity, due to the fact that learning the integrated skill guarantees solution of simple and complex problems.

Considerations on the Cognitive Domain and Its Teaching

In the following sections, we will explain how the tutorial modulus reactivity was implemented based on the needs of these three subtutors, and how the environment characteristics were obtained. Remember that from reactive principles, we derived the need to obtain the environment dynamic characteristics to be monitored in order to get the system reaction and, therefore, the emergent functionality.

When we shift to ITS, the cognitive environment leads us to consider other important aspects. Next, we will analyze the importance of the domain representation which, in our case, is the effective teaching of certain skills.

Expert vs. Novice

Many things could be said concerning the differences in problem-solving between experts and novices. During the last decade, AI and cognitive psychology have investigated a set of fundamental aspects related to the subject, such as the quantity of knowledge that could be organized and how this organization can play a very important role in the efficient recognition of the application of knowledge during problem solution.

Both branches of knowledge have reached the conclusion that the expert solution to the problem depends much more on having the appropriate knowledge of the specific domain than on any extraordinary intellectual skills.

Besides, other studies have shown that experts not only know much more on the subject but also that there is a qualitative difference concerning the organization of knowledge and the way that the knowledge is used (Elio & Scharf, 1990).

There are two main differences between the way experts and novices solve problems in the field of structural mechanics.

First Difference

Novices start to work on the solution by choosing a way that allows them to solve the unknowns and the equations step-by-step. This forward method of solving is carried out iteratively, until all the unknowns are solved or until they realize that this way is not suitable. In this case they have wasted a lot of time.

Experts also carry out this forward solution. By starting from known variables, they manage to generate equations with unknowns, which become known after the solution. The basic difference is that novices are not able to identify those aspects that facilitate the task, producing one solving path without surprises.

The great difference between novices and experts shows us that there are qualitative differences in the inference process, which generate information that cannot be explicitly found in the problem and which is fundamental for the solution of the problem. This information represents in RR the dynamic characteristics of the environment allowing the system to react. To achieve it in ITS, we think of errors. When the student begins learning the skill, some errors are due to not properly identifying the things which can be found implicitly. For example, the principles of general physics, if properly used, help to determine whether the structure is stable or statically determinate or indeterminate, and the capacity of the power of the simplification rules in structural geometry is also useful. Therefore, it is extremely important that the student recognize these aspects. The error characterization gives us the way to generate the dynamic characteristics of the cognitive environment; these errors are identified and classified in Table 3.

Second Difference

This difference is represented by the knowledge and knowledge organization to solve the problem. Novices classify the problem according to superficial aspects, thereby obtaining a very informal representation. On the other hand, experts classify the problem according to deeper aspects. They have a stronger mental scheme, allowing a qualitative analysis marked by domain inferences that generate additional useful information about the problem situation that was not explicitly stated in the problem statement. The experts can concentrate on principles of physics and the restrictions on applying them (Elio and Scharf, 1990).

One of our goals in splitting the tutorial module was to be able to represent deeper aspects based on the information that cannot be explicitly found in the problem, and which only the expert gets, in order to be able to convey them.

Knowledge Representation

A tutorial process model with knowledge representation based on the knowledge units to represent the expertise is required, in order to monitor the solution of the problem.

The Makatsiná goal is that the user should be able to acquire the procedural and tactical knowledge of problem-solving skills; that is, to be able to create patterns of specification that show which particular actions should be considered, and under which particular conditions these actions have to be taken into account and carried out. In the field of robotics, this fact could be understood as the capacity to react when faced with certain characteristics, which will be synthesized by different agents according to their specific functions.

In other words, this is equivalent to finding the dynamic characteristics of the environment which triggered the agents' actions.

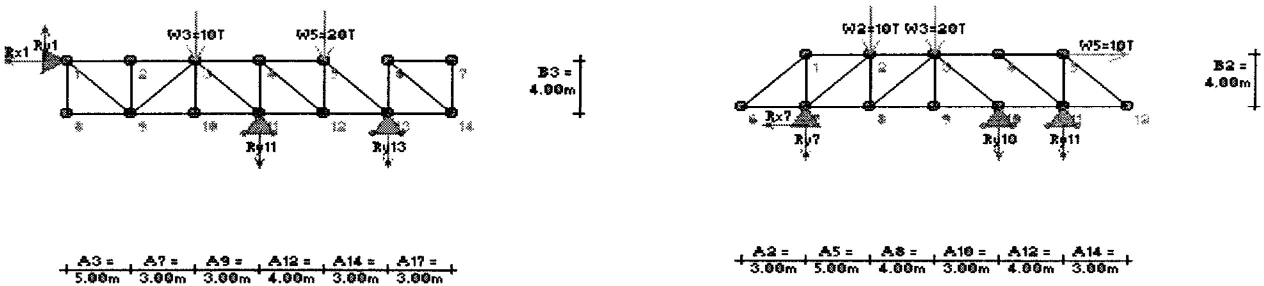

Figure 5. Trusses with more than three reactions.

