

Using Ontologies for Modeling Virtual Reality Scenarios

Mauro Dragoni^{1(✉)}, Chiara Ghidini¹, Paolo Busetta², Mauro Fruet²,
and Matteo Pedrotti²

¹ FBK–IRST, Trento, Italy

{dragoni,ghidini}@fbk.eu

² Delta Informatica, Trento, Italy

{paolo.busetta,mauro.fruet,matteo.pedrotti}@deltainformatica.eu

Abstract. Serious games with 3D interfaces are Virtual Reality (VR) systems that are becoming common for the training of military and emergency teams. A platform for the development of serious games should allow the addition of semantics to the virtual environment and the modularization of the artificial intelligence controlling the behaviors of non-playing characters in order to support a productive end-user development environment. In this paper, we report the ontology design activity performed in the context of the PRESTO project aiming to realize a conceptual model able to abstract the developers from the graphical and geometrical properties of the entities in the virtual reality, as well as the behavioral models associated to the non-playing characters. The feasibility of the proposed solution has been validated through real-world examples and discussed with the actors using the modeled ontologies in every day practical activities.

1 Introduction

Serious games with 3D interfaces are a branch of VR systems and are often used for the training of military personnel (in individual as well as team coordination danger situations) and, more recently, for the training of civilian professionals (firefighters, medical personnel, etc.) in emergency situations using tools such as VBS3¹ and XVR².

A crucial step towards the adoption of VR for training is the ability to configure scenarios for a specific training session at reduced costs and complexity. By looking at state of the art technologies, it is already possible to do so for physical landscapes, physical phenomena, and crowds (including their behaviors), and trainers and system integrators can assemble and customize serious game products for a specific scenario using commercial products and libraries that need to be (easily) adapted to the specific landscapes and needs of the clients.

Not so advanced is the technology for enriching the scenarios with non playing characters, that is, those characters (people, animals, vehicles, small teams,

¹ <https://www.bisimulations.com/>.

² <http://futureshield.com/xvr-ensemble.shtml>.

and so on) directly involved in game playing in collaboration with (or in opposition to) human players, but whose behavior is entirely animated in an artificial manner. Here the problem is (at least) twofold. A first problem is the lack of configuration environments for trainers and system integrators to complement the “physical landscape” with descriptions of the non playing characters at a high level. As an example, an environment that makes possible the configuration of a scenario for fire emergency training in a hospital ward that contains, in addition to the physical reconstruction of the ward building and of the fire, a set of non playing characters composed of three nurses, of which one expert, one doctor from another ward who is not familiar with the safety procedures of the ward, and eight patients among which a child and a blind patient. A second problem is the lack of algorithms for the generation/selection of realistic and plausible behaviors for non playing characters, able to adapt themselves to the evolution of the game. While this is not a problem for entertainment games, it is a serious problem for serious games as it has the effect of discouraging and disengaging the trainee, so that it is not uncommon the recruitment of experts to impersonate additional characters (such as team mates, enemies, victims, dogs, injured people, and so on) in the simulation, making it more complex and expensive.

Current attempts to the programming of non playing characters rely on ad hoc specifications/implementations of their behaviors done by VR developers. Thus, a specific behavior (e.g., a function emulating a panicking reaction) is hardwired to a specific item (e.g., the element “Caucasian_boy_17” of a VR such as XVR) directly in the code. This generates a number of problems typical of ad hoc, low level solutions: the solution is scarcely reusable, it often depends on the specific knowledge of the code of a specific developer, and is cumbersome to modify, since every change required by the trainer has to be communicated to the developers and directly implemented in the code in a case by case manner. The existence of high level specifications of non playing characters and modular behaviors, described in a manner that is independent from the specific VR, and available for both trainers and developers, would be an important step towards the definition of reusable, flexible, and therefore cheaper, scenarios that include non playing characters.

In this paper, we focus on the experience of using Semantic Web techniques, and in particular lightweight ontologies, for the high level description of the artificial entities (including characters) and their behaviors in gaming in order to uncouple the description of scenarios performed by the trainers from their physical implementation in charge to the developers. Differently from a number of works in literature that often uses ontologies for a detailed description of the geometrical properties of space and objects, the focus of our work is on the description of the entities of a VR scenario from the cognitive point of views of the trainers and the developers alike, in a way that is semantically well founded and independent of a specific game or scenario [1], and with the goal of fostering clarity, reuse, and mutual understanding [2].

