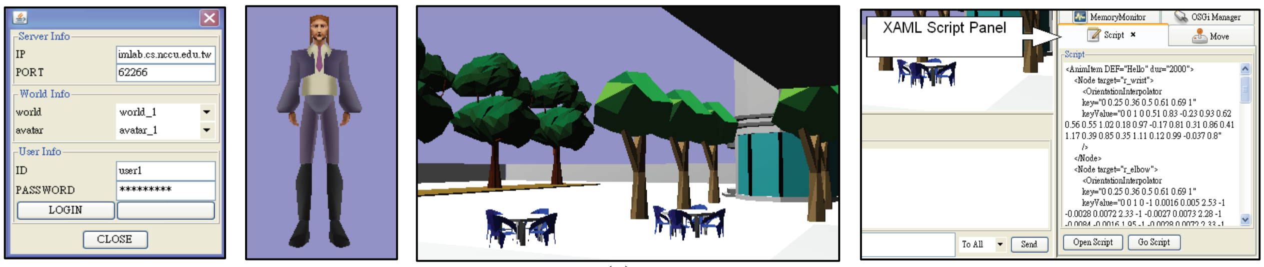
Appear in Proceedings of the 7th International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI), 2008

Using Pluggable Procedures and Ontology to Realize Semantic Virtual Environments 2.0

Yu-Lin Chu Tsai-Yen Li
Computer Science Department, National Chengchi University
{g9505, li}@cs.nccu.edu.tw



IP	imlab.cs.nccu.edu.tw
PORT	62266
world	world_1
avatar	avatar_1
ID	user1
PASSWORD	*********

```
<AnimItem DEF="Hello" dur="2000">
  <Node target="r_wrist">
    <OrientationInterpolator
    key="0 0.25 0.36 0.5 0.61 0.69 1"
    keyValue="0 0 1 0 0.51 0.83 -0.23 0.93 0.62
0.56 0.55 1.02 0.18 0.97 -0.17 0.81 0.31 0.86 0.41
1.17 0.39 0.85 0.35 1.11 0.12 0.99 -0.037 0.8"
    />
  </Node>
  <Node target="r_elbow">
    <OrientationInterpolator
    key="0 0.25 0.36 0.5 0.61 0.69 1"
    keyValue="0 0 1 0 -1 0.0016 0.005 2.53 -1
-0.0028 0.0072 2.33 -1 -0.0027 0.0073 2.28 -1
```

(a) (b) (c) (d)

Figure 1. Common components in a multi-user virtual environment: (a) login; (b) choose an avatar; (c) interact with virtual world; (d) a scripting interface (optional)

Abstract

Allowing users to design animation procedures and share their designs with other users is a crucial function for creating personalized 3D avatar behaviors on multi-user virtual environments (MUVE's). By describing the ontology of the virtual objects in the environment to animation procedures, we allow these procedures to create customized animations for an avatar to interact with the environment or other avatars. In this paper, we attempt to realize a semantic virtual environment (SVE) in our MUVE system (called IMNET) with the concept of web 2.0 through three mechanisms: dynamical installation of animation procedures, ontology design of semantic description for virtual objects and avatars, and interaction-oriented message delivery. Based on our demonstrative applications, we have designed example semantics for the objects in the virtual world by the use of ontology. We have used two application scenarios about automatic generation of navigation path and casual interactions between users to illustrate the potential of creating sharable rich contents in a semantic virtual environment.

CR Categories:

Keywords: Semantic Virtual Environment, Multi-user Virtual Environment, OSGi, Ontology for Virtual Environment

1 Introduction

In recent years, Multi-User Virtual Environment (MUVE) has attracted much attention due to the increasing number of users and potential applications. Figure 1 shows the common components that a MUVE system may provide. Generally speaking, a MUVE refers to a virtual world that allows multiple users to log in concurrently and interact with each other by texts or graphics provided by the system. On-line games can be considered as a special kind of virtual environment with specific characters, episode and ways of interactions. Other MUVE systems such as SecondLife[1] provide a general framework for users to design their own 3D contents and interact with other users through their avatars in a more general way. Although the users are allowed to build their own world, the animations that can be displayed are limited to those that have been prepared by the system. In addition, due to the lack of semantic information, it is also not feasible to design virtual avatars, controlled by the computer, to interact with other avatars.

