A Basic Semantic Common Level for Virtual Environments

Jesús Ibáñez Martínez and Carlos Delgado Mata

***Abstract*—This paper describes our proposal for a new representation model for virtual worlds. Our model provides a layer of semantic information for virtual environments that does not intend to substitute application dependant approaches, but to constitute a common lower level for all of them. Our model is simple in that it consists of a reduced number of different elements and most of them can be automatically annotated. The paper shows the ideal annotation process. In this sense, we show our own experiences in annotating worlds. The model has already been successfully used in two different applications.**

***Index Terms*—Virtual representation models, semantic virtual environments, and intelligent virtual environments.**

I. INTRODUCTION

Current virtual worlds representation models describe the worlds in such a manner that browsers can efficiently visualize the geometry of the worlds and, in some cases, can support low level interactivity. There is a gap between this low level representation of the worlds and the way we conceptualize them (and therefore the way we think and talk about them). Thus, a high level representation model (including semantics descriptions of the objects in the worlds) is desirable in order to support much richer user interactions at a more abstract level (for example querying for contents) and to deploy agents reasoning about the environments they inhabit.

We think the semantic level of information in virtual worlds must be seen as a component of the worlds. Therefore, the construction of virtual worlds must include semantic annotation of the environment, apart from the classical graphic edition.

There is no standard (not even a de facto standard) to model the semantics of virtual worlds. As a result, some groups define their particular way to model it (see [1] for a survey). However, usually these approaches are application dependant.

In this paper we describe a simple not application-dependant semantic common level for virtual environments. This level is constructed by adding meta-contents to the information usually describing the worlds If we want our model to be useful, it should be employed by the world creators. In turn, if we want the model to be used by world creators (that is, if we want them to annotate the worlds with the necessary meta-contents), the model should not require a great effort from them. Obviously, the less features to annotate, the less effort from annotators. Another way to reduce the effort is by using meta-contents that can be automatically annotated. In this sense, our model is simple. It consists of a reduced number of different features, and the majority of them can be automatically annotated.

In short, our proposed meta-contents model every virtual object as uniquely identified on the Internet, corresponding to a particular object type according to a concrete ontology and geometrically viewed as the minimum box able to contain it. Furthermore, they also establish spatial containment relations among objects and they specify a navigation network.

The model has already been successfully utilized in two different kinds of applications:

1) the construction of query solvers that allow users to search for particular objects or scenes in the worlds they visit [2]
2) the construction of a virtual guide which is aware of the world they inhabit [3]

The structure of the paper begins with an overview of the representation model. Following this, we present both the meta-contents which should be manually annotated and the meta-contents which can be automatically extracted. We also detail the annotation process, including our experiences annotating virtual worlds. We then outline the conclusions.

II. REPRESENTATION MODEL OVERVIEW

Fig. 1 shows how the proposed representation model is intended to be used. The usual 3D model (geometry, textures, etc.) of a particular world is extended with meta-contents which compose a layer of semantic information. Users can still directly interact with the 3D environment by using an appropriate 3D browser. However, they can also interact with agents which, in turn, can access the information contained on the semantic level. Thus, for example, a user could query an agent for a particular object in the world. The agent could solve the query by employing the information on the semantic level. Another agent could autonomously inhabit the world navigating it by using the semantic information.

Manuscript Received on September 27, 2006

Jesús Ibáñez Martínez, Departamento de Tecnología, Universidad Pompeu Fabra, Barcelona, Spain, jesus.ibanez@upf.edu .

Carlos Delgado Mata, CINAVI, Universidad Bonaterra, Aguascalientes, Mexico, cdelgado@bonaterra.edu.mx .

