Semantic information visualization

Martin Klima (xklima@fel.cvut.cz), Pavel Halabala (halabab@fel.cvut.cz), Pavel Slavik (slavik@fel.cvut.cz)
Czech Technical University in Prague, Czech Republic

Abstract

In this work we will introduce a system for semantic information visualization and interactive manipulation implemented at CTU Prague. Explicit semantic information is necessary for automatic data processing. Commonly used data formats do not natively include such semantic information. The semantics must be therefore supplied in a separate file. The goal of our system is to enable creation of such semantic metadata for adaptation of multimedia content for use on mobile devices. The adaptation should modify the audio/video and vector graphics data in a logical manner to fulfill the user's requirements while limiting the system resources necessary for data transport and rendering.
The data themselves are designed for representation of the visual aspect only, therefore an additional semantic layer must have been added in a form of external metadata. This is achieved by the introduced system.
The metadata creation itself is an interactive process where the human must be present. Nevertheless the system may support the process by appropriate visualization and assemble tools.
In our case, we have concentrated on adaptation of complex 3D (VRML) and 2D (SVG) scenes for browsing and annotating on mobile devices. The named formats need to be adapted with the use of additional semantic metadata in MPEG-7 format.
We are introducing a tool for creation of the MPEG-7 description. The system supports direct interconnection between the data (VRML) and the semantic visualization. Thanks to extendable system core supporting a plug-in mechanism we can easily add new visualization components and data formats.
The implemented 3D visualization is fully interactive. For more efficient manipulation we have created a set of supportive tools for visualization of semantic relations, relation iteration, filtering and reasoning.
The functionality was successfully tested and is currently used in the Mummy Mobile Knowledge Management project (IST-2001-37365).

Introduction

The are many matured graphical data formats that can very well deal with any kind of 3D or 3D graphics. We can manipulate the graphical objects in any imaginable way. The commonly used graphical formats like VRML or DXF are designed to express the geometrical and visual aspects of the modeled scene. Due to the nature of these data formats, the internal file structure does not allow to include additional non geometrical data that would enhance the scene topology.
Although in many cases the geometrical information is exactly what we need, in some cases we want to manipulate the scene in a way that takes into account additional semantic meaning. This can be easily solved by either extending the existing data formats to contain the

semantic information or by adding the information into an external proprietary data file that is linked on the application level with the graphical data.

Semantic Data Use Case

In our primary use case, the user is located in a mobile environment, for example on a large construction site. The user is equipped with a PDA (Personal Digital Assistant) device with a wireless connectivity. The user needs to solve some problem that has just arisen. He connects to the central server and requests a 3D scene of the building he is currently located in. On the server side, the central database is searched for the document and for its semantic description. When the document is found it passes through a process called *adaptation*. In this process the data are modified to be renderable on the particular mobile device. The modification may consist of (in case of the 3D data) data simplification, removal or simplification of textures, removal or modification of scripts and animations, filtering out of unnecessary objects, specification of level of detail and more. The geometrical data in this case are not sufficient to proceed the adaptation. Therefore, the additional semantic information is used with advantage. The system can for example use it for filtering the relevant objects in the scene and determining the level of details that should be used to display them. An example of adaptation of a complex 3D scene of a building is shown in Figure 1. The user has in this case specified that he wants to see all objects on the way from point A to point B.

