A Multiagent Extension for Virtual Reality Based Intelligent Tutoring Systems

Ricardo Imbert, Leticia Sánchez, Angélica de Antonio, Gonzalo Méndez, Jaime Ramírez
Computer Science School, Universidad Politécnica de Madrid (Spain)
{rimbert,angelica,jramirez}@fi.upm.es, leticia@bermudas.ls.fi.upm.es,
gonzalo@gordini.ls.fi.upm.es

Abstract

Applying Virtual Reality (VR) in combination with Intelligent Tutoring Systems (ITSs) is a promising approach to computer based learning and training. However, the classical structure of ITSs has not been conceived to deal with the new sources of information and interaction provided by VR environments. The resulting structures combining both technologies often renounce to the traditional modular separation of ITSs in pursuit of increasing efficiency, but making very difficult the reusing of their components.

In this paper it is described the extension of the classical ITS structure proposed in the MAEVIF model to build reusable and adaptable VR based ITSs.

1. Introduction

The use of Virtual Reality (VR) and Virtual Environment (VE) technology combined with Intelligent Tutoring Systems (ITSs) has shown as a valuable and promising approach to computer based teaching and training. Together with the traditional properties of ITSs instruction (personalization of the contents and presentation, adaptation of the tutoring strategy to the student needs…), VR allows the student to gain practice and skills interacting with virtual scenarios similar to the real ones, but without assuming any of their potential risks. The instant response to his actions, the possibility of activating any kind of simulation, the advantage of having the chance to experiment alternative actions, just to see what happens, are some other features that VR adds to ITSs. In addition, as not only the trainee, but also the tutor may have a virtual representation (virtual mannequin or avatar) in the 3D environment, it can be used to perform demonstrations about how to carry out specific tasks, just in the same way in which a human tutor may proceed with his students. Finally, VR technology potentially increases the student's presence, even when using non immersive interface devices.

However, the ITS classical structure fails in coping with VR-based educational processes, since it does not fit properly in the new technology needs, and, more important, because it does not consider the interaction between the user and the VE, but only the interaction of the user with the tutoring system.

On the other hand, if developing an ITS is not a trivial task, adding VR to it makes it much more intricate. Curiously, a big amount of the work to be done to develop a new system of this kind is always similar, given that the underlying mechanisms are ultimately common. However, the theoretical richness of ITS, separating and isolating properly the different kinds of knowledge needed for the learning process (namely expert, student, tutoring and communication), is usually lost when context specificities and VR technology restrictions end up "infecting" the other elements of the system. The resulting systems present increased effectiveness, but their basic components, or at least, part of them, are not suitable for being conveniently reused.

Some previous approaches (e.g. [1-9]) have tried to cope with these two big problems but there is still a long way to achieve it successfully. As an answer to them, and from the experience gained in the development of several VR-based ITS, the Decoroso Crespo Lab research group (Universidad Politécnica de Madrid), has been developing during the last years a reusable and easily configurable model for the application of VR technology to education, called MAEVIF [10]. Its structure, together with its operational schema, is described in detail in the following sections.

2. Extending the classical ITS architecture

ITSs have been traditionally conceived to give support to individual students. The student interacts with the software tutor carrying out the activities proposed by it and, according to the evolution of that interaction,

the tutor adapts its didactical process to the student's specific characteristics.

However, when the user interaction capabilities with the ITSs are enriched by means of a more immersive interface, the tutor receives information coming from new sources. These sources are of high relevance from the educational point of view, and must be taken into account to evaluate the student's performance. Thus, the software tutor can consider information emerging from the interaction between the student and the 3D environment, such as the objects he manipulates, where he is looking at, or the path he is following to reach a point in the virtual scenario.

In addition, group and collaborative training in distributed virtual environments asks for a redefinition of the concept of personalized tutoring.

The classical ITS structure turns to be insufficient to deal with this new setting since it does not fit with the new interaction possibilities and group tutoring needs. This motivated us to extend the classical ITS structure.

To cope with these issues, MAEVIF first introduces a new module, called *world module* (because it manages knowledge about the virtual *world*). Thus, this module is responsible for: maintaining awareness of the state of the virtual scenario, the objects positioned in it and the virtual characters that inhabit it; providing perceptual capabilities to the possible synthetic or virtual characters introduced in the virtual environment to support the learning process, including the virtual representation of the tutor (*virtual tutor*) and other team members; and the planning of the ideal paths to be followed by the student in the virtual environment, which enables their tracking in real time.

In addition, we believed that the different kinds of knowledge and the processes that manage them should appear clearly distinguished inside the system, to maintain the coherence and essence of ITSs. This independence has been pursued in MAEVIF through the conceptual model itself, and then through its design and implementation as a multiagent system (MAS), as it is shown in Figure 1. Software agents are a technology that fits appropriately to the need of maintaining the independence between all the ITS modules, acting collaboratively to achieve the common goal of teaching the student. In addition, this design enables the ITS customization, allowing the reuse and adaptation of agents if it is needed with low development effort. The MAS designed for MAEVIF and its operation are described in the next sections.

3. Multiagent system description

As it was said above, the MAEVIF architecture has been developed following an agent-based approach,

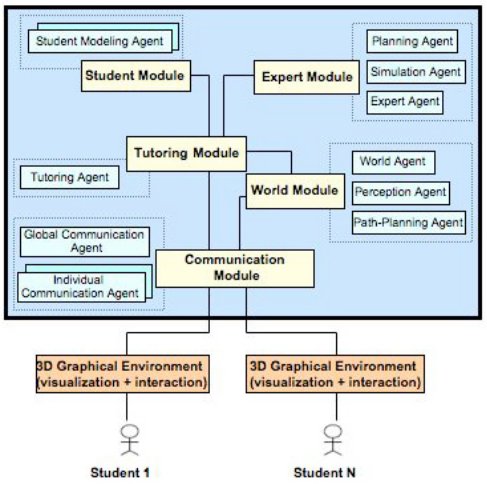

Figure 1. MAEVIF's ITS extension

identifying one or more software agents for every ITS module (marked with dotted-squares in Figure 1):

- *Planning Agent*: Builds plans for every defined activity, made up of each individual action the students are expected to execute to reach successfully the activity's objectives.
- *Tutoring Agent*: When it receives a request from the students to initiate a training session, it provides them with a list of the available activities in accordance to their knowledge about the subject that is being taught and their possible roles. It controls the team formation in collaborative activities, together with the assignation of the character to be interpreted by every student.
  It also performs the student tracking, controlling every one of his actions and comparing them with the ones defined in the activity plan generated by the Planning Agent. The Tutoring Agent also manages the assistance and help system, providing the student with timely information and answering his questions, related both to the scenario and to the process to be followed to end successfully the activity.
- *Path-Planning Agent*: Performs the tracking of the student paths along the VE comparing them with the optimal one. This information may be important in certain training activities.
- *Expert Agent*: Gathers information related to the actions the students can perform in the VE, more exactly, the preconditions and the consequences of the actions. This information will be used in the validation and execution of the student actions.
- *World Agent*: Maintains and controls geometrical and semantic information about the VE objects and inhabitants (students, auxiliary characters and

virtual tutors). All this information is represented in an ontology.

- *Perception Agent*: Provides the virtual characters (the virtual tutor or other auxiliary autonomous virtual characters) with perceptual capabilities. This agent monitors a visual field for every character, providing them with information about the places, objects and characters they are watching.
- *Simulation Agent*: Simulates the consequences of every student action. It also provides the student with suitable information in answer to questions like: "What will it happen if...?". This agent also supports the Expert Agent in the action validation process.
- *Communication Agent*: Is the communicative interface between the students and the tutoring system. Every student has an associated Communication Agent.
- *Student Agent*: Records the actions executed by a specific student, the evaluation of the activities carried out, and a register of his physical behaviors, including the places and objects being watched, and the paths followed in each student's movement action. In addition, it provides the Tutoring Agent with information to advise the students if their itinerary does not match the optimal path calculated by the Path-Planning Agent, or to check if the student is confused or does not know how to finish the activity. There are as many Student Agents as students connected to the system.

Figure 2 shows the internal dependencies among these agents.

4. A dynamic view of the tutoring process

To give a deeper understanding of the way in which MAEVIF carries out its tutoring tasks, and to illustrate the internal working of its MAS, in this section we review the operation of the system during a fragment of a training task. The selected scenario is part of the process of producing vinegar, in particular, the moment in which operators must collect samples of concentrate acetic acid, an intermediate product dangerous because of its acidity. Briefly, the trainee has to take two tools to perform the task: a sample receptacle and a radio transmitter. Then, he has to go to a valve and check the concentrate pressure in a manometer. If the pressure is adequate, he opens the valve and collects the concentrate sample. Finally, he closes the valve and labels the collected sample. On the contrary, if the acid pressure is not adequate, he must use the transmitter to communicate the incidence to the control post, and ask for instructions.

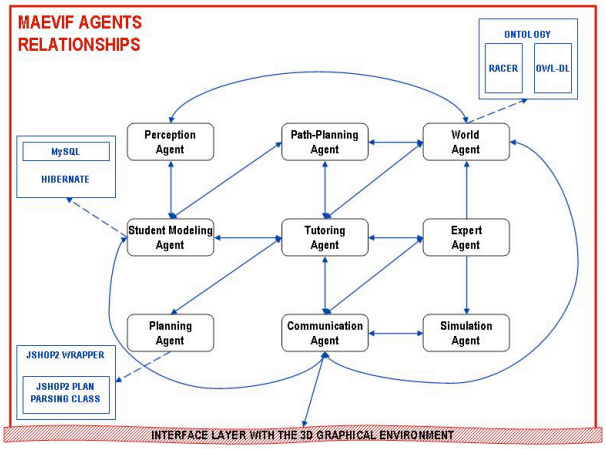

Figure 2. Dependencies and relationships among MAEVIF's agents

Next, we are going to review the training process of a student in this scenario, analyzing the tutoring activities involved.

The student takes the receptacle and the transmitter. When the trainee gets to the starting point of the proposed activity, the Tutoring Agent, which controls the course syllabus, asks the Planning Agent to design a plan to achieve the desired goal, i.e. the sample collection by the student. The generated plan is a sequence of actions or blocks of actions. The actions inside a block may be ordered or unordered, which means that it is possible to have, in a given moment, several actions to be performed, but without caring about the exact order in which they will be done. This is exactly the case of the initial activities of the proposed operation, so the Planning Agent provides the Tutoring Agent with a plan that contains an initial action block consisting of two unordered actions: *take the receptacle* and *take the transmitter*. The Tutoring Agent, then, waits for the next student action.

If the student takes the receptacle, this generates an event, captured by the interaction devices and transmitted to the MAEVIF MAS. In particular, the Communication Agent is informed about the student's intention of applying the operator *take_object* on the object *receptacle*.

The action execution has to be validated, i.e., its *preconditions* must be checked. With this purpose, the Communication Agent informs of the student intention to the Tutoring Agent. In order to do so, it must transform the intended action execution, with the help of the World Agent, into the language the MAS agents use, translating, for instance, raw coordinates of the 3D environment into conceptual places. Then, the Tutoring Agent asks the Expert Agent to check the action preconditions. Since the *take_object* action does not in-

volve any underlying simulation, the Expert Agent only has to ask the World Agent to validate its preconditions against the *semantic model of the environment* (the ontology) it manages.

The validation results are then transmitted to the Tutoring Agent and to the Communication Agent. If the preconditions have not been validated, the Communication Agent solicits the 3D graphical environment to not allow the execution of the action. In addition, the Tutoring Agent is informed about the unfulfilled precondition(s), in order to be able to provide the student with appropriate assistance. On the other hand, if the preconditions have been all validated, the Communication Agent allows the 3D Graphical Environment to execute the consequences of the action (a visual representation of the receptacle grasping).

The Tutoring Agent, in turn, initiates the necessary updating of the world state. It asks the Expert Agent to execute the consequences of the validated action, who delegates this task to the World Agent, in order to maintain the semantic model of the environment updated, and to the Simulation Agent, to launch any simulation related to the action consequences.

Once the feasibility of the action has been assessed, the Tutoring Agent checks if the executed action effectively matches any of the expected (planned) ones. If it was an unexpected action, it would ask the Planning Agent to determine if it is innocuous, still allowing the execution of the existing plan, or to find an alternative plan, if the current one has become invalid. If the Planning Agent was unable to find a new plan, the Tutoring Agent would inform the student about his fatal action, which, for instance, may suppose putting and end to the training activity.

However, if the executed action matches one of the expected ones, the Tutoring Agent will remain waiting for the next student action. If the student now decides to take the transmitter, the whole process will be repeated.

In any case, whenever the student begins and finishes the execution of an action, this is notified to the Student Agent, to incorporate the information to the *student model*.

The student leads to the valve. Every time the execution of an action is validated and finished, the Tutoring Agent checks if any of the possible following actions is a translation (i.e., the student will have to move to another place in the virtual environment). In this case, it must inform the Path-Planning Agent of the students that may initiate a movement, together with their expected destinations, to begin their movement tracking.

The Path-Planning Agent then asks the World Agent about the current position of those students, and calculates the *optimal path* to arrive to their destinations. It also subscribes to a service offered by the Communication Agent, to obtain the students coordinates while they are moving around the virtual environment. From that data, the Path-Planning Agent checks if the student follows the optimal path. If he deviates more than an established range, it informs the Tutoring Agent to provide the student with *assistance*.

The Path-Planning Agent will keep on checking the students' positions till the Tutoring Agent stops it. This will happen when the Tutoring Agent receives a from the student an object manipulation action (in our example, *read_manometer*), coming through the Communication Agent. In that moment, the Tutoring Agent asks the Path-Planning Agent if any of the possible destination points has been reached. If it has, it checks which one of the possible movement actions has been performed, and it continues processing the object manipulation action received. On the contrary, if none of the expected destination points has been reached, the Tutoring Agent asks the Planning Agent to find a new plan to reach the desired goals, if it is still possible.

Assisting the student. After having read the manometer lecture, the student is intended to open the valve if the acid pressure is adequate. But, what will it happen if he does not remember exactly what he has to do?

The Tutoring Agent has been designed to provide the student with different kinds of assistance, and also with different levels of detail. This aid may be offered on-demand or proactively, and it is always subject to the tutoring strategy followed by the Tutoring Agent.

For instance, in the proposed situation, the student may ask the tutor a *basic question* such as: "*Which is the next action I have to perform?*" The Tutoring Agent would give to him a high-level-of-detail answer, like: "*Now, you have to collect the sample in the receptacle*." If the student feels that he still needs some more help, he could ask an *advanced question* such as: "*Which are the objects that I have to use for the next action?*" The Tutoring Agent would then answer: "*You need the receptacle and the valve*". The level of detail of the answers would increase as the student continues asking about the action.

That is not the only kind of questions the student can formulate. If his problem is that he cannot identify an object, he can ask *general questions* to the ITS: "*What is this object?*" with an answer such as "*It is a valve*". Or *expert questions*: "*What is the valve good for?*" with "*You can use it to collect acid samples*" as an answer… If the Tutoring Agent has a virtual representation in the environment (virtual tutor), it can even show the student the right way to execute the next action (*practical demonstration*).

The Tutoring Agent may also provide the student with some proactive information. For instance, it can

give him some *clues* if it perceives that he is lost, is looking to different objects to those needed for the action… As it happened with questions, MAEVIF Tutoring Agent may offer clues of several levels of detail depending on the student and on the number of previous clues given to him about the same subject. E.g. "*Now, you have to handle an object*". "*Next thing to do is to handle the valve*". "*You have to use the receptacle below the valve*". And so on.

Every time the Tutoring Agent assists the student, it also informs the Student Agent, to include this information into the student model. Several agents, including the Tutoring Agent, continuously access the student model, mainly to allow the adaptation of the teaching-learning process to the student (e.g. if the student frequently asks questions, the Tutoring Agent may decide to give clues more often or to answer them with a higher level of detail) and to influence his evaluation (too many clues during the training should reduce the final mark).