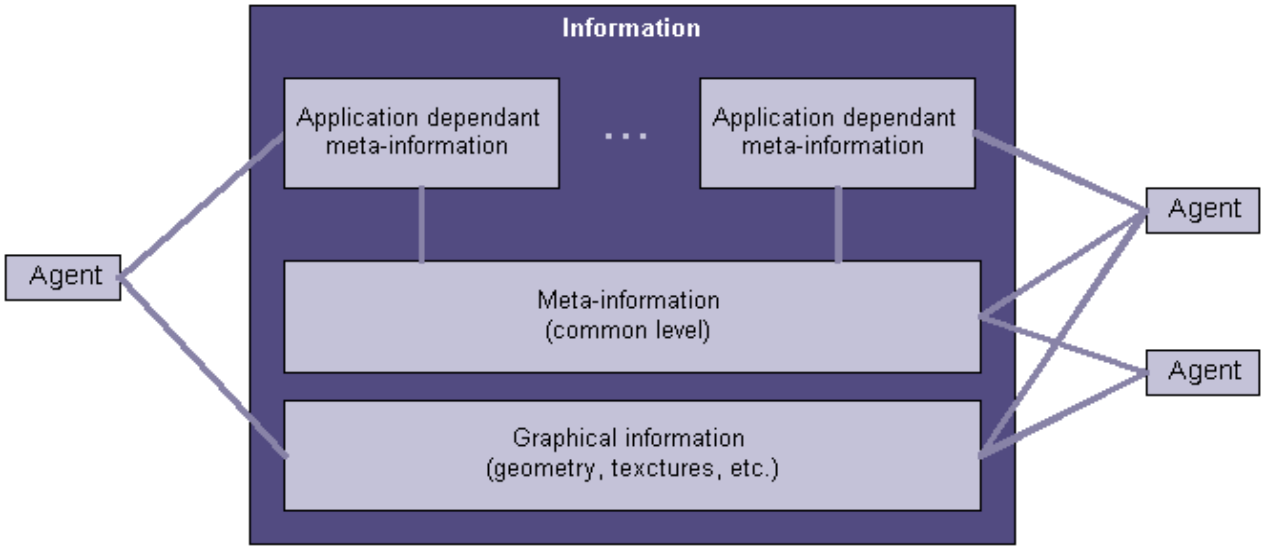

Fig. 1. Diagram showing how the meta information is used.

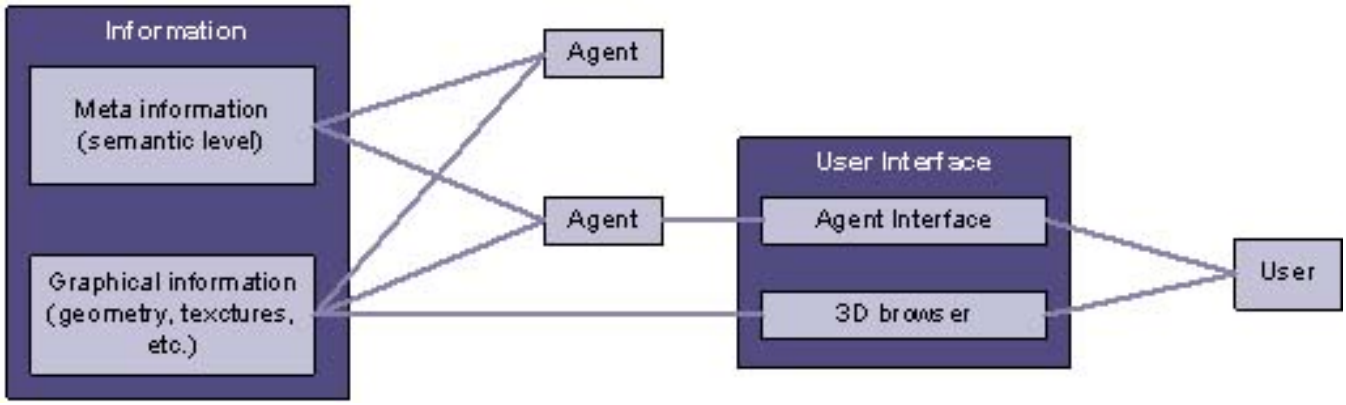

Fig. 2. Levels of meta information.

As there is no standard to model the semantics of virtual worlds, some groups define its particular way to model it. Usually these approaches are application dependant. We think application approaches are necessary, and we think they are different from one another in nature. Thus, the model we propose is situated at a lower level than application dependant approaches. Our model does not intend to substitute application dependant approaches, but to constitute a common lower level for all of them (see Fig. 2). This is demonstrated through the systems we describe in [2] and [3]. Both of them define an application dependant level on top of the common level we describe in this paper.

The representation model we propose is composed of ten elements: *object type*, *navigation network*, *object identifier*, *location,orientation*, *width*, *height*, *depth*, *spatial containment relations*, and *minimal paths*. These elements can be classified according to different criteria. In particular, we consider two classifications:

1) elements which can be automatically generated (usually quantitative and objective) versus elements which should be manually annotated (usually qualitative and subjective).
2) elements which are related to a particular object versus elements which are not related to a particular object.

Fig. 3 shows how elements are classified according to both of these criteria. The next sections consider the elements classified according to the former criteria and describe each one of the elements in detail.

III. META-CONTENES WHICH SHOULD BE MANUALLY ANNOTATED

The meta-contents we propose to be manually annotated are: *object type*, and *navigation network*. The first one is related to every object in the world being considered. The last one provides information in order to ease the navigation process in virtual worlds.

A Object Type

Every virtual object is annotated with its object type according to a particular ontology. That is, every object is annotated with both its type and a reference to the ontology employed to annotate it. Although there are other frameworks for structuring information (such as taxonomies and thesaurus),

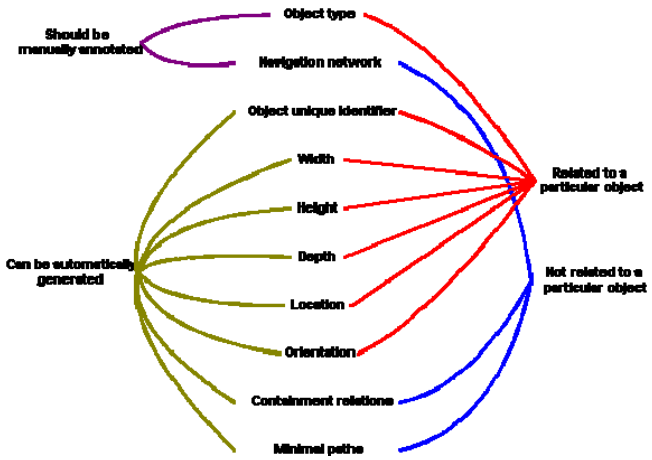

Fig. 3. Elements of the representation model classified according to two different criteria.

we propose to use ontologies because really complex relations can only be represented by them [4]. Furthermore, the effervescence of both the *multi-agent systems* and *semantic web* areas has fuelled efforts towards the development of interesting methodologies and tools to deal with ontologies.

The American Heritage Dictionary defines "ontology" as the branch of metaphysics that deals with the nature of being. The term has been adopted by the artificial intelligence community to refer to a set of concepts or terms that can be used to describe some area of knowledge or to build a representation of it.

As pointed out in [5], the distinction between an ontology and a knowledge base should be clarified. To us, an ontology provides the basic structure or armature around which a knowledge base can be built. An ontology provides a set of concepts and terms for describing some domain, while a knowledge base uses those terms to represent what is true about some real or hypothetical world. Thus, a medical ontology might contain definitions for terms such as "flu" or "infectious disease", but it would not contain assertions that a particular patient had some disease, although a knowledge base might.

Although differences exist within ontologies, general agreement exists between ontologies on many issues [6]:

- There are *objects* in the world.
- Objects have *properties* or *attributes* that can take *values*.
- Objects can exist in various *relations* with each other.
- Properties and relations can change over *time*.
- There are *events* that occur at different *time instants*.
- There are *processes* in which objects participate and that occur over time.
- The world and its objects can be in different *states*.
- Events can *cause* other events or states as *effects*.
- Objects can have *parts*.

B Navigation Network

Every world is annotated with meta-contents in order to make it more navigable. These meta-contents represent a *navigation network* defined by:

- A set of *accessible nodes*. Each accessible node is a reachable three-dimensional point in the virtual environment.
- A set of *connectivity segments*. Every connectivity segment connects two accessible nodes, indicating the possibility of moving from one of them to the other one.

In the applications we have developed so far, the navigation network is used, for example, by:

- An intelligent virtual assistant which takes the user to the destination he wishes when he selects one of the solutions provided by the query solver to answer his query.
- A virtual storyteller to navigate the world she inhabits, while telling stories at the same time.

IV. META-COTENTS WHICH CAN BE AUTOMATICALLY EXTRACTED

The automatically extracted meta-contents we propose to be annotated are: *object identifier*, *location*, *orientation*, *width*, *height*, *depth*, *spatial containment relations*, and *minimal paths*. The first six ones indicate characteristics of every object in the world being considered.

A Object Unique Identifier

Every virtual world which is accessible from the Internet is also uniquely identified in the Net by its related *uniform resource location* (*URL*). In our model every object is uniquely identified in the world where it is contained by an *object identifier* (*OID*). Therefore every annotated object *O* in a world *W* can be uniquely referenced on the Internet and in the knowledge base by the sequence

$$OUI_{O,W} = URL_W + \text{'\#'} + OID_{O,W} \qquad (1)$$

where $OUI_{O,W}$ stands for *object unique identifier* of the object *O* in the world *W*, + is the string concatenation operator, URL_W is the URL that uniquely identifies *W* and $OID_{O,W}$ is the unique identifier of *O* in *W*. Fig. 4 shows an example where *OUI*'s of some objects of a virtual world are annotated.

Note that the OID (and therefore the OUI) can be automatically annotated by using a string composed of the type of the object concatenated to an increasing counter of objects of this type in the world.

If a particular object *O* is contained in a virtual world which is not available on the Internet, but only on an unplugged computer, then its *object unique identifier* is

$$OUI_{O,W} = OID_{O,W} \qquad (2)$$

B Object Width, Height and Depth

Width, height and depth of an object are defined as the extents of the object along the X, Y, and Z axis (please note that we use a Cartesian, right-handed, three-dimensional coordinate system) respectively (see Fig. 5). These lengths can be

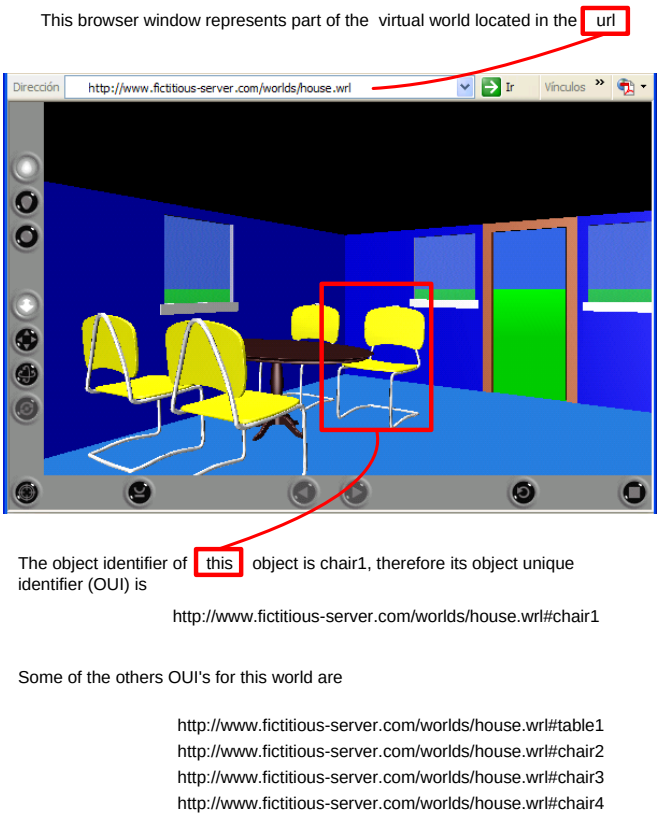

Fig. 4. OUI is annotated for several objects in a world. See Color Plate 9.

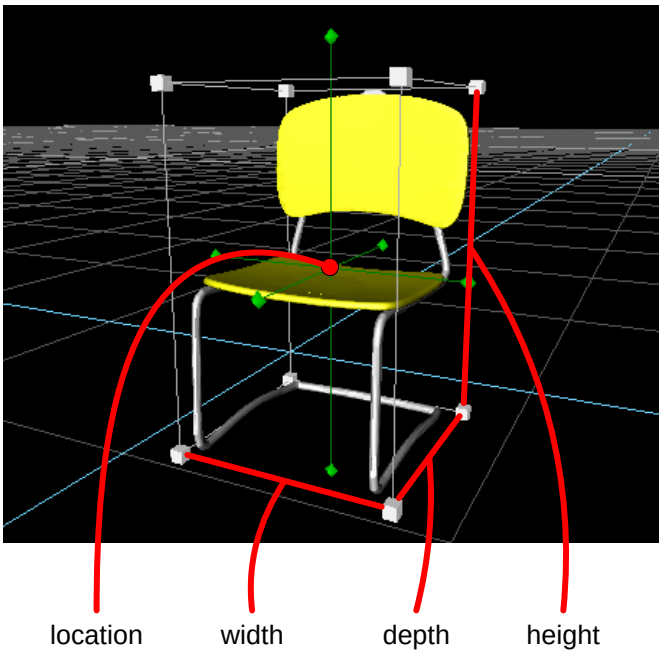

Fig. 5. Width, height, depth and location of an object. See Color Plate 10.

calculated from the object maximum and minimum values in each axis, as we show below

$$\begin{aligned} Width &= maximum_X - minimum_X \\ Height &= maximum_Y - minimum_Y \\ Depth &= maximum_Z - minimum_Z \end{aligned} \qquad (3)$$

where $maximum_X$, $maximum_Y$ and $maximum_Z$ are the maximum values of the object in *X*, *Y* and *Z* axis respectively. $minimum_X$, $minimum_Y$ and $minimum_Z$ are the minimum values of the object in the *X*, *Y* and *Z* axis respectively.

However, often this calculation is not needed because the extents of the object are explicitly specified in the geometric models, in order to optimise the visualisation. For example, several VRML nodes include a bounding box specification [7] comprised of two fields, *bboxSize* and *bboxCenter*. A bounding box is a rectangular parallelepiped of dimension *bboxSize* centred on the location *bboxCenter* in the local coordinate system. This is typically used by grouping nodes to provide a hint to the browser on the group's approximate size for culling optimisations. The interesting thing is that *bboxSize* explicitly indicates the extents of the object along the X, Y, and Z axis.

C Object Location

The location of a particular object is defined by coordinates X, Y, Z corresponding to the object location with respect to a global reference system which is unique for the world where the object is located in. These coordinates can be either directly extracted from the geometric model or obtained by transforming data directly extracted from the geometric model (for example, if the object is referenced with respect to a relative reference system). In particular, we define the object location, $Loc = (Loc_X, Loc_Y, Loc_Z)$, as the centre of mass of the object. Therefore

$$\begin{aligned} Loc_X &= minimum_X + \frac{Width}{2} \\ Loc_Y &= minimum_Y + \frac{Height}{2} \\ Loc_Z &= minimum_Z + \frac{Depth}{2} \end{aligned} \qquad (4)$$

where $minimum_X$, $minimum_Y$ and $minimum_Z$ are the minimum values of the object along the *X*, *Y* and *Z* axis respectively. *Width*, *Height* and *Depth* correspond to the features described previously in Section Ⅲ(B)

Please note that this location corresponds to the centre of the minimal box containing the object, i.e. in terms of VRML, the location of the object corresponds to the field *bboxCenter* of the bounding box containing the object (see Fig. 5).

D Object Orientation

Object orientation is defined by a *rotation axis* and a *rotation angle*. The rotation axis is an imaginary line about which the object is rotated. To specify a direction for the rotation axis, two coordinates are provided, indicating the extremes of the axis. The rotation angle specifies the angle, in radians, that the object is rotated about the rotation axis. Object orientation data can be either directly extracted from the geometric model or obtained by transforming data directly extracted from the geometric model.

E Spatial Containment Relations

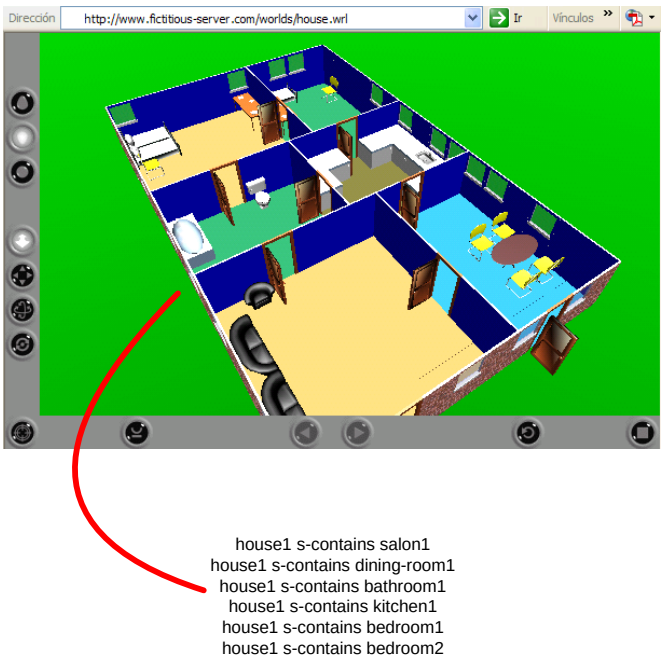

Fig. 6. Relations of containment are annotated for several objects in a world.

Spatial containment relations are relations of the type *X s-contains Y* expressing that the object *Y* is spatially contained by the object *X*. These relations can be calculated from the geometric models. In fact, these relations can be often easily extracted from the world geometric models, due to the fact that the world description formats are usually based on hierarchical models.

For example, a VRML file is hierarchical [8]; node statements can contain *SFNode* or *MFNode* field statements that, in turn, contain node (or USE) statements. This hierarchy of nodes is called the scene graph. Each arc in the graph from *A* to *B* means that node *A* has an *SFNode* or *MFNode* field whose value directly contains node *B*. It is interesting to constrain the node hierarchy so that it also becomes a spatial hierarchy, in order to obtain better visualisation optimisations thanks to the *bounding boxes* described above. If a parent box is culled within a hierarchy of bounding boxes, all of their children are culled as well. In such a case, the containment relations could automatically be extracted from the hierarchical model.

Fig. 6 shows an example where relations of containment between pairs of objects of a virtual world are annotated.

F Minimal Paths

The minimal path from an accessible node *A* to an accessible node *B* is the shortest path (constructed as a sequence of accessible nodes/connectivity segments) to reach *B* from *A*. The minimal paths between every pair of accessible nodes of the navigation network are stored in a table of minimal paths. Note that this table can be built by utilising any shortest path algorithm. In particular we apply the Floyd algorithm.

V. ANNOTATION PROCESS

Fig. 7 shows the ideal annotation process that we propose to follow in order to annotate a virtual world. This process assumes that the world creator utilises an editor prepared to annotate worlds with the meta-contents which can be automatically generated.

First the user creates 3D objects by employing an appropriate

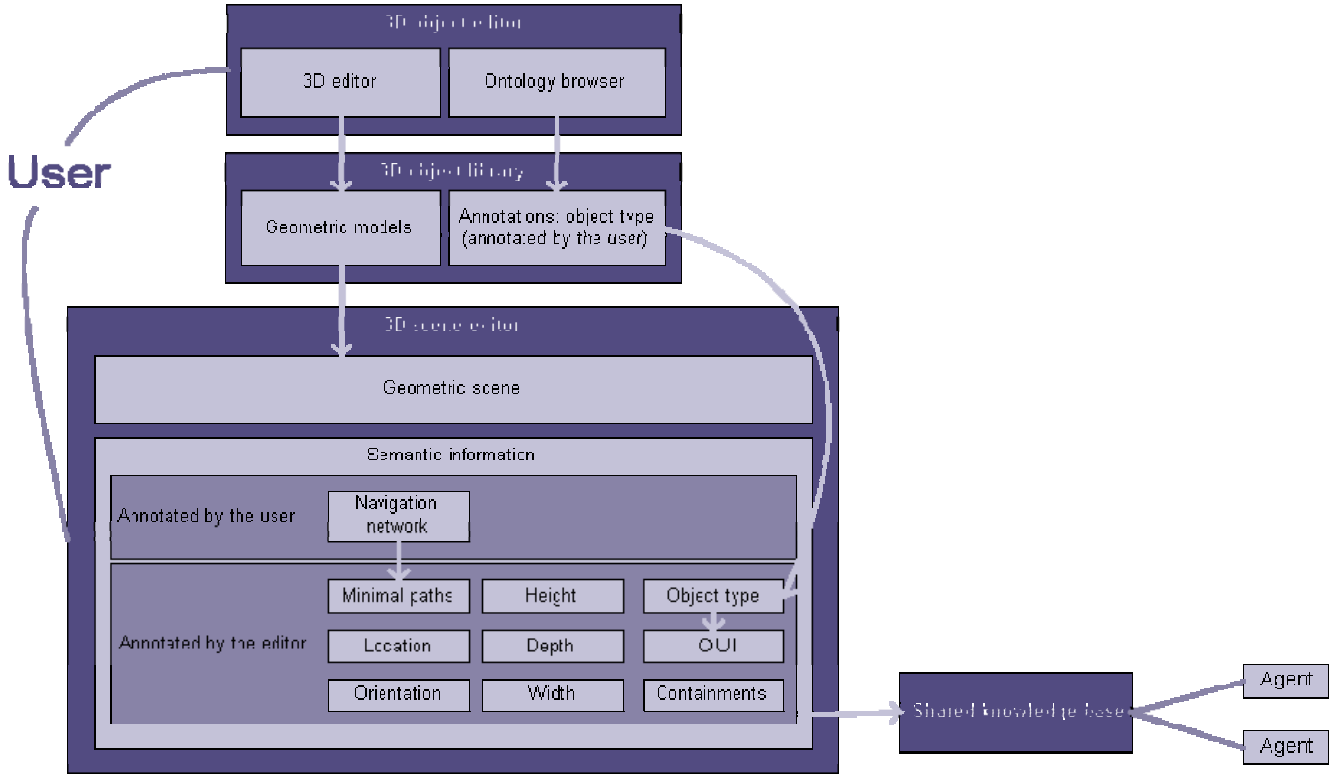

Fig. 7. Annotation process.

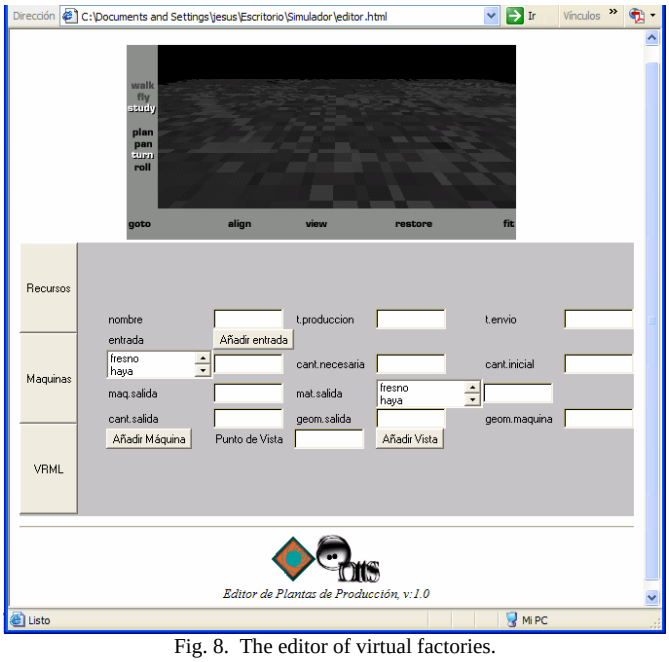

Fig. 8. The editor of virtual factories.

object editor. The user annotates every object with a meta-content indicating its type through an ontology browser. The objects created in this way, along with its related annotations, are stored in an object repository consisting of a 3D object library.