Anderson et al. (1981) proposed a pattern generalization process that compares features of both problem statements and solutions, thus forming some new unit of knowledge that associates common problem features with goals and subgoals. This is a way of dividing the tasks into smaller blocks, marking fundamental aspects of the task. These blocks correspond to single aspects to be learned by the user. This is the key to teaching the student to discover implicit aspects of the problem. This is a fundamental part of the instructional process development.

Agents (subtutors) in our multiagent architecture represent the knowledge units coming from the mental model and all the analysis process, while the implicit aspects are represented in the sets of errors monitored for each agent.

ERROR TREATMENT

The errors were obtained through global considerations of the task. Once the agents and their capabilities were chosen, those potentialities were spread out among them. Once again, we have to point out that Makatsiná is an ITS interested in helping the student to learn the use of a determined strategy. Consequently, it is necessary to treat the errors from a didactic point of view.

A Common Error

A common error in truss analysis is to think of a truss as being statically determinate or indeterminate without considering the relation between the number of joints and the number of reactions.

Many students have the very common idea that if a truss has more than three reactions it is statically indeterminate. The students are careless and do not take time for the verification of this assumption. The trusses in Figure 5 have this feature.

All these trusses are statically determinate, but externally indeterminate, which does not mean statically indeterminate, so, therefore, they can be analyzed by the method of joints.

In some other case, there are trusses with three reactions which indeed are statically indeterminate. Figure 6 presents a truss with this characteristic.

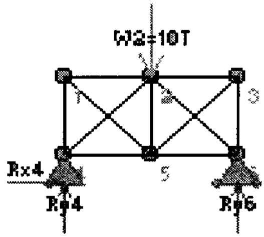

Figure 6. Truss with three reactions.

In this case, the structure is statically indeterminate. It cannot be analyzed by the method of joints, but it is externally determinate, so the reactions can be established. Figure 7 shows more than two unknown forces per joint.

This is an important error. It has been classified as fatal (Table 3). It implies a significant misconception leading to the failure of the analysis method.

In the case of this error, a complete module will come into action to teach the general statements on the relationship between joints and reactions. Makatsiná is able to create an environment to help the student by means of didactic tactics. These tactics will guide the tutor intervention until the student is brought again to the domain environment. This intervention is made up of an explanation followed by an example showing the correct use of the skill.

Error Classification

We have four error types within Makatsiná, which can be seen in Table 3. They are:

1. The fatal errors, implying an important conceptualization mistake leading to the failure of the resolution method.
2. The light errors, indicating lack of knowledge. In other words, the user has specific knowledge for the purpose, but due to lack of attention the user becomes disoriented and in consequence does not finish that part of the process.
3. The superficial errors, being those errors due to lack of specific or local attention, as in an incorrect sign in one equation or a badly constructed projection equation in the Cartesian plane.

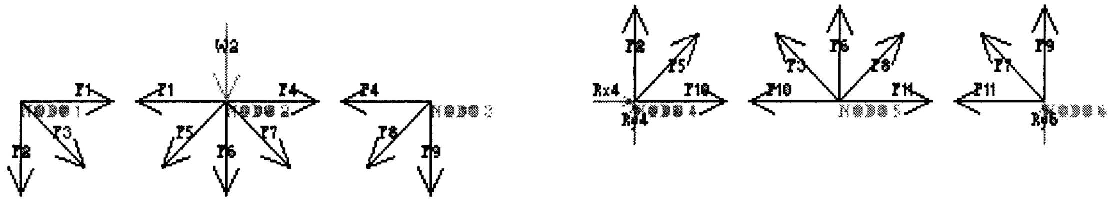

Figure 7. Free body diagram of joints.

Table 3. Critical Errors to Control the Intervention of SUB-TUTORS (Obtained from the CT Global Development)

Sub-tutors (c)	Error 1	Error 2	Error 3	Error 4	Error 5	Error 6	Error 7	Error 8	Error 9	Error 10
Isostatic and Stable	Statically indeterminate is not detected and it exists **F**	Instability is not detected and it exists **F**	Statically indeterminate is detected and it does not exist **F**	Instability is detected and it does not exist **F**						
Bar Simplification Rules	This option has not been executed **F**	Bars are missing according to rule-1- **L**	Bars are missing according to rule-2- **L**	Bars are missing according to rule-3- **L**	No agreement in some or all cases **A**					
Force Evaluation	In an inadequate support **L**	Wrong sign convention **L**	Projections have been wrongly used **S**	Wrong interpretation of the consequences of compression and /or tension **F**	Incorrect sign equation **S**	Wrongly substituted variables **S**	Equilibrium forces transmission has not been used **F**	Moments sum is used and it is not needed **L**	Moments sum is needed but not used **L**	Wrong equation **A**

Errors Type: F: Fatal; L: Light; S: Superficial; A: Abort.

4. The abort errors, showing a total lack of knowledge. In consequence, it is impossible to continue, given that initial knowledge of the resolution method is taken for granted.

THE REACTIVE ARCHITECTURE FOR THE TUTORIAL MODULE

Finally, as a result of the domain analysis, we include the reactive architecture for the tutorial module in Figure 8. Each subtutor represents a skill and these skills will be monitored independently, meaning that subtutors do not communicate among themselves.

Each subtutor has its own perception of the world and each one is able to synthesize those global characteristics which are interesting to it, thereby building its internal representation and acting in consequence.

SYSTEM EVALUATION

Makatsiná has been used for two weeks by a team of three human experts in structural analysis, who are also experienced lecturers in the subject. They have individually run different examples while purposely making different errors, checking not only the knowledge included in the solution method, but also the microworlds used by Makatsiná to properly correct the detected errors. After the experiment, the experts gathered together to discuss the results and, finally and separately, filled in a questionnaire with specific details of system operation. The obtained results show:

- Quality of the domain knowledge included: very good (98/100);
- Significant errors included in the system: all of them (100/100);
- Tactics included to correct them: very suitable and appropriate (99/100);
- System interface: good interactivity (96/100) and adequate for the students (95/100).

The system has been used as a learning aid within a structure course. Two groups, randomly chosen and having the same grade point average profile, have been involved in an experiment. One of the groups (the experimental one) used the system intensively, while the other one (the control group) had the same number of hours devoted to solving