The student finds it impossible to collect samples. If the acid pressure is adequate, the student has to perform a sequence of four object manipulating actions: *situate_receptacle*, *open_valve*, *close_valve* and *label_sample*. Their processing follows the schema of any manipulation action as the one described above.

However, an inadequate acid pressure lecture would provoke the abortion of the active plan and a change in the goals of the student. It is impossible for him to reach the initial goal and a new goal emerges: to inform the control post of the incident. The Planning Agent is asked to elaborate a new plan, which, in this case, consists of two ordered actions: *notify_incidence* and *ask_for_instructions*.

5. Conclusions and ongoing work

Developing VR-based ITSs is a complex task that involves a big amount of effort in areas of different nature. Thus, it is desirable to reuse the result of that effort from system to system, and ITS philosophy seems to fit well with this purpose. However, the classical structure of ITSs is unable to cope with the richness of interaction and knowledge that VR provides to the teaching-learning process. Forcing the ITS structure to deal with this new information provokes inefficiencies and important coupling among its modules, what goes against the essence of ITSs and makes reusing quite difficult.

MAEVIF is born to tackle with these two difficulties: to exploit all the new technology possibilities and to allow reusing, thanks to its convenient model. It extends the classical ITS structure to include the new kinds of knowledge and its processing, and presents a highly configurable internal structure designed as a flexible MAS.

MAEVIF has been first tested in an environment to train workers in the context of nuclear power plants, during a project funded by the Spanish Government. Currently, an upgraded version of MAEVIF is being applied in conjunction to haptic devices and new reasoning models in the bosom of the Spanish Government funded *ENVIRA Project*.

6. References

[1] E. Shaw, R. Ganeshan, W.L. Johnson, and D. Millar, "Building a Case for Agent-Assisted Learning as a Catalyst for Curriculum Reform in Medical Education". *Procs. of the 9th Intl. Conf. on Artificial Intelligence in Education AIED'99*, IOS Press, 1999, Vol. 50, pp. 509–516.

[2] D.M.Zhang, L. Alem, and K. Yacef, "Using Multi-agent Approach for the Design of an Intelligent Learning Environment", Springer-Verlag, 1998, *LNCS*, vol. 1441, pp.221-230.

[3] M. Hospers, E. Kroezen, A. Nijholt, R. op den Akker, and D. Heylen "An Agent-based Intelligent Tutoring System for Nurse Education", *Whitestein Series in Software Agent Techs*., chap. 9, pp. 143-159, 2004, Birkhauser Publishing.

[4] J. Rickel, and W.L. Johnson. "Animated agents for procedural training in virtual reality: Perception, cognition and motor control", *Applied AI*, 13:343-382, 1999.

[5] J.C. Lester, B.A. Stone, and G.D. Stelling, "Lifelike pedagogical agents for mixed initiative problem solving in constructivist learning environments", *User Modeling and User-Adapted Interaction*, 9(1–2):1-44, 1999.

[6] J.C. Lester, J.L. Voerman, S.G. Towns, and C.B. Callaway, "Cosmo: A life-like animated pedagogical agent with deictic believability", In *Procs. of the IJCAI'97 Workshop on Animated Interface Agents*, Nagoya, Japan, 1997.

[7] J.C. Lester, L.S. Zettlemoyer, J.P. Grégoire, and W.H. Bares. "Explanatory lifelike avatars: performing user-centered tasks in 3D learning environments", in *Procs. of the Third Annual Conf. on Autonomous Agents*, pp. 24-31, 1999.

[8] W. Swartout et al. "Towards the holodeck: Integrating graphics, sound, character and story". *Proc. of the 5th International Conf. on Autonomous Agents*, pp. 409-416, 2001.

[9] J.L. los Arcos, et al., "LAHYSTOTRAIN: Integration of Virtual Environments and ITS for Surgery Training", *Intelligent Tutoring Systems*, Springer-Verlag, 2000, *LNCS*, vol. 1839, pp. 43-52.

[10] A. de Antonio, J. Ramírez, R. Imbert, and G. Méndez. "Intelligent virtual environments for training: An agent-based approach", Springer, 2005, *LNAI*, vol. 3690, pp. 82-91.