The Ghost in the Shell Paradigm for Virtual Agents and Users in Collaborative Virtual Environments for Training

Thomas Lopez, Florian Nouviale, Valérie Gouranton, Bruno Arnaldi

To cite this version:

Thomas Lopez, Florian Nouviale, Valérie Gouranton, Bruno Arnaldi. The Ghost in the Shell Paradigm for Virtual Agents and Users in Collaborative Virtual Environments for Training. VRIC 2013, Mar 2013, Laval, France. pp.29. hal-00809070

HAL Id: hal-00809070
https://hal.archives-ouvertes.fr/hal-00809070
Submitted on 8 Apr 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
The Ghost in the Shell Paradigm for Virtual Agents and Users in Collaborative Virtual Environments for Training

Thomas Lopez
INSA de Rennes, IRISA,
INRIA Rennes, France
thomas.lopez@irisa.fr

Florian Nouviale
INSA de Rennes, IRISA,
INRIA Rennes, France
florian.nouviale@irisa.fr

Valérie Gouranton
INSA de Rennes, IRISA,
INRIA Rennes, France
valerie.gouranton@irisa.fr

Bruno Arnaldi
INSA de Rennes, IRISA,
INRIA Rennes, France
bruno.arnaldi@irisa.fr

ABSTRACT
In Collaborative Virtual Environment for Training (CVET), different roles need to be played by actors, i.e. virtual agents or users. We introduce in this paper a new entity, the Shell, which aims at abstracting an actor from its embodiment in the virtual world. Thus, using this entity, users and virtual agents are able to collaborate in the same manner during the training. In addition to the embodiment’s control, the Shell gathers and carries knowledge and provides interaction inputs. This knowledge and those inputs can be accessed and used homogeneously by both users and virtual agents to help them to perform the procedure. In this paper, we detail the knowledge mechanism of the Shell as this knowledge is a crucial element for both collaboration and learning in the CVET context. Furthermore, we also validate our exposed model by presenting an operational implementation in an existing CVET and discuss of its possible usages.

Categories and Subject Descriptors
[Software and its engineering]: Software organization and properties - Virtual worlds training simulations; [Human-centered computing]: Human computer interaction - Collaborative interaction

Keywords
Virtual Environment for Learning / Informed virtual environments / Serious games for knowledge generation / Collaborative Learning environment

1. INTRODUCTION
The use of Virtual Reality for training provides many advantages. It reduces the costs and risks for the trainees as much as for the equipment and allows those trainees to repeat a procedure as many times as necessary. Another asset of virtual training is the possibility to practice team training and learn collaborative procedures, along with other team members, in Collaborative Virtual Environments for Training (CVET). Our study focuses on CVETs where virtual agents and real users collaborate towards a common goal as equal team members in the simulation. The constraint on the number of trainees required to perform a procedure is released as virtual agents can be used to replace the missing team members and act as collaborators, tutors or even perturbators. In such situations, a virtual agent should be able to interact in an analog way and with the same autonomy as a user in the virtual environment.

For the rest of this paper, we refer to both virtual agents and users as actors, and their representations in the virtual world as their mannequins.

Application examples
In the context of CVETs, actors are learning a team procedure. Contrary to mono-agent trainings, the actors, both real and virtual, have to understand how the team is working and what the others are doing during the collaborative procedure. Thus, it is essential for a user to have some feedbacks on the knowledge he has assimilated during the training. It can also be interesting for it to exchange the control of its
embodiment with another actor in order to change role and access to the knowledge associated to this new role. For instance, in the industrial area, a trainer could take over the control of a trainee’s embodiment to check if this one has got in its knowledge the necessary pieces of information before a critical task. In another scenario, a trainee could exchange its embodiment with another actor in order to learn a new part of a collaborative procedure. Looking at those examples, we identified two essential needs. The first need is to have access on the knowledge assimilated by a particular role during the procedure. The second one is the possibility to exchange the embodiment of two actors during a training to understand a collaborative procedure in its whole.

**Contribution**

The purpose of this paper is to present a new entity, the Shell, that abstracts an actor from its embodiment in the virtual world. Moreover, this entity encompasses a knowledge representation associated to the considered embodiment of an actor over the simulation. Thus, it acts as a controller of this virtual entity and can retrieve information concerning its Shell, its procedure, its teammates and the virtual world to support its decision process. Using this entity also adds the advantage to abstract the virtual or real nature of the actor by proposing a homogenous entry point for both of them to control the mannequin (and its associated knowledge) through the Shell.

The term Shell refers to the Japanese manga series "Ghost in the Shell" written by Masamune Shirow. In these futurist books, the idea is that a "Ghost" is a thinking mind that can be embodied in a "Shell", i.e. a body, and takes control over it to act in the world. In our context, the Ghost is an actor, no matter its real or virtual nature, and the Shell is an entity gathering the control over a mannequin, a perception module and a knowledge base.

We introduced the term of Shell to emphasize the difference with the term of Avatar, commonly used in the Computer Graphics field, that usually designates only the visual representation (i.e. the mannequin) of an actor in the Virtual World.

**Organisation**

We start by a review of related works in section 2 followed by the core of our contribution, the definition of the Shell and the knowledge base in section 3. We then describe an implementation of our models in our test platform in section 4 before wrapping things up in our conclusion section 5.

### 2. RELATED WORK

Our related works are divided in three parts. We first examine how virtual agents are currently used in CVETs in section 2.1. We then look how the knowledge is modeled in CVETs, agents and multi-agents settings (section 2.2). Finally, section 2.3 presents some interesting patterns proposed to represent an actor as the combination of a mind and a body in the virtual world.

#### 2.1 Virtual agents in CVETs

In the CVET literature, autonomous agents generally interact with users in three different manners:

- as a personal assistant assigned to a single trainee to take care of basic tasks and help him,
- as a team assistant ensuring the communication between users in the simulation, helping them to coordinate their actions and better focus on the important tasks requirements,
- as an equal team member operating as an autonomous entity that performs a collaborative procedure alongside users and other virtual agents.

In our context, we focus on the last situation where virtual agents interact with other actors in CVETs as full-fledged team members and take part in the procedure. Thus, they have to be able to perform tasks, interact with the objects of the environment and to some extent, communicate with other teammates. Regardless of the nature of the procedure – scenario-based or goal oriented –, the virtual agents and the users work towards a shared objective. In most CVETs, virtual agents are able to perform their task independently. In those CVET, they are generally able to play different roles such as collaborators, instructors or assistants that help the trainee. They can also replace missing team members needed for a training. Unfortunately, interactions between team members and particularly between virtual agents and users are limited. They perform parallel tasks, working towards the team’s shared goal but cannot neither interact collaboratively on a same object nor exchange their roles during the simulation.

Some platforms allow collaborative interactions between teammates. This is the case of the Generic Virtual Training (GVT) in its collaborative version. The MASCARET model proposes a framework for the design of pedagogical software in a team training setting. Although this model can potentially handle collaborative interactions, the current implementations do not deal with the issue. Moreover, none of these collaborative models allows an exchange of role or embodiment between trainees during the procedure. However, the MASCARET model does give the tutor the possibility to let a trainee do his/her job.

To sum up, the interactions between virtual agents and users are pretty much summarized by their participation in parallel tasks working towards the shared goal of the team. Their contribution mainly consists in advancing the procedure on their own, with limited interactions with others. We also found that at the best of our knowledge, no exchange mechanism is fully implemented in the current CVETs.

#### 2.2 Knowledge models

In user/agent collaboration, actors need to understand each other in order to collaborate effectively. Thus, all actors need to represent their knowledge in a way usable and understandable by both users or agents. In this respect, various theories and methods of depicting team knowledge and shared knowledge in a mixed team setting exist. According to the psychology and cognitive science literature, people use internal representations of the world to understand and act upon it. These representations are known as mental models. They serve as a reasoning mechanism that exists in a person’s mind and allow to predict, reason and form explanations of the surrounding environment.

Build from this theories, the shared mental model is a popular method in teamwork related works, as it is said essential to improve team performance. The shared mental model notion and team knowledge have been used in various multi-agents setting, such as or . In those works,
virtual agents share knowledge such as team structure, common recipes or plans [7]. These techniques have been used to form a shared understanding between humans and agents in mixed teams [25]. However, most of these works focus on a mixed team where agents are here to help users, by guiding their work or by means of user interfaces [19], and at the best of our knowledge, no work has attempted to build from the ground up a unified knowledge model where virtual agents and users work together towards the same goal. Unfortunately, none of the CVETs we know tried to handle this issue.