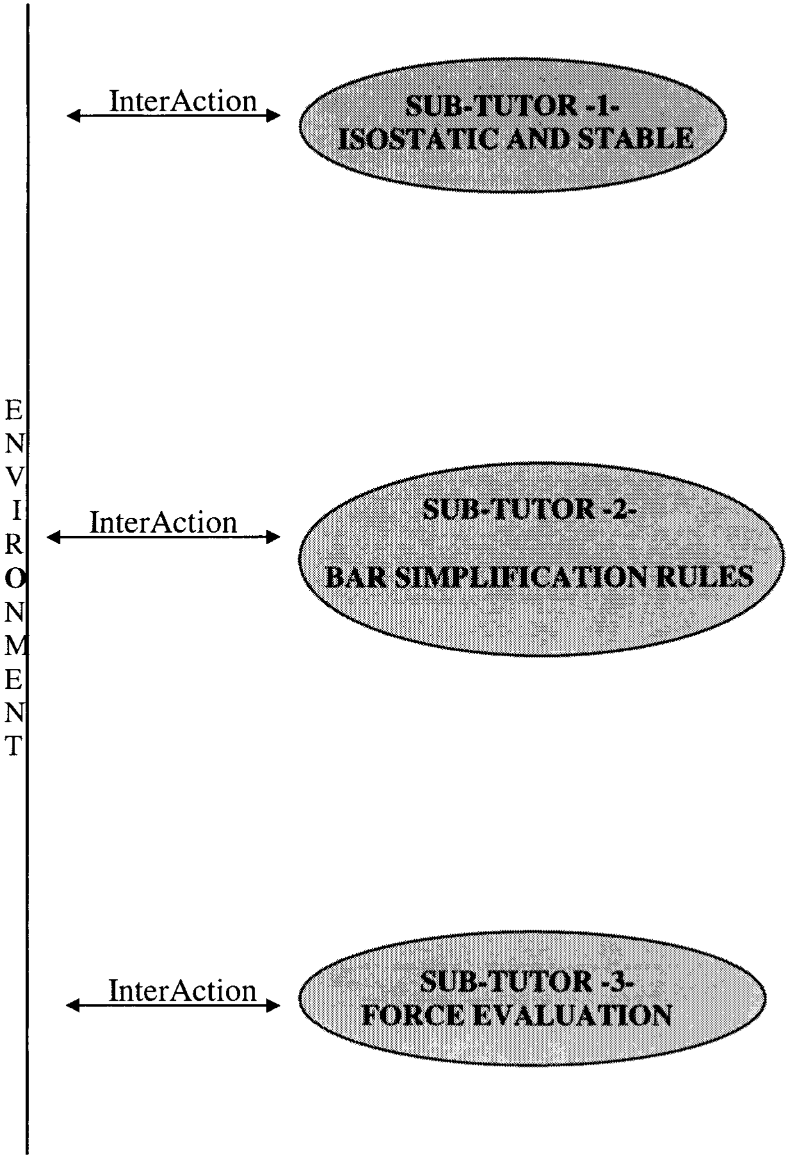

Figure 8. Reactive architecture for the tutorial module.

regular course problems by hand. The main differences obtained between the two groups were the following: a 25% increase in the grade point average mean for the experimental group; a 28% reduction in the estimated learning time of the course for the experimental group; and, a 30% increase in the student interest in structural analysis, (experimental group) measured by the number of enrollments in other structural analysis courses.

RELATED WORK

The following systems complement the list of those already cited so far:

- The multiagent architecture developed by Girard, Gauthier, and Levesque (1992). Its concept could be defined as quasitraditional made up of the planner, curriculum, microworld, and tutor modules but with each module under the care of an agent.
- Néhémie (1992) uses one agent to represent and analyze the student module within a multiagent-aided learning environment. His main concern is the obtainment of the student module conceived as a system.
- More recently, Hexmoor (1995) studied different agent interactions and how the GLAIR architecture helps learning. Finally, El Alami et al. (1997, 1998) have presented an architecture based on cognitive agents; they studied the migration of novices to experts with the aid of ENTrenador, a multiagent system for designing specific training exercises.

CONCLUSIONS

In summary, we would like to point out the following main conclusions:

- Makatsiná is an example of the application of a cognitive approach including the expert mental models of the learning environment, which allows the practical consideration of the problem of turning novices into experts.
- This computer solution is also designed according to cognitive considerations, which provide desirable qualities like flexibility, robustness, and modularity. The number of agents can easily be increased according to the resolution level or the number of tasks to be executed.

- The architecture can easily support the consideration of different subtutors, each one in charge of a microworld, if the problem domain does not preclude doing so.
- The system behaves in a reactive manner before the spontaneous manifestations of the student or the user errors.
- Reactive systems have limitations because we found simple, as well as complex, tasks needing a long reasoning process, or even planning activity, to obtain their objectives but in many cases of simple tasks, reactive agents can be used without planning.
- Based on student errors, we can obtain a representation of the implicit aspects of the problem for a detailed treatment of the errors made by the student during the session. This representation allows us to clearly analyze not only errors but also teaching plans corresponding to different tutorial tactics to be handled by each subtutor.
- Another advantage of this kind of multiagent architecture consists in being structured or divided into submodules. That way it can have an incremental design and the ITS could be taken as a part of another larger one dealing with this CT as a prerequisite or component of another task. For example, we can think of an ITS dealing with bridges. Furthermore, the modules can use software techniques to facilitate programming efforts, such as abstract data types and object-oriented programming, to obtain robustness, correctness, etc. (Laureano, 1993a; Laureano & Ortíz, 1996; Oktaba, 1993).
- According to the assessment, Makatsiná, built according to this approach, seems capable of good tutorial work for the solution of truss analysis problems.

ACKNOWLEDGMENTS

The ideas presented here were based on the work done by Ana Lilia Laureano-Cruces during the research of her Ph.D. degree at the Instituto de Investigaciones Biomédicas de la Universidad Nacional Autónoma de Mexico, where she was advised by Dr. José Negrete Martinez. Thanks also to Javier Ramírez, Fabián García, Alejandro Aldama and Santiago Ledesma for fruitful discussions and to the anonymous referee for valuable comments and suggestions that have led to a much improved paper.

REFERENCES

Alonso, C., and D. Gallego. 1994. *Los Estilos de Aprendizaje*. Bilbao, Spain: Ediciones Mensajero.

Anderson, J. 1982. Acquisition of cognitive skill. *Psychology Rev*. 89: 369–406.

Anderson, J. 1988. The expert module. In *Foundations of intelligent tutoring systems*, eds. Martha C. Polson and J. Jeffrey Richardson. London: Lea Hove.

Anderson, J., C. F. Boyle, and B. J. Reiser. 1985. Intelligent tutoring systems. *Science* 228:456–462.

Anderson, J., J. G. Greeno, P. J. Kline, and D. M. Neves. 1981. Acquisition of problem solving skill. In *Cognitive skills and their acquisition*, ed. J. R. Anderson, 191–230. Hillsdale, NJ: Erlbaum.

Anderson, J. R., and B. J. Reiser. 1985. The lisp tutor. *BYTE* 10:159–175.

Arruarte, A., and I. Fernández. 1995. A two-phased development shell for learning environments: A design proposal. In *Proceedings World Conference Computer in Education VI (WCCE '95)*, eds. J. D. Tinsley and T. J. Van Weert, 53–65. London: Chapman & Hall.

Bajo, M. T., and J. Cañas. 1991. *Ciencia cognitiva*. Editorial Debate. Madrid, Spain.

Barr, A., M. Beard, and R. C. Atkinson. 1976. The computer as a tutorial laboratory: The Stanford BIP Project. *International Journal of Man Machine Studies* 8:567–596.

Beer, F., and E. Jhonson. 1980. *Mecánica Vectorial para Ingenieros TOMO I*. New York: McGraw-Hill.

Beer, R. D. 1990. *Intelligence as adaptive behavior: An experiment in computational neuroethology*. New York: Academic Press.

Bermond, J. C. 1993. Theories des Graphes. *Le Courrier du CNRS*. No. 80, February. Meudon, France.

Borillo, M. 1993. De l'informatique a la cognition. *Le Courrier du CNRS*. No. 80, February. Meudon, France.

Bratko, I. 1992. Machine learning and qualitative modelling in medical applications. In *artificial intelligence and intelligent tutoring systems*, eds. Danny Kopec and R. Brent Thompson. England: Ellis Horwood.

Bretch, B., and M. Jones. 1988. Student models: The genetic graph approach. *Internat. J. Man Machine Studies* 28:483–504.

Breuker, J. 1992. Generality watching: ITS caught between science and engineering. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada, eds. C. Frasson, G. Gauthier, and G. I. McCalla, 11–20. New York: Springer-Verlag.

Brooks, R. 1986. A robust layered control system for a mobile robot. *IEEE Robotics and Automation* RA-2(1):14–23.

Brooks, R. 1991a. Intelligence without representation. *Artificial Intelligence* 47:139–159.

Brooks, R. 1991b. Intelligence without reason. Memo 1293, April, *MIT Artificial Intelligence Laboratory.*

Brown, J. S., R. R. Burton, and A. G. Bell. 1975. SOPHIE, a step towards a reactive learning environment. *Internat. J. Man Machine Studies* 7:675–696.

Brown, J. S., R. R. Burton, and J. deKleer. 1982. Pedagogical, natural language, and knowledge engineering techniques in SOPHIE I, II, and III. In *Intelligent tutoring systems*, eds. D. Sleeman and J. S. Brown, 227–282. New York: Academic Press.

Burton, R., and J. Brown. 1982. An investigation of computer coaching for informal learning activities. In *Intelligent Tutoring Systems*, eds. D. Sleeman and J. S. Brown, 79–98. New York: Academic Press.

Cameron, D., and B. Rosenblatt. 1992. *Learning GNU EMACS.* O'Reilly & Associates (A NutshellBook).

Carbonell, J. 1970. AI in CAI: An artificial intelligence approach to computer assisted instruction. *IEEE Transactions on Man Machine Systems* 11(4):190–202, December.

Castañeda, S. 1993. Procesos Cognitivos y Educación Médica. *Serie Seminarios No. 1.* Facultad de Medicina/Universidad Nacional Autónoma de México. México, D. F.

Castañeda, S. 1994. *Cátedra José Gómez Robleda, UNIDAD 3 y UNIDAD 4.* Faculad de Psicología/Universidad Nacional Autónoma de México. México, D. F.

Choquet, Ch., T. Mengelle, G. Gouarderes, and B. Malsallez. 1992. L'Assistance a l'Usager dans DIGITEF: un Mécanisme Réutilisable. In *Proceedings Intelligent Tutoring Systems ITS '92.* Montréal, Canada. eds. C. Frasson, G. Gauthier, G. I. McCalla, 148–155. New York: Springer-Verlag.