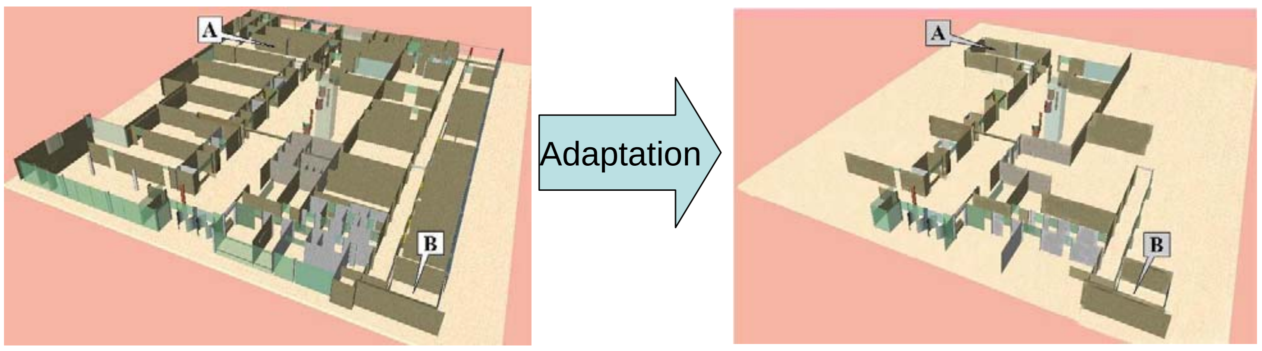

Figure 1 3D Scene Adaptation Example

All objects along the shortest path computed using the semantic metadata are preserved. Also objects that have some logical relation to the path are preserved. All other objects are removed from the scene. Both the file size and the rendering time decreases dramatically making it possible to browse on the mobile device.

Problem Specification

There are several problems when we want to use the semantics. The first problem is that the graphical data formats commonly used in the 3D and 2D modeling do not natively contain the semantic information. They are not prepared for inclusion of such information and are difficult to extend. There are several standardized data formats like EXPRESS or IFC that can contain both the geometrical and the semantic information but these formats are not commonly used and are not supported by the mobile platforms. For our introduced scenario we need to find a solution that will preserve the original data (VRML) and link it with external semantics.

When we want to take advantage of the semantic information, we need to know its internal structure and need to have a tool for either deriving the semantics from some sources or to explicitly create it using a semantic editor.

Our Solution

We were looking for a solution that would enable usage of VRML data format as the container for 3D graphics and the semantic description stored in an external standardized data format. We were especially looking for a data format capable to describe relations between objects, object and relation description. From the existing standardized formats we have chosen the MPEG-7. The advantages of this format are summarized in the following bullets:

- XML based – easy to parse and validate
- Standardized ISO/IEC developed by MPEG (Moving Picture Experts Group)
- Especially designed for description of multimedia content
- Defines a basic set of objects, their types and relations
- Easily extensible

The MPEG-7 format is in many aspects similar to other semantic formats. Therefore we can abstract from the particular language and understand the semantic as a general graph consisting of a number of nodes and relations. Each node represents either an existing object from the real world (wall, door) or can represent an abstract object (room). Relations between the nodes are represented by oriented edges. The relation can be of hierarchical type (is component of, contains) or of different type (connects to). The MPEG-7 format is intended for description of multimedia content. The relations and properties are given by the MPEG-7 standard which can be on one hand a limiting factor, on the other hand it helps us to create an editing tool and to guide the user during the creation process.

The graph visualization

We can understand the semantics as a general graph and therefore the visualization can have many forms. In our scenario, the user is usually interested into the hierarchical relations within the graph. These relations form a tree which is easy to understand and visualize. In case we organize the objects in the tree into further logical groups that we will call levels of abstraction, we can obtain very well readable graphs.

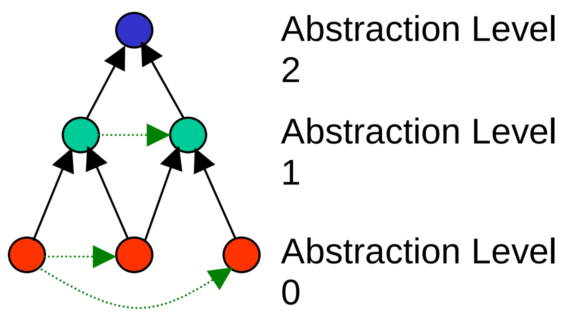

Figure 2 Abstraction levels

In Figure 2 is an example of a hierarchical structure formed by the relations (solid line arrows). Relations of other type are depicted by dashed arrow. If we only consider the solid line arrow skeleton, we get a tree structure that can be organized into individual abstraction levels. For example the abstraction level 0 can represent individual walls in a building, abstraction level 1 can represent rooms in the building and the abstraction level 2 can represent the building itself. Visually is such structure well organized and easy to manipulate. Problems arise when the number of nodes increases and the number of relations of non-hierarchical type grows. In this case a visualization as seen in Figure 2 can get unreadable.