Under the concept of web 2.0, we think future virtual environments will also depend on how easily the users can share their own designs of procedures for customized animations and high-level behaviors. However, it is a great challenge to design an extensible virtual environment system that allows the users to write their own customized procedures that can dynamically acquire the information of the virtual environment and other users. In our previous work, we have succeeded in extending a MUVE system developed by ourselves, called IMNET (Li et al., 2005), to allow user-defined animation procedures to be specified, downloaded, and executed on the fly (Chu et al., 2008). However, in order to enable these user-defined procedures to create richer animations for interactions, we must be able to describe the semantics of the objects in the world in a standard way accessible to all potential animation/behavior designers.

In this paper, we aim to use ontology to describe the semantics of the objects in the virtual environment such that users can design their own animation procedures based on the information. For examples, if we can acquire object information such as 3D geometry, height, and 2D approximation, we can design a motion planning procedure that can generate a collision-free path for the avatar to walk to a given destination. In addition, we also have designed the ontology specifically for information exchange between avatars. We also have added a new information query mechanism

[1] Second Life, http://secondlife.com

to facilitate the communication between avatars. These new functions will be demonstrated through several examples where the user-designed programs acquire application-specific semantics in the standard ontology format after the programs have been deployed dynamically to other clients' machines.

The remaining of the paper is organized as follows. In the next section, we will review the research related to our work. In the third section, we will describe the example design of ontology for the objects in the virtual worlds and for the avatars. In Section 4, we will describe the improved communication protocol allowing on-demand query of information among avatars. Then, in the following two sections, we will give examples of how to design animation components that can take advantage of the semantic information to generate richer user behaviors. Finally, we will conclude the paper with future directions.

2 Related Work

In this work, we aim to provide a smart virtual environment that can enable richer contents and interactions. Before the concept of semantic virtual environment emerges, there has been much research about how to integrate AI technologies into a virtual environment system. R. Aylett et al. (2001) found that this type of virtual environments have several common features. For example, these systems add components for solving problems such as configuration, scheduling and interaction. These systems constructed knowledge-level representation of the virtual scene and supported high-level processing with a natural language interface. These systems also used causal behaviors to replace the simulation of physical features.

In order to facilitate advanced applications that allow better communications between agents and the environment, the concept of semantic virtual environment was proposed (Otto, 2005; Kleinermann et al., 2007). Unlike traditional virtual environments that were designed mainly for visual effects, semantic virtual environments should contain richer structural semantic information that is adequate for a computer to process (Otto, 2005). For example, a plate on top of four sticks may be easily interpreted as a table by a human but it will be more difficult for a machine to infer that without low-level geometry reasoning. An analogy of this unstructured information for visual interpretation is the vast home pages on the web. This is also why semantic web was proposed to facilitate automatic processing and communication among web servers. Similarly, in order to facilitate the reuse of 3D design and animation procedures, it is crucial to annotate the objects in a virtual world with semantic information in a standard format such that the same contents can be reused in different worlds or in different MUVE systems.

The idea of SVE has been realized from various aspects by much work in the literature. For example, the SEVEN system was proposed by Otto (2005) as an example of realizing SVE with the concept of software components. It focuses more on the reusability of system components on different MUVE systems instead of components designed by the users. Gutierrez et al. (2005) also have proposed an ontology for virtual human by incorporating semantics into human shapes. They also regarded that the design of ontology is a continuous process where richer semantic information about human attributes should be added in a collaborative fashion. Instead of dealing with human shapes, the work in (Abaci et al., 2005) focuses on adding action semantics in smart objects to facilitate the interaction between the avatars and with the objects in the virtual environment. Garcia-Rojas et al. (2006) also proposed to add semantic information, such as emotion and expressiveness, to animations in order to facilitate the selection of appropriate animation clips for a specific scenario.

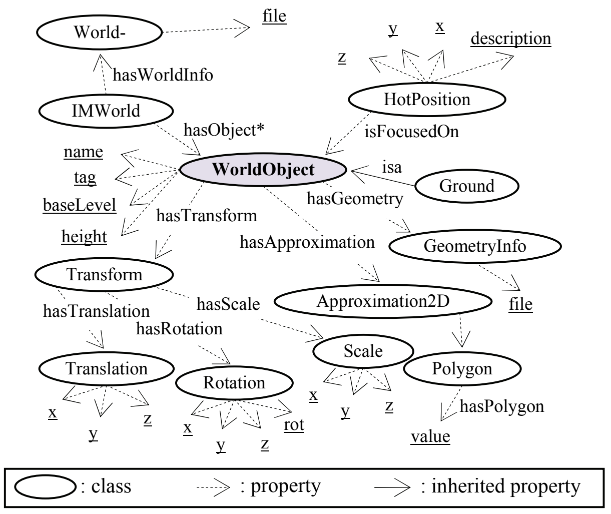


Figure 2. Ontology design for virtual world

3 Design of Ontology in IMNET

For developing and sharing animation procedures in the IMNET system, in this section we will describe our ontology design for virtual objects and avatars in IMNET. The topology described in this section is to serve the demonstrative purpose of potential applications; therefore, the design is by no means complete.

3.1 Ontology design of virtual environment

The objective of ontology design for virtual environment in this work is two-fold. First, we would like to keep the information that exists in the original IMNET such as object geometry and transformation. Second, we hope to use an example to show that more semantic information about the virtual objects can facilitate the computation of advanced reasoning procedures such as a path planner that may be designed by the users.

Our ontology design of the virtual environment is shown in Figure 2. The root of the world document is the `IMWorld` node, which contains world information (`WorldInfo`) and all the virtual objects (`WorldObject`) in the world. In order to retain the semantic information of the virtual objects existing in the original IMNET, we have designed the `GeometryInfo` and `Transform` nodes. Each object also has some additional attributes such as `name`, `tag`, `baseLevel`, and `height`. The `tag` attribute is designed for the user to denote application specific properties for virtual objects. For example, in the example of path planning, one can tag certain objects as `sidewalk` and `crosswalk` such that these regions can be treated appropriately by the path planner according to their meanings in the world. Each object may also have the attribute of `Approximation2D`, which is a polygon that can be used to define 2D approximation of obstacles in the environment for the path planner. In addition, if 2D approximation is over simplified for the application, one can also use the `baseLevel` and `height` attributes to define 3D approximation regions where the obstacles are located. If these attributes are not available for some objects, they still can be computed from the given 3D ge-

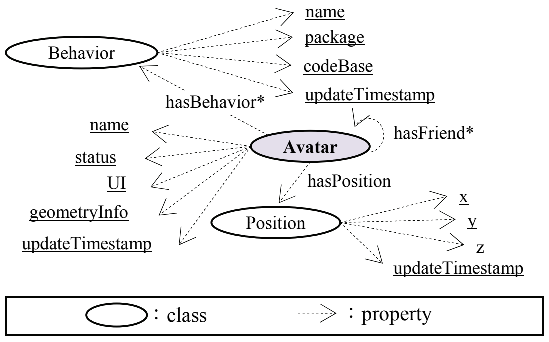


Figure 3. Ontology design for avatars

ometry and transformation of the objects. Some objects may also serve as the ground of the world by the node of `Ground` to define the boundary of the world. In addition, some objects could also be treated as `HotPosition` when they are the foci of interest in the application.

3.2 Ontology design for avatars

In MUVE's, an avatar could be controlled by a real user or by a computer program (called *virtual user*) if the system provides such a function. Virtual users can be used by the designer of the virtual world to perform simple tasks such as a watching a gate or offering guided tours. In this section, we describe the basic ontology classes and attributes (as shown in Figure 3) that we have designed for the applications of the avatar interactions. Although an avatar is also an object in a virtual world, they have more active and complicated roles to play. For example, a user may choose to use his/her own animation for a specific behavior by using the `hasBehavior` property to connect to the `Behavior` class. This `Behavior` class defines the procedure (with `name`, `package`, and `codebase`) for generating the desired animation. In addition, an avatar may contain some basic attributes such as `name`, `geometryInfo`, and `status`. We also use the `hasFriend` and `hasPosition` properties to get the friendship and current position information of avatars.

3.3 Using ontology to load the virtual world

In the two subsections above, we have defined an ontology for the objects and avatars in the virtual world. However, in the original IMNET system, the geometry of the virtual world is loaded from a single VRML file. The geometry is parsed and converted into the underlying format for 3D display by a module called VRMLModelFactory as shown in Figure 4. In order to augment the system with semantic information, we have split the geometry into several VRML files with one file for each object. This file is specified in the `geometryInfo` attribute of every `worldObject`. We have adopted the Web Ontology Language (OWL) established by W3C as the file format for the ontology of the virtual world. As shown in Figure 5, the system first loads and parses the OWL file into an object format through the automatic generated Java class and the Protégé API. The geometry file for each object is then retrieved from the ontology and loaded into the system by the VRMLModelFactory module.

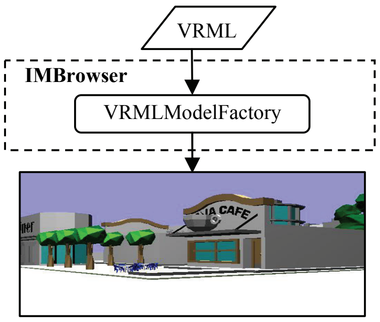


Figure 4. Processing a single VRML file to generate the virtual world

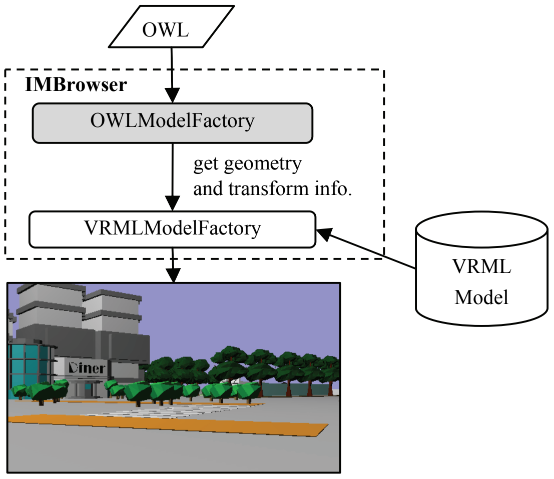


Figure 5. Processing an OWL file and loading multiple VRML files to generate the virtual world

```
<IMNET from="user1" to="user2">
  <Chat>…</Chat>
</IMNET>
```

Figure 6. An example of IMNET message

4 Communication Protocol for Information Query

In a semantic virtual environment, we think semantic information should not only be used by internal modules, but should also be accessible to other clients through user-defined pluggable modules. In the previous section, we have described the ontology of the world and how it is loaded into the IMNET system. However, the clients are not required to specify all attributes defined in the ontology of the avatar. In addition, not all information described in the ontology will be broadcast to all clients. Therefore, we need to have a flexible way for the avatars to communicate semantic information with each other. In this subsection, we will describe how we modify the current communication protocol of IMNET to take information query into account.

The application protocol in the original IMNET is similar to other MUVE's that only encapsulates pre-defined message types in the XML format. The underlying animation scripting language, XAML, is an example of message type (Li et al., 2004). Another example is the message for textual information used in the chat module. For instance, in Figure 6, we show an example where

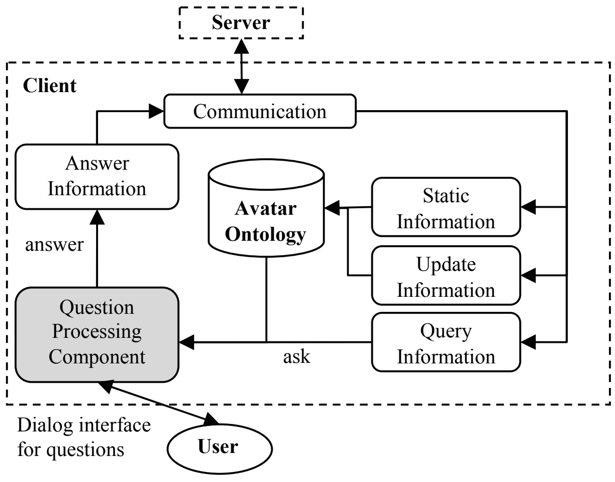


Figure 7. Client architecture for the processing of three types of information

```
<Info from="user1" to="user2" timestamp="1215476323">
  <queryInfo ask="make friend"/>
</Info>
```

Figure 8. An example of query message

```
<Info from="user2" to="user1" timestamp="1215476330">
  <queryInfo askId="1215476323"  answer="yes"/>
</Info>
```

Figure 9. An example of responding message

User1 wants to send a <Chat> message to user2. However, in the original design there is no way for the clients to query the information of other avatars that may be defined by the avatar designers instead of the system. This function is crucial for the avatars to exchange information for richer interactions in a semantic virtual environment.

In the work, we have enhanced the communication protocol of IMNET to incorporate a broader range of message types. We distinguish three types of information exchange between avatars as shown in Figure 7. The first one is the *static* information, such as the id and name properties that are delivered only once at the beginning when a user logs in. The second type is the *update* information, such as the position of the avatar, which is voluntarily pushed to all clients in a more frequent way. The third type is the *query* information, such as optional attributes or questions, which is sent to the inquirer only upon requests. We have integrated these three types of messages with the tag <Info> and distinguish their types in an internal tag. In particular, we have designed a mechanism for query-response processing as shown in Figure 7. For example, in Figure 8, we show a scenario where user1 asks user2 if user2 wants to be a friend of user1. The message is sent with a <queryInfo> internal tag and with a timestamp property. This timestamp property can be used as the id of the query. The query processing component in Figure 7 may prompt user2 (if user2 is a real user) with a dialog box or evoke an auto-responding component (if user2 is a virtual user) for a response. Then he/she can use this id to indicate this query (askId) in his/her reply message to user1 as shown in Figure 9.

```
<MoPlan package='imlab.osgi.bundle.interfaces'
  codebase='http://imlab.cs.nccu.edu.tw/plan.jar'>
  <param name="-s" value="1.1  2.3"/>
  <param name="-g" value="5.2  3.8"/>
</MoPlan>
```

Figure 10. An example of specifying a motion planning problem in a user-defined component, MoPlan.

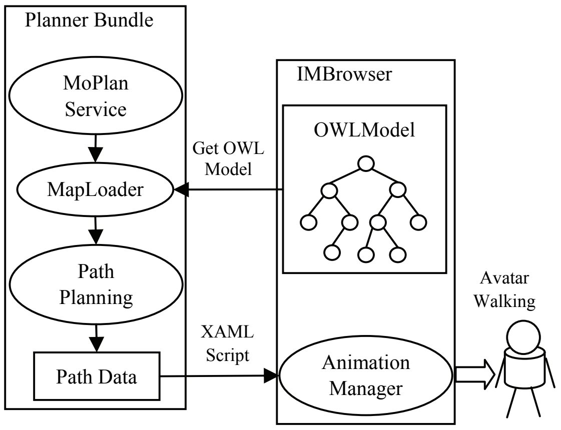


Figure 11. The process of how the motion planning component generates animations

5 Demonstrative Examples

In this section, we will give two examples of using semantic information in the virtual world to enhance the functions and behaviors of the avatars.

5.1 Example 1: motion planning for avatars

A common way for a user to navigate in a virtual environment is by controlling his/her avatar by input devices such as keyboard or mouse. However, it is a low-level task that may not be easy for a novice user to control his/her avatar to reach a destination. There has been much research that proposed to use motion planning techniques to generate collision-free navigation paths for the avatar to follow (Salomon et al., 2003). However, in order to define a motion planning problem, we need to obtain the geometric information of the objects in the environment such that we know where the boundary of the world is and which of the objects needs to be treated as an obstacle.

In this subsection, we will use a motion planning component as an example to illustrate how a user-defined animation procedure can be installed dynamically and retrieve necessary world information for the needs of the application. A user first prepares the procedure as a software bundle according to the specification of OSGi[2]. Then he/she can use a XAML script, such as the one shown in Figure 10, to indicate where he/she wants to move to. In this script, he/she needs to specify the name of the package, the initial (optional) and goal locations, and the URL for downloading the bundle if it is not already installed. In this case, the bundle will be installed dynamically and evoked through the OSGi mechanism (Chu et al., 2008).

[2] OSGi Alliance, http://www.osgi.org/

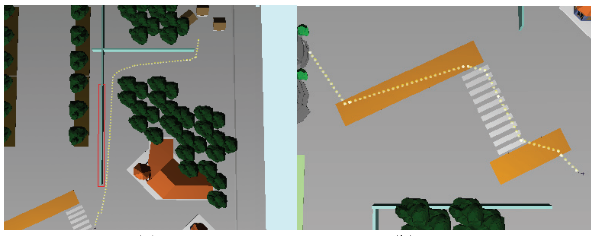

(a) (b)

Figure 12. The example paths generated by the path planner: (a) avoiding the obstacles described by the 2D approximation in semantics; (b) the path generated by taking crosswalk and sidewalk into account.

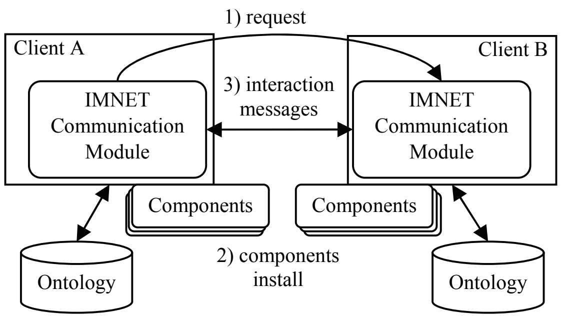


Figure 13. The steps for initiating an interaction between avatars

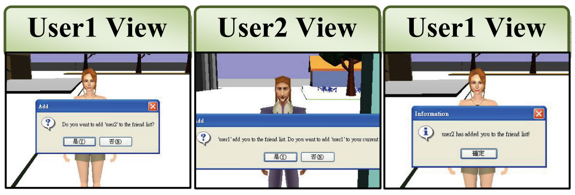
In order to generate a collision-free path, a motion planner needs to acquire obstacle information from the world. In our system, the motion planning component obtains this semantic information through the ontology of the virtual world defined in Section 3. According to the obtained obstacle information, the planner first converts the obstacle information into a 2D bitmap and then computes a potential field that is commonly used in a motion planner. And then the planner performs a best-first search to find a feasible path to the goal according to the potential field. Finally, the planner component translates the path to a XAML script and assigns it to the avatar to generate the walking animation as depicted in Figure 11.

Obstacle information can not only be inferred from low-level geometry but also be given as approximation by the scene designer. In Section 3, we have designed an optional attribute called Approximation2D in the ontology of a virtual object. In Figure 12(a), we show an example of the collision-free path generated by the planner by the use of the 2D approximation of the objects in the world. If the planner can find this 2D approximation for an object, it will use it to build the 2D bitmap needed in the planner. If not, it still can build the convex hull of the 3D geometry and project it into the ground to form a 2D approximation. In other words, semantic information could be designed to facilitate automatic reasoning but it is not mandatory. The designers of virtual objects are not obligated to define all attributes in an ontology that could be large in collaborative creation. In addition, the user-defined animation procedures do not easily break down in such a loosely coupled distributed environment either since they can take this into account in the design stage.

However, some semantic information cannot be inferred directly from geometry. For example, in the virtual environment, there could be some crosswalk or sidewalk regions that need to be used whenever possible. One can tell this kind of objects from their appearance but it would be difficult for the machine to infer their functions through geometry. In this case, the planner has to acquire this information through the semantics defined in the ontology of the virtual world. In the example shown in Figure 12(b), the planner knows where the sidewalk and crosswalk through object tagging in the ontology and makes the regions occupied by these objects a higher priority when planning the path for the avatar. The potential values in these regions are lowered to increase the priority during the search for a feasible path. Consequently, a path passing through these regions was generated in the example shown in Figure 12(b). In addition, according to this semantic information, appropriate animations, such as looking around before moving onto the crosswalk region, could be inserted into the motion sequence of walking to the goal.

```
<SocialService package='imlab.osgi.bundle.interfaces'
   codebase='http://imlab.cs.nccu.edu.tw/social.jar'/>
```

Figure 14. Starting the mechanism of the interaction between avatars through XML script



Add

Do you want to add 'user2' to the friend list?

是(Y) 否(N)

'user1' add you to the friend list. Do you want to add 'user1' to your current

是(Y) 否(N)

Information

user2 has added you to the friend list!

確定

(a) (b) (c)

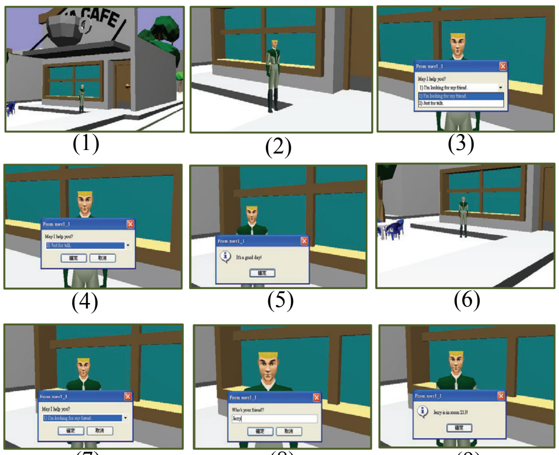
Figure 15. Interaction between the real users

5.2 Example 2: interaction between avatars

An objective of this work is to allow different animation components owned by different clients to interact with each other. The users can communicate through customized tags to acquire the avatar ontology of each other and use this information to perform specific interactions. The user behind an avatar could actually be a real user or a virtual user controlled by a computer program. In this subsection, we will use two scenarios to illustrate these two types of interactions. The first scenario is to demonstrate the interaction between two real users, and the second scenario is for the interaction between a real user and a virtual user.

To facilitate the interaction between the avatars, we have designed a component called SocialService. There are three steps for initiating an interaction between avatars as shown in Figure 13. A user who would like to initiate the interaction first sends a customized XAML script shown in Figure 14 to the other avatar (step 1) for it to install this social interaction component (step 2). Once the component has been installed, interaction queries related to social activities can be delivered through the communication protocol described in Section 4 and processed by the SocialService component (step 3).

In the first scenario, both users are real users. First, user1 would like to invite user2 to be his friend (Figure 15(a)). Therefore, a query message: "'User1' added you to his friend list. Do you want to invite 'user1' to be your friend as well?" appeared in user2's interface (Figure 15(b)). If user2 choose 'yes', user1 would be added into her friend list and a confirmation message would be



From user1_1
May I help you?
1) I'm looking for my friend.
2) Just for talk.
It's a good day!
Who's your friend?
Jerry
Jerry is in room 213!

Figure 16. Interaction between a real users and a virtual user (doorkeeper)

sent back to user1 (Figure 15(c)). Through the interaction between the two real users, the friend information was updated into the ontology of both avatars.