The outcome of such an experience is a shared vocabulary, presented in Sect. 3, grounded in the foundational ontology DOLCE that helps in identifying

the basic entities of a VR scenario, together with their mappings to items of a specific VR implementation (such as XVR). An evaluation of the usefulness of such a shared vocabulary in a real-world use case (the PRESTO project described in Sect. 2) is presented in Sect. 5, while a discussion concerning lessons learned about the feasibility of the proposed system is reported in Sect. 6.

To the best of our knowledge, the construction and evaluation of the ontology presented in this paper provides a first experience towards the description of a virtual world from a cognitive level that can highlight the potential and criticality of using Semantic Web techniques, and existing ontologies, to describe a VR from a cognitive point of view and can provide the basis for further developments.

2 The PRESTO Project

The objective of PRESTO (Plausible Representation of Emergency Scenarios for Training Operations) research project is the creation of a system for the customization of serious games scenarios based on virtual reality. The advantage of this system, compared to the state of the art, resides in the richness and the ease of defining the behavior of artificial characters in simulated scenarios, and on the execution engines able to manage cognitive behaviors, actions, and perceptions within a virtual reality environment. One of the main outcome of the project is the possibility of specifying procedures, psychological profiles, and other factors that influence the behavior of individuals and/or small groups in any role (emergency teams, victims, observers, terrorists, criminals, etc.) and to build scenarios, for instance a car accident, in which part or all of the people involved are simulated by artificial characters. To this end, the system has to include an environment for building the training scenarios by the VR trainer, tools for the specification of cognitive and perceptual models used for augmenting psychological profiles of non-player characters, and execution engines able to manage cognitive behaviors, actions, and perceptions within a virtual reality environment.

The system can be used, for example, for training safety personnel, for the verification and the optimization of operational procedures, and for the analysis of work environments. The system has been tested in a pilot use case selected in a specific application domain of large interest in both commercial and research fields: training for emergency management within close environments (such as fires, evacuations, overload of users due to external factors such great disasters scale, etc.). The pilot has been conducted in collaboration with the Health Services of the Trentino local government (APSS).

The open problems addressed by this project may be summarized as follows:

1. the perception of the virtual environment by an artificial character and the execution of its models and procedures must be able to adapt to the context, to its history and status (fatigue, emotions, intake of stimulants such as caffeine or depressants such as alcohol) and must maintain a level of variability (i.e. in the accuracy of the vision, the rate of reaction, in the choices among alternatives) such that the behavior is plausible but not trivially predictable;

2. the representation of procedures and patterns of behavior must be independent of one specific usage scenario and accessible to training specialists (i.e. industrial safety or civil protection) rather than just a computer, in an environment facilitating the definition and configuration of training scenarios by such specialists.

The first open problem relates to aspects such as the usage of a BDI (Beliefs-Desire-Intention) multi-agent system with cognitive extensions, CoJACK [3], as the artificial intelligent engine for the generation/selection of behaviors in serious games [4], that go beyond the scope of this paper.

What we present in this work, instead, is the experience of using Semantic Web techniques, and in particular lightweight ontologies, to contribute to the second open problem, that is the development of a programming environment for serious game platforms thanks to end-user development tools [5] and the ability to mix and match scenario components (including behavioral components) taken off-the-shelf from a market place.

3 PRESTO Ontology Design

The development of programming environment for the high level description of artificial entities (including characters) and their behaviors in scenarios of serious games requires the ability to represent a wide range of entities that *exist* in the (artificial) world. The approach taken in PRESTO is to use ontologies to represent this knowledge, in a way that is semantically well specified and independent of a specific game or scenario [1].

The construction of the PRESTO ontology therefore is driven by typical questions that arise when building ontological representations of a domain, that is:

- “What are the entities that exist, or can be said to exist, in a Virtual Reality scenario?”
- “How can such entities be grouped, related within a hierarchy, and subdivided according to similarities and differences?”

Differently from Ontology in philosophy, where these questions are motivated from the need to investigate the nature and essence of being, we have looked at these questions from the pragmatic point of view of computer science, where ontologies and taxonomic representations have been widely proposed and used to provide important conceptual modeling tools for a range of technologies, such as database schemas, knowledge-based systems, and semantic lexicons [2] with the aim of fostering clarity, reuse, and mutual understanding.

A serious problem we had to face in PRESTO was the lack-of/limited-availability of training experts and software developers, and the broad scope of items and behaviors that can occur in an arbitrary scenario of VR, that can range from terrorist attacks in a war zone, to a road accidents in a motorway, to a fire alarm in a nuclear plant or hospital and so on. Because of that reason, building everything from the ground up by relying on domain experts and using one of the state of the art ontology engineering methodologies such as METHONTOLOGY [6] was deemed unfeasible. Thus the process followed in PRESTO has been driven by an

attempt to: (1) maximize the reuse of already existing knowledge and (2) revise and select this knowledge with the help of experts by means of more traditional ontology engineering approaches such as the one mentioned above. The choice of already existing knowledge has lead us to consider the following two sources:

- state of the art foundational ontologies which provide a first ontological characterization of the entities that exist in the (VR) world; and
- the concrete items (such as people, tools, vehicles, and so on) that come with virtual reality environments and can be used to populate scenarios.

Our choices for the PRESTO project were the upper level ontology DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) [7], and the classification of elements provided by XVR. DOLCE was chosen as this ontology not only provides one of the most known upper level ontologies in literature but it is also built with a strong cognitive bias, as it takes into account the ontological categories that underlie natural language and human common sense. This cognitive perspective was considered appropriate for the description of an artificial world that needs to be plausible from a human perspective. The decision to use the classification of elements provided by XVR was due to the extensive range of item available in their libraries (approximatively one thousand elements describing mainly human characters, vehicles, road related elements, and artifacts like parts of buildings) and the popularity of XVR as virtual reality platform.

The construction of the first version of the ontology of PRESTO was therefore performed by following a middle-out approach, which combined the reuse and adaptation of the conceptual characterization of top-level entities provided by DOLCE and the description of extremely concrete entities provided by the XVR environment. More in detail,

- we performed an analysis and review of the conceptual entities contained in DOLCE-lite [7] together with the Virtual Reality experts (both trainers and developers) and selected the ones referring to concepts than needed to be described in a VR scenario; this analysis has originated the top part of the PRESTO ontology described in Sect. 4.1.
- we performed a similar analysis and review of the XVR items, together with their classifications, in order to select general concepts (e.g., vehicle, building, and so on) that refer to general VR scenarios; this analysis has originated the middle part of the PRESTO ontology described in Sect. 4.2.
- as a third step we have injected (mapped) the specific XVR items into the ontology, thus linking the domain independent, virtual reality platform independent ontology to the specific libraries of a specific platform, as described in Sect. 4.3.

A reader could ask now why we didn't simply/mainly rely on the XVR classification in order to produce the, so called, PRESTO ontology. The reason is twofold: first of all, the XVR classification mainly concerns with objects. It provides therefore a good source of knowledge for entities “that are” (in DOLCE called Endurants), but a more limited source of knowledge on entities “that happen” (in DOLCE called Perdurants). Second, the XVR libraries contain objects

described at an extremely detailed level whose encoding and classification resembles more to a Directory structures built to facilitate the selection of libraries rather than a well thought is-a hierarchy and therefore presents a number of problems that prevent its usage ‘as such’. In the following, we review the most common problems we found in the categorization of the XVR items:

- Concepts names are used to encode different types of information. For instance the concept name “Caucasian_male_in_suit_34” is used to identify a person of Caucasian race, dressed in suit and of 34 years of age. Encoding the information on race, age, and so on via e.g., appropriate roles enables the definition of classes such as e.g., “Caucasian_person”, “young adult”, “male” and so on and the automatic classification (and retrieval) of XVR item via reasoning.
- The terminology used to describe concepts is not always informative enough: for instance, it is difficult to understand the meaning of the entity “HLO_assistant” from its label and description and to understand whether this item may suggest a type of “assistant” that may be useful in several scenarios and could therefore be worth adding to the ontology.
- The level of abstraction at which elements are described varies greatly. For instance the library containing police personnel items classifies, at the same hierarchical level the general concept of “Police_Officer” and the rather specific concept of “Sniper_green_camoouflage”.
- the criteria for the classification is not always clear: for instance, the “BTP_officer” (British Transport Police) concept is not a subclass of “Police_Officer”.
- Certain general criteria of classification are not present in all the libraries. As an example, the general concept “Adult_Male” should be a general concept used for the classification of male characters. Nonetheless, it is present in e.g., the library of “Environment_humans” (that is, the library that describes generic characters) and is not present in e.g., the libraries of “Rescue_humans” and “Victims” (that is, the libraries of characters impersonating rescuers and victims, respectively).
- Unclear classification: for instance, in the XVR original classification a “sign” is a “road_object”, and a “danger_sign” is an “incident_object”. By considering that no relations are defined between the entities “sign” and “danger_sign”, it is not possible to infer any relation between “danger_sign” and “road_object”.
- Duplication of concept names: for instance, the label “police_services” is used to describe both human police characters in the library “environment_human”, and police vehicles, in the library “rescue_vehicle”.

In the next section we provide an overview of the PRESTO ontology and of its top-level, middle level and XVR specific components in detail.

4 The PRESTO Ontology

As introduced in Sect. 3, the PRESTO ontology³ is composed of three parts: (i) a top level part constructed with the help of DOLCE; (ii) a middle level describing

³ The current version of the PRESTO ontology cannot be published due to copyrights constraints. A preliminary version, from which it is possible to observe the rational used for modeling it, may be found here: <https://shell-static.fbk.eu/resources/ontologies/CorePresto.owl>.

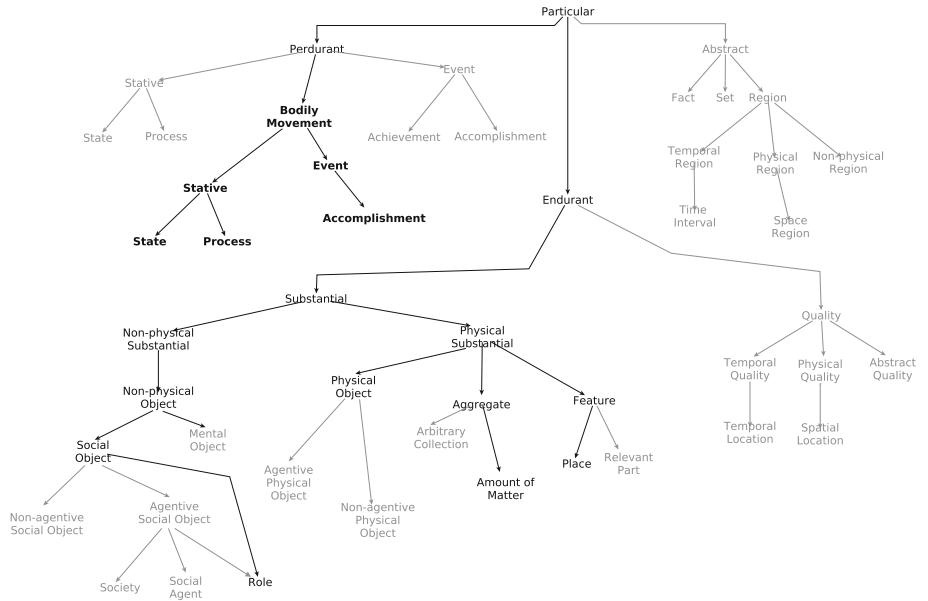


Fig. 1. The top-level PRESTO ontology.

general entities that can occur in a VR scenario, and (iii) a specific set of entities representing objects and “behaviors” available in a concrete VR.

4.1 The Top-Level Ontology: DOLCE Entities

Figure 1 shows the taxonomy of DOLCE entities taken from [7] revised and customised to the needs of PRESTO.

Entities in gray where not included in the PRESTO ontology, while entities in boldface where added specifically for PRESTO.

Among the first level of entities we selected **Endurants** and **Perdurants**: endurants are indeed useful to describe the big number of physical and non-physical objects that can occur in a serious game, including avatars, vehicles, tools, animals, roles and so on; perdurants are instead useful to describe what happens in a scenario. Concerning endurants the diagram in Fig. 1 shows the ones we selected to be included in PRESTO; note that we did not include the distinction between agentive and non-agentive physical objects because of an explicit requirement by the PRESTO developers. In fact, they require the possibility to treat every object in a VR as an agentive one for the sake of simplicity⁴. While perdurants can be useful in a VR to describe a broad set of “things that happen”, in the current version of the ontology they were mainly used to describe

⁴ A typical example is vehicle, which the developers prefer to treat as an agentive objective, rather than a non agentive object driven by an agent, for the sake of simplicity of the code.

animations (that is, “bodily_movements”) of avatars. From an ontological point of view we felt it was appropriate classify them according to the categories of stative and eventful perdurants included in DOLCE. In fact, we can have *state* bodily movements (e.g., being sitting), *process* bodily movements (e.g., running), and *accomplishment* bodily movement (e.g., open a door). The investigation of animations did not show examples of *achievement* bodily movements, which were therefore not included in the ontology.

The current version of the ontology does not contain **Qualities**, but current work (not described in this paper) is devoted to investigate how to include them in a further revision. Instead **Abstracts** do not seem to play a role in the PRESTO ontology.

4.2 The Middle-Level Domain Ontology

This part augments the top level ontology described above with concrete, but still abstract, entities that may appear in a broad range of virtual reality scenarios for serious games. The current version of the ontology is composed of 311 concepts, 5 object properties and 3 annotations properties. Concerning the Endurant part the main entities modeled in the middle-level ontology pertain classifications of persons (avatars), buildings, locations, tools/devices, vehicles, and roles. Concerning perdurants the ontology contains concepts describing *state*, *process* and *accomplishment* bodily movement. An excerpt of the middle-level ontology can be seen in Fig. 2.

4.3 Injecting the Bottom-Level Ontology

The linking of the bottom-level ontology, representing the classification scheme used for organizing the items contained in the 3D-library, is not a trivial task.

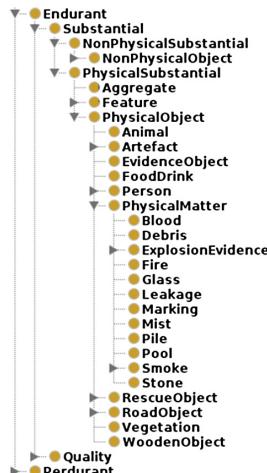


Fig. 2. The middle-level PRESTO ontology.

Indeed, the correct alignment of these levels enables the transparency of the system with respect to the actual content of the 3D-library.

While the creation of the top and middle-level of the PRESTO ontology is meant to create a stable knowledge source, the definition of the alignments with the bottom-level elements is an activity that has to be done every time a new 3D-library is plugged into the system.

To ease this injection we decided to accomplish it in two separate steps: (i) an automatic definition of alignments by using an ontology alignment tool and (ii) a manual refinement of the alignments before using the complete ontology in the production stage.

The output of the alignment task is the linking between the abstract concepts contained in the middle PRESTO ontology and the concrete items contained in the underlying 3D-library implemented in the system. Indeed, such alignments allow the access to the entire set of items defined in the 3D-library and that are physically used for building the virtual reality scenario.

For sake of clarification about the alignment process works, let's consider the following example. In the middle-level of the ontology we have defined the concept "Tent" representing a general tent that may be used for building a virtual reality scenario. By plugging, for example, the XVR library, we need to find an alignment between the entity "Tent" and the specific tent items contained in XVR, such as "Decontamination_Tent_Zone_1", "Family_tent_blue", "Treatment_Area", and so on. To do that, as first step, we execute the Alignment API library [8]: for the entity "Tent", the XVR item identified in the 3D-library and aligned with it is "Tents". Such an alignment, classifies the bottom-level ontology "Decontamination_Tent_Zone_1", "Decontamination_Tent_Zone_2", "Decontamination_Tent_Zone_3", "Family_tent_blue", "Family_tent_orange", "Festival_tent", and "Treatment_Area" as children of the concept "Tents". As a consequence, all these elements can be retrieved and used at run time to produce a specific scenario which requires the presence of a tent, while the scenario can still be described using the abstract term "tent". Also, the same high level scenario may be easily adapted to the usage of other 3D-libraries, simply by exploiting the (different) mappings of such libraries with the middle level "Tents" concept.

In some cases the automatic alignment we used fails: for example, the middle-level entity "Weapon" is automatically aligned with the bottom-level entity "Baton" instead of being aligned with the bottom-level entity "Service-weapon". In these cases, a manual refinement of the generated alignments was done afterwards for pruning wrong axioms.

By considering the XVR use case, the automatic alignment procedure allowed a time-effort reduction, with respect of doing everything manually, of around 65 % in the definition of the alignment between the middle-level and the bottom-level ontologies, thus showing the potential of using ontology mapping technologies in the concrete scenario of virtual reality libraries.

5 PRESTO Ontology in Action

In this section, we present how the PRESTO ontology has been applied in the development process of virtual reality scenarios. We will start with a brief

description of the virtual reality environment used in the PRESTO project, the XVR 3D-library. Then, we will present code samples showing how the ontology has been exploited in practice for development purposes and how it has been used in the definition of agent behavioral scripts.

5.1 XVR 3D-Library

XVR is a virtual reality training software used to educate and train operational and tactical safety and security professionals.

One of the most important features of XVR is that the trainer can build an incident scenario and has full control over the course of events in the scenario during the exercise. He can also give feedback, for instance by role-playing the control room or other rescue staff. The trainer, for example, is able (i) to respond to the student's decisions by activating events in the virtual scenario, (ii) decide to condense time and jump to a next phase in the incident, (iii) to ask the student to assess the new situation and respond appropriately, and (iv) to use his full experience and creativity to influence the scenario during an exercise to optimize his learning objectives.

The XVR engine includes an extensive 3D-model database, the XVR library. The XVR library contains dozens of 3D environments and hundreds of virtual objects such as rescue professionals, people, victims, vehicles, wrecks, fires, leaks, and countless other objects. Such objects may be “static” and they cannot be changed over time, or “dynamic” such as adjustable fires or an adjustable explosion range as well as victims to which triage can be applied. These adjustable objects allow the trainer to make changes to the scenario (without the students noticing) so the scenario changes dynamically over time.

From the technical point of view, the XVR library is integrated as extension of Unity3D that is a framework developed for making the creation of video games more easy. It contains a complete game development ecosystem: (i) a powerful rendering engine fully integrated with a complete set of tools and workflows for creating interactive 3D and 2D content; (ii) multi-platform publishing; and (iii) thousands of ready-made assets. For independent developers, Unity's democratizing ecosystem smashes the time and cost barriers to creating games. From the development point of view, Unity3D provides a set of APIs⁵ that can be used for software development.

XVR provides enhanced APIs allowing the control of specific feature of the XVR items that, otherwise, would not be possible with the classic Unity3D APIs (for example the power of fire extinguishers, or escalator movements). Besides this, the XVR library inherits the all functionalities of the Unity3D engine.

5.2 Ontology Enhanced Development

Below, it is possible to observe some examples about how the use of the ontology allows to work by maintaining a high level of abstraction and transparency with respect to the underlying 3D-library used by the system.

⁵ <http://docs.unity3d.com/ScriptReference/>.

```

1: IPhysicalEntity entity = (IPhysicalEntity) scene.GetEntity(id.as_string());
2: if(entity.getModel().equals("asset_object_int_chair"))
3: {
4:     currentDistance = agentPosition.Distance((Vector) position.as_object());
5:     if (currentDistance < minDistance)
6:     {
7:         seat = entity;
8:         minDistance = currentDistance;
9:     }
10: }
11: }
```

Algorithm 1. Usage of the Unity3D API without the PRESTO ontology.

```

1: IEntity entity = scene.GetEntity(id.as_string());
2: if(ontology.IsA(entity, CorePresto.seatable))
3: {
4:     currentDistance = agentPosition.Distance((Vector) position.as_object());
5:     if (currentDistance < minDistance)
6:     {
7:         seat = entity;
8:         minDistance = currentDistance;
9:     }
10: }
11: }
```

Algorithm 2. Usage of the Unity3D API with the PRESTO ontology.

These first two source code examples show the difference in using, or not, the PRESTO ontology in the development process. Algorithm 1 shows a branch of code where a Model-Based Object Creation strategy, concerning in the direct access to the objects contained in the 3D-library, has been used. On the contrary, Algorithm 2, shows how the Ontology-Enhanced Object Creation strategy helps in the abstraction for accessing the elements defined in the virtual reality scenario. The main difference can be seen at line 2 of both algorithms: while in Algorithm 1, the model name of the entity to check is hard-coded in the source code, in Algorithm 2 the type of the entity is checked by invoking the API in charge of mapping the type of the current entity with the concepts defined in the PRESTO ontology.

The second set of examples, shown below, concerns the definition of scripts used for describing the behavior of the characters that are placed in the virtual reality scenario. Briefly, such characters, based on the values of some parameters or based on the “situation” of the scenario, have to act in a certain way. This way of acting is described by some behavioral models like the ones proposed below.

Script 1 shows an example of the assignment of the goal “GoToLocation”, to all elements placed in the scenario referring to the concept “<http://www.Presto/UnityItems#Robot>”. In this case, the entities placed in the virtual reality scenarios are linked through the use of the ontological concepts.

```

<scenes>
  <scene name="Mech approaching">
    <updateAgent concept="http://Presto/UnityItems#Robot">
      <role>BOT</role>
      <behaviouralModel>Interpreter:Robot</behaviouralModel>
      <goal>GoToLocation</goal>
    </updateAgent>
  </scene>
</scenes>

```

Script 1. Example of behavioral model developed with the support of the PRESTO ontology.

The equivalent version, where the name of instances are used instead of the name of concepts, is presented in Script 2. The problem here is that the name of the instances to which the goal has to be applied is completely specified within the behavioral model. This way, the developer has to specify, before to know how the scenario has been composed, the entire list of entities placed in it.

```

<scenes>
  <scene name="Mech approaching">
    <updateAgent name="Mech1">
      <role>BOT</role>
      <behaviouralModel>Interpreter:Robot</behaviouralModel>
      <goal>GoToLocation</goal>
    </updateAgent>
    <updateAgent name="Mech2">
      <role>BOT</role>
      <behaviouralModel>Interpreter:Robot</behaviouralModel>
      <goal>GoToLocation</goal>
    </updateAgent>
    <updateAgent name="Mech3">
      <role>BOT</role>
      <behaviouralModel>Interpreter:Robot</behaviouralModel>
      <goal>GoToLocation</goal>
    </updateAgent>
  </scene>
</scenes>

```

Script 2. Example of behavioral model without using the PRESTO ontology.

6 Lessons Learned

In the previous sections, we presented which are the goals of the PRESTO project and we explained how and why the use of ontologies simplifies the development of virtual reality scenarios, as well as, improves the re-usability of the source code when the developed software is plugged to different underlying 3D-libraries.

In this Section, we sum up the experience of the PRESTO project by reporting which aspects have been perceived as advantages by the people actively involved in the project and, on the contrary, which ones have been considered as criticalities that need to be analyzed more in details for future perspectives.

The evaluation of the usefulness and effectiveness of the proposed ontology-based system has been conducted by interviewing developers and modelers concerning the usability of the proposed system with respect to previous version of the platform where ontologies were not adopted. From the research point of view, the two questions that we want to answer are:

- RQ1: “Is the use of an ontology-based system useful for simplifying the development of virtual reality scenario?”
- RQ2: “Is the use of an ontology-based system enough for managing the behavior of the characters deployed in the scenario?”

Below, we report the outcomes of the discussions about the most important aspects done with both the developers and the modelers involved in the project.

Code Re-usability. The high level of code re-usability observed by the developers was the most perceived advantage of the proposed system. As described in Sect. 5, the use of the ontology allows to develop the structure of virtual reality scenario, as well as, characters behavioral models, by abstracting the references to the physical entities. This way, the implementation remains completely independent by the libraries used for modeling the actual 3D-element or for defining ad-hoc behavioral models.

The result is that every time a new classification scheme, describing the content of a 3D-library or a set of behavioral models, is plugged to the system, the effort requested for migrating the source code to the new libraries is strongly reduced. The industrial nature of the project make this aspect the most important one, especially from the business point of view by considering the economical saving in using the proposed technology.

Development Effort. The second point, that is directly connected with the previous one, is related to the effort saved by the developer during his work. In particular, there are two aspects that have been highlighted:

- speed-up the development process: by using a “fixed” set of concepts, “fixed” in the sense that the set of concepts remains the same independently by the 3D-library used, developers do not need to learn, every time a new library is plugged to the system, the classification scheme of the items or of the behavioral models contained in the plugged library.
- developer knowledge limited to part of the ontology structure: as direct consequence of the aspect presented above, by using the ontology, the developer is not demanded to know what has been modeled “under” the middle-level. This because the alignment between the middle and the bottom levels of the ontology is delegated to the modeler; therefore, the developer does not have to know the entire structure of ontology, but his knowledge may be limited to the top and middle levels. Indeed, the developer expresses each reference to entities by using exclusively the concept contained in the middle-level of the ontology without knowing any detail related to the physical description of the 3D-items, as well as, of the behavioral models.

Criticalities. Besides the positive consideration described above, the discussions with all the people involved in the project rose some criticalities during the usage of the platform. The first criticality is related to evaluation of the effort needed for maintaining the ontology by plugging new classification schemes when needed with respect to the hard-coding of the entities in the source code. The rose issue concerns that the plug of a new classification scheme to the ontology requires the accomplishment of two tasks: (i) the transformation of the 3D-items (or behavioral models) classification scheme to a bottom-level ontology and (ii) the definition of the alignments between the entities modeled in the middle-level to the plugged ones. About the first task, the effort for completing it may vary based on the quality of the classification scheme. In Sect. 3, as example of the difficulties that might be found in such an activity, we presented which were the issues detected in adapting the XVR classification schema. Instead, concerning the alignment task, we discussed, always in Sect. 3, how the use of automatic ontology alignment tools may help in reducing the effort needed for completing the plug of a new classification scheme to the ontology. On the other hand, by having the entity labels hard-coded in the software, the work of migrating the code from one library to another is unsafer unless to find some development solutions that, by the way, would request an effort comparable, if not higher, with the plug of a new classification scheme.

The second criticality concerns the management of ambiguities when the plugged classification schemes are richer with respect to the vocabulary modeled in the middle-level ontology. The risk is that during the development of a scenario, the developer, through the ontology, is not able to access to all items contained in the 3D-library. However, by considering that in the use cases addressed until now in the project this event did not happen, the resolution of this weak point has been demanded as future work for the next version of the system.

Finally, by summing-up all the collected observations, we may positively answer to both research questions. We may state that the use of the ontology made the development process easier with respect of the hard coding alternative. The same perception has been reported also for what concerns the management of the behavioral models.

7 Related Work and Conclusion

In this paper, we focused on the experience of using Semantic Web techniques, and in particular lightweight ontologies, for the description of the artificial entities and their behaviors in gaming with the aim of uncoupling the description of virtual reality scenarios from their physical implementation in charge to the developers.

With respect to the literature, where ontologies are often used for a detailed description of the geometrical properties of space and objects [9], we focused more on how the description of the entities of a VR scenario can be easily represented and managed from the practical point of view. Indeed, the literature addressed such problems only marginally by focusing mainly on the use

of ontologies for managing the representation of virtual reality scenarios themselves [10,11], even if in some cases a clear target domain, like the management of information related to disasters [12], is took into account. Also the description of character behaviors have been supported by using ontologies for different purposes like as support for UML-based descriptions [13] or as a “core” set of structural behavioral concepts for describing BDI-MAS architectures [14].

However, all these works do not take into account issues concerning the practical implementations of flexible systems for building virtual reality scenarios. The proposed solution demonstrated the viability of using Semantic Web technologies for abstracting the development of virtual reality scenarios either from the point of view of the 3D-design and from the modeling of character behaviors.

References

1. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing? *Int. J. Hum. Comput. Stud.* **43**(5–6), 907–928 (1995)
2. Guarino, N., Welty, C.: Identity and subsumption. In: Green, R., Bean, C.A., Myaeng, S.H. (eds.) *The Semantics of Relationships: an Interdisciplinary Perspective*, vol. 3. Kluwer, Dordrecht (2001)
3. Ritter, F.E., Bittner, J.L., Kase, S.E., Everts, R., Pedrotti, M., Busetta, P.: CoJACK: a high-level cognitive architecture with demonstrations of moderators, variability, and implications for situation awareness. *Biol. Inspired Cogn. Archit.* **1**, 2–13 (2012)
4. Everts, R., Pedrotti, M., Busetta, P., Acar, H., Ritter, F.: Populating VBS2 with realistic virtual actors. In: Conference on Behavior Representation in Modeling & Simulation (BRIMS), Sundance Resort, Utah, 30 March–2 April 2009
5. Paternò, F.: End user development: survey of an emerging field for empowering people. *ISRN Software Engineering* **2013**, 1–11 (2013)
6. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: Methontology: from ontological art towards ontological engineering. In: Proceedings Symposium on Ontological Engineering of AAAI (1997)
7. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) *EKAW 2002. LNCS (LNAI)*, vol. 2473, p. 166. Springer, Heidelberg (2002)
8. David, J., Euzenat, J., Scharffe, F., dos Santos, C.T.: The alignment API 4.0. *Semant. Web* **2**(1), 3–10 (2011)
9. Chu, Y.L., Li, T.Y.: Realizing semantic virtual environments with ontology and pluggable procedures. In: Lanyi, C.S. (ed.) *Applications of Virtual Reality*. InTech, Rijeka (2012)
10. Bille, W., Pellens, B., Kleinermann, F., Troyer, O.D.: Intelligent modelling of virtual worlds using domain ontologies. In: Delgado-Mata, C., Ibáñez, J. (eds.) *Intelligent Virtual Environments and Virtual Agents, Proceedings of the IVEVA 2004 Workshop*, ITESM Campus Ciudad de Mexico, Mexico City, D.F., Mexico, 27 April 2004. CEUR Workshop Proceedings, vol. 97. CEUR-WS.org (2004)
11. Xuesong, W., Mingquan, Z., Yachun, F.: Building VR learning environment: an ontology based approach. In: First International Workshop on Education Technology and Computer Science, ETCS 2009, vol. 3, pp. 160–165, March 2009

12. Babitski, G., Probst, F., Hoffmann, J., Oberle, D.: Ontology design for information integration in disaster management. In: Fischer, S., Maehle, E., Reischuk, R. (eds.) Informatik 2009: Im Focus das Leben, Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Lübeck, 28 September–2 October 2009. Proceedings LNI, vol. 154, pp. 3120–3134. GI (2009)
13. Bock, C., Odell, J.: Ontological behavior modeling. *J. Object Technol.* **10**(3), 1–36 (2011)
14. Faulkner, S., Kolp, M.: Ontological basis for agent ADL. In: Eder, J., Welzer, T., (eds.) The 15th Conference on Advanced Information Systems Engineering (CAiSE 2003), Klagenfurt/Velden, Austria, pp. 16–20, June 2003. CAiSE Forum, Short Paper Proceedings, Information Systems for a Connected Society. CEUR Workshop Proceedings, vol. 74. CEUR-WS.org (2003)