The Knowledge Engineering Review, Vol. 23:4, 369–388. © 2008, Cambridge University Press
doi:10.1017/S026988890800009X Printed in the United Kingdom

Simulating socially intelligent agents in semantic virtual environments

FRANCISCO GRIMALDO, MIGUEL LOZANO,
FERNANDO BARBER and GUILLERMO VIGUERAS

*Computer Science Department, University of Valencia, Dr. Moliner 50, 46100 Burjassot, Valencia, Spain;
e-mail: francisco.grimaldo@uv.es, miguel.lozano@uv.es, fernando.barber@uv.es, guillermo.vigueras@uv.es*

Abstract

The simulation of synthetic humans inhabiting virtual environments is a current research topic with a great number of behavioral problems to be tackled. Semantical virtual environments (SVEs) have recently been proposed not only to ease world modeling but also to enhance the agent–object and agent–agent interaction. Thus, we propose the use of ontologies to define the world's knowledge base and to introduce semantic levels of detail that help the sensorization of complex scenes—containing lots of interactive objects. The object taxonomy also helps to create general and reusable operativity for autonomous characters—for example, liquids can be poured from containers such as bottles. On the other hand, we use the ontology to define social relations among agents within an artificial society. These relations must be taken into account in order to display socially acceptable decisions. Therefore, we have implemented a market-based social model that reaches coordination and sociability by means of task exchanges. This paper presents a multi-agent framework oriented to simulate socially intelligent characters in SVEs. The framework has been successfully tested in three-dimensional (3D) dynamic scenarios while simulating a virtual university bar, where groups of waiters and customers interact with both the objects in the scene and the other virtual agents, finally displaying complex social behaviors.

1 Introduction

Socially intelligent agents are autonomous problem solvers that have to achieve their goals by interacting with other similarly autonomous entities (Hogg & Jennings, 2001). Bearing this in mind, multi-agent systems are normally referred to as *societies of agents*, and provide an elegant and formal framework to design social behaviors for three-dimensional (3D) autonomous characters. A major goal in behavioral animation is the construction of an intelligent system able to integrate the different techniques required for the realistic simulation of the behavior of virtual humans. Among them, we can include perception, motion control, goal selection, action execution, communication between agents, their interaction with the environment, etc. (Iglesias & Luengo, 2004). This paper introduces a multi-agent framework dealing with these issues, which is oriented to simulate socially intelligent characters in 3D virtual environments.

Many AI-based models have been applied to virtual worlds in order to increase the behavioral complexity of the objects and the actors involved. When designing such agents, the main concern has normally been with the decision-making mechanism, as it is responsible for the actions that will be finally animated. For example, game narrative (Harris & Young, 2005) and storytelling systems (Charles *et al.*, 2003) have productively applied planning techniques to generate dynamic and interactive stories. Although these approaches are well known to be *knowledge intensive*, there is no common formalism to manage the great amount of information associated with the

environment. Instead, all that knowledge is generally assigned only to the actors' plans and the agents can hardly reuse their skills. For instance, whereas a real barman has the general ability to serve drinks to customers, a corresponding virtual barman would normally define one operator for each of the tasks that the role can carry out. Hence, an agent might be able to serve an orange juice and a glass of wine while not having any idea about how to serve a cup of tea. This happens because virtual agents do not actually have general but particular understanding about the situations they face. Nevertheless, regardless of the nature of the application—educational, entertainment, training, etc.—the definition of a semantic knowledge base would benefit production, visualization and interaction within intelligent virtual environments (IVEs) (Luck & Aylett, 2000). Scene graphs are insufficient to specify knowledge associated with virtual entities beyond simple relationships between objects. The need for richer expressiveness has led to semantic virtual environments (SVEs) that take into account not only the spatial structure but also high-level semantics. Therefore, we propose the use of ontologies to enhance both agent–object interaction and manageability of complex scenes that contain interactive objects. For instance, we manage containers such as a tray with glasses, a cabinet containing dishes, shelves with bottles on top, etc.

Semantics can enrich planning in knowledge-rich application domains (Gil, 2005); however, rationality is not the only issue to address when designing virtual humans in 3D scenarios (Conte, 1999). This kind of agents usually operate in dynamic resource-bounded contexts where obstructions rapidly appear since they compete for the use of shared resources—that is objects in the environment or events to fulfill such as a new order placed by a customer in a virtual bar that can be served by several waiters. According to this, social simulations normally require group coordination, as self-interested agents—uniquely devoted to accomplish a set of goals—easily come into conflicts even though their goals are compatible, producing low-quality simulations. Furthermore, since characters normally represent human roles in the scenario, such as a waiter or a customer in a bar, the social network formed by the relations among the members of the society should be considered when animating their behavior. We also propose the use of ontologies to define such relations and to enhance sociability. To incorporate human-style social reasoning in virtual characters, we have developed a market-based social model that follows the multi-agent resource allocation approach presented in Hogg & Jennings (2001), where agents express their preferences using utility functions. This model coordinates the activities of groups of virtual humans and also includes social actions in the agent decision-making. The dynamics of social interactions are inspired by the theory of Piaget (1995) over which we have implemented reciprocal task exchanges between agents.

The remainder of the paper is organized as follows. Section 2 provides an overview of previous works on the simulation of inhabited virtual environments. In Section 3, we have introduced our multi-agent simulation framework. Section 4 presents the ontologies that have been defined and their usage to manage sensorization of complex scenes. Section 5 defines socially intelligent agents that use the semantic knowledge to enrich their operativity and sociability. Section 6 describes an illustrative example modeled to test our framework and Section 7 analyzes some results extracted from it. Finally, conclusions are drawn in Section 8.

2 Simulation of inhabited virtual environments

2.1 Semantic virtual environments

The use of semantics as a means to develop virtual worlds is becoming familiar within the graphics community. In VR-Wise (Pellens *et al.*, 2005), a non-virtual reality specialist can design a virtual environment using a set of DAML + OIL ontologies. Afterwards, the system maps concepts of a world domain onto VRML graphical representations that are used to render the scene. The semantic model presented by Gutierrez (2006) defines the abstract concept of virtual entity. This concept includes the geometry and the user interface of the objects and is mainly used to visualize

and interact with the items in different contexts/devices. Additionally, a preliminary version of ontology for virtual humans is also introduced. This XML-based ontology is aimed at modeling and animating the human body as well as at the interaction of virtual humans with smart objects (Kallmann & Thalmann, 1999). Therefore, agent–object interaction is governed by the environment since all the information relative to it are stored in the objects themselves. Unfortunately, smart objects, as introduced by Kallmann, are strictly targeted at animation and support neither high-level reasoning nor symbolic description of the properties of the objects. Synoptic objects in STARFISH (Badawi & Donikian, 2004) follow the same philosophy as smart objects; however, the agent's decision cycle is not dealt with. The idea behind these approaches is to reduce the complexity of the agent tasks by transferring animation-oriented information to the virtual environment. Similarly, the notion of coordination artifacts (Viroli *et al*., 2006) has been proposed as an environment-based alternative for structuring interactions and providing collaborating agents with specifically designed coordination tasks.

Populated virtual cities have also used semantic information to define population profiles and topologies of complex urban environments (Costa de Paiva *et al*., 2005; Farenc *et al*., 1999; Thomas & Donikian, 2000). These *informed environments* define space partitions—buildings, streets, road intersections, etc.—and the objects contained in it, for example, crosswalks, bus stops and so on. The structured information is then used by intelligent agents such as pedestrians or car drivers basically for navigation purposes.

When developing autonomous agents, the notion of *Knowledge in the World* (Doyle, 2002) proposes annotated environments as a way to allow believable agents to act across different worlds. More recently, the SeVEn platform (Otto, 2005) aims at developing system-independent software that can be reused over several virtual environments. The platform supports the definition of SVEs based on a W3C resource description framework (RDF) and categorizes objects using an explicit-type field. Depending on this field, task-relevant information about an object can be extracted. However, static annotations do not suffice for dynamic environments.

From the agents' community, a semantically enhanced approach attains the annotation of the environment on the fly using an ontology-based concept layer (Chang *et al*., 2005). The concept model helps intelligent planning agents to achieve flexible behavior through ontological inference. Furthermore, ontologies can be used to model the social relations between the agents (Kao *et al*., 2005) and support social reasoning—that is social constraints and conflict resolution. However, the action scheme adopted here has several expressiveness limitations since the effect of an action is restricted to one single object, that is, it always implies placing the affected object as an instance of another different qualitative concept. Instead of that, actions sometimes can affect multiple objects or change an object property without modifying any concept. An example could be *to fill up a glass using a bottle of wine*. In such an action, the quantity of liquid contained in both objects should be modified but they should remain instances of the same concepts.

Agent–object interaction has been enriched using semantic information in more complex action schemes. Parametrized action representation (Badler *et al*., 2000) defines uninstantiated actions (UPARs) that do not specify either the virtual character or the objects involved in the action, thus representing all the actions possible in the system. This way, instantiated actions can basically extend UPARs with the virtual entities involved. The task definition language (Vosinakis & Panayotopoulos, 2003) aims at defining context-independent tasks using high-level language. The main advantage of the proposed language is that it enables tasks to be easily constructed and reused by different agents and in different environments. The object-specific reasoner (OSR) (Levison, 1996) classifies the objects into taxonomies and then uses that information to decide what to do with it—for example containers can be opened by hands. Another interesting use of semantics is shown in Soto & Allongue (2002), where the separation of actions and consequences brings interoperativity and reusability to virtual entities. However, the authors do not propose a formalized ontology that can be used to build and manage virtual worlds. We propose the use of ontologies to help sensorization and actuation of intelligent virtual agents in complex scenes (Grimaldo *et al*., 2006).

2.2 Behavioral animation of virtual humans

Much research has been done on the behavioral animation of virtual agents over the last few years—see Badler *et al.* (1993) for a good introduction to the field. The pioneer work of Tu & Terzopoulos (1994) showed how to design a framework to animate natural ecosystems with minimal input from the animator. He simulated *Artificial fishes* in the natural complexity of virtual underwater worlds. However, human behavior is clearly different and more complex to emulate. Possibly, the more relevant works in this field came from Thalmann's group (Farenc *et al.*, 1999; Raupp & Thalmann, 2001; Thalmann & Monzani, 2002). The goal of these previous works was to design agents with a high degree of autonomy without losing control. Their agents are an extension of the BDI architecture described in Rao & Georgeff (1991), and they include internal states as emotions, reliability, trust, etc. BDI-based agents have been also used in games such as Epic Game's Unreal Tournament (Kim, 2003) or Lionheads Studios's Black&White (Molyneux, 2001).

Some research has been done on the believability issues of groups of synthetic characters, usually centered on the interactions either between a human user and a single character (Bickmore & Cassell, 2001) or among the synthetic characters (Tomlison & Blumberg, 2002; Schmitt & Rist, 2003). These interactive scenarios often present tasks to the participants that must be solved collaboratively (Prada & Paiva, 2005). From the interaction among synthetic characters emerges the notion of an artificial society. MAS-SOC (Bordini *et al.*, 2005) aims at creating a platform for multi-agent-based social simulations, which is similar to our purposes. In this context, work is ongoing to incorporate social-reasoning mechanisms based on exchange values (Ribeiro *et al.*, 2003).

Behavioral animation of artificial societies has also been tackled from the field of coordinated multi-agent systems. For example, in generalized partial global planning (GPGP) (Decker & Lesser, 1997), agents merge the meta-plans describing their operational procedures and they figure out the better action in order to maximize the global utility. Task delegation has been also considered to obtain coordination in hierarchical societies. Among these approaches, no negotiation mechanism is considered (such us bidding or contracting), since leader agents are allowed to delegate tasks to subordinated agents. TAEMS framework (Decker & Lesser, 1997) describes the coordination relationships between the tasks that are carried out by different agents. It represents tasks at multiple levels of abstraction, from the grouping of tasks that share explicit inter-relationships to executable methods such as fully instantiated plans. Collaboration is supported in the RETSINA system (Giampapa & Sycara, 2002), thanks to the use of communicative acts that synchronize tasks and occasionally manage conflicts. A DECAF framework (Iglesias & Luengo, 2004) is an agent toolkit devoted to easily design MAS solutions. Here, planning is achieved using hierarchical task networks (HTNs), as in the RETSINA system, while communication issues are based on KQML. Team formation and task coordination have been studied for heuristic search planning (HSP)-based virtual humans in Ciger (2005) and Grimaldo *et al.* (2005) to adapt better to the dynamism of shared environments.

Although the results obtained by the previous approaches show realistic simulations of many task-oriented behaviors, autonomous characters should also display social behaviors—such as interchanging information with their partners or grouping and chatting with their friends. This kind of socially intelligent animation agents is required in many complex simulation environments: military/civil simulations, social pedestrians in virtual cities, games and probably very soon in large-scale distributed environments such as Second Life.

3 Multi-agent simulation framework

This paper presents the multi-agent framework oriented to simulate socially intelligent characters in 3D virtual environments. The framework has been developed over Jason (Bordini & Hübner, 2007), which allows the definition of BDI agents using an extended version of

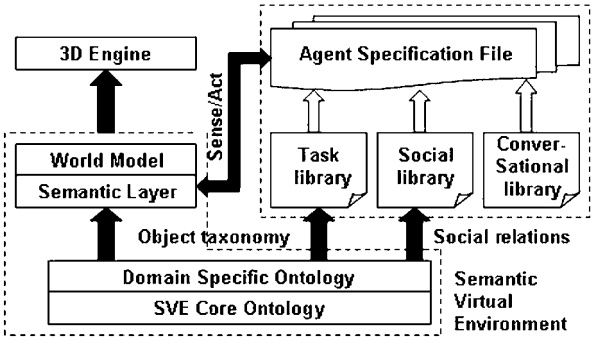

Figure 1 Multi-agent simulation framework

AgentSpeak(L) (Rao, 1996). Figure 1 shows the architecture of the system, which can be divided into several parts:

- Ontologies define the world knowledge base as well as the set of all possible relations among the agents within an artificial society. We distinguish two levels of representation: the *SVE Core Ontology* is a unique base ontology suitable for all virtual environments that can be extended by different *Domain-Specific Ontologies* in order to model application-specific knowledge. Then, environments can be constructed by instantiating the classes of these ontologies. For example, in Section 6, we will create groups of waiters and customers that inhabit a virtual bar with a large number of objects—such as bottles, glasses, etc.
- Environment is handled by the *Semantic Layer*, which acts as an interface between the agent and the world. When sensing knowledge-rich objects such as a tray with 20 glasses, it uses the ontology to reduce the information flow (see Section 4). Besides, this layer is in charge of executing the actions requested by the agents as well as maintaining their semantic effects. The animation system—virtual characters, motion tables, etc.—is located at the *3D Engine* that can extract graphical information from the *World Model* database, thus performing visualization.
- Socially intelligent agents receive sensorial information from the *Semantic Layer* and calculate the appropriate sequence of actions in order to achieve their goals. The agent decision-making is defined in the *Agent Specification File*. This file contains the agent's finite state machine. The Task Library contains the operators that sequence the actions needed to animate a task. In this context, the object taxonomy defined in the ontology is used to generalize operators, which, in turn, will increase the interoperativity of the agents across different scenarios (see Section 5). As stated above, only rational behaviors are not enough to simulate agent societies. Therefore, we included a Social library to manage different types of social situations. This library is based on an auction model and uses social welfare concepts to avoid conflicts and allow the agents to behave in a coordinated way. The social library also incorporates a reciprocity mechanism to promote egalitarian social interactions. Finally, the conversational library contains the set of plans that handle the animation of the interactions between characters (e.g. ask someone a favor, planned meetings, chats between friends, etc.).

4 Ontology-based semantical virtual environment

4.1 SVE core ontology and domain-specific ontologies

Ontologies in our SVE define the abstract world model, that is, the hierarchy of classes as well as their properties and possible inter-relationships. We have adopted the Web ontology language (OWL) (W3C, 2004) to implement them. In particular, the *OWL DL* sublanguage, as it provides maximum expressiveness without losing computational completeness.

The *SVE Core Ontology* defines the basic classes needed to create any virtual environment and allows us to use inheritance to define the object taxonomy. According to this, Figure 2 shows a

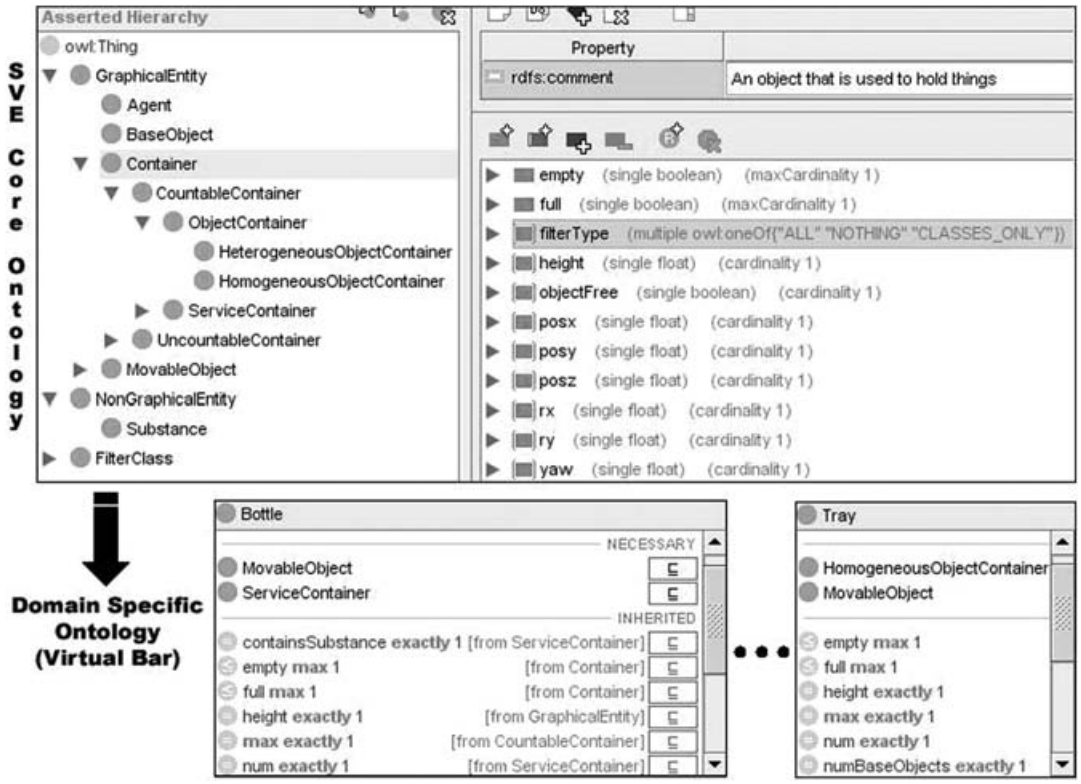

Figure 2 SVE core ontology and virtual bar ontology

fragment of the core ontology[1] focused on the classes inherited to represent different *Container* objects. Here, containers of uncountable substances—*UncountableContainers*—such as a pot with salt are distinguished from those that hold countable elements—*CountableContainers*. Among them, *ServiceContainers* provide a number of services of entities with no associated visual model—for example, a bottle of whisky—while *ObjectContainers* contain graphical objects. Moreover, *HeterogeneousObjectContainers* such as a cabinet with different items—for example, books, bottles, etc.—are differentiated from *HomogeneousObjectContainers*, which only contain undistinguishable objects—for instance, a shelf with clean glasses. Figure 2 also shows the properties shared by all *Containers*, both quantitative—such as *location* or *height*—and qualitative/semantic—e.g. *empty* or *full*. The core ontology also defines the set of possible relations for *MovableObjects*: currently, *in*, *on* and *pickedBy* (see Figure 4).

The *SVE Core Ontology* provides a high-level classification; however, certain application domains contain objects with special traits. Then, a *Domain-Specific Ontology* can be used to extend the core ontology and to represent new classes in a particular scenario. This way, for the virtual bar example of Section 6, we have defined classes such as *Bottle*, which inherits from *MovableObject* and *ServiceContainer* (see Figure 2).

Social relations among the agents within an artificial society can also be ontologically represented in the form of inter-relations between classes of agents. Figure 3 shows the extensions made to the object ontology in order to hold agent relations. We distinguish two basic levels of social relations: the level of individuals—*agentSocialRelations*—and the institutional level—*groupSocialRelations*. When one agent is related with another single agent, an *agentSocialRelation* will link them. Different application domains can need specific relations; thus, domain-specific ontologies are used to inherit particular relations from the core ontology. For instance, the

[1] Protege (Stanford Medical Informatics, 2006) has been used to develop the ontologies.

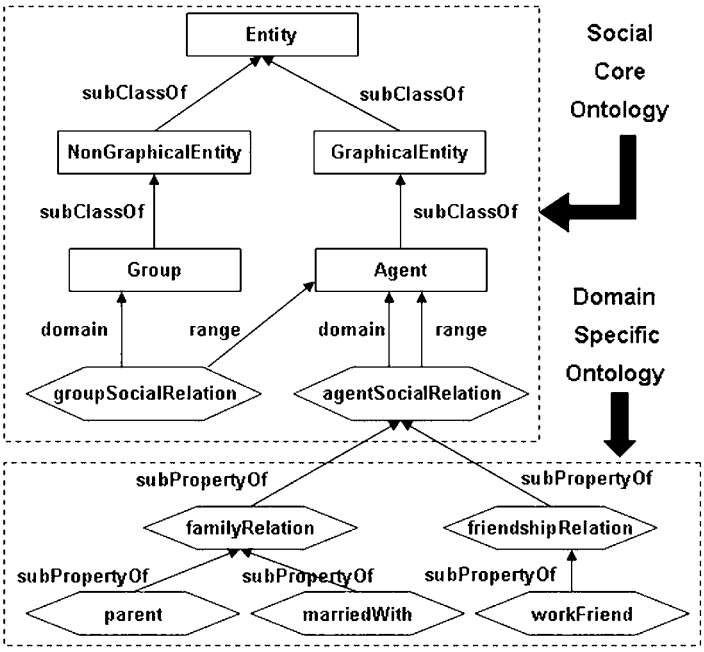

Figure 3 Social ontology

property *workFriend* is used by the waiters in the virtual bar (see Section 6) to model the characteristic of being a friend of a workmate. Other examples of individual relations are family relations such as being parent of or being married to another agent. In this case, there is not only semantic but also structural difference, since parent is a unidirectional relation whereas *marriedWith* is bidirectional.

On the other hand, *groupSocialRelations* can be used to represent an agent belonging to a group. The social network created by this type of relation can be explored to get the rest of the agents of the same group, thus modeling a one-to-many relation. The *Group* class is an abstraction of any kind of aggregation. Therefore, we can model from physical groups such as the players of a football team to more sophisticated mental aggregations such as individuals of a certain social class or people of the same religious ideology. The dynamics of how these relations are created, modified and terminated falls outside the scope of this paper. Thus, at the moment relations are set off-line and do not change during the simulation.

4.2 Semantic layer

The *Semantic Layer* uses the ontology to properly manage agent–object interaction during simulation. It is mainly concerned with two issues: sensing and actuation.

Knowledge-rich environments can be hard to sense for intelligent agents—for example, a shelf in a bookstore contains too much information for any planning search state. To properly balance expressiveness and manageability, this layer uses a tree to hierarchically extract the scene state (see Figure 4). Links in this tree represent sensorial dependencies between objects, which are modeled through *senseDepends* relations. *SenseDepends* has been modeled as an OWL superproperty defined in the *SVE Core Ontology*. The range associated with this superproperty is the superclass *FilterClass*, which filters the information in accordance with the value of the property *filterType*. This property is configured initially and it can be changed dynamically. For instance, relation *in* relates objects with containers; so, each *Container* can manage the information that is sensed about the contained objects. Currently, we have implemented three kinds of *FilterClass* filters: *ALL*, *NOTHING* and *CLASSES-ONLY* (see Figure 2). Any information can pass through the *ALL* filter whereas the *NOTHING* filter blocks everything. On the other hand, an object with the

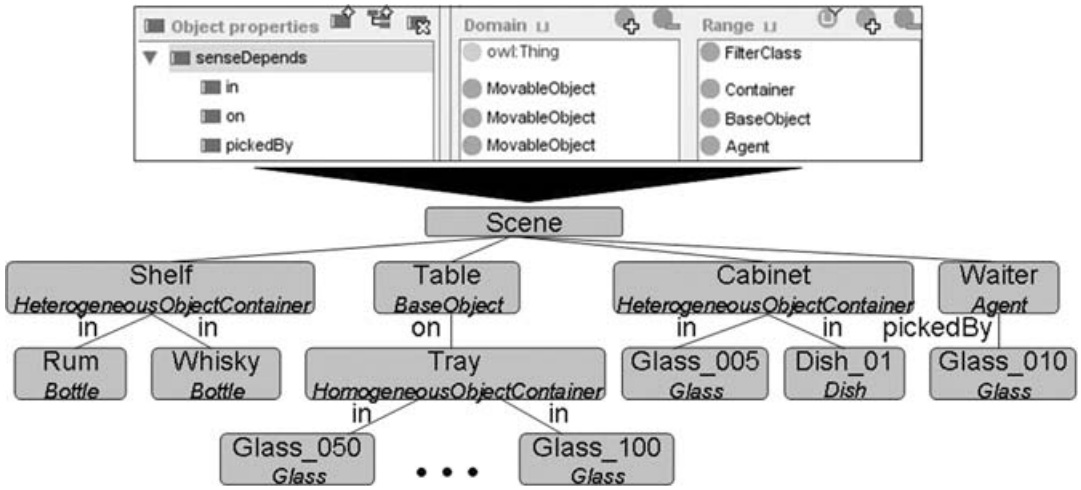

Figure 4 Sensorial dependency tree

CLASSES-ONLY filter only publishes the classes of their subordinates. These filters allow the definition of semantic levels in detail of complex objects; for instance, a cabinet can be modeled as a *HeterogeneousObjectContainer* that only publishes the classes of the objects inside while it is closed. Additionally, the *Semantic Layer* implements a specific sensorial behavior for *HomogeneousObjectContainers*. Within them, objects are supposed to be undistinguishable; thus, this kind of containers can only publish the information about a reduced number of k interactive objects[2]. This way, interaction with these objects is guaranteed at anytime while the amount of information is lowered. In the example of the virtual bar, this behavior will be applied to a shelf with a lot of clean glasses (see Section 6).

The *Semantic Layer* also executes the actions requested by the agents. It supports the procedures that check action success/failure and applies the effects to the objects involved. Thus, actions can change object relations, quantitative properties and also semantic properties. In our virtual bar, for example, when a waiter grabs one glass from a shelf the action also decreases the number of glasses contained in the shelf and changes the *empty* property to true if the shelf contains none.

5 Socially intelligent agents

5.1 Task library

Goal-oriented virtual actors need an action scheme to plan their tasks and to achieve their goals. Usually, BDI-based agents manage their operativity through a set of plans that addresses the situations that can arise during the lifetime of the agent. BDI agents using propositional planning instead of pre-built plan libraries are possible (Meneguzzi *et al.*, 2004) but not common. Planning-based agents are normally inspired by the STRIPS (ADL/PDDL) action language, where activities are represented using a classic state model with a conjunction of grounded literals. This approach has been used in storytelling domains (Charles *et al.*, 2003) and other task-oriented 3D agents (Lozano *et al.*, 2004). In these contexts, the usual representation of the actions suffers from different drawbacks when an actor is interested in reusing his/her operators with different objects. Thus, the actors' activity should be reviewed to incorporate semantic information from the ontology, which represents the abstract world model for the actors. As demonstrated by Levison (1996) and Badler *et al.* (2000) the classification of objects into taxonomies can be utilized to define general operators that can be instantiated over different objects.

[2] Where k is also a dynamically configurable object property.

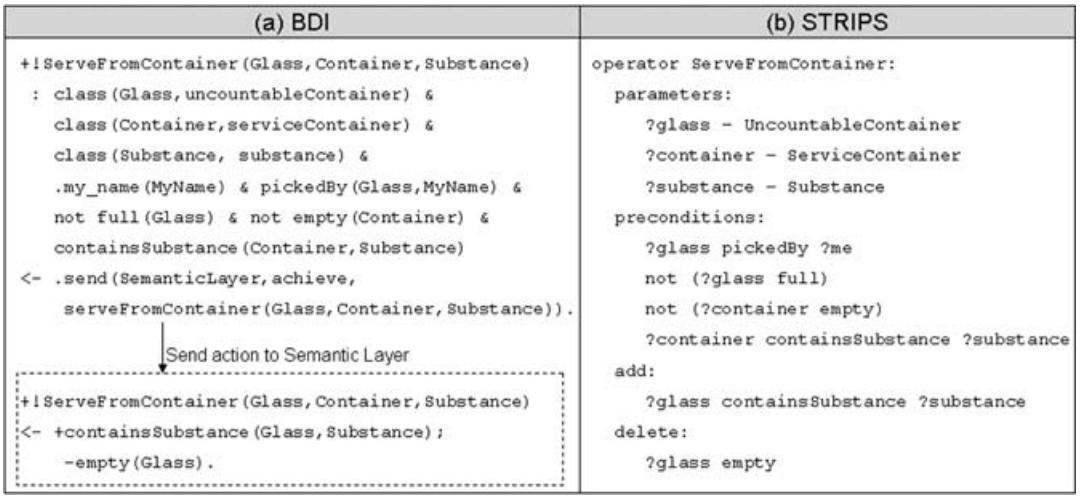

Figure 5 General and reusable operators expressed as BDI plans (a) or in STRIPS-based logic (b)

In this section, we show how the object taxonomy defined in the ontologies of our SVE can be used to create reusable operators regardless of the formalism being taken to implement the agent decision-making—BDI or propositional STRIPS-like planning (see Figure 5). A frequent action for a waiter in a virtual bar is serving any *Substance* contained in a *ServiceContainer* into an *UncountableContainer*. The operator *ServeFromContainer* in Figure 5a defines the preconditions that the objects of the environment have to satisfy in order to execute the action upon them. For example, it allows both ice cubes to be served from an ice container and donuts to be served from a tray, providing they are instances of the class involved. The object taxonomy defined in the ontology is used to classify the interactive objects and to get their properties. As stated previously, the *Semantic Layer* will finally execute the action, thus applying the effects to the objects involved. On the other hand, Figure 5b shows the STRIPS definition of the *ServeFromContainer* operator using the semantic information from the ontology. Now, the parameters section contains the objects involved, and the *preconditions*, *add* and *delete* lists follow normal STRIPS assumptions.

This action scheme that basically introduces variables and types referred to the hierarchy of classes avoids the definition of an operator for each object, which finally reflects in a higher degree of interaction. Furthermore, agent interoperability between different scenarios is enhanced since they will be able to manage any object of the world provided that it is an instance of a class defined in the ontology.

5.2 Social library

The simulation of worlds inhabited by interactive virtual actors normally involves facing a set of problems related to the use of shared limited resources and the need to animate pure social behaviors. Both types of problems are managed by the Social library by using a multi-agent resource allocation approach (Hogg & Jennings, 2001). This library allows any agent to auction tasks in order to reallocate them so that the global social welfare can be increased. Tasks are exchanged between agents using a first-price sealed-bid (FPSB) auction model where the agents express their preferences using performance and social utility functions.

The performance utility function $U^i_{\text{perf}}(\langle i \leftarrow t \rangle)$ of a bidder agent i reflects the efficiency achieved when the task t is allocated to the agent i ($\langle i \leftarrow t \rangle$). There could be many reasons for an agent to be more efficient: it may be performing the task faster than others because of his know-how or it may be using a resource that allows several tasks to be performed simultaneously—for example, a coffee machine in a virtual bar can be used by a waiter to make more than one coffee at the same time. The utility function has to favor the performance of the agents, but high

performances can also be unrealistic for the animation of artificial human societies. For example, if all agents work as much as they can, they will display unethical or robotic behaviors. Furthermore, agents should also show pure social behaviors to animate the normal relations between the members of a society.

Whereas the performance utility function modeled the interest of an agent to exchange a task from an efficiency point of view, we introduce two additional social utilities to represent the social interest in exchanging a task. The aim of social utilities is to promote task allocations that lead the agents to perform social interactions with other agents—for example, planned meetings with their friends. Therefore, these functions take into account the social relations established between the agents and defined in the ontology to compute the value that expresses their social preferences. Negotiation of long sequences of actions is not very interesting for interactive characters, as plans are likely to be thwarted due to the dynamism of the environment and to other unpredictable events. Thus, we define the following social utility functions:

(a) Internal social utility ($U^i_{\text{int}}(\langle i \leftarrow t, j \leftarrow t_{\text{next}}\rangle)$) is the utility that a bidder agent i assigns to a situation where i commits to do the auctioned task t so that the auctioneer agent j can execute his next task t_{next}.
(b) External social utility ($U^i_{\text{ext}}(\langle j \leftarrow t\rangle)$) is the utility that a bidder agent i assigns to a situation where the auctioneer agent j executes the auctioned task t while i continues with his current action.

The winner determination problem has two possible candidates coming from performance and sociability. In Equation (1), the welfare of a society is related to performance; hence, the winner of an auction will be the agent that bids the maximum performance utility. On the other hand, Equation (2) defines the social winner based on the maximum social utility received to pass the task to a bidder ($U^*_{\text{int}}(t)$) and the maximum social utility given by all bidders to the situation where the task is not exchanged but performed by the auctioneer j ($U^*_{\text{ext}}(t)$):

$$winner_{\text{perf}}(t) = \left\{ k \in Agents | U^i_{\text{perf}}(t) = \max_{i \in \text{Agents}} \{U^i_{\text{perf}}(\langle i \leftarrow t\rangle)\} \right\}, \tag{1}$$

$$winner_{\text{soc}}(t) = \begin{cases} j & U^*_{\text{ext}}(t) = U^*_{\text{int}}(t), \\ i & U^*_{\text{ext}}(t) < U^*_{\text{int}}(t) \wedge U^i_{\text{int}}(t) = U^*_{\text{int}}(t). \end{cases} \tag{2}$$

To balance task exchange, social utilities are weighted with a reciprocity matrix (see Equations (3) and (4)). We define the reciprocity factor w_{ij} for two agents i and j, as the ratio between the number of favors—that is tasks—that j has made to i (see Equation (5)):

$$U^*_{\text{int}}(t) = \max_{i \in \text{Agents}} \{U^i_{\text{int}}(\langle i \leftarrow t, j \leftarrow t_{\text{next}}\rangle) w_{ji}\}, \tag{3}$$

$$U^*_{\text{ext}}(t) = \max_{i \in \text{Agents}} \{U^i_{\text{ext}}(\langle j \leftarrow t\rangle) w_{ij}\}, \tag{4}$$

$$w_{ij} = \frac{Favours_{ji}}{Favours_{ij}}. \tag{5}$$

At this point, agents can decide whether to adopt this kind of social allocations or to be only rational as explained previously. They choose between them in accordance with their *Sociability* factor, which is the probability to select the social winner instead of the rational winner. *Sociability* can be adjusted in the range [0,1] to model intermediate behaviors between efficiency and total reciprocity. This can provide great flexibility when animating characters, since sociability can be dynamically changed, thus producing different behaviors depending on the world state.

5.3 Conversational library

The auction-based model presented above represents a useful technique to obtain group coordination through social commitments. Apart from the auction carried out internally, it is clear that

Table 1 Conversations to animate social agreements

Winner	Preconditions	Action	Response
	Performance winner		
j	None	None	None
	Near (*i*,Resource), not (near(*i*,*j*))	Shout (*j*,*i*,make(*t*))	Shout (*i*,*j*,no)
	Near (*i*,Resource), near (*i*,*j*)	Tell (*j*,*i*,make(*t*))	Tell (*i*,*j*,no)
i	Noise (high)	Approach (*i*), tell (*j*,*i*,make (*t*))	Tell (*i*,*j*,yes)
	Noise (low)	Shout (*j*,*i*,make(*t*))	Shout (*i*,*j*,yes)
	Social winner		
j	None	Plan_meeting (*j*,*i*), chat (*j*,*i*)	Chat (*i*,*j*)
i	Noise (high)	Approach (*i*), tell (*j*,*i*, make (*t*)),	Tell (*i*,*j*,yes)
		Plan_meeting (*j*,*i*), chat (*j*,*i*)	Chat (*i*,*j*)
	Noise (low)	Shout (*j*,*i*,make(*t*)),	Shout (*i*,*j*,yes)
		plan_meeting (*j*,*i*), chat(*j*,*i*)	Chat (*i*,*j*)

the agreements should be animated according to the situation being simulated. To manage the animation of social commitments, we have developed a set of plans that uses several conversations to display the agreement reached.

Table 1 summarizes the actor plans depending on the winner of an auction. On the upper half of the table, when a task (*t*) is auctioned, a bidder (*i*) will be the performance winner only if he can do it faster than the others. When an agent *i* wins an auction, the auctioneer (*j*) can approach *i* and animate the agreement starting a dialog. For instance, he can shout at the winner: '*Please, make a cup of coffee for me!*'; where a positive answer is necessary to be consistent with the agreement previously reached. When the auctioneer is also the winner, no social agreement is normally produced. However, these situations can be also useful to animate some failures in social commitments, which will add more variability to the character's behavior—for example, '*Sorry I can not do it now because I have a lot of work to do*'.

The lower half of the table shows the actions performed when the winner is a social winner, that is, an agent that obtains the best social reward in accordance with the utility values received from its friends. In this case, a conversation always occurs and the auctioneer can *approach and tell* or *shout at* its partner the commitment made—for example, '*Go to the counter, please, I want to chat with you!*'. Planning social meetings is a mechanism oriented to animate short chats between two actors; therefore, they will start chatting when they are close enough and a meeting was previously planned between them. In both cases, the animated actions have preconditions, such as the level of noise in the environment or the distance between actors or resources—*near*(*x*,*y*). The noise level can be easily derived from the whole number of actors in the bar and those who are near the winner.