2.3 Actor representation

Different architectures have been proposed to represent virtual humans in environments. Some architectures notably proposed to split this representation in two parts. The body can act on the world and perceive events in it and the mind can take decisions and influence the emotive state of the actor [11, 12]. However, as far as we know such models where only used for virtual actors and the mind part was not handling the knowledge associated to the actor.

In the Interactive Story Telling field, the actor is a central concern. A character can be controlled by either a human or the system. Thus, different methods proposed an unified interface for both real and virtual actors to act upon a character [13, 14]. In this works a memory concerning the role of the character and its perceptions is attached to the character which allows to guide it during a movie or to help a user to fit to the role’s behaviour. However, the character are considered as omniscient and the memory is only used in the case a virtual agent. In the case of a user, this memory is not accessible and is used by the system as a pedagogy to give him pieces of advice to better stick to its roleplay. Moreover, this method does not propose to exchange the control of a character during the simulation.

2.4 Synthesis

Our conclusions after our review of the related works are the following:

- There are limited interactions between virtual agents and users in current CVETs.
- CVETs lack a model of team knowledge, especially in regard to a unified knowledge for virtual agents and users.
- A few techniques have been used to model the team knowledge, but at the best of our knowledge, none handle the issue of virtual agents and users working together as equals in a collaborative task.
- Homogenous architectures have already been proposed to model actors but the knowledge available relies on the actor’s nature and the abstraction is generally simplified to an informative help.

3. SHELL

This section presents our main contribution, the Shell, in three main steps, after explaining our motivations in section 3.1. Section 3.2 explains the concept and some of the different usages of the Shell and section 3.3 details the knowledge model inside the Shell.

3.1 Motivation

Designing a new collaborative procedure requires to acknowledge various elements concerning the participants, like the knowledge they acquire and use during the procedure. While learning a collaborative procedure, it is important for a trainee to understand the decisions made by his teammates, which can be virtual agents or users. In order to meet these requirements, we introduce the Shell, an entity containing a knowledge base and which control can be exchanged between virtual agents and real users. Although this does not prevent an actor from having his own knowledge base, it is essential that the Shell can provide assimilated knowledge to either allow an actor to resume its actions or to consider the knowledge acquired so far.

3.2 Concept

The Shell contains three main elements presented in Fig.1:

- An interaction capability that handles the inputs from both virtual agents and users and that controls the mannequin that acts on the virtual world.
- A knowledge model that stores pieces of knowledge of various nature.
- An acquisition module that generates pieces of knowledge and transmit them to the knowledge model.

We find this concept to be particularly relevant in CVETs with mixed agent/human teams, where each actor has a role.
to play in the collaborative task, independently of its nature. Thus, using this method, we are able to abstract the real or virtual nature of the actors during the task. Moreover, using this representation, the knowledge base of a Shell can be accessed by both type of actors, for example to help their decision process.

### 3.2.1 Interaction model

We describe in [21] an interaction model that allows objects to be collaboratively manipulated. It defines interactive objects, that present manipulable parameters and interactors that can manipulate these parameters. The Shell is designed as both an interactive objects as it is controllable by an actor and an interactor as it can act on the objects of the virtual world. Thus, both type of actors share the same representation and capacities in the virtual world as they use the Shell to interact with the virtual world. With this model, two or more mannequins can manipulate the same object, like a piece of furniture, by using identical inputs.

### 3.2.2 Exchange protocol

The exchange protocol [22] basically consists in exchanging the controls from the interaction model between the actors. It also provides new useful usages to CVET. For example, a teacher performing a medical procedure can pick a student and exchange their roles, allowing this student to perform a part of the medical procedure under his teacher's supervision. In industrial training applications, a trainer can take the control of an embodiment controlled by an autonomous agent and either help or perturb the work of the trainees. Using our protocol, this can be done without the trainees noticing the active intervention of their teacher.

### 3.3 Knowledge model

In the Virtual Environment for Training context, an actor needs to acquire new knowledge on the procedure it has to perform and also to use previously assimilated knowledge. To do so, we propose to associate to the Shell a unified knowledge model that can be understood by both real and virtual actors collaborating in a CVET. Our model aims to construct a knowledge base inspired by the mental model theory [5, 9]. According to this theory, humans use internal representations of the world that surrounds them to understand and act upon it. Those mental models serve as a base for understanding and reasoning about the world. In our model, the Shell's knowledge model is a database that sorts and stores information all along the simulation. Once acquired, this knowledge is accessible by the actor controlling the embodiment to fuel its decision process.

