International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:3, 2007

An Interactive Ontology Visualization Approach for the Networked Home Environment

Ilkka Niskanen, Jarmo Kalaoja, Julia Kantorovitch, and Toni Piirainen

Abstract—Ontologies are broadly used in the context of networked home environments. With ontologies it is possible to define and store context information, as well as to model different kinds of physical environments. Ontologies are central to networked home environments as they carry the meaning. However, ontologies and the OWL language is complex. Several ontology visualization approaches have been developed to enhance the understanding of ontologies. The domain of networked home environments sets some special requirements for the ontology visualization approach. The visualization tool presented here, visualizes ontologies in a domain-specific way. It represents effectively the physical structures and spatial relationships of networked home environments. In addition, it provides extensive interaction possibilities for editing and manipulating the visualization. The tool shortens the gap from beginner to intermediate OWL ontology reader by visualizing instances in their actual locations and making OWL ontologies more interesting and concrete, and above all easier to comprehend.

***Keywords*—**Ontologies, visualization, interaction.

I. Introduction

ARTIFICIAL intelligence refers to programs and procedures that are able to automatically solve problems heretofore solved by humans [1]. To enable this automated problem solving, recent work in artificial intelligence has been concentrated on exploring ways to specify content-specific agreements for the sharing and reuse of knowledge among software entities [25]. Ontologies have become a popular research topic among several artificial intelligence research communities because of their ability to provide a shared and common semantic understanding [2].

A commonly agreed definition of ontology, made by Gruber [3] is the following: “An ontology is an explicit and formal specification of a conceptualisation of a domain of interest”. Furthermore ontology is defined as a controlled vocabulary that describes objects and the relations between them in a formal way; ontology resembles faceted taxonomy but uses richer semantic relationships among terms and attributes, as well as strict rules about how to specify terms and relationships [4], [5]. Ontologies represent a shared meaning of a domain and they can be used to describe almost any kind of domain explicitly.

Manuscript received April 15, 2007.

VTT Technical Research Center of Finland, Kaitoväylä 1 Oulu, Finland (e-mail: {Ilkka.Niskanen, Jarmo Kalaoja, Julia Kantorovitch, Toni Piirainen}@vtt.fi; phone: +358 20 722 2030).

Ontologies are broadly used in the context of the networked home environments. As Gu et al. [25] have pointed out, with ontologies it is possible to define and store context information, as well as to model different kinds of physical environments. Ontologies are central to networked home environments, as they carry the meaning. With ontologies some of the most important problems in the development of the pervasive computing environments can be overcome [26]. OWL semantic mark up language was created for publishing and sharing ontologies and it provides mechanisms for creating all the components of an ontology: concepts, instances, relations and axioms [8].

However, working with ontologies and the OWL language is often complex. Looking at an OWL ontology for the first time can be overwhelming and the gap from beginner to intermediate OWL ontology reader is cumbersome. Information visualization by definition is the process of turning abstract data into a visual shape easily understood by the user, making it possible for him/her to generate new knowledge about the relations between the data [9]. Visualization can also be the key to better understanding of the data contained by ontologies. A visual version of an ontology allows users to visually follow a concept to its nearest neighbours or analyze the overall space for interesting related or unrelated concepts. [10].

Several ontology visualization approaches have been delivered by research community in some past. Common feature of these approaches is that they are graph based and domain independent. However, the graph based visualization is not capable of effectively visualizing every kind of data. As Wehrend & Lewis [11] have mentioned, the chosen visualization technique should always be relevant to the given problem and support the user’s goal in viewing the representation. Another shortcoming of these approaches is their limited interaction features. In order to gain full understanding of the visualized data, users should be able to interact with the visualization [6].

The main research problem of this study is to find out how can ontologies be visualized, particularly in the domain of networked home environments, and how can interactive visualization elements be included in the visualization. To answer this problem, three research questions must be answered:

1. What special requirements does the networked home domain set for the visualization approach?
2. How should context information be presented and managed in the visualization?
3. How should the interaction be implemented and managed?

In this study a visualization tool called VantagePoint is constructed. VantagePoint is able to interactively visualize ontologies particularly in the domain of networked home environments. Special attention is given to the interaction between the user and the visualization. The constructed approach should act as an example of how the actual research problem can be solved. The approach visualizes OWL ontologies graphically, making them more interesting and concrete, and above all easier to comprehend.