Afterwards the world creator interacts with a scene editor to edit the virtual environment. While editing the scene, he adds objects from his own graphics library. Each time the creator adds an object, the editor automatically adds the related identifier, width, height, depth, location, orientation and containment information to the semantic level of the scene. In the same way, every time the user changes any object property (for example he changes the location of the object by moving it), the editor accordingly modifies its related value in the semantic representation. In addition, the editor adds the object type as annotated by the user when the 3D object was created. If the user did not annotate it, the scene editor asks the user for the object type.

The creator must construct a navigation network. Some current world editors (for instance the Unreal Tournament scene editor) allow creators to do that. Finally, a table of minimal paths is automatically generated from the navigation network. Thus, in the best case, users only will have to annotate the type of the virtual objects and the navigation network.

Once the edition process finishes, the annotated information is added to a global knowledge base containing information related to several worlds. This knowledge base is made available for agents, so that they can take advantage of the semantic conceptualisation of the environments.

A Our Experiences Annotating Virtual Worlds

We had three different kinds of experiences with annotation of virtual worlds employing our semantic model: first we extended a simple scene editor so that it would support the just described annotation process; second we manually annotated a complex virtual environment we previously developed; third we created a program that automatically generates particular virtual worlds along with its related semantic annotations.

At first we modified a simple scene editor we had previously developed [9] (see Fig. 8). It was intended to allow users to create virtual factories. The editor was implemented in Java and VRML. It was a simple prototype, so its functionality was quite reduced. Basically the editor allowed users to add objects from a graphic objects library to the scene being edited. Users could also define basic behaviours for some objects. We successfully added the mechanisms involved in the just described annotation process to our simple editor.

After that, we designed a large virtual environment representing the real Center of Environmental Education (Centro de Educación Medioambiental, CEMACAM) of the Mediterranean Savings Bank (Caja de Ahorros del Mediterráneo, CAM) located in Torreguil, Murcia, Spain. The environment, which is shown in Fig. 9, was composed of five different buildings. The graphic edition of the complete environment took about six months. The edition was carried out by employing commercial products such as Autocad, 3D Studio Max and Cosmo Worlds, as well as Java programs developed by ourselves to produce some curved ceilings and walls. Once it was finished, we manually annotated meta-contents for all the objects in the environment. This process lasted only one week. One week is little compared to six months. Therefore, from our experience, we can say that the effort necessary to annotate virtual environments following our model, even in the worst case (annotation of a previously created world), is relatively small. We later utilized the annotated virtual CEMACAM to preliminary test some of our applications.

Finally, we developed a Java application which automatically generates virtual worlds composed of several parks with objects of different types (like trees, park benches, etc.) and sizes (see Fig. 10). The application not only generates the geometrical representation of the worlds, but also its related semantic information. The system relies on a library of pre-defined VRML prototypes (parks, trees, park benches, etc). A world is constructed as a combination of instantiations of the prototypes. These instantiations scale the prototypes by employing automatically generated values which fulfil certain rules such as not exceeding a particular maximum height. The instantiations are spatially located taken into account pre-specified constraints such as maximum number of trees per park, minimum distance among parks, etc. The application was developed in order to generate several annotated virtual worlds to test some of our applications.

VI. CONCLUTIONS

We have described our proposal for a new representation model for virtual worlds. Our model provides a layer of semantic information for virtual environments which does not intend to substitute application dependant approaches, but to

Fig. 9. Snapshots of the virtual CEMACAM. See Color Plate 11.

constitute a common lower level for all of them. Our model is simple in that it consists of a reduced number of different elements and most of them can be automatically annotated. The model was conceived in that way to encourage the world creators to use it, as it does not require great efforts from them. Conversely, if world creators agree on using a common semantic layer like the one we propose, their creations would take advantage of intelligent agents (like virtual guides, personal assistants, etc.) developed by other groups.

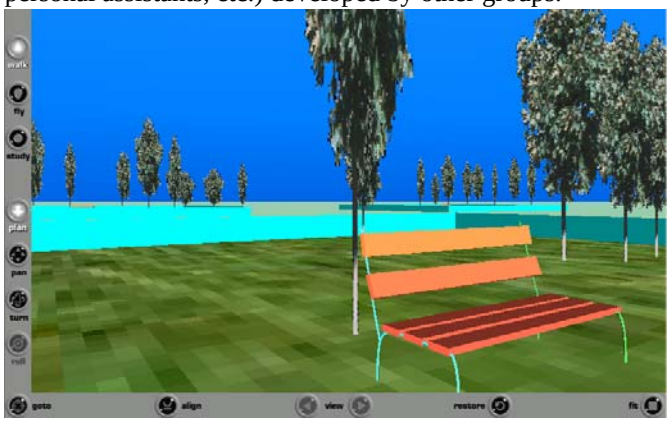

Fig. 10. Annotated world of parks, automatically generated by a Java application developed by ourselves. See Color Plate 12.

We have also shown the ideal annotation process, which requires 3D graphics editors that are especially prepared. However, even without these editors, annotating a world is not too demanding. In this sense, we have shown our own experiences in annotating worlds.

The model has already been successfully used in two different applications. In this sense, we have extended the model with two different application dependant upper semantic levels. The first one supports the construction of query solvers that allow users to search for particular objects or scenes in the worlds they visit [2]. The second one supports the construction of virtual storytellers which are aware of the world they inhabit [3].

ACKNOWLEDGMENT

We would frankly like to thank Leticia Lipp for generously proofreading. This work has been partially funded by the Spanish Ministry of Science and Technology (via the project "PLANET: Plataforma de colaboración aumentada para el acceso y distribución de contenidos educativos", TIC-2003-09288-C02-00).

REFRENCES

[1] Jesús Ibáñez and Carlos Delgado-Mata. Virtual Environments and Semantics, UPGRADE, *The European Journal for the Informatics Professional*, vol. 7, no. 2, pp. 18-24, April 2006.

[1] Jesús Ibáñez, Antonio F. Gómez-Skarmeta and Josep Blat. Fuzzy approach to the intelligent management of virtual spaces, *IEEE Transactions on Systems*, Man, and Cybernetics, Part B: Cybernetics, vol. 36, no. 3, pp. 494-508, June 2006.

[1] Jesús Ibáñez, Ruth Aylett and Rocío Ruiz-Rodarte. Storytelling in Virtual Environments from a Virtual Guide Perspective, *Journal of Virtual Reality*, Springer, vol. 7, no. 1, pp. 30-42, December 2003.

[1] M.N. Huhns and Munindar P. Singh. Ontologies for Agents, *IEEE Internet Computing*, pp. 81-83, November/December 1997.

[1] William Swartout and Austin Tate. Ontologies - Guest editors' introduction, *IEEE Intelligent Systems*, pp. 18-19, January/February 1999.

[1] Balakrishnan Chandrasekaran, John R. Josephson and V. Richard Benjamins. What Are Ontologies, and Why Do We Need Them? , *IEEE Intelligent Systems*, pp. 20-26, January/February 1999.

[1] Rikk Carey and Gavin Bell. Bounding boxes, in *Annotated VRML 97 Reference Manual*, *Addison-Wesley*, 1997, ch.2.6.4. Available: http://www.web3d.org/resources/vrml_ref_manual/ch2-26.htm#2.6.4 .

[1] Rikk Carey and Gavin Bell. Scene graph hierarchy, in *Annotated VRML 97 Reference Manual*, Addison-Wesley, 1997, ch. 2.6.2. Available: http://www.web3d.org/resources/vrml_ref_manual/ch2-24.htm#2.4.2

[1] Antonio F. Gómez-Skarmeta and Jesús Ibáñez. Un entorno para la Edición y Simulación de Plantas de Producción Virtuales, in *Proceedings XI Congreso Internacional de Ingeniería Gráfica*, Ingegraf, Logroño, Spain, 1999, pp. 638-647.

Jesús Ibáñez Martínez received the Ph.D. degree in computer science from the University of Murcia, Murcia, Spain, in 2004.He is a Visiting Professor in the Department of Technology at the University Pompeu Fabra, Barcelona, Spain. Among others, he was granted a Marie Curie fellowship at the Centre for Virtual Environments at the University of Salford, Manchester, U.K. His research interests include the application of intelligent systems to improve user interaction, especially with virtual worlds. He has published extensively in this area.

Carlos Delgado-Mata received his PhD from University of Salford, United Kingdom. He is an Assitant Professor and head of the CINAVI (Centre for Research in Intelligent Virtual Environments) at Universidad Bonaterra, Aguascalientes, México. He is co-chair of the International Conference Intelligent Virtual Environ -ments and Virtual Agents (IVEVA) 2006 and he was co-chair of the International workshop IVEVA 2004. He is author of more than 25 peer refereed articles on international conferences and journals. He is partner of the startup computer games developer Nibbo Studios.