In this section, we detail the properties of our knowledge model as follows: section 3.3.1 presents the different mechanisms that can provide new knowledge to the model, 3.3.2 explains how the data is stored. Finally, section 3.3.3 addresses the issue of accessing the knowledge stored inside the database and the section 3.3.4 sums up our contribution.

#### 3.3.1 Acquisition

The Shell provides new knowledge to its database using various mechanisms as presented in Fig. 2:

- **The Perception** provides information about objects of the virtual world by emulating various sensory channels, such as vision or hearing.
- **The Introspection** is centered on the embodiment. It contains for example knowledge about the actions the mannequin is currently performing, or about its current state.
- **The Inference** allows to generate new pieces of knowledge using existing pieces of knowledge. It can either perform on new data only for simple and fast computation or use the whole knowledge database of the Shell. The generated pieces of knowledge are then added to the knowledge for later use.

![Figure 2: Detail of the acquisition mechanism dataflow. New pieces of knowledge are provided to the inference module (1) and to the knowledge database (2)](image)

The pieces of knowledge acquired using these mechanisms are of various nature. In order to better organize the knowledge, this information is filtered and dispatched in different parts of the knowledge model.

#### 3.3.2 Storage

To optimize the representation of the information, some mental models perform a fragmentation of the data. Different categories such as team task, tools and members are generally used to sort the pieces of knowledge. This mechanism is interesting as it first allows to have different accesses and representations for each category of knowledge. Then, it also makes it easier to compare the knowledge assimilated by different actors. We thus propose a fragmentation for the knowledge model of the Shell suited for collaborative training, as follows:

- **The Ego** contains all the information about the self. This category regroups data such as internal resource states (status of the hand), task-specific skills and knowledge (operating a machine, using a screwdriver, the action currently performed), interaction skills (lift, push), internal states (level of stress or exhaustion) or collaborative profile (disrupter, tutor).
- **The Others** contains all the information about the other Shells, i.e. the teammates’ egos and their actions. An action can here be represented for example by its name, its completion, the number and the identity of involved teammates and the different world objects used. It can also contain information about the requirements and effects of the performed action.
- **The Task** contains all task-related knowledge. Depending on the simulation, it can contain data such as the steps of the procedure that need to be performed, short or long term goals or task requirement and effect.
Along with each task can be associated the identity of the person that needs to perform the task if known, the objects involved in the task (tools, machines) and other properties such as task difficulty, estimated time of completion and information about how to perform the action.

- The World contains all the data about the world objects. It can contains elements such as last known position (absolute or relative), type (tool, machine,...), properties (weight, diameter,...) or interactive capacities (liftable, pushable, number of people required,...).

### 3.3.3 Access

There is a growing interest in collaboration between virtual agents and users. This type of cooperation not only requires compatible means of representing the world and the tasks, but also that those representations can be interpreted and used by actors of both nature [2]. Thus, humans need to be able to understand the collaborative task from the virtual agent’s perspective [4], and vice versa. As pointed out earlier, this problematic is especially important after an exchange of embodiment, as the actors involved in the process need to be able to carry on the work of their predecessor, regardless of its nature.

By accessing to the data structure directly, virtual agents can retrieve the needed knowledge in a very straightforward way. However, as human users do not have this option, it is necessary to provide them a way to browse the contents of the knowledge base hosted by the Shell. Both users and virtual agents must be aware of what action team members are performing, their reasons for performing it, and where it is on the team members’ current agenda [1]. This type of information can be given to the user by a GUI listing all the collected knowledge, a distinct display like a tablet pc or via specific metaphors among other possibilities. For example, knowledge about objects could be printed in a pop-up window on the screen while the selected object is highlighted in the scene or knowledge about an action could trigger a replay of the action.

### 3.3.4 Summary

We defined in this section the Shell as an common entity for virtual agents and users. It implements an interaction model allowing the actors to act on the virtual world. It also contains a knowledge model fueled by an acquisition module. These elements are compatible with and enhance the exchange protocol that allows actors to swap their embodiments.

### 4. IMPLEMENTATION

In this section we describe an implementation of the concepts presented in section 3. The different contributions exposed in this article have been integrated in an existing Collaborative Virtual Environment for Training, GVT Corvette platform based on GVT [2].

A video showing our implementation can be found at the following webpage: [http://vimeo.com/57677805](http://vimeo.com/57677805)

The GVT platform has been used to create our test scenario for various reasons. First, this platform possesses two branches: an industrial version and a research prototype that allows us to easily integrate our research work in this CVET. Second, this platform is already used in an industrial context for a variety of applications, from machine maintenance to railroad stations and military training, allowing a wide range of possible scenario. Moreover, its latest version, GVT Corvette, allows collaborative teamwork between multiple real users and autonomous agents where those agents can participate in the procedure as authentic team members.

We start by describing the architecture of the Shell in section 4.1, then we focus on the knowledge model through section 4.2 and present a usecase in section 4.3.

#### 4.1 Architecture

![Figure 3: Architecture of the Shell](image)

According to our model, the implementation of the Shell gathers the control, the assimilation and the knowledge attached to a mannequin in one single entity. In our implementation, the architecture of this entity, presented in Fig. 3 is composed of three main parts:

1. The acquisition modules as presented in figure [2] that allow the entity to assimilate new knowledge.
2. The interface with the environment that allows the Shell to interact with the environment using the interaction protocol on the mannequin.
3. Finally, the knowledge database that stores and classifies the knowledge acquired by the Shell for later use.

The Shell is regularly running the acquisition modules and dispatch the new pieces of knowledge in the different parts of its knowledge database based on their nature, as described in section 4.2. The Shell also provides several inputs and outputs. The more important are the ones linking the actor to its controlled Shell. The actor can thus control the mannequin by sending commands to the Shell, access the knowledge database content and receive events when new pieces of knowledge are assimilated.

#### 4.2 Knowledge Model

In this section we describe our implementation of the knowledge model. The assimilative abilities of the Shell are described in section 4.2.2, the knowledge storage in section 4.2.2 and its access in 4.2.3.
4.2.1 Acquisition
For each acquisition module described in section 3.3.1, we implemented the following solutions:

- The omniscient perception, which gets all the data present in the virtual world. This solution corresponds to the one generally used in the current CVET to fill the knowledge model.
- The limited perception, which simulates the vision of the team member by only gathering data about objects and members that are visible in a limited field of view.
- The action inference, which uses knowledge about any perceived member to create a new piece of knowledge containing the action currently performed by the said team member.
- The introspection, that provides new pieces of knowledge created by the module driving the mannequin of Shell, like the action previously performed.

![Figure 4: Screenshot of our test scenario: A real user is accessing to the knowledge of its Shell to have information about the item "Pulling mass". This item is selected in the list (A) and highlighted in the environment (B) while the knowledge is displayed in another window (C).](image)

4.2.2 Storage
In order to have an efficient perception, we attach various data to the objects of the virtual world, like their types, their locations, their properties (such as their weights and sizes), ... This data composes the knowledge associated with the perceived object. When a new piece of knowledge is presented to the knowledge model, it will first remove any similar piece of knowledge already in the database if the new data is more complete or more up to date, before adding this new knowledge to the database.

We presented in section 3.3.2 the decomposition of our knowledge model in different subcategories: Ego, Team, World and Task. We implemented in our test scenario a first version of this knowledge model that divides the knowledge using these categories (see Fig. 3). The Ego is currently filled by the knowledge obtained through introspection. The World is filled by the knowledge coming from the perception, while the Team gets its data from the inference that uses the perception output to generate knowledge about the actions performed by the other Shells. The last type of knowledge, (Task), concerns the completion of the task the user has to perform. This knowledge is related to the actor’s procedure and the skills associated to the Shell as it indicates different means to perform a precise part of the procedure. For instance, in our scenario, a back seat has to be moved by a trainee and this action can be executed in three different ways. An virtual agent is then able to use the knowledge about this action in order to choose how to perform this task.

4.2.3 Access
Both virtual agents and users need to access the knowledge database of the Shell they are controlling. In the case of virtual agents, the queries depend on their decision algorithm that may need to access to specific pieces of knowledge. In the case of users, the knowledge is there to complete their expertise on the training, and they can use it only when needed. For them, we can rely on different display methods to access to this knowledge. We currently use a GUI that displays either all of the pieces of knowledge possessed by the Shell or only the recent ones. This list allows a user to select an item that will be highlighted in the environment while another GUI windows shows the information stored about it. Fig. 4 illustrates this display for an element of the scenario where extra information has been previously added. The user can also filter the whole knowledge database content by selecting some types amongst the ones available, like all the object with the type: "door" or "tool". He or she has the possibility to directly enter text in a search field. That will filter the knowledge database and only show the elements that names contain the given text.

Finally, information concerning the other trainees is directly displayed in the environment. When a user's Shell has another mannequin in its field of view, a pop-up appears above the head of the mannequin showing what it is doing and how far it is in the execution of its current task, as shown in Fig. 5. This data correspond to the knowledge previously generated by the inference module. This display is particularly helpful for collaboration.

Various other displays could be imagined for the objects or the other users actions, notably in order to determine what would be the better media to access and interact with these knowledge for users, but this problem is not in the scope of...
Shell is given control of the scenario. Once all the trainees are ready, they are first to select one of the roles at the beginning even if there are less trainees than roles to provide. Each agent allows the trainees to execute the learning process and fill in the five roles. The use of autonomous agents allows the trainees to execute the learning process even if there are less trainees than roles to provide. Each trainee has first to select one of the roles at the beginning of the scenario. Once all the trainees are ready, they are given control of the Shell corresponding to their role and execute their part of the training scenario. When the scenario begins, the roles that have not been taken by real users are given to virtual agents that will be able to execute their part of the work in the training procedure. Throughout the procedure, discrete visual effects warn the user when a new object has been perceived by the Shell. He or she can also access the knowledge database of the Shell, using the dedicated interface for knowledge concerning objects and through the display of the visible actors current actions details. At any time, a user can switch his or her embodiment and proceed with the procedure of his or her new role by exploiting the knowledge of the newly controlled Shell.

4.3 Usecase

To better highlight the real benefits of our contribution, we present our results in a test scenario resulting from a concrete CVET application defined with specialists of the domain. This test scenario consists in a team of five members, each with one different role, learning a procedure. Each member has to perform a sequence of actions that only him is able to perform in the environment. Thus, the presence of all of the team members is required to carry out the entire procedure and fill in the five roles. The use of autonomous agents allows the trainees to execute the learning process even if there are less trainees than roles to provide. Each trainee has first to select one of the role at the beginning of the scenario. Once all the trainees are ready, they are given control of the Shell corresponding to their role and execute their part of the training scenario. When the scenario begins, the roles that have not been taken by real users are given to virtual agents that will be able to execute their part of the work in the training procedure. Throughout the procedure, discrete visual effects warn the user when a new object has been perceived by the Shell. He or she can also access the knowledge database of the Shell, using the dedicated interface for knowledge concerning objects and through the display of the visible actors current actions details. At any time, a user can switch his or her embodiment and proceed with the procedure of his or her new role by exploiting the knowledge of the newly controlled Shell.

5. CONCLUSION

Virtual agents and real users have often been considered in the literature as two different problems in virtual environments. In this paper, we proposed to consider them as two aspects of the same problem and we defined a common entry point for the real and virtual actors in the virtual world. To do so, we introduced the Shell that separates the thinking mind, i.e. the actor, from its embodiment in the virtual environment, i.e. its mannequin.

This Shell implements an interaction model to be controlled and to control the mannequin its associated with and contains a knowledge model based on acquisition components and a sorted database. Knowledge is thus accumulated during the procedure and accessible by the actor that controls the Shell. This knowledge can be used in order to exchange embodiments between actors of both nature, understand the decision process of a team member or enhance the collaboration with the other members. We finally proposed a functional implementation of our model in an existing CVET proving our Shell model as a valid abstraction model for both users and virtual agents.

Future works will first focus on proposing different uses of the Shell. For instance, it would be interesting to have a simultaneous access on a Shell by two or more actors. A real or virtual actor could then follows a user as a passive controller, monitor his actions or takes over a part of the procedure when needed. This control sharing could also allow a trainer to fine-tune a manipulation performed by a trainee. Other research will focus on the knowledge, notably by looking for different means to acquire new knowledge, such as oral communication between the different actors of the procedure. A second research axis will concern new ways for a real user to browse and use the knowledge of the Shell. We intend to make user experiments in order to compare various representation metaphores and determine the easiest and more efficient way to display the knowledge depending on their nature and the context. Finally, the perception and the deductions modules used by the Shell should also be refined. However, this task relies more on Artificial Intelligence techniques than on CVET and should thus be explored according to this research area.

6. ACKNOWLEDGMENTS

This work was supported by the French Research National Agency project named CORVETTE (ANR-10-CONTINT-CORD-012) and the French Unique Interdepartmental Funds SIFORAS (FUI 11).

7. REFERENCES