In the second scenario, user1 arranged a virtual user called doorkeeper to watch the door and provide information to potential guests (Figure 16(1~2)). When user2 entered a designated region, the doorkeeper would turn to face user2 and ask: "May I help you?" At the first encounter, user2 just entered this area by accident and therefore chose the answer: "Just look around." The doorkeeper replied: "Have a good day!" (Figure 16(3~5)) The state of the doorkeeper in this interaction was then set to FINISH. After user2 left the area, the state was restored to IDLE (Figure 16(6)). Assume that after some period of time, user2 approached the doorkeeper again for the second time. This time user2 chose: "I'm looking for my friend." The doorkeeper replied: "Who's your friend?" Then user2 answered: "Jerry." At this moment, the doorkeeper queried the avatar ontology of user1 (named Jerry) to see if user2 is in his friend list. If so, the doorkeeper would inform user2 the current position of user1. Otherwise, the doorkeeper would answer: "Sorry, Jerry does not seem to know you." If there is no such a user called Jerry, the doorkeeper would answer: "Jerry is not in this world." (Figure 16(7~9))

6 Conclusions and Future Work

Sharing of user-designed software components is a key function for enabling richer contents and behaviors in the future development of MUVE systems. In our previous work, we have designed a mechanism under OSGi to facilitate dynamic installation of user-designed software components in IMNET. In this work, we have extended the MUVE system to allow the semantics of the objects and avatars in the virtual environment to be described in the form of ontology. This provides a standard way for the software components to acquire semantic information of the world for further reasoning. We have used two types of examples: path planning and social interaction, to show how users can design their own code to facilitate richer or autonomous behaviors for their avatars (possibly virtual). We hope that these examples will shed some lights on the further development of object ontology and more sophisticated applications.

7 Acknowledgement

This research was funded in part by National Science Council under contract no. NSC96-2221-E-004-008.

References

ABACI, T. C'IGER, J. AND THALMANN, D. 2005. Action semantics in Smart Objects. In Proc. of Workshop towards Semantic Virtual Environments.

AYLETT, R. AND CAVAZZA, M. 2001. Intelligent Virtual Environments - A State-of-the-art Report. In Proc. of Eurographics.

CHU, Y.L. LI, T.Y. AND CHEN, C.C. 2008. User Pluggable Animation Components in Multi-user Virtual Environment. In Proc. of the Intl. Conf. on Intelligent Virtual Environments and Virtual Agents, China.

GARCIA-ROJAS, A. VEXO, F. THALMANN, D. RAOU-ZAIOU, A. KARPOUZIS, K. AND KOLLIAS, S. 2006. Emotional Body Expression Parameters In Virtual Human Ontology. In Proc. of the 1st Intl. Workshop on Shapes and Semantics, pp. 63-70, Matsushima, Japan.

GUTI'ERREZ, M. GARCÍA-ROJAS, A. THALMANN, D. VEXO, F. MOCCOZET, L. MAGNENAT-THALMANN, N. MORTARA, M. SPAGNUOLO, M. 2005. An Ontology of Virtual Humans: incorporating semantics into human shapes. In Proc. of the Workshop towards Semantic Virtual Environments.

KLEINERMANN, F. TROYER, O.D. CREELLE, C. AND PELLENS, B. 2007. Adding Semantic Annotations, Navigation paths and Tour Guides to Existing Virtual Environments. In Proc. of the 13th Intl. Conf. on Virtual Systems and Multimedia (VSMM'07), Brisbane, Australia.

LI, T.Y. LIAO, M.Y. AND LIAO, J.F. 2004. An Extensible Scripting Language for Interactive Animation in a Speech-Enabled Virtual Environment. In Proc. of the IEEE Intl. Conf. on Multimedia and Expo (ICME2004), Taipei, Taiwan.

LI, T.Y. LIAO, M.Y. AND TAO, P.C. 2005. IMNET: An Experimental Testbed for Extensible Multi-user Virtual Environment Systems. In Proc. of the Intl. Conf. on Computational Science and its Applications, LNCS 3480, O. Gervasi et al. (Eds.), Springer-Verlag Berlin Heidelberg, pp. 957-966.

OTTO, K.A. 2005. The Semantics of Multi-user Virtual Environments. In Proc. of the Workshop towards Semantic Virtual Environments.

SALOMON, B., GARBER, M., LIN, M. C., MANOCHA, D. 2003. Interactive Navigation in Complex Environment Using Path Planning. In Proc. of the 2003 Symposium on Interactive 3D graphics.