6 Application example

In order to test the multi-agent simulation framework previously presented, we have created a virtual university bar where waiters take orders placed by customers (see Figure 6). The typical objects in a bar—such as a juice machine—behave like resources that have an associated time of use to supply their products—for example, 2 min to make an orange juice—and they can only be occupied by one agent at a time. We have used the ontologies introduced in Section 4 to properly classify the interactive objects present in the world. For instance, the class *ServiceContainer* has been used to model a coffee machine, a tray with donuts and also an ice container. Similarly, several objects are represented as instances of the class *HeterogeneousObjectContainer*, among them *shelf-B*, a shelf with alcoholic bottles on top. Other *HeterogeneousObjectContainers* such as the *refrigerator*, the *cabinet* or the *cupboard* have been configured with the *CLASSES-ONLY* filter type to only publish the classes of the objects inside while they are closed. A domain-specific

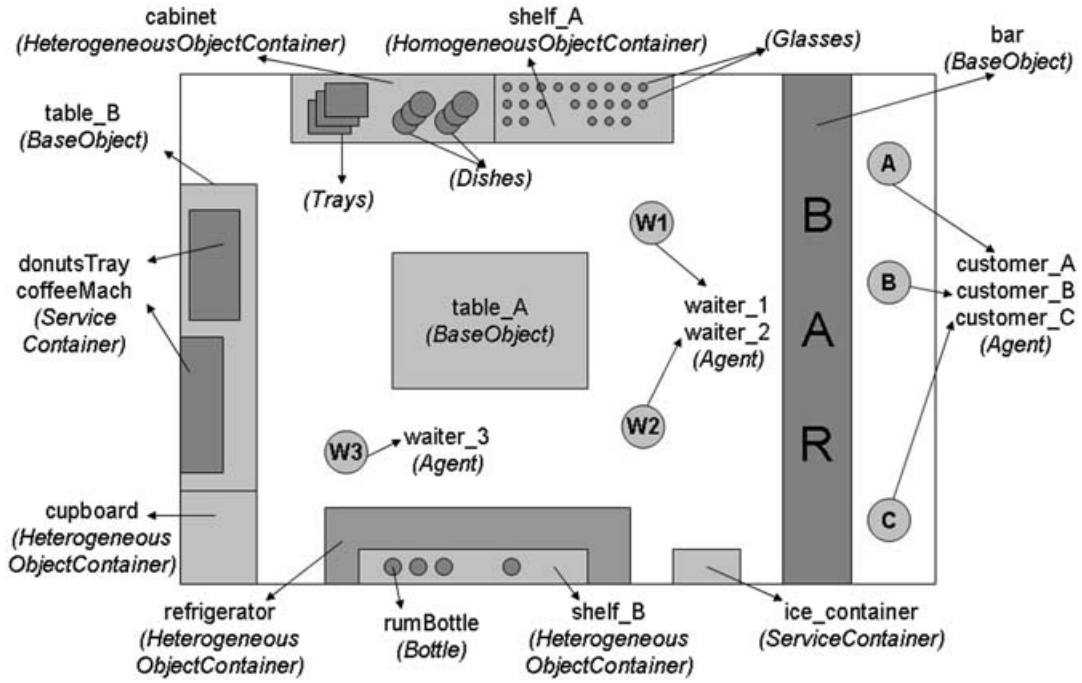

Figure 6 Virtual university bar environment

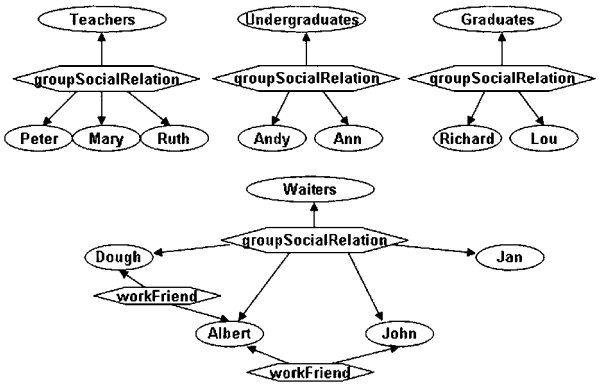

Figure 7 Social relations between agents in the virtual bar

ontology has been implemented that defines common objects in a bar such as *Bottles*, *Trays*, *Dishes* and so on.

Agents can be socially linked using the concepts defined in the ontology. According to them, all waiters are related through a *groupSocialRelation* to *Waiters*, a group representing their role (see Figure 7). Moreover, they can be individually related with other waiters through *workFriend*. This relation semantically means that the agents are workmates and, in this application, this relation has been modeled as bidirectional but not transitive. For example, in Figure 7, Albert is a friend of Dough and John but the latter are not friends. Moreover, we have also specified three possible groups of customers: teachers, undergraduates and graduates. The social network specified by them is used to promote social meetings among customers in the university bar.

The waiters are governed by the finite state machine[3] shown in Figure 8a, where orders are served basically in two steps: first, using the corresponding resource—for example, the grill to

[3] Specified by means of plans in Jason's extended version of AgentSpeak(L).

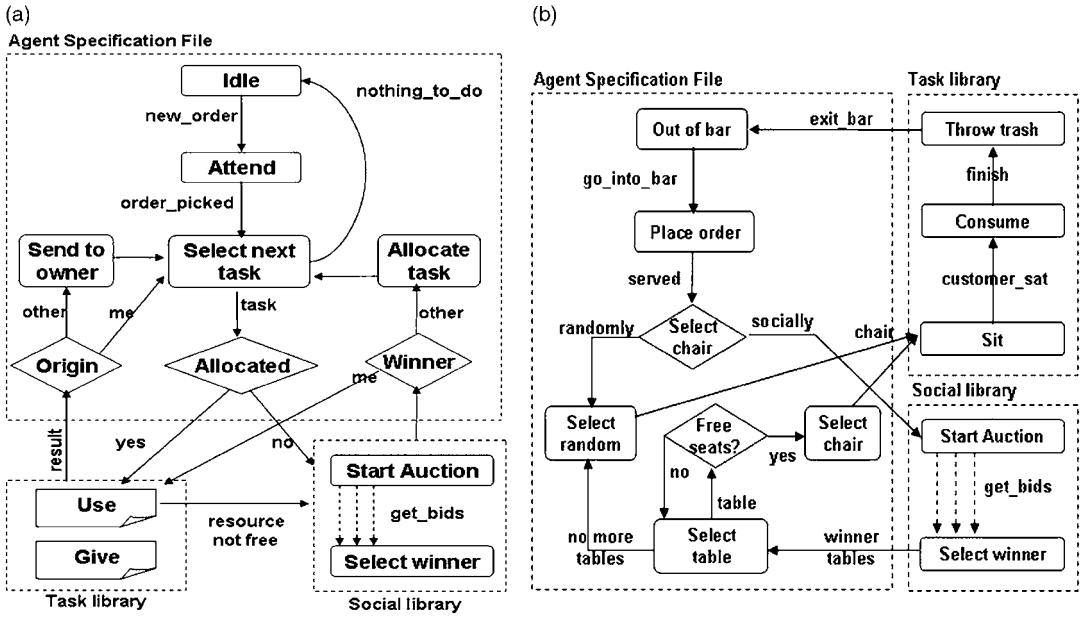

Figure 8 (a) Waiter specification and (b) customer specification

produce a sandwich—and second, giving the product to the customer. Tasks are always auctioned before their execution in order to find good social allocations. Equations (6) and (7) define the utility values returned by the performance utility function for these tasks. This function aims at maximizing the number of tasks being performed at the same time and represents the waiters' willingness to serve orders as fast as possible. Social behaviors defined for a waiter are oriented to animate chats among his friends at work. Therefore, waiters implement the internal and external social utility functions detailed in Equations (8) and (9), where *Near* computes the distance between the agents while they are executing a pair of tasks. These functions evaluate social interest as the chance to meet a *workFriend* in the near future—thus performing a planned meeting.

$$U^i_{\text{perf}}(\langle i \leftarrow' Use'\rangle) = \begin{cases} 1 & \text{if } [(i = Auctioneer) \wedge IsFree(Resource)] \vee \\ & [IsUsing(i, Resource) \wedge \text{not}(IsComplete(Resource))], \\ 0 & \text{Otherwise,} \end{cases} \tag{6}$$

$$U^i_{\text{perf}}(\langle i \leftarrow \text{'Give'}\rangle) = \begin{cases} 1 & \text{if } [(i = Auctioneer) \wedge nextAction = NULL] \vee \\ & [currentTask = \text{'Give'} \wedge not(handsBusy < 2)], \\ 0 & \text{Otherwise,} \end{cases} \tag{7}$$

$$U^i_{\text{int}}(\langle i \leftarrow t, j \leftarrow t_{\text{next}}\rangle) = \begin{cases} 1 & \text{if } IsFriend(i,j) \wedge Near(t, t_{\text{next}}) \wedge \\ & ExecTime(t_{\text{next}}) RemainTime(currentTask), \\ 0 & \text{Otherwise,} \end{cases} \tag{8}$$

$$U^i_{\text{ext}}(\langle j \leftarrow t\rangle) = \begin{cases} 1 & \text{if } IsFriend(i,j) \wedge Near(currentTask, t), \\ 0 & \text{Otherwise.} \end{cases} \tag{9}$$

On the other hand, customers place orders and consume them when served. At the moment, we are not interested in improving customer performance but in animating interactions between the members of a social group—that is teachers, undergraduates and graduates. The finite state machine in Figure 8b governs the actuation of customers that use auctions to solve the problem of *where to sit*. Depending on his or her sociability factor, a customer can randomly choose a chair or start an auction to decide where to sit and consume. This auction is received by all customers in the bar, which use the external social utility function defined in Equation (10) to promote social

meetings. This function uses the *groupSocialRelations* to determine whether two individuals belong to the same group. We define the performance and the internal social utility functions as 0 since task passing is not possible in this case—no one can sit instead of another customer. Finally, when a social meeting emerges, both waiters and customers use the plans in the *Conversational Library* to sequence the speech-acts needed to animate commitments, greetings or simple conversations:

$$U^i_{\text{ext}}(\langle j \leftarrow \text{'Sit'} \rangle) = \begin{cases} 1 & \text{if } \mathit{IsSameClass}(i,j) \wedge \mathit{IsConsuming}(i, \mathit{auctionedTable}), \\ 0 & \text{Otherwise.} \end{cases} \tag{10}$$

7 Results

In this section, we present some graphical and numerical results oriented to evaluate the simulations obtained with different configurations of our framework.

Firstly, the use of semantics has been tested during sensorization of complex scenes—such as the environment shown in Figure 6. For example, we have modeled a shelf with 20 clean glasses (*shelf-A*) as an instance of the class *HomegeneousObjectContainer*. A full representation of it would easily inundate any search state, besides, it would be unnecessary since all the glasses are similar. Therefore, our *Semantic Layer* summarizes the necessary information during the simulation. Although it informs about the total number of glasses on top, it only gives a full description of a reduced number of glasses. The scenario depicted in Figure 6 contains 68 interactive objects. Without any filtering, the amount of information necessary to describe—in propositional logics—the properties and inter-relations of all these objects would be up to 612 literals. However, the ontology-based filtering made by the *Semantic Layer* lowers this quantity to 252 literals only referred to 28 interactive objects, hence producing a reduction of the 60% in the information flow. Thus, we guarantee reasonable size states for the intelligent virtual agents involved.

Secondly, results on social behavioral animation—using friendship and group relations extracted from the ontology—are now explained. The *Sociability* factor is the main parameter of the social model presented here. It allows both customer and waiter agents to balance their efficiency and sociability. For example, the plots depicted in Figure 9 correspond to two simulations where two waiters that are friends—that is *workFriends*—take the orders placed by four customers in the virtual bar. In these simulations, all customers want to have a sandwich but the grill—which is needed to prepare them—is shared by the waiters. Although the grill cannot be used by more than one agent at a time, we have provided this object with the capability of making four sandwiches simultaneously. Therefore, waiters can coordinate to serve customers faster. In Figure 9a, waiters are defined with $\mathit{Sociability} = 0$—elitist agents—thus, coordination is purely based on performance. In this way, while *Waiter1* makes the sandwiches, *Waiter2* takes orders and serves the customers. That is, they work as efficiently as they can—maximum waiter coordination.

Nevertheless, an excess of elitism is not always common in human societies. The *Sociability* factor can be adjusted to incorporate social interactions in accordance with the social relations defined among the agents. For example, waiters in Figure 9b define $\mathit{Sociability} = 0.6$. Hence, *Waiter2* first decides to be efficient and he passes the action of making a sandwich to *Waiter1*—who was already using the grill to serve *Customer1*. However, when serving *Customer3*, *Waiter2* opts to chat with his *workFriend Waiter1*, while the latter finishes preparing his sandwiches. In this case, customers wait more time to be served but the waiter's behavior is enriched thanks to the variability derived from the new social situations that are being animated.

Figure 10 shows the 3D animation of the plot in Figure 9b. Here, waiters can chat while no customer is waiting to be served. Then, waiters serve customers in order of appearance using a standard dialog as seen in snapshot 10a. When a resource that is needed to perform a task—such as the grill to make a sandwich—is already in use, a waiter has two possibilities: (a) try to pass the task and continue serving; or (b) animate social chats—in our scenario, casual conversations—while waiting for the resource to be free. The decision is made probabilistically depending on the *Sociability* factor of the agent. For example, in snapshot 10b *Waiter2* notices that *Waiter1* is also

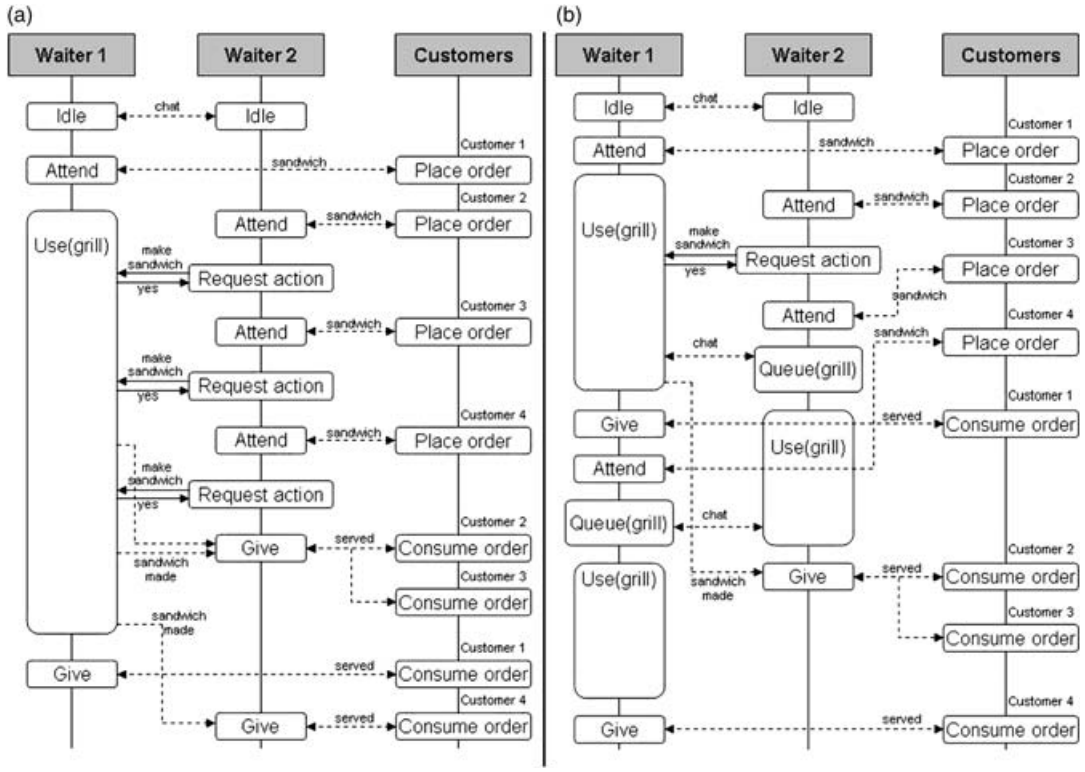

Figure 9 Examples of simulation plots: (a) maximum waiter coordination and (b) balancing efficiency and sociability

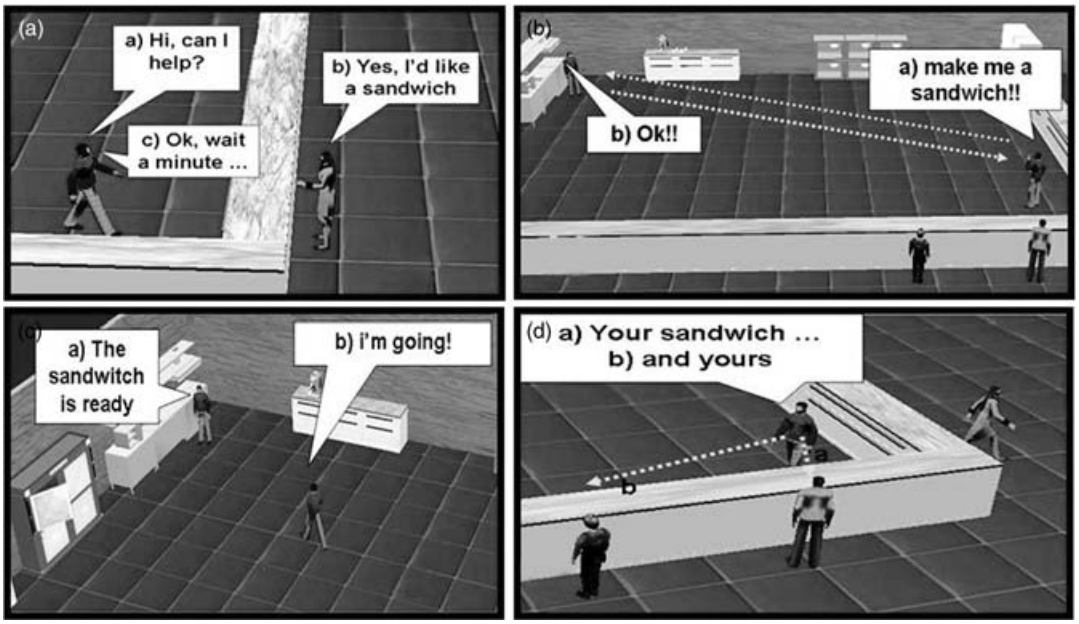

Figure 10 Animating interaction situations: (a) attend situation, (b) action request situation, (c) inform result situation and (d) serving situation

preparing a sandwich and asks him to make another sandwich. Once the reallocated task has been completed, the waiter that has performed it informs the applicant agent about its result. In snapshot 10c, *Waiter1* tells *Waiter2* that the sandwich is ready. Similar to using a resource for solving several tasks simultaneously, the waiters can use their two hands to carry more than one

product at the same time. As an example, *Waiter2* in snapshot 10d carries two sandwiches that are given to *Customer2* and *Customer3*.

The effects of sociability over customers are shown in Figure 11. Here, seven waiters serve 16 customers belonging to the three social groups previously defined: teachers, undergraduates and graduates. We use distinct avatars to distinguish the members of each social group. For this example, customers have a value of *Sociability* = 0.7; thus, once they are served, they usually try to sit with other customers of their same group. See how avatars of the same type sit at the same table whenever it is possible.

To estimate the effects of the social techniques being applied, we have simulated the virtual university bar example with up to 10 waiters serving 100 customers both with different sociability factors. We measure the efficiency of a group of waiters as the ratio between the optimal simulation time and the real simulation time (see Equation (11)). *Throughput* is an indicator in the range [0,1] that estimates how close a simulation is to the ideal situation in which the workload can be distributed among the agents and no collisions arise:

$$Throughput = \frac{T^*_{\text{sim}}}{T_{\text{sim}}} = \frac{N_{\text{tasks}} \times \overline{T_{\text{task}}}/N_{\text{agents}}}{T_{\text{sim}}}. \tag{11}$$

Figure 12a shows the *Throughput* obtained by different types of waiters versus self-interested agents—that is, agents with no social mechanisms included. Self-interested agents collide as they compete for the use of the shared resources and these collisions produce high waiting times as the

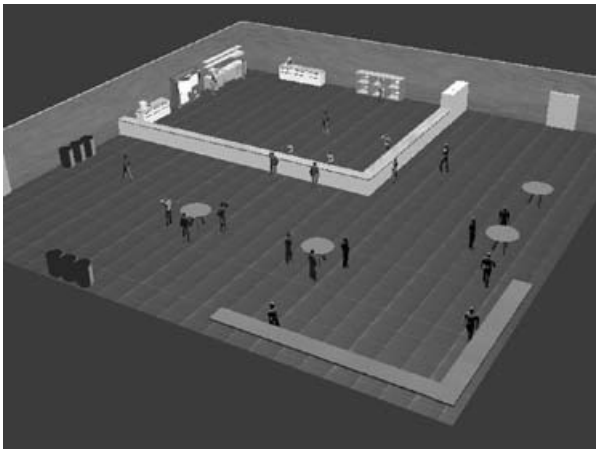

Figure 11 Customer sociability in the three-dimensional virtual university bar example

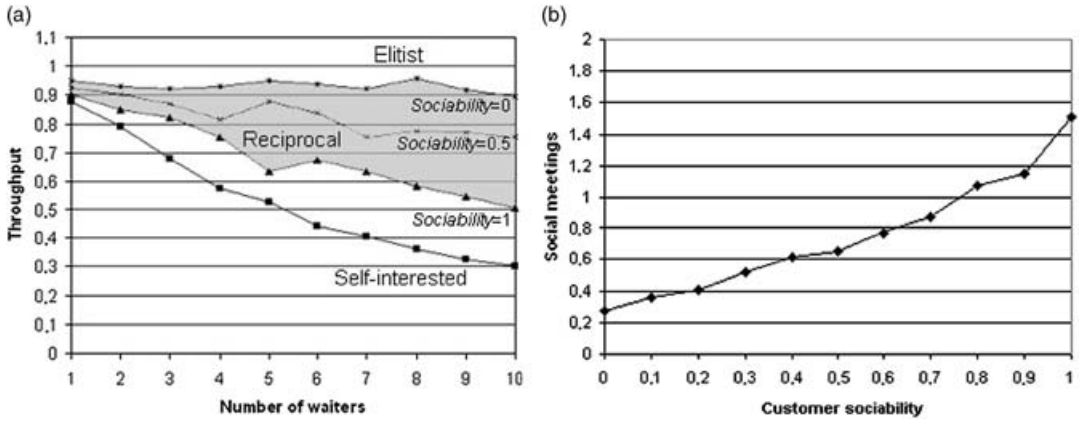

Figure 12 (a) Waiter *Throughput* and (b) number of customer social meetings

number of agents grows. We can enhance this low performance with efficiency-centered agents—*Sociability* = 0—which exchange tasks with others that can carry them out in parallel, thus reducing the waiting times for resources. Nevertheless, they produce robotic outcomes since they are continuously working if they have the chance, leaving aside their social relationships—in our example, chats between friends. The *Sociability* factor can be used to balance rationality and sociability. Therefore, the *Throughput* for the sort of animations we are pursuing should be placed somewhere in between elitist and fully reciprocal social agents—with *Sociability* = 1. On the other hand, Figure 12b demonstrates that the higher the *Sociability* factor is, the larger the number of social meetings that will be performed by the customers when they sit at a table.

Throughput is an estimator for the behavioral performance but, despite being a basic requirement when simulating groups of virtual characters, it is not the only criterion to evaluate when we try to create high-quality simulations. Therefore, we have defined another estimator that takes into account the amount of time that the designer of the simulation wants the agents to spend on their social interactions. According to this, we define the following simulation estimator:

$$Animation = \frac{T^{*}_{\text{sim}} + T_{\text{social}}}{T_{\text{sim}}}, \tag{12}$$

where T_{social} represents the time devoted to chatting and to animating social agreements among friends. In our virtual bar, we have chosen T_{social} as the 35% of T^{*}_{sim}. Figure 13 shows the animation values for 10 reciprocal social waiters with four degrees of friendship: all friends, 75% of the agents are friends, half of the agents are friends and only 25% of the agents are friends. As we have already mentioned, low values of *Sociability* produce low-quality simulations since the values obtained for the animation function are greater than the reference value (*Animation* = 1). On the other hand, high values of *Sociability* also lead to low-quality simulations, especially when the degree of friendship is high. In these cases, the number of social conversations being animated is too high to be realistic and animation is far from the reference value. The animation function can be used to extract the adequate range of values for the *Sociability* factor, depending on the situation being simulated. For example, in our virtual bar, we consider as good-quality animations those that fall inside ±10% of the reference value (see shaded zone in Figure 13). Hence, when all the waiters are friends, good animations emerge when *Sociability* ∈ [0.1, 0.3].

Finally, Table 2 compares the amount of time devoted to executing each type of task in executions with 10 elitist waiters (*Sociability* = 0) and 10 fully reciprocal social waiters (*Sociability* = 1). The irregular values in the columns T_{use} and T_{give} on the left-hand side of the table demonstrate how some agents have specialized in certain tasks. For instance, agents 2, 5, 9 and 10 spend most of their time giving products to the customers while agents 3 and 7 are mainly devoted

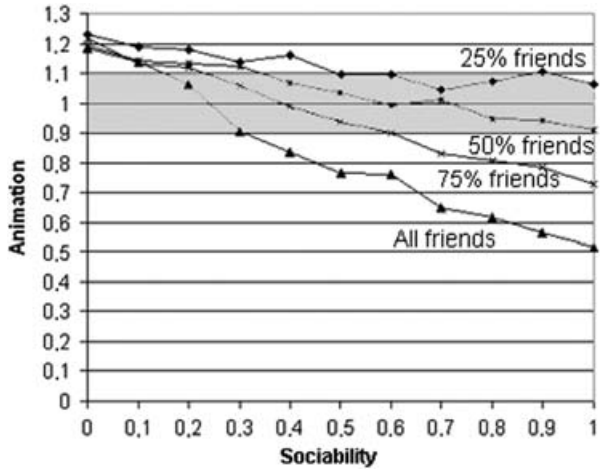

Figure 13 Animation results obtained for waiters

Table 2 Time distribution for reciprocal social waiters (time values are in seconds)

	Sociability = 0					*Sociability* = 1				
Agent	T_{wait}	T_{chat}	T_{use}	T_{give}	*Balance*	T_{wait}	T_{chat}	T_{use}	T_{give}	*Balance*
1	0	0	32	19	−6	16	36	69	34	−2
2	3	0	4	26	−3	18	62	58	24	−2
3	14	0	52	1	28	41	66	45	16	0
4	3	0	16	28	−3	48	61	60	27	3
5	0	0	7	30	−16	34	68	58	12	−1
6	3	0	37	17	−1	48	74	64	14	−2
7	0	0	67	4	21	18	66	48	24	1
8	0	0	45	17	1	33	76	45	24	4
9	7	0	5	23	−11	46	58	36	21	0
10	1	0	6	41	−10	27	69	56	20	−1

to using the resources of the bar—such as the coffee machine. Although specialization is a desirable outcome in many multi-agent systems, egalitarian human societies need also to balance the workload assigned to each agent. On the right-hand side of the table, fully reciprocal social waiters achieve equilibrium between the time they are giving products and the time they are using the resources of the environment (see columns T_{use} and T_{give}). Furthermore, the reciprocity factor balances the number of favors exchanged among the agents (see columns T_{use} and T_{give}). Furthermore, the reciprocity factor balances the number of favors exchanged among the agents (compare *Balance* columns). A collateral effect of this equilibrium is the increase in the waiting times, since social agents will sometimes prefer to meet his friends in a resource than to reallocate the task (compare columns T_{wait}). As a consequence, a new percentage of the execution time appears (T_{chat}) within which agents can animate pure social interactions—for example, chats between waiters that are friends.

8 Conclusions

We have presented a semantic-based framework for the simulation of groups of intelligent characters, a fast-developing area of research that integrates computer graphics and artificial intelligence solutions. Throughout the paper, we showed the importance of semantics when modeling dynamic virtual environments inhabited by groups of role-oriented agents. Beyond physical aspects, these roles are normally supported not only by the actions that the agents perform upon the objects in the environment but also by their interactions with other agents. Therefore, sociability issues should be included in the agent decision-making. According to this, we aim at incorporating human style social reasoning for the virtual humans simulated.

In the first part of the paper, after a short introduction to the field, we have highlighted the two major research points: semantic environments and behavioral animation of virtual humans. They jointly demonstrate the wide scope of problem tackled. The second part of the paper focuses on the multi-agent framework. The framework has two main parts. Firstly, the SVE, which incorporates core and specific ontologies and a semantic layer to sense and interact with the objects in the scene. Secondly, the agent architecture that is represented by the tasks, the social relations and the conversational skills that are handled by the agents. All these modules are explained in depth and applied to an application example: the virtual university bar. The third part of the paper concentrates on the results extracted from the example and it shows how semantics can help the agents to manage knowledge-rich environments, which are normally hard to sense for them. Furthermore, the object taxonomy extracted from the ontology allows the actors to reuse their operators in different contexts and to model social relations. The *Sociability* factor included in the agent architecture allows to easily regulate the creation of different types of social agents: from

elitist agents—which use their interactions to increase the global performance—to reciprocal agents—which can also interchange their tasks according to their sociability needs. Behavioral results are explained in a set of plots, where the paths executed by several waiters and customers are commented.

Finally, to estimate the level of cooperation achieved when regulating social and task-oriented behaviors, some informative metrics have also been included. Different simulations have been done with groups of up to 10 waiters serving 100 customers—also varying the sociability factor. The numerical results illustrate how social roles—from elitist to reciprocal—emerge, and how we can control the behavior to produce in the end higher quality social animations.

Acknowledgments

This work was jointly supported by the Spanish MEC and the European Commission FEDER funds under Grants Consolider Ingenio-2010 CSD2006-00046 and TIN2006-15516-C04-04.

References

Badawi, M. & Donikian, S. 2004. Autonomous agents interacting with their virtual environment through synoptic objects. In *Proceedings of the 17th Conference on Computer Animation and Social Agents (CASA2004)*.

Badler, N., Bindiganavale, R., Bourne, J., Palmer, M., Shi, J. & Schuler, W. 2000. *A Parametrized Action Representation for Virtual Human Agents*. MIT Press, 256–284.

Badler, N., Philips, C. & Webber, B. 1993. *Simulating Humans: Computer Graphics, Animation and Control*. Oxford University Press.

Bickmore, T. & Cassell, J. 2001. Relational agents: a model and implementation of building user trust. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI'2001)*. ACM.

Bordini, R. H., da Rocha, A. C., Hübner, J. F., Moreira, A. F., Okuyama, F. Y. & Vieira, R. 2005. MAS-SOC: a social simulation platform based on agent-oriented programming. *Journal of Artificial Societies and Social Simulation* **8**(3), 133–160.

Bordini, R. H. & Hübner, J. F. 2007. Jason, available at `http://jason.sourceforge.net/`.

Chang, P. H.-M., Chien, Y.-H., Kao, E. C.-C. & Soo, V.-W. 2005. A knowledge-based scenario framework to support intelligent planning characters. In *Proceedings of the 5th International Workshop of Intelligent Virtual Agents (IVA05)*, Lecture Notes in Artificial Intelligence **3661**, 134–145. Springer Verlag.

Charles, F., Lozano, M., Mead, S., Bisquerra, A. & Cavazza, M. 2003. Planning formalisms and authoring in interactive storytelling. In *First International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, Fraunhofer IRB Verlag, 216–225. Springer Verlag.

Ciger, J. 2005. *Collaboration with Agents in VR Environments*. PhD thesis, École Polytechnique Fédérale de Lausanne.

Conte, R. 1999. Artificial social intelligence: a necessity for agent systems' developments. *The Knowledge Engineering Review* **14**, 109–118.

Costa de Paiva, D., Vieira, R. & Musse, S. R. 2005. Ontology-based crowd simulation for normal life situations. In *Proceedings of the Computer Graphics*. IEEE.

Decker, K. S. & Lesser, V. R. 1997. Designing a family of coordination algorithms. *Readings in Agents*.

Doyle, P. 2002. Believability through context using "knowledge in the world" to create intelligent characters. In *First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS02)*. Academic Press, 342–349.

Farenc, N., Boulic, R. & Thalmann, D. 1999. An informed environment dedicated to the simulation of virtual humans in urban context. In *Computer Graphics Forum (Eurographics '99)*, **18**(3), Brunet, P. & Scopigno R. (eds.). The Eurographics Association and Blackwell Publishers, 309–318.

Giampapa, J. A. & Sycara, K. 2002. *Team-oriented Agent Coordination in the RETSINA Multi-agent System*. Technical report cmu-ri-tr-02-34, Robotics Institute, Carnegie Mellon University.

Gil, Y. 2005. Description logics and planning. *AI Magazine* **26**(2), 73–84.

Grimaldo, F., Barber, F. & Lozano, M. 2006. An ontology-based approach for IVE+VA. In *Proceedings of the IVEVA'06: Intelligent Virtual Environments and Virtual Agents*.

Grimaldo, F., Lozano, M., Barber, F. & Orduña, J. 2005. Integrating social skills in task-oriented 3D IVA. In *Proceedings of the IVA'05: International Conference on Intelligent Virtual Agents*, Lecture Notes in Artificial Intelligence. Springer Verlag.

Gutierrez, M. 2006. *Semantic Virtual Environments*. PhD thesis.

Harris, J. & Young, M. 2005. Proactive mediation in plan-based narrative environments, *IVA'05*.

Hogg, L. M. & Jennings, N. 2001. Socially intelligent reasoning for autonomous agents. *IEEE Transactions on Systems Man and Cybernetics* **31**(5), 381–393.

Iglesias, A. & Luengo, F. 2004. Intelligent agents in virtual worlds. *Third International Conference on CyberWorlds IEEE Computer Society*, 62–69.

Kallmann, M. & Thalmann, D. 1999. Direct 3D interaction with smart objects. In *VRST '99: Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. Academic Press, 124–130.

Kao, E. C.-C., Chang, P. H-M., Chien, Y.-H. & Soo, V.-W. 2005. Using ontology to establish social context and support social reasoning. In *Proceedings of the IVA'05: International Conference on Intelligent Virtual Agents*.

Kim, I.-C. 2003. KGBot: A BDI agent deploying within a complex 3D virtual environment. In *Intelligent Virtual Agents, Lecture Notes in Computer Science*, Rist, D. B. T., Aylett, R. & Rickel J. (eds.). Springer Verlag, 192–196.

Levison, L. 1996. *Connecting Planning and Acting via Object-specific Reasoning*. PhD thesis, University of Pennsylvania.

Lozano, M., Grimaldo, F. & Barber, F. 2004. Integrating minimin-HSP agents in a dynamic simulation framework. In *Third Helenic Conference on AI, SETN 2004*, Lecture Notes in Computer Science **3025**, 535–544. Springer Verlag.

Luck, M. & Aylett, R. 2000. Applying artificial intelligence to virtual reality: intelligent virtual environments. *Applied Artificial Intelligence* **14**(1), 3–32.

Meneguzzi, F., Zorzo, A. & Da Costa Móra, M. 2004. Propositional planning in BDI agents. In *SAC'04: Proceedings of the 2004 ACM Symposium on Applied Computing*. ACM Press, 58–63.

Molyneux, P. 2001. Black&white, Postmortem: Lionhead studio.

Otto, K. A. 2005. Towards semantic virtual environments. *Workshop Towards Semantic Virtual Environments (SVE'05)*.

Pellens, B., Bille, W., De Troyer, O. & Kleinermann, F. 2005. VR-wise: a conceptual modelling approach for virtual environments. *Methods and Tools for Virtual Reality (MeTo-VR 2005) Workshop*.

Piaget, J. 1995. *Sociological Studies*. Routledge.

Prada, R. & Paiva, A. 2005. Believable groups of synthetic characters. In *AAMAS '05: Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM Press, 37–43.

Rao, A. S. 1996. AgentSpeak(L): BDI agents speak out in a logical computable language. In *Proceedings of the MAAMAW'96*, No. 1038, Lecture Notes in Artificial Intelligence, 42–55. Springer Verlag.

Rao, A. S. & Georgeff, M. P. 1991. Modeling rational agents within a BDI-architecture. In *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*. Morgan Kaufmann Publishers Inc., 473–484.

Raupp, S. & Thalmann, D. 2001. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics* **7**(2), 152–164.

Ribeiro, M., da Rocha, A. C. & Bordini, R. H. 2003. A system of exchange values to support social interactions in artificial societies. In *AAMAS'03: Autonomous Agents and Multi-agent Systems*. ACM Press.

Schmitt, M. & Rist, T. 2003. Avatar arena: virtual group-dynamics in multicharacter negotiation scenarios. In *Proceedings of the IVA'03: International Conference on Intelligent Virtual Agents*, Lecture Notes in Artificial Intelligence. Springer Verlag.

Soto, M. & Allongue, S. 2002. Modeling methods for reusable and interoperable virtual entities in multimedia virtual worlds. *Multimedia Tools and Applications* **16**, 161–177.

Stanford Medical Informatics 2006. Protege, available at `http://protege.stanford.edu/`.

Thalmann, D. & Monzani, J. 2002. Behavioural animation of virtual humans: what kind of law and rules? In *Proceedings of Computer Animation*. IEEE Computer Society Press, 154–163.

Thomas, G. & Donikian, S. 2000. Virtual humans animation in informed urban environments. *Computer Animation*.

Tomlison, B. & Blumberg, B. 2002. Social synthetic characters. In *Proceedings of the Computer Graphics* **26**.

Tu, X. & Terzopoulos, D. 1994. Artificial fishes: physics, locomotion, perception, behavior, *SIGGRAPH*. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1994*. ACM, 43–50.

Viroli, M., Ricci, A. & Omicini, A. 2006. Operating instructions for intelligent agent coordination. *The Knowledge Engineering Review* **21**, 49–69.

Vosinakis, S. & Panayotopoulos, T. 2003. A task definition language for virtual agents. *Journal of WSCG* **11**, 512–519.

W3C 2004. OWL Web ontology language guide, available at `http://www.w3.org/TR/owl-guide/`.