This paper is structured as follows. Chapter 2 presents some of the tools currently available for ontology visualization identifying their shortcomings and benefits. The developed ontology visualization tool is described in more detail in Chapter 3 and finally, the conclusions are presented in Chapter 4.

II. The State of Art in Ontology Visualization Tools

Currently there exist several approaches to visualizing ontologies. Many of these approaches are embedded in tools that support the development process of ontologies. The intended users of these tools are ontology engineers that need to get an insight into the complexity of the ontology. Therefore, these tools employ schema visualization techniques that primarily focus on the structure of the ontology, i.e. its concepts and their relationships [13]. The examples of such ontology visualization approaches are Ontoviz [14] and Jambalaya [15], which are visualization components of ontology editor Protégé [12], and Cluster Map [13], which is a component of navigation engine Spectacle.

Common to these approaches is that they are domain independent and graph based. Ontologies are visualized only by representing their entities with nodes and the relationships between entities with arcs. Furthermore, these approaches contain quite limited graphical editing operations. Only TGVizTab offers real possibilities to graphically edit the visualization. In the other two approaches, the interaction operations are restricted to navigation and the selection of features to be visualized. Furthermore, considering how important a role queries play in data analysis, the search features in these approaches are surprisingly modest.

The tool presented by this research addresses the above enumerated shortcomings by visualizing the semantic information models in a realistic and concrete manner. It allows users to dynamically manipulate the underlying models through various graphical editing operations. By visualizing environments realistically, users are able to see the different operations as in real life.

III. The Most Important Requirements for the Approach

The domain of networked home environments sets some special requirements for the visualization approach. In this chapter those requirements are described.

As discussed above, the ontologies considered in this study model different kinds of physical environments. Thus, it was thought important to present contextual information and spatial relationships effectively in the visualization. Another requirement was that the graphical appearance of the visualization should be realistic. Realism breeds the expectation of accuracy, reliability and authority in the representation, especially when considering computer visualizations which aspire to simulate real environments [27].

The next requirement highlighted the importance of interactive elements. The interaction between the user and the visualization helps us to understand the visualization better [6] and thus VantagePoint was designed to offer such editing operations as adding, moving and deleting of instances. These operations were defined not just to enable the editing of the visualization itself, but also to give direct access to the underlying ontological data behind the visualization.

The ability to view the data from different perspectives and from different angles enhances understanding of the data [16]. To realize this, a requirement for multiple angles and perspectives was added to VantagePoint. By defining this requirement, it was assured that users would get a good overall picture of the data, as well as being able to perform accurate editing operations.

The last requirement was to ensure that VantagePoint offers extensive possibilities to construct and execute queries. As was concluded in Chapter 2, in the existing ontology visualization approaches the search features were surprisingly modest. VantagePoint addresses this shortcoming by providing two different query methods: a graphical query and a free query. With the graphical query option inexperienced users are also able to execute queries without having any particular knowledge of the ontology query languages RDQL [28] or SPARQL [29]. With the textual query option, advanced users are able to define their own query statements without any constraints.

IV. The Main Components of VantagePoint

VantagePoint core contains the most important functions of the software: model visualization, model manipulation and model simulation. These functions are implemented in Java and controlled through Swing GUI. The VantagePoint core is presented in Fig. 1.

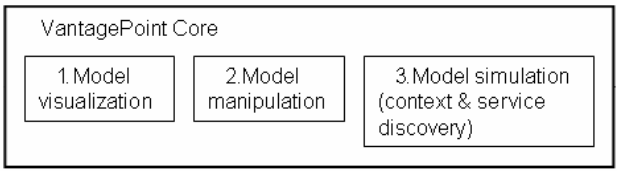

Fig. 1 The main components of the VantagePoint

1. As discussed before, VantagePoint forms semantic models of networked home environments. The *Model visualization* component is responsible for the visualization of these models. It reads OWL files and searches for individuals that belong to predefined VisualComponent class. Only individuals that belong to the VisualComponent class are visualized, all other data that is irrelevant or impossible to visualize is ignored. The VisualComponent class is divided into two separate subclasses - *Item* and *Area*. All things visualized fall into these two classes. If the ontology does not contain the class VisualComponent, it cannot be visualized with VantagePoint. In such cases a blank visualization is created and users can examine the ontological model through the query features provided by VantagePoint.

2. The *Model manipulation* component enables changes to be made to the visual representation of the model. The manipulation consists of editing operations such as removing, adding, moving and rotating of instances. Whenever editing operations are made in the visualization the ontological data changes accordingly.

3. The *Model simulation* component defines ways of implementing context and other discovery operations. These operations are carried out by utilizing the SPARQL and RDQL query interfaces provided by VantagePoint.

V. Architectural Framework

The Model-View-Controller (MVC) framework was selected as a starting point for the architectural design of VantagePoint. The MVC framework is a widely used architectural approach for interactive applications. It divides functionality between objects involved in maintaining and presenting data to minimize the degree of coupling between the objects [17]. In the Model-View-Controller architecture, objects of different classes take over the operations related to the application domain (the model), the display of the application's state (the view), and the user interaction with the model and the view (the controller) [18]. Modularity of components has enormous benefits, especially when building interactive applications. Isolating functional units from each other as much as possible makes it easier to understand and modify each particular unit, without having to know everything about the other units. This three-way division of an application entails separating the parts that represent the model of the underlying application domain from the way the model is presented to the user and from the way the user interacts with it [17].

The architectural design of the VantagePoint is a slightly modified version of the Model-View-Controller framework. Contrary to the Model-View-Controller paradigm, the model class do not notify the views when it changes. This is mainly because the model class in VantagePoint architecture is adopted from the class library provided by Jena [30] and therefore it was considered better not to implement any new functionality in the OntModel class, but to leave it as it was. Instead, a new class called VPEditor was created to implement some of the functionalities that would normally belong to the model class in the MVC architecture. Furthermore, it was decided to implement the user interface elements of the application in the VPEditor class. In this way the WorldManager class, which acts as a controller class in VantagePoint architecture, could be maintained as a simple interface to the model. This was considered to increase the versatility and the reusability of the approach. To illustrate the functionality of the architectural framework of the approach, a detailed description of VantagePoint's interaction cycle is presented in Fig. 2.

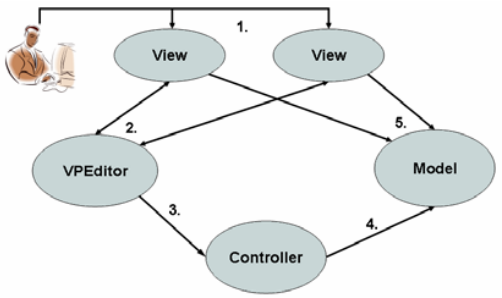

Fig. 2 A standard interaction cycle of VantagePoint

1. The basic interaction cycle starts when the user performs an editing operation in some of the views. For example, the user may change the location of an instance by dragging it to another position.

2. The VPEditor class 'listens' to the views and tracks the changes occurring in them.

3. VPEditor forwards the necessary information about the editing operation to the controller class.

4. The controller class (WorldManager) acts as an interface to the model (OntModel) and changes the model according to the editing operation.

5. Finally, VPEditor calls the views' update methods to set the views in sync with the current state of the model.

As can be seen from Fig. 2, different components in VantagePoint architecture implement their own strictly defined responsibilities. Separating responsibilities among model, view, and controller objects reduces code duplication and facilitates handling of the data, whether adding new data sources or changing data presentation, as business logic is kept separate from data [18].

VI. The VantagePoint

The software tool presented in this study allows to interactively visualize ontological models particularly in the domain of networked home environments. The approach is written in Java and it uses Jena interface to manage OWL

ontologies, and Java 2D graphics to visualize them. Rather than visualizing the abstract structural relationships of ontology classes and instances, VantagePoint presents contextual information and spatial relationships effectively by visualizing instances in their actual locations. VantagePoint also offers the possibility to view the visualization from multiple angles and with different perspectives and it exploits both two- and three-dimensional views.

The edit view is a 2D 'ground plan' view of the ontology that has been visualized, whereas the isometric view visualizes ontologies more impressively from a three-dimensional perspective. The purpose of the two-dimensional edit view is to enable more accurate editing operations. In general, 2D views are considered better for navigating and measuring distances precisely, establishing precise relationships and performing spatial positioning [19], [20]. As can be seen from Fig. 3, the appearance of the edit view is somewhat rough. Items are represented with symbols, which include a textual description of the item, and an arrow indicating the current direction of the item. The edit view is presented from a bird's eye perspective, which does not exploit the three-dimensional representation. Instead, it enables the possibility to accurately create areas with exact measurements and locate items in their correct positions.

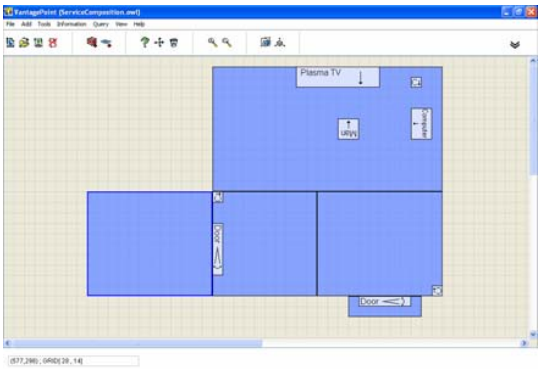

Fig. 3 A VantagePoint screenshot where the edit view is visible

In the isometric view the environment is presented from an isometric projection which offers a better general view of the house. However, the isometric view does not offer as accurate editing and adding operations as the two-dimensional edit view. Three-dimensional displays are said to be good for gaining an overview of a 3D space, understanding 3D shape, and navigating approximately [21]. The isometric view is presented in Fig. 4.

The isometric projection was selected as a representation method, because it allows users to have a general view of the visualized world at a glance. It is also stated that by using the isometric projection, spatial relationships between objects can be seen within wide environments [22]. Furthermore, the isometric projection has proven its effectiveness in visualizing three dimensional spaces in such popular computer games as 'The Sims' [23] and 'SimCity' [24].

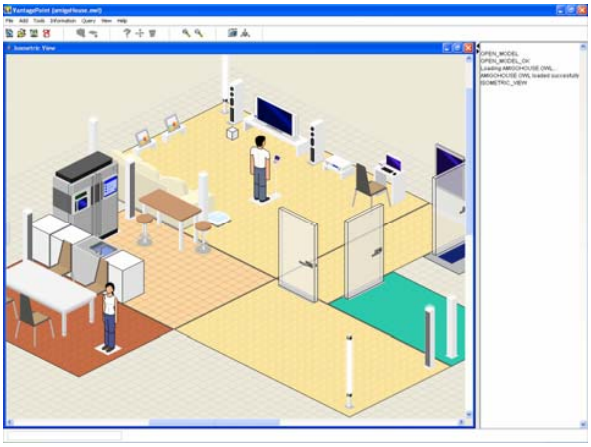

Fig. 4 A VantagePoint screenshot where the isometric view is visible

The exploitation of both visualization types, 2D and 3D, was thought to be a necessity, since VantagePoint was required not only to visualize ontologies, but also to offer means to accurately edit them.

VII. The Interactive Features

As discussed, one of the most important requirements of VantagePoint was to offer extensive interaction possibilities to the user. In this chapter the interaction operations provided by VantagePoint are described in more detail.

To begin with, VantagePoint offers operations for adding and removing of instances. In VantagePoint, instances are either areas or items. Areas are used to represent different kinds of spaces, like rooms and hallways in the house model, whereas items represent either devices or persons and they are represented with realistic 3D icons or 2D symbols that are supposed to make items recognizable. Both areas and items can be added and removed through simple graphical operations.

The items are added by selecting an item from the text list that enumerates all the items offered by VantagePoint. After the item has been selected, the location for the item can be determined by dragging it to a desired position.

The adding of areas is performed by assigning the corner points of the area. Once the desired points have been selected (three at least), the area must be named. It is also possible to determine a floor material for the area. Different floor materials are represented with different textures in the isometric view (see Fig. 4).

Delete operations are performed simply by selecting the desired instance and pressing the 'garbage can' button in the control panel. As the selected instance disappears from the screen, it is also removed from the ontological model. If an area is deleted, all areas and instances that were contained by this area are also deleted.

In addition, to the operations described above, VantagePoint offers interaction operations such as moving of instances, printing of models and getting additional information about instances.

In VantagePoint all visualized elements are movable and the moving operation is executed simply by dragging an instance to a new location. The printing operation enables the visualized model to be printed in a textual form. In this way it is possible to examine the structure of the OWL file and see how the changes made in the visualization have affected the model. The final interaction operation described here is called "getting additional information about instances". This operation enables certain additional information about selected instances to be quickly obtained.

As it was earlier discussed, queries are an effective way of retrieving data from ontologies and in the existing visualization approaches the query construction features are quite restricted. VantagePoint stands out from other ontology visualization approaches by providing extensive support for query construction. It supports two ontology query languages SPARQL and RDQL. In addition, VantagePoint offers two distinguished query methods: a graphical query and a free query.

Graphical querying in VantagePoint means that users can define queries that will be executed when an instance is being clicked on in the visualization. By means of graphical querying it possible to retrieve information about, for example, what services are offered in a certain area. VantagePoint provides also a possibility to save query sets in text file to be reused later and thus the same queries can be executed through graphical user interface with minimal knowledge about the query languages needed.

The queries that will be executed when an instance is being clicked can be defined in the query settings dialog box, which is presented in Fig. 5.

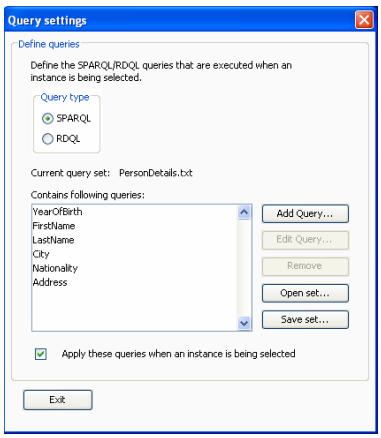

Fig. 5 A dialog to define queries executed in graphical querying

In the free query, the queries to be executed are not restricted in any way. With the free query it is possible to retrieve any kind of information from the model, even data that could not have been visualized.

As presented in Fig. 6, the free query is constructed by writing the query statement in to the upper text area and pressing the execute button. Results will appear to the lower text area.

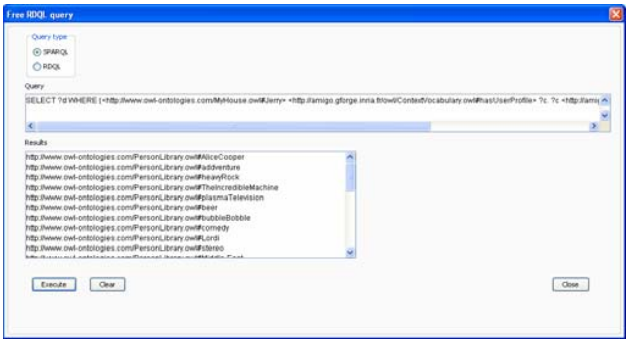

Fig. 6 The dialog for defining free queries

VIII. Conclusion

This paper presents a prototype software tool called VantagePoint. VantagePoint is able to interactively visualize ontologies, particularly in the domain of networked home environments. VantagePoint fills a certain niche that is not supported by the other ontology visualization tools by visualising ontologies in a domain specific way. VantagePoint exploits such representation techniques that are particularly effective at showing the physical elements of networked home environments, and spatial relationships between entities. Currently, there are no similar approaches available.

VantagePoint provides two distinct views to examine the visualized models. The two-dimensional edit view is a "ground plan" view of the environment being visualized. This view enables effective and accurate discovery of the spatial relationships between different elements, perception of the exact positions of various instances, and accurate editing operations. The isometric view represents environments impressively from an isometric perspective. This 3D view enables to obtain a better overall picture of an environment.

Furthermore, VantagePoint provides extensive interaction possibilities. With various editing operations users are able to edit both the visualization and the underlying ontological model. Through these operations, users are able, for example, to add new instances and thus graphically create their own semantic models from scratch. In addition, VantagePoint provides extensive query construction possibilities to extract any kind of information from the visualized model.

VantagePoint helps people who are not familiar with ontologies and the OWL language. It provides an easy access to the complex world of ontologies and OWL language. VantagePoint shortens the gap from beginner to intermediate OWL ontology reader by visualizing ontologies realistically and making OWL ontologies more interesting and concrete, and above all easier to comprehend.

References

[1] R. J. Schalkoff, "Artificial Intelligence: An Engineering Approach", McGraw-Hill, Singapore, 1990.

[2] D. Fensel, I. Horrocks, F. van Harmelen, D. McGuinness, and P. F. Patel-Schneider: OIL: Ontology Infrastructure to Enable the Semantic Web, IEEE Intelligent Systems, 16(2), 2001.

[3] T. Gruber, A translation approach to portable ontologies. Knowledge Acquisition 5(2):199–220, 1993. http://ksl-web.stanford.edu/KSL_Abstracts/KSL-92-71.html
[4] M. Uschoold, M. Cruninger, Ontologies: Principles, Methods and Applications, Knowledge Engineering Review, 2, 1996.
[5] L. Xie, Y. Zheng, B. Shen, Ontology Construction for Scientific Visualization, imsccs , pp. 778-784, 2006.
[6] M. Lanzenberger, J. Sampson, AlViz - A Tool for Visual Ontology Alignment, iv , pp. 430-440, 2006
[7] E.M. Petriu, N.D. Georganas, D.C. Petriu, D. Makrakis, V.Z. Groza, Sensor-based information appliances, IEEE Instrumentation and Measurement Magazine 3 (4) , 31–35., 2000
[8] J Davies, R Studer, P Warren - Semantic Web Technologies: Trends and Research in Ontology-based Systems 2006 - John Wiley & Sons p.4
[9] R. Spence, Information Visualization. Addison- Wesley. 2001.
[10] D.E O'Leary, Using AI in knowledge management: knowledge bases and ontologies. IEEE Intelligent Systems, 34-39, May/June 1998.
[11] R. Wehrend and C. Lewis. A problem-oriented classification of visualization techniques. In Proceedings of the First IEEE Conference on Visualization: Visualizaiion 90, pages 139-143. IEEE, Los Ahunitos, CA, October 1990.
[12] Protege ontology editor and knowledge-base framework, http://protege.stanford.edu/
[13] C. Fluit, M. Sabou, and F. van Harmelen. Ontology-based Information Visualisation. In V. Geroimenko, editor, Visualising the Semantic Web. Springer, 2002.
[14] OntoViz ontology visualization plug-in for Protégé, http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz
[15] Jambalaya ontology visualization plug-in for Protégé http://www.thechiselgroup.org/jambalaya
[16] J.X. Chen, D. Rine, and H.D. Simon, “Advancing Interactive Visualization and Computational Steering,” IEEE Computational Science and Engineering, vol. 3, no. 4, pp. 13-17, 1996.
[17] G.E. Krasner and S.T. Pope, “A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 system,” Journal of Object Oriented Programming, vol. 1, no. 3, pp. 26-49, 1988.
[18] Singh, I., Stearns, B., Johnson, M., et al.: Designing Enterprise Applications with the J2EE Platform, 2nd Edition. Addison-Wesley, 2002
[19] M. Tory, A. E. Kirkpatrick, M. Stella Atkins, and T. MÄoller. Visualization task performance with 2D, 3D, and combination displays. IEEE Trans.Visual. Comput. Graphics, 12(1), pp. 2-13, 2006.
[20] M. St. John, M.B. Cowen, H.S. Smallman, and H.M. Oonk, “The use of 2D and 3D displays for shape-understanding versus relative-position tasks,” Human Factors, vol. 43, no. 1, pp. 79-98, 2001.
[21] L.R. Wanger, J.A. Ferwerda, and D.P. Greenberg. Perceiving spatial relationships in computergenerated images. IEEE Computer Graphics and Applications, 12(3), pp. 44-58, May 1992.
[22] C. Fernández-Vara, J. Zagal, and M. Mateas, Evolution of Spatial Configurations In Videogames in International DiGRA Conference 2005, Vancouver, Canada, 2005.
[23] The official web page of the game The Sims, http://thesims.ea.com/
[24] The official web page of the game SimCity, http://simcity.ea.com/
[25] T.R. Gruber, Toward principles for the design of ontologies used for knowledge sharing. Int. J. Hum. Comput. Stud. 43, 5/6, pp. 907–928, 1995.
[26] A. Ranganathan, R. E. McGrath, R. Campbell, and D. M. Mickunas. Ontologies in a pervasive computing environment. Workshop on Ontologies in Distributed Systems, IJCAI 2003, 2003.
[27] D. T. Luymes, The rhetoric of visual simulation in forest design in Sheppard S. R Forest and Landscapes, International Union of Forest Research (IUFRO), CABI International, UK 2000.
[28] http://www.w3.org/TR/rdf-sparql-query/
[29] http://www.w3.org/Submission/RDQL/