Chouraqui, E., and C. Inghilterra. 1992. Résolution par Analogie de Problemes Géometriques dans une Perspective Tutorielle. In *Proceedings Intelligent Tutoring Systems ITS '92.* Montréal, Canada. eds. C. Frasson, G. Gauthier, and G. I. McCalla, 156–163. New York: Springer-Verlag.

Chu, R., and C. Mitchell. 1992. A pedagogical and architecture for intelligent tutoring and aiding in supervisory control of complex systems. In *Proceedings Intelligent Tutoring Systems ITS '92.* Montréal, Canada. eds. C. Frasson, G. Gauthier, and G. I. McCalla, 164–171. New York: Springer-Verlag.

Clancy, W. J. 1982. Guidon. In *the handbook of artificial intelligence.* ed. A. Barr and E. A. Fegenbaum, 267–278. Los Altos, CA: William Kaufamann.

Clancy, W. J., and R. Letsinger. 1984. *NEOMYCIN: Reconfiguring a Rule-based Expert System for Application to Teaching.* Massachusetts: Addison-Wesley.

Dion, P., and R. Lelouche. 1992. Application de la méthodologie du traçage de modele a un environment d'apprentissage utilisant une stratégie pédagogique non directive. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada, eds. C. Frasson, G. Gauthier, and G. I. McCalla, 421–434. New York: Springer-Verlag.

El Alami, M., F. de Arriaga, and A. Ugena. 1997. SIMUL: Intelligent simulation environment for interdisciplinary project work. In *Proceedings Jubilee International Conference* II: 53–60. Denmark: Roskilde University.

El Alami, M., F. de Arriaga, and A. Ugena. 1998. Multi-agent simulation as an aid for decision making and learning. In *Proceedings 2nd KFUPM Workshop on Information & Computer Science (WICCS '98)*, 121–130. Dhahran, Saudi Arabia.

Elio, R., and P. Scharf. 1990. Modeling novice-to-expert shifts in problem-solving strategy and knowledge organization. *Cognitive Science* 14:579–639.

Elorriaga, J., I. Fernández, and J. Gutiérrez. 1995. Sistemas Tutores Inteligentes y Aprendizaje Automático. Revisita *Informática y Automática* 28(4):3–18. Madrid, Spain.

Faletti, J. 1982. PANDORA: A program for doing commonsense planning in complex situations. In *Proceedings of the National Conference on Artificial Intelligence*, 185–188, AAAI Press.

Fernández, I. 1989. *Estrategias de Enseñanza en un Sistema Inteligente de Enseñanza Asistida por Ordenador*. Doctoral thesis (3rd Cycle), Universidad del País Vasco, San Sebastián, Spain.

Ferber, J. 1993. L'intelligence Artificielle Distribuée. *Le Courrier du CNRS*. No. 80, February. Meudon, France.

Ferber, J., and A. Drogoul. 1992. Using reactive multi-agent system simulation and problem solving. In *Distributed Artificial Intelligence Theory and Praxis*. Computer and information science 5. eds. Nicholas M. Avouris and Les Gasser. Norwell, MA: Kluwer Academic Publishers.

Gagné, E. 1985. *The cognitive psychology of school learning*. Boston, MA: Little Brown & Company.

García, A. 1993. Agentes Computacionales Cognitivos Reactivos en la Modelación de la Conducta Visuomotora de la Rana. M. Sc. Thesis, IIMAS-UNAM. México, D.F.

García-Alegre, M. C., P. Bustos, and D. Guinea. 1995. Complex behavior generation on autonomous robots: A case study. In *Proceedings of the IEEE International Conference on Systems, Man & Cybernetics*, Vol. II, 1729–1734 October. Vancouver, BC, Canada.

García-Alegre, M. C., and F. Recio. 1998. Basic and motor agents for increasingly complex behavior generation on a mobile robot. *Autonomous Robots* 5:19–28. Norwell, MA: Kluwer Academic Publishers.

Gegg-Harrison, T. 1992. ADAPT: Automated debugging in an adaptive prolog tutor. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. eds. C. Frasson, G. Gauthier, and G. I. McCalla, 111–115. New York: Springer-Verlag.

Genesereth, M. 1978. *An automated user consultant for MACSYMA*. Ph.D. Thesis, Harvard University.

Ghallab, M. Le Robot Qui Décide Seul. 1993. *Le Courrier du CNRS*, No. 80, February. Meudon, France.

Ginsburg, H., and S. Opper. 1986. *Piaget y la Teoría del Desarrollo Intelectual.* Englewood Cliffs, NJ: Prentice-Hall.

Girard, J., G. Gauthier, and S. Levesque. 1992. Une architecture multiagent. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla, 172–182. New York: Springer-Verlag.

Goldstein, I. 1976. The computer as a coach: An athletic paradigm for intellectual education, Memo 389, December. *MIT Artificial Intelligence Laboratory.*

Goldstein, I. 1977. The genetic graph: A representation for the evolution of procedural knowledge. In *Man Machine Studies* 11:51–77.

Gutiérrez, J. 1994. *INTZA: Un Sistema Tutor Inteligente para Entrenamiento en Entornos Industriales*. Doctoral Thesis (3rd Cycle), Universidad del País Vasco, San Sebastián, Spain.

Halff, H. 1988. Curriculum and instruction in automated tutors. In *Foundations of Intelligent Tutoring Systems*. Eds. Martha C. Polson and J. Jeffrey Richardson. Lea Hove & London.

Hayes-Roth, B. 1997. Introducción. In *Proceedings of First International Conference on Autonomous Agents*. Marina del Rey California: ACM Press.

Hexmoor, H. 1995. Learning routines. In *Proceedings International Conference on Intelligent Agents II*. Ed. Wooldridge, 178–186. Montreal, Canada.

Huang, T. C. 1980. *Mecánica para Ingenieros-TOMO I*. Editorial Representación y Servicios de Ingeniería S.A.

Kaelbling, L. P. 1987. An architecture for intelligent reactive systems. In *Reasoning about actions and plans*. Eds. Michael P. Georgeff and Amy L. Lansky, 395–410. Morgan Kauffmann.

Kaelbling, L. P., and S. J. Rosenschein. 1993. Action and planning in embedded actions. In *Designing autonomous agents: Theory and practice from biology to engineering and back*. Cambridge, MA: MIT Press.

Kono, Y., M. Ikeda, and R. Mizoguchi. 1992. To contradict is human. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla, 451–458. New York: Springer-Verlag.

Laureano, A. 1992. Introducción a la Mecánica Estructural por Medio de la Animación por Comuptadora. Reporte Interno No. 286. *Introducción a la*

Mecánica Estructural por Medio de la Animación por Computadora. Departamento de sistemas. Universidad Autónoma Metropolitana-Azcapotzalco. México, D.F.

Laureano, A. 1993a. Tecnologías Orientadas a Ojetos. Revista *EN LINEA Universidad Autónoma Metrolitana-Azcapotzalco* 1(2):26–30, July. México, D.F.

Laureano, A. 1993b. Multimedios v Cognición. Revista *Perfiles Educativos*. No. 62, 38–41. CISE-UNAM. México, D.F.

Laureano, A. 1994. Inteligencia Artificial e Inducción. Revista *EN LINEA Universidad Autónoma Metrolitana-Azcapotzalco* 1(4):7–12, March. México, D.F.

Laureano, A. 1995. Herramientas para la Representación del Conocimiento en Sistemas con Inteligencia Artificial. Revista *EN LINEA Universidad Autónoma Metrolitana-Azcapotzalco* 1(5):14–22, May. México, D.F.

Laureano, A. 1997. Multiagentes en el Diseño de un Sistema de Enseñanza Inteligente. Revista *Perfiles Educativos* XIX(75):23–33. CISE-UNAM, México, D.F.

Laureano, A., and F. de Arriaga. 1996. Diseño del Domino en Sistemas de Enseñanza Inteligentes: basado en la reactividad de los agentes. In *Proceedings II Jornadas Nacionales de Innovación en las Enseñanzas de las Ingenierías* II: 987–999. ICE-UPM, Madrid, Spain.

Laureano, A., and F. de Arriaga. 1997. Un Sistema MultiAgente Reactivo en los Sistemas de Enseñanza Inteligentes. Revista *Informática y Automática* 30(4):71–84. Spain.

Laureano, A., and F. de Arriaga. 1998. An intelligent tutoring system to teach an integrated skill. In *Proceedings 2nd KFUPM Workshop on Information & Computer Science (WICCS '98)*, 71–84. Daharam, Saudi Arabia.

Laureano, A., and J. Negrete. 1996. Una Arquitectura Reactiva en Sistemas de Enseñanza Inteligentes. In *Proceedings Informática Educativa '96, UNED*. Madrid, Spain.

Laureano, A., and T. Ortiz. 1996. *Introducción a la Programación Orientada a Objetos: Un Enfoque con Tipos Abstractos de Datos y Estructuras de Datos*. Ed. Universidad Autónoma Metropolitana-Azcapotzalco. México, D.F.

Maes, P. 1993a. Designing autonomous agents. In *Designing autonomous agents: Theory and practice from biology to engineering and back*. Cambridge, MA: MIT Press.

Maes, P. 1993b. Situated agents can have goals. In *Designing autonomous agents: Theory and practice from biology to engineering and back*. Cambridge, MA: MIT Press.

Michie, D., and M. Bain. 1992. Machine acquisition of concepts from sample data. In *Artificial intelligence and intelligent tutoring systems*. Eds. Danny Kopec and R. Brent Thompson. England: Ellis Horwood.

Minsky, M. 1987a. *La Sociedad de la Mente*. Editorial, Galápagos, Spain.

Moore, J. 1992. Winch simulation: Multiple, linked representations of linear functions. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla, 111–115. New York: Springer-Verlag.

Negrete, J. 1992. *De la Filosofía a la Inteligencia Artificial*. Ed. Grupo Noriega Editores. Mexico.

Néhémie, P. 1992. A systematic approach for student modeling in a multi-agent aided learning environment. In Proceedings Intelligent Tutoring Systems ITS '92. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla. 475–482. New York: Springer-Verlag.

Nigay, L., and J. Coutaz. 1991. Building user interfaces: Organizing software agents. *A cookbook for organizing software agents*. ESPIRIT Basic Research Action 3066 AMODEUS (Assimilating Models of Designers, Users, and Systems).

Oktaba, H. 1993. *Diseño Orientado a Objetos: Método de Booch*. Notas de la Maestría en Ciencias de la Computacion de la UACP y P del CCH, con sede en el IIMAS-UNAM, March. Ciudad Universitaria, Mexico.

Rasmussen, J. 1986. *Information processing human-machine interaction. An approach to cognitive engineering*. Amsterdam: North Holland.

Redding, R. E. 1989. Perspectives on cognitive task analysis: The state of the state of the art. In *33rd Annual Proceedings of the Human Factors Society*. Ed. Human Factors, 1348–1352. Santa Monica, CA.

Redding, R. E. 1992. A standard procedure for conducting cognitive task analysis. *ERIC Documentation Reproduction Service*. No. DE 340-847.

Rickel, J. 1989. Intelligent computer aided instruction: A survey organized around systems components. *IEEE Transactions on Systems, Man, and Cybernetics* 19(1):40–57.

Ryder, J. M., and R. E. Redding. 1993. Integrating cognitive task analysis into instructional systems development. *Educational Technology Research & Development. (ETR & D)* 41(2):75–96.

Shortliffe, E. H. 1976. *Computer-based medical consultations: MYCIN*. New York: American Elsevier.

Sime, J., and R. Leitch. 1992. A learning environment based on multiple qualitative models. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla, 116–123. New York: Springer-Verlag.

Sleeman, D., and J. S. Brown. 1982. *Intelligent tutoring systems: An overview*, 1–11. New York: Academic Press.

Sombé, L. 1993. Informations Incomplétes, Informations Contradictoires. *Le Courrier du CNRS No. 80*, February. Meudon, France.

Urretavizcaya, M. 1991. *Sistema para la Detección, Diagnóstico y Corrección de Errores en un Sistema Tutor Inteligente*. Doctoral thesis (3rd Cycle), Universidad del País Vasco, San Sebastián, Spain.

VanLehn, K. 1988. Student modeling. In *Foundations of intelligent tutoring systems*, eds. Martha C. Polson and J. Jeffrey Richardson. Lea Hove & London.

Wang, H., and A. Kushniruk, 1992. The UNIX Tutor. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla, 317–324. New York: Springer-Verlag.

Wild, M. 1996. Mental model and computer modelling. *J. Computer Assisted Learning* (12):10–21.

Woods, P. M., and J. R. Hartley. 1971. Some learning models for arithmetic task and their use in computer-based learning. *British J. Education Psychology* 41:35–48.

Yibin M., and L. Jianxiang. 1992. Intelligent tutoring system for symbolic calculation. In *Proceedings Intelligent Tutoring Systems ITS '92*. Montréal, Canada. Eds. C. Frasson, G. Gauthier, and G. I. McCalla, 111–115. New York: Springer-Verlag.