We have decided to extend the visualization of the graph into the 3D space which has brought new possibilities of how to visualize it. The navigation in a 3D space is usually difficult for the user who can get lost or miss manipulate the graph geometry. Therefore we have combined the advantages of 3D space with a number of 2D subspaces where the user can easily navigate.

In this case each abstraction level is represented by a plane and nodes belonging to it can move within this 2D space only.
The visualization is shown in Figure 3 and can be summarized like this:

- Each plane could have arbitrary shape. For simplicity, we will assume flat planes (rectangular polygons) only.
- When adding nodes into the semantic graph, for each node must be specified the plane it belongs to. The planes can be thought of as containers.
- Node movement is restricted on the surface and the border of the plane. Later, the nodes can be moved among planes, in order to reorganize the logical layout of the graph.
- Hierarchical relations should head from one plane to another
- Non hierarchical relations should be placed within individual planes only

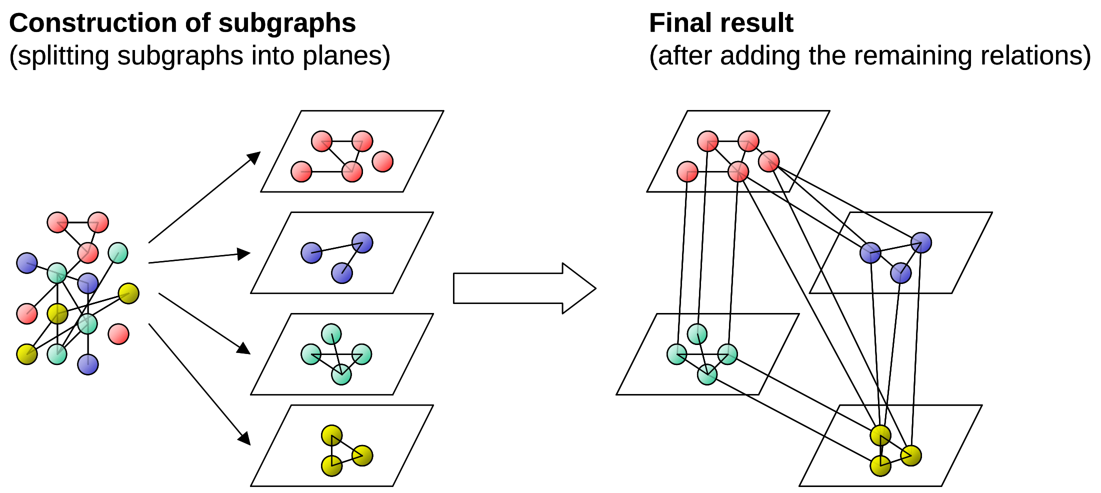
Construction of subgraphs
(splitting subgraphs into planes)
Final result
(after adding the remaining relations)

Figure 3. Semantic graph creation

As a result we get a set of planes distributed in the 3D space. Each plane can be moved, rotated, or zoomed independently from the others, while the position of the nodes on the surface is transformed accordingly to preserve their relative distances.
The logical layers represented by the planes can be understood as 2D graphs, because the degree of freedom of nodes is reduced from 3 to 2 (the nodes can be moved tangentially along the surface represented by 2 vectors, but not in perpendicular direction).
As can be seen from the previous example, it is easier to understand the topology and the meaning of the subgraph when it is visualized together with the underlying plane. The usage of the plane increases the visual perception of compactness of the subgraph.
We will demonstrate the mechanism on an example. The example scene is composed of four rooms connected by doors (see Figure 4). The rooms are represented by surrounding walls in red color and connections among them by yellow arrows. Semantic graph representations of the scene are constructed in the next two figures. The semantic graph is split into layers (planes) according to following criteria:

- Level of abstraction – abstract / concrete classes (Figure 5)

- Classes – Each class and its instances are put into a different plane (Figure 6)

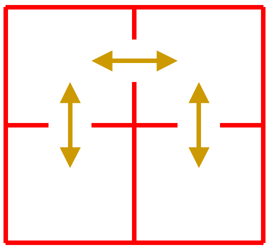

Figure 4. Test scene of a building

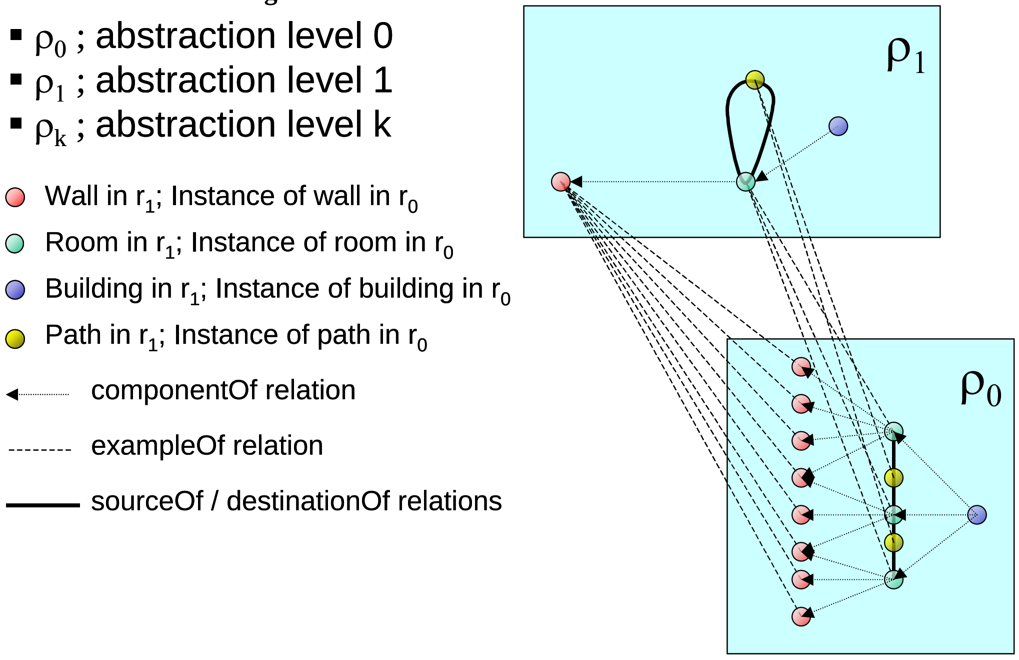
ρ_0 ; abstraction level 0
ρ_1 ; abstraction level 1
ρ_k ; abstraction level k
Wall in r_1; Instance of wall in r_0
Room in r_1; Instance of room in r_0
Building in r_1; Instance of building in r_0
Path in r_1; Instance of path in r_0
componentOf relation
exampleOf relation
sourceOf / destinationOf relations
ρ_1
ρ_0

Figure 5. Semantic graph split into layers according to the level of abstraction

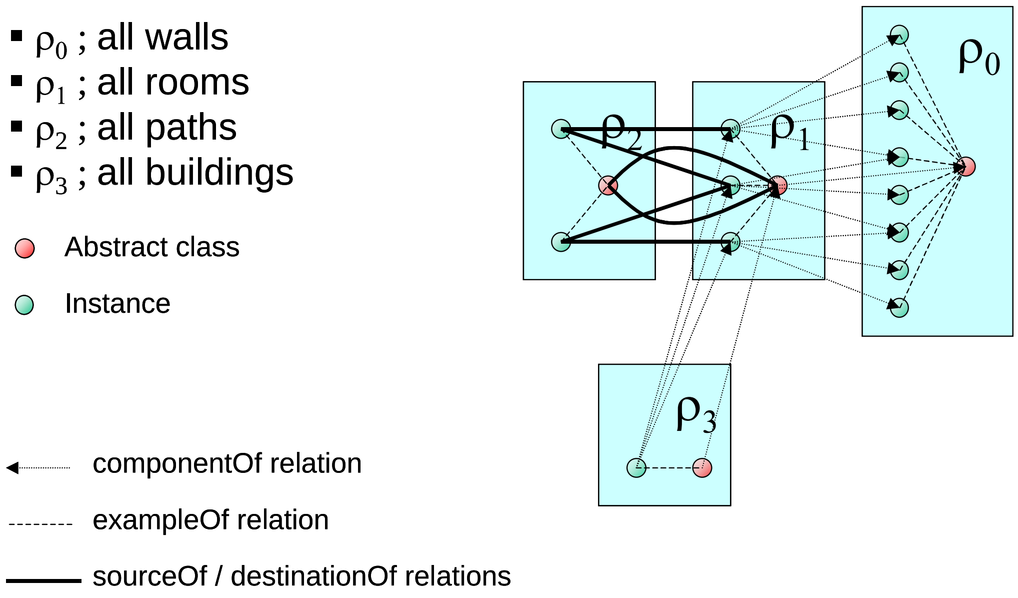

Figure 6. Semantic graph split into layers according to the classes and their instances

Figure 5 and Figure 6 illustrate identical semantic graphs. The only difference is in the distribution of nodes among layers (planes). This approach enables creation of as many layers (planes) in the system, as required by the user, each layer (plane) representing an illusion of a 2D graph as a subgraph of the whole semantic graph. The main user's attention is aimed at the subgraphs in the planes. However, subgraphs in different plane are still visually interconnected by relations (edges), but these relations are usually beyond the user's focus. The planes can be represented parametrically as follows:

$$\rho: <0,1> \times <0,1> \rightarrow E3$$

that is, as a mapping from closed two-dimensional space of parameters into three-dimensional space. In general, the planes need not to be flat. Due to the parametric representation, any known parametric surface can be used. (e.g. NURBS surface could be applied to approximate a sphere, cylinder, cone, or any possible shape) Due to practical reasons we have implemented 2D planes only.
After splitting the semantic graph into the planes, each plane could still contain many nodes. These nodes can be spread stochastic on the surface. The user can move the nodes to the appropriate positions, in order to get a "good looking" layout of the graph. Implementation of some self-organizing algorithm will remain in the scope of future work.

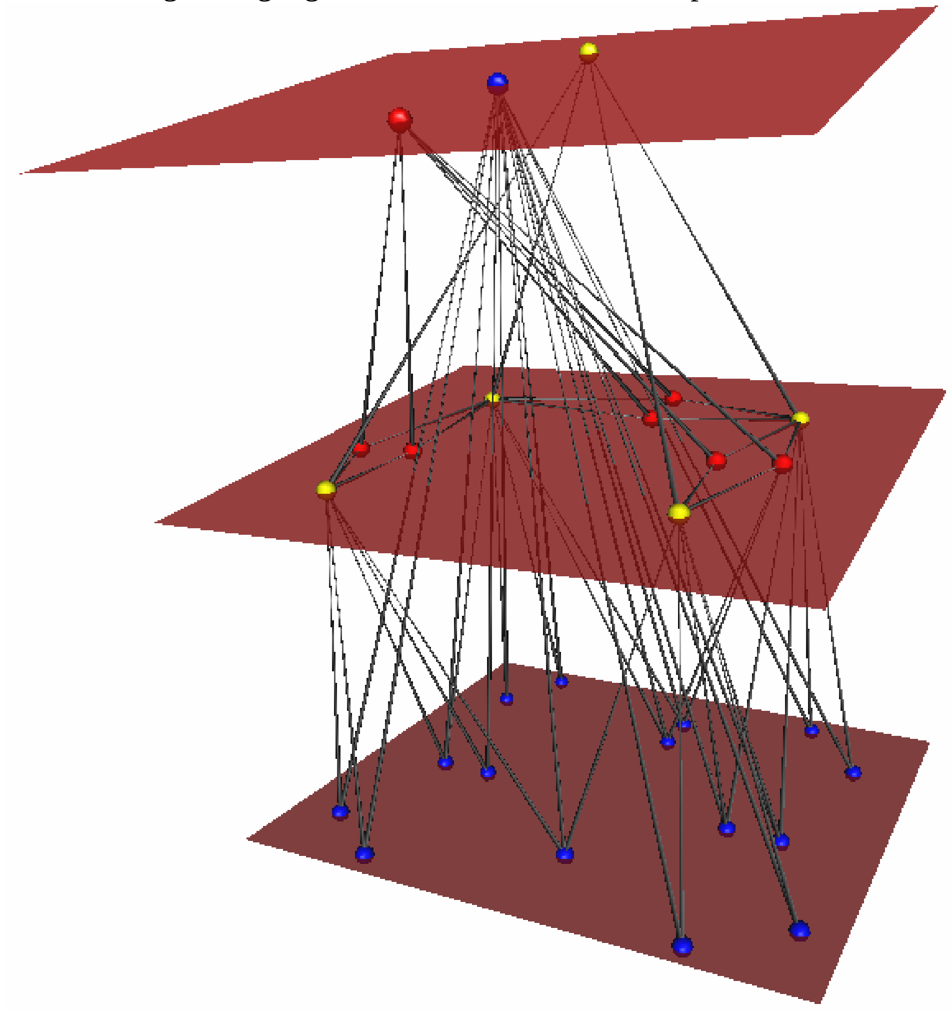

Figure 7 Example of a 3D visualization of a semantic graph

Implementation

When implementing the system we were focusing on these requirements:

- General core for semantic manipulation
- General XML based graphical data format interface
- Interactive designer

- Visualization of semantic and geometrical structures
- Interconnection between the visual components
- Interactive tools for semantic manipulation
    - Object filtering
    - Relation filtering

System architecture

The system architecture is divided into several independent units that can communicate via defined interface. The system core is responsible for manipulating the semantic information while the multimedia content browser and the semantic graph visualization unit are responsible for visualizing the graphical and semantic data. This flexible architecture allows to substitute the VRML visualization unit with SVG or any other that implements the given interface.
Any change in the graphical data are through the system core immediately propagated to the semantic visualization unit and vice versa. The semantic design is therefore fully interactive. The system core can be controlled by the user via its own graphical interface.

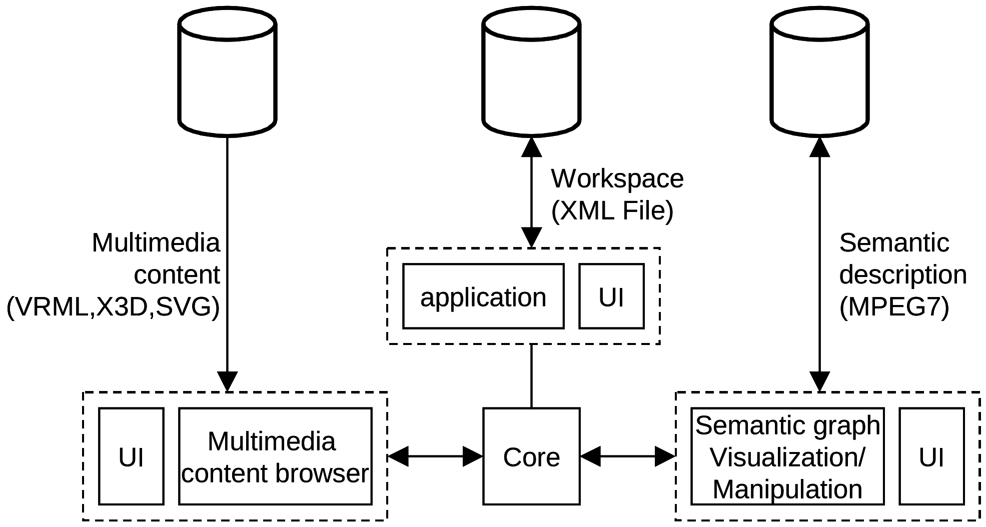

Figure 8 System architecture

The system architecture is depicted in Figure 8. Individual components are represented by boxes. Besides the system core and its user interface, all other components can be freely added or removed using a plug-in mechanism. This makes the system very flexible for creating semantic metadata for any graphical xml based format. The interconnection between components helps to visualize the creation process as also seen in Figure 9. The application is split into two basic windows. The left one displays the graphical data in any suitable form (in this case we can see the 3D scene an its scene graph). The right window displays the semantic description visualization. The application core mediates the communication between the components by distributing notification about changes in any of them to all other instances of visual components (plug-ins). Actions like selection, insertion or deletion can be immediately visualized.
The system core can control the application and provides additional features like node filtering, relation filtering, tracing and others.

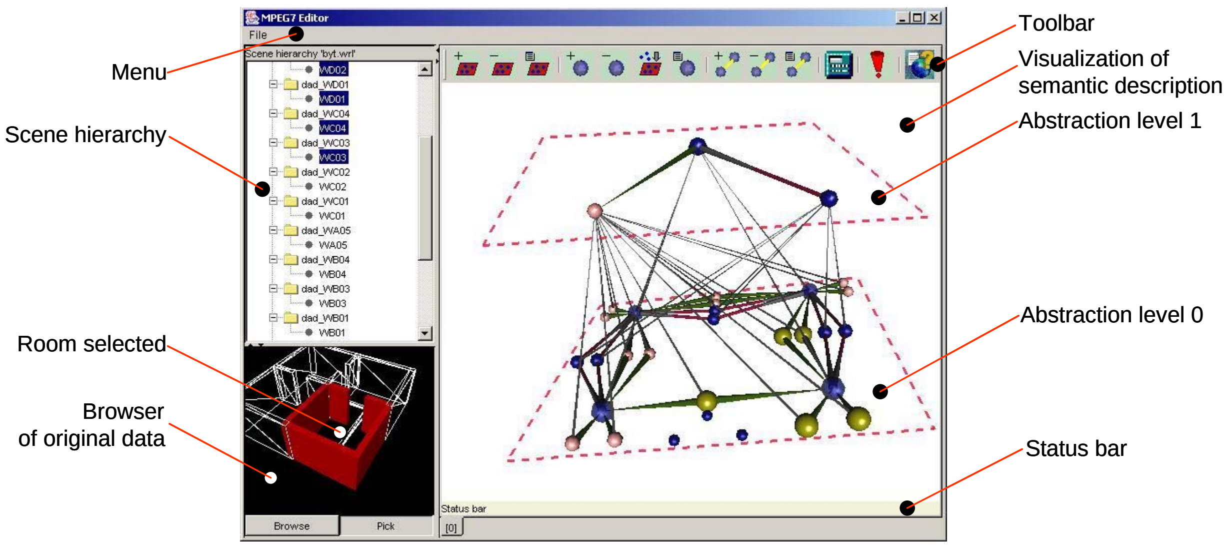

Figure 9 System graphical interface

Conclusion

We have designed and implemented a graphical tool for visualization and manipulation with semantic metadata for the use of data adaptation in the mobile environment. The tool is capable of visualizing of both the 3D (2D) data and the semantic graphs. The semantics and geometrical information are interconnected so the manipulation with one is immediately propagated to the other. The interactivity and appropriate visualization greatly help the user to create or modify complex semantic descriptions.

References

[1] **Mummy project** [WWW Document] `http://mummy.intranet.gr/`

[2] *Balfanz D.*: **Automated Geodata Analysis and Metadata Generation**, in Proceedings of SPIE Conference on Visualization and Data Analysis Vol. 4665 (pp. 285-295)

[3] *Li J. Z., Ozsu M. T., Szafron D.*: **Query Languages in Multimedia Database Systems**, Technical Report, The University of Alberta Edmonton

[4] *Mikovec, Z., Klima, M., & Slavik, P.*: **Manipulation of Complex 2D/3D Scenes on Mobile Devices**, In Proc. of the 2nd IASTED International Conference Visualization, Imaging and Image Processing (pp. 161-166), Anaheim: Acta Press. ISBN 0-88986-354-3

[5] *Mikovec, Z., Klima, M., & Slavik, P.*: **Structural and semantic dialogue filters**, In Proc. of the 2nd International Workshop Text, Speech and Dialogue (pp. 280-285), Plzen: Springer-Verlag, ISBN 3-540-66494-7

[6] *Mikovec Z., Klima M., Foldyna R.*: **GUI for Graphical Data Retrieval by Means of Semantic Filtering** 2003, In: Human-Computer Interaction: