Simulate pedagogical reasoning in a virtual environment for training.

Cédric Buche, Ronan Querrec
Centre Européen de Réalité Virtuelle
BP 38 - 29280 Plouzané
France
[buche, querrec]@enib.fr

ABSTRACT
In the framework of decisional behaviour training within collaborative procedures, we define the implementation, by the way of agents, of basic models of Intelligent Tutoring Systems (ITS). The objective is to assist a teacher to take pedagogical decision, suggesting pedagogical assistances. Considering information on the learner and on the simulation, we simulate a pedagogical reasoning using three levels of abstraction. The model integrates an artificial learning mechanism.

KEY WORDS
Virtual Environment for Training, pedagogical agent, Intelligent Tutoring System, artificial learning.

1. Introduction
This article takes place in the formalization of processes for the development of virtual environments for training (VET).

Our works focus around the use of the virtual reality for the implementation of a differentiated pedagogy focused on the decisional behaviour training for procedural and collaborative work. We build an educational environment where a teacher has in charge several learners. In this context, every student is immersed in a virtual environment, simulating his physical environment and realizes his social environment [1].

Imagine that every learner has a personal training assistant who pays attention to the participant's learning needs, assesses and diagnoses problems, and provides assistance as needed. The assistant could alert the instructor of learning problems; therefore the teacher would be free to concentrate on training issues that require greater expertise. This is the goal of Intelligent Tutoring System (ITS): to provide the benefits of one-on-one instruction automatically.

Many applications integrate an ITS in a virtual environment for training. [2] is a good example of such environment. However, they are focused on the simulation of the environment and do not incorporate explicit knowledge for education reasoning. [3] incorporates a knowledge representation but only on the domain and on the learner. The most evolved integrates a diagnosis model [4], where the error notion and the suggested pedagogical assistances are specified for every exercise, and therefore are not generic.

2. The system
Our approach divides our work in two parts.

First, we provide an “informed virtual environment” where the knowledge on the learner, the domain, the interface and the pedagogy are represented, and updated online. For that, we propose an ITS. Our model defines our ITS items in the form of a multi-agents system. This part of the work is presented in section 2.1 and 2.2.

Second, we define a pedagogical agent. It provides an educational reasoning considering the knowledge of our ITS. This part of the work is presented in section 2.3.
2.1. Represent the knowledge for the pedagogy

In this section, we show how we represent the knowledge used by the pedagogical reasoning of our system.

Components
This knowledge is contained in models inspired from classical ITS [5]:
1. the domain model, representing the expert knowledge on the domain, useful for the pedagogy;
2. the learner model, allowing to establish the state of his knowledge, at a given instant, as well as his profile;
3. the errors model, proposing to identify the errors and containing a knowledge base on classical errors;
4. the interface model, allowing the information exchange between system and users;
5. the pedagogical model, taking the teaching decisions.

Dynamic
Models are represented by an autonomous agent constituted of behaviours and knowledge. Every agent (ExpertAgent, LearnerAgent, ErrorAgent, InterfaceAgent, PedagogicalAgent, TeacherAgent) is in interaction with one or several agents, and consequently is able to update its knowledge. The teacher intervenes using the TeacherAgent. Models interact and exchange data extracted from the simulation or deducted from an internal reasoning.

2.2. ITS process

We propose a process in 5 steps cycle, to be performed for every learner (figure 2):
1. Observe (InterfaceAgent): using the interface model, the system analyzes the learner activities. Relevant elements for the training are provided to the learner model: learner actions, elements observable by the learner and learner motions.
2. Detect and identify (LearnerAgent, ExpertAgent, ErrorAgent): the system analyzes the learner actions (learner model) and compares them to actions to carry out (domain model). This confrontation allows to detect an eventual error. If an error is detected, an error identification mechanism acts (using the errors model). Moreover, the custom rules of the domain model are checked (independent rules of the procedure scheduling).
3. Propose pedagogical assistances (PedagogicalAgent): using the learner model (features, activities, error) and the domain model (knowledge on organisational structures), a mechanism simulating a pedagogical reasoning recommends pedagogical assistances adapted to the context. This point is presented in the next section.
4. Pedagogical assistance choice (TeacherAgent): the teacher selects a pedagogical assistance among those recommended.
5. Represent the pedagogical assistance (InterfaceAgent): the selected pedagogical assistance is presented to the learner in the virtual environment.

Figure 2: The pedagogical process.

The models of our ITS contain the knowledge base used by our system pedagogical reasoning, by the way of the domain model, the learner model, the errors model and the interface model. Details on available data can be found in [6]. Therefore, the PedagogicalAgent has a knowledge base allowing to simulate a pedagogical reasoning. Next section explains the PedagogicalAgent functionalities.

2.3. Pedagogical decision

The pedagogical decision is divided into three parts (figure 3):
1. The automatic construction of the “pedagogical situation”. The objective is to propose to the teacher pedagogical assistances adapted to the learner. For that, the pedagogical agent (PedagogicalAgent) constructs a knowledge base, called the pedagogical situation. It corresponds to relevant elements to take into account to choose pedagogical assistances.
2. Using such elements, the *pedagogical model* controls the *PedagogicalAgent* behavior, simulating a pedagogical reasoning. The goal is to suggest pedagogical assistances to the teacher.

3. The *artificial learning* mechanism modifies the pedagogical model, taking into account the teacher's choices. Consequently, it modifies futures pedagogical assistances suggestions.

**Pedagogical situation**

The pedagogical situation is a knowledge base, built in order to provide data to the pedagogical model (to simulate pedagogical reasoning). It is constituted by elements present in models of our ITS previously described (error, domain, learner features and activities). It is based on the analysis of knowledge from others agents (ExpertAgent, LearnerAgent, ErrorAgent and InterfaceAgent).

It also has more information based on the context of the current action. The context of an action (figure 4) is constituted by 3D objects (pre-condition, post-condition, target and performer). For example, the action "the firefighter has to move the car to reach the fire". "Firefighter" is the performer, "car" is the target, "reach the fire" is the post-condition. This pool of 3D objects constituted the context of the action, means associated elements. We could use them to reason or to launch action.

**Figure 4: The context of action data.**

The pedagogical situation also has information on the context of actions connected to the current action. In fact, the training aims at procedural work. Procedures and actions are the keys points. The objective is to reduce the distance to the end of the procedure. In order to provide knowledge to help the learner to reduce this distance, the pedagogical situation provides knowledge on the context of:

- the current action being achieved,
- the correct action,
- the next action,
- the previous action,
- the linked actions.

Linked actions mean actions with a pre-condition matching correct action post-condition. In fact, the correct action effect (post-condition) is requested to launch linked actions and therefore the correct action is necessary to continue the procedure. Use such information will be useful to explain why the learner should do the correct action.

**Pedagogical situation knowledge uses**

Knowledge from pedagogical situation can be used by two ways:

1. *launch a pedagogical assistance*: we have information connected to the learner action (3D objects). Consequently, it is possible to provide pedagogical assistance using pedagogical virtual reality possibilities. For example, we can surround 3D objects connected to the correct action target,

2. *check information*, for example we can verify if the learner is a novice using LearnerAgent information or check if the current action is the correct action.

**Figure 5: The pedagogical situation information.**

**Pedagogical model**

The pedagogical model controls the *PedagogicalAgent* behavior, this one is specified by a pedagogical specialist. He is not a computer scientist; therefore we need an expressive representation, as "if <condition> then <action>s" rules.

In addition, simulate a pedagogical reasoning is complex and "if-then" rules can not connect directly data from the pedagogical situation and pedagogical assistances.
Therefore, our model structures the pedagogical reasoning in 3 levels of abstraction:

1. pedagogical approaches,
2. pedagogical attitudes,
3. pedagogical techniques.

Moreover, rules are managed by set. A set of rules is situated in a level (figure 6). A set of rules has an intensity parameter.

**Dynamic**

The dynamic of the system is as follows:

1. Using knowledge from the pedagogical situation, conditions rules are checked.
2. Matched rules in the first level increase the set of rules intensity in the second level thanks to their effect part. It is the same process from the second level to the last level.
3. In the last level (techniques), we obtain pairs intensity-pedagogical assistance, thanks to the effect part. Intensity comes from the parent set of rules intensity.
4. Pedagogical assistances are suggested to the teacher with an ordered list, using pairs intensities.
5. The teacher selects one of the pedagogical assistances. He does not have to select the first one.

**Representation**

Classifiers system is an induction self-learning system based on a set of simple logical rules called classifiers. Each rule has the following structure: "if <condition> then <action>".

In order to introduce the three levels and to use set of rules, we use hierarchical classifiers system [7]. A classifiers system (CS_System) is a set of rules. In our model (figure 7), a level (ClassifierSystemSet) is defined as a set of classifiers systems. It manages several classifiers systems. The pedagogical model contains three levels.

Classically, classifiers system is optimized using reinforcement learning mechanism.

**Artificial learning**

In order to adapt the pedagogical model to the pair learner/teacher, we want to introduce artificial learning. The goal is to adapt suggested pedagogical assistances order. We use the bucket brigade algorithm [8]. It is the classifiers system reinforcement learning mechanism.
Elements
To implement the algorithm, each classifier is assigned a quantity called its strength. The bucket brigade adjusts the strength to reflect the classifier's overall usefulness to the system. In our case, each classifier has an activity parameter. The strength is used to compute the rule activity. Consequently, the classifier activity is not only based on the intensity of the parent set of rules:
\[ \text{activity} = \text{strength} \times \text{(intensity of the parent set of rules)} \]

Dynamic
The strength is used as the basis of a competition in the last level:
1. When condition parts of classifiers coincide with the ITS knowledge, classifiers compute a bid proportionally to their strength.
2. Matched classifiers in a level increase the set of rules intensity in level downstairs, using their activity parameter.
3. At the last level, classifiers are ordered using bid competition.
4. The reward from the environment is added to the classifiers strength responsible of the selected action in last level. In our case, the reward comes from the choice of the teacher in the list of pedagogical assistances. \[ \text{reward} = 2 \times (\text{three levels bids}) \]
5. "Winning" classifiers, in the last level, pay a portion of their earnings to other classifiers that assisted them to be activated (second and first level).
6. All matched classifiers decrease their strength thanks to their bid value.

This technique increases rules strengths contributing to pedagogical assistance selected by the teacher, whereas matched rules decrease their strength thanks to bid taxes. This way, the pedagogical model adapts itself to the pair learner/teacher.

3. Conclusion and futures works
We present the models of our ITS. We present the dynamics between the models, allowing to update their knowledge and we define a process bringing to a pedagogical decision. Our ITS provides a knowledge base allowing to make a pedagogical reasoning.

The pedagogical agent simulates a pedagogical reasoning. The objective is to suggest pedagogical assistances to the teacher. The model is constituted by three parts. First, it constructs knowledge necessary to take pedagogical decision (called pedagogical situation). Such information are from agents interactions of our ITS and from the analysis of the current action being achieved by the learner. Secondly, our pedagogical model simulates a pedagogical reasoning with a structured architecture based on “if-then” rules. The model representation uses a hierarchical classifiers system. The dynamic of the system uses the pedagogical situation knowledge to check rules (condition is fulfilled?) and to suggest pedagogical assistances. Thirdly, we add a bucket brigade algorithm to adapt our pedagogical model to the pair learner/teacher.

We will apply our ITS model to a virtual environment for training simulating aircraft traffic on an aircraft carrier. The application will be designed for officers. They have to manage and order teams to solve incidents that they can't have in real situation for training exercises.

Futures works will also evaluate the impact of our system on learner’s performances, comparing result with and without our ITS.

References: