KNOWLEDGE REPRESENTATION
An Ontology for Managing a Virtual Environment

Lydie Edward, Kahina Amokrane
Heudiasyc Laboratory UMR 6599 CNRS, University of Technology of Compiègne, France
Centre de Recherches de Royallieu, 60200 Compiègne, France

Domitile Lourdeaux, Jean-Paul Barthès
Heudiasyc Laboratory UMR 6599 CNRS, University of Technology of Compiègne, France

Keywords: Ontology, Virtual environment for training, Multi-agent systems, Knowledge representation.

Abstract: This paper presents an ontology developed in order to manage a virtual environment for risk prevention. This ontology represents the objects composing the environment, the agents operating in the environment and the events that can happened. In the virtual environment, different entities cohabit: virtual operators represented by cognitive agents and the learner's avatar that represents a real operator. They can interact with the objects. It is therefore useful to have on one hand a managing system that well define the framework in which the interactions or actions can be allowed and on the other hand a representation of the knowledge involve in such interactions. To do this, we combine artificial intelligence and knowledge engineering to propose agent COLOMBO. It is composed with the developed ontology and a set of reasoning rules.

1 INTRODUCTION

In our work we are interested in virtual environments with enough credibility that offer human users an attractive platform through which they can gain some experience for training and decision-making (Edward et al., 2007). For training and even more for decision-making, it is necessary to reproduce the criteria/events that are useful for the operator to decide (for example if he does not close the gate properly we must have a leak). We seek to model autonomous virtual characters interacting with avatars in such environments to perform a task at a high-risk industrial plant and show the results of their activity and the risks incurred. It is therefore useful to have on one hand a world model and a managing system and on the other hand a representation of the knowledge involve in such interactions. Above all existing approaches, we chose to develop a world model using ontology. The goal of creating such ontology is to allow users to have access to various information (*example: what are all the actions possible on an object? What is the link between object i and objet j*). The representation of the environment integrates knowledge on the objects that composed the environment but also on the activity that agents or avatars have to realized. We combine artificial intelligence and knowledge engineering to propose agent COLOMBO[1] associated with an ontology that permits us to represent the working environment (objects, agents), actions and events that can happened in the environment. COLOMBO manages the actions and interactions with the environment in the way that it determines if an action is possible and if all the conditions are satisfied to realize the action.

2 STATE OF THE ART

Kallmann proposed *Smart Objects* that integrate the information needed for creating the character's behaviors. They provided information are oriented toward the interaction between the characters and the objects (Kallman, 2004). He distinguishes several types of information: (i) intrinsic properties of the object (semantic, physics), (ii) information on interaction (actions, positions, gestures), (iii) object behavior in response of an interaction. In line with the research of Kallman, Chevallier and Querrec proposed an extension of UML 2.1 model and developed a metamodel

[1]French Translation : Ontological Creation linked to the Modeling of the Objects

called VEHA[2]. VEHA is used for modeling semantic, structural, geometrical and topological properties and agents reactive behaviors (Chevaillier et al., 2009).

3 DEVELOPED ONTOLOGY

The approaches presented above are not sufficient for our needs. The Smart Objects proposed by Kallmann fill this lake but do not let the learner to do some errors. The object behaviors are scripted and correct. The UML approach gives a good framework to specify relations between objects and actions but it is not a generic language. Furthermore, it does not gives us the opportunity to add rules on objects. Thus, in our approach, we chose ontology as an effective structure for knowledge representation : "An ontology is an explicit specification of conceptualization " (Gruber, 1993). Noy and McGuinness defined an ontology as a formal explicit description of concepts in a specific domain, properties of each concept describing various features and attributes of the concept and restrictions on properties (Noy and McGuinness, 2001). In addition to these elements, an ontology may contain different types of relations between its concepts.

Our domain knowledge is situated in the industrial activity on SEVESO sites. To construct the ontology, we used knowledge provided by ergonomists. In addition, knowledge concerning risks are provided by experts of INERIS[3]. During the process of ontology creation we paid attention to be agree on the vocabulary. The world domain is different from the activity domain. We distinguished the ontology that is made of the concepts (screw, gate, handle) and the knowledge base that is composed with the individuals or instances of the different concepts ($screw_0 5, gate_0 7, handle_0 2$). We explain in the following the main concepts and relations of our ontology (Figure 1).

3.1 Main Concepts

- Object-V3S
  The main components of the environment are the objects. They are represented in the ontology by the root concept "Object-V3S" (Figure 2). There is also subconcepts such as: opening-object (valve, door), container-object (toolbox, cupboard), tools (screwdriver, hammer, Swiss army knife). Each real object in the environment is represented by an instance in the knowledge base. During a working session, the state of a given object may change from a value to another (opened, shut, taken) and its decay characteristic (normal, rusted, broken) may change also, and thus we added respectively the attributes "state" and "status". At each moment, an object has one or several states but only one status. An object is also described by its position and its orientation in the world. We distinguish several categories of objects according to their function and utility. If the object is linked to another object (i.e a door and its handle) we called it a cognitive object otherwise it is a reactive object.

- Action-V3S
  Each autonomous agent or learner may perform one or more tasks on objects in the virtual environment. We have thus added the "Action-V3S" concept that represents all the actions that can be performed in the environment (open, close, unscrew). These actions are classified according to the objects on which they can be applied. We have for example the subconcepts: valve-action, loading-arm-action. This distinction is useful to abstract the relations between concepts. During the execution of an action, it changes from one state to another. It can be current (active), awaiting (pending), ended (finished) or in failure (failed).

- Agent
  To represent the work team (team leader, operator, manager) at a site, we added the "Agent" concept. Team members are represented by autonomous virtual agents and the avatar of the learner in the virtual environment. We have different categories of agents according to their function. It corresponds to the following subconcepts: Operator, Manager and Project Manager. An agent has a position and orientation in the environment, a toolbox containing its tools and an equipment box with its protection clothes.

3.2 Ontology Relations

In addition to the classical subsumption relation between concepts, we enriched our ontology with other (horizontal) relations. We describe below some of these relations :

- "has-main-resource" relation between an action and an object. It specifies the appropriated objects needed to realize the action. For example, the action ”loosen” on the object ”screw” has for main resource the object ”screwdriver”.

- "has-secondary-resource" relation between an action and an object. It specifies the other objects

[2]Virtual Environment supporting Human Activities
[3]Industrial Environment and Risk National Institute

that can be used to realize the action. For example, the action ”loosen” on the object ”screw” has for secondary resource a sort of screwdriver such as a ”Swiss army knife”.

- “has-target” relation between an action and an object. It means the instance of the object concerned by the action.
- “has-main-action” : relation between an action and an object. Main actions are the tools main function (i.e screwdriver : unscrew, screw).
- “has-in-hand” : relation between an agent and an object. It specifies which tools are in the agent hand.

The relation has-secondary-resource between an object and an action is created automatically at the loading of the knowledge base. We added a rule-based system that identifies according to certain constraints, the objects to link with an action. For example all the objects with a certain weight and geometry will be secondary-resource-of the action knock. The rules are specifically based on the objects properties. This system is very helpful to maintain the ontology. If we add some new concepts, with their attributes and properties we do not need to specify for which action they can be secondary resource.

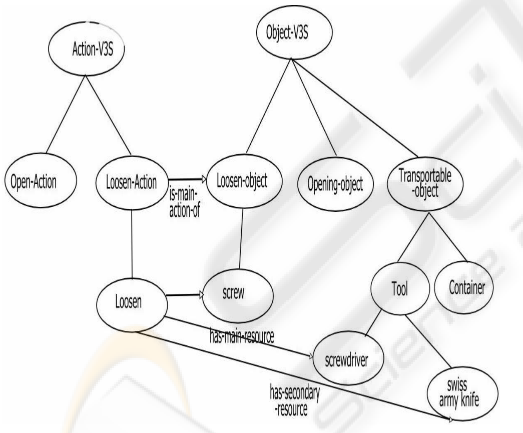
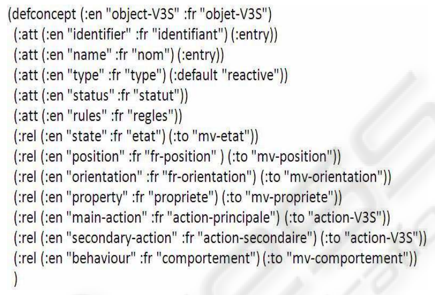


Figure 1: Ontology.

3.3 Implementation

Our ontology has been constructed with MOSS (Barthès, 2009), a modeling language implemented with the Lisp language. MOSS has a powerful engine that permits to access to the entire ontology and the knowledge base through simple or complex queries. The query system takes a formal query as input, focusing on a particular concept : the question ”Persons whose name is Dupond” gives the query (”person” (”name” :is ”Dupond”)). It then computes a set of possible candidates, and tries to validate each candidate in turn. Creating a concept is simple and can be done with the defconcept macro as follows: (defconcept ”Gate”). MOSS objects have two kinds of properties: immediate values that qualify the object, and links to other objects. The first kind is called an attribute (:att), the second kind is called a relationship or relation (:rel).



```
(defconcept (:en "object-V3S" :fr "objet-V3S")
 (:att (:en "identifier" :fr "identifiant") (:entry))
 (:att (:en "name" :fr "nom") (:entry))
 (:att (:en "type" :fr "type") (:default "reactive"))
 (:att (:en "status" :fr "statut"))
 (:att (:en "rules" :fr "regles"))
 (:rel (:en "state" :fr "etat") (:to "mv-etat"))
 (:rel (:en "position" :fr "fr-position" ) (:to "mv-position"))
 (:rel (:en "orientation" :fr "fr-orientation") (:to "mv-orientation"))
 (:rel (:en "property" :fr "propriete") (:to "mv-propriete"))
 (:rel (:en "main-action" :fr "action-principale") (:to "action-V3S"))
 (:rel (:en "secondary-action" :fr "action-secondaire") (:to "action-V3S"))
 (:rel (:en "behaviour" :fr "comportement") (:to "mv-comportement"))
 )
```

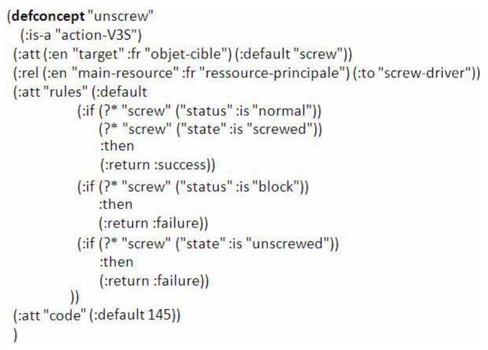
Figure 2: Concept Object-V3S.

4 RULES

We do not aim at only describing the objects and actions. But we want to let the learner and the autonomous agents to do some errors (pedagogical choice). But we avoid the errors that can not be done in real life. For example to unscrew with a hammer. It is not possible. Thus, if the learner or the agent decides to do an action, it does not mean that the action will be done. For example, if the agent wants to close the gate and the gate is already closed, the action will not be done (no changing state). Thus we add information into the objects and also into the action in order to manage how an object can change its state and when an action can be executed. We distinguish two kind of objects, reactive objects (screw) and cognitive objects (loading-arm). The reactive objects are just like reactive agent, they respond directly to a certain stimuli (an action). For example, if the learner decides to unscrew the screw, the result of the action will not depend on other parameters than the concerned object state. The cognitive objects are more complex as cognitive agents (Demazeau, 1995) (Ferber, 1995). For example, let us take the loading-arm. There is a link called ”connect” between the concept loading-arm and the concept gate. In order to remove the loading-arm, we have to be sure that the loading-arm’s handle is turned and that the appropriated gate is unlocked. To manage these different situations, we add two kind of rules into the cognitive objects and into the actions : execution rules and transition rules.

- Execution rules

In response to the agent or learner actions the environment should be able to respond correctly. If the agent wants to do an action we need to know if this action is possible. This role is done by agent COLOMBO. When the agent or the learner wants to do an action, COLOMBO receives a query. Then it verifies the execution rules linked to the action (Figure 3). If the conditions are satisfied then the action is possible and an animation is started in the environment.

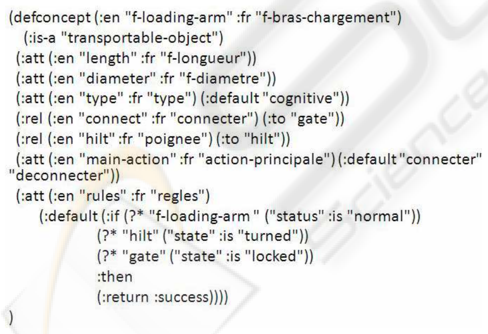


```
(defconcept "unscrew"
   (:is-a "action-V3S")
  (:att (:en "target" :fr "objet-cible") (:default "screw"))
  (:rel (:en "main-resource" :fr "ressource-principale") (:to "screw-driver"))
  (:att "rules" (:default
            (:if (?* "screw" ("status" :is "normal"))
                 (?* "screw" ("state" :is "screwed"))
                 :then
                 (:return :success))
            (:if (?* "screw" ("status" :is "block"))
                 :then
                 (:return :failure))
            (:if (?* "screw" ("state" :is "unscrewed"))
                 :then
                 (:return :failure))
           ))
  (:att "code" (:default 145))
  )
```

Figure 3: Concept action unscrew.

- Transition rules

  The transition rules are the conditions that permit an object to change its state. They are stored in the object concept (Figure 4). As we said above, the realization of an action does not necessary implies that the target object change its state. Thus, only if they transition rules are satisfied, COLOMBO sets the new value of the object state.



```
(defconcept (:en "f-loading-arm" :fr "f-bras-chargement")
   (:is-a "transportable-object")
  (:att (:en "length" :fr "f-longueur"))
  (:att (:en "diameter" :fr "f-diametre"))
  (:att (:en "type" :fr "type") (:default "cognitive"))
  (:rel (:en "connect" :fr "connecter") (:to "gate"))
  (:rel (:en "hilt" :fr "poignee") (:to "hilt"))
  (:att (:en "main-action" :fr "action-principale") (:default "connecter"
"deconnecter"))
  (:att (:en "rules" :fr "regles")
      (:default (:if (?* "f-loading-arm " ("status" :is "normal"))
                 (?* "hilt" ("state" :is "turned"))
                 (?* "gate" ("state" :is "locked"))
                 :then
                 (:return :success))))
)
```

Figure 4: Concept object loading-arm.

5 CONCLUSIONS

In this paper we present the development of an ontology for managing a virtual environment. Our ontology is not only a formal description of the concepts of a domain but we also aim at building a intelligent system that decides : (i) if the action is possible, (ii) if the object can change its state. To reach this goal, we proposes two kind of rules : (i) execution rules defined in the action concept and (ii) transition rules defined in the cognitive object concept. The next step of our work will be to enrich the system and to found solutions for the maintenance of the ontology.

ACKNOWLEDGEMENTS

Authors want to thank the ergonomists of the University of Paris Descartes; INERIS for providing the necessary information. We also thank the persons who worked with us on the development of the ontology and the virtual environment.

REFERENCES

Barthès, J.-P. (2009). OMAS - A Flexible Multi-Agent Environment for CSCWD. *CSCWD*.

Chevaillier, P., Querrec, R., and Septseault, C. (2009). Veha : Un méta-modèle d'environnement virtuel informé et structuré. *Revue des Sciences et Technologies de l'Information, série Techniques et Sciences Informatiques (RSTI-TSI).*

Demazeau, Y. (1995). From interactions to collective behaviour in agent-based systems. *European Conference on cognitive sciences*.

Edward, L., Lourdeaux, D., Lenne, D., Barthès, J., Burkhardt, J., and Camus, F. (2007). V3S: A Training and Decision Making Tool to Model Safety Interventions on SEVESO Sites. *VRIC*.

Ferber, J. (1995). Les systèmes multi-agents - vers une intelligence collective. *Paris : InterEditions*.

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5:199–220.

Kallman, M. (2004). Interaction with 3-d objects. *In Magnenat-Thalmann, N. and Thalmann, D. editors, Handbook of Virtual Humans, chapter 13, pages 303322.*

Noy, N. and McGuinness, D. (2001). Ontology development 101 : A guide to creating your first ontology. *Technical Report KSL-01-05 and SMI-2001-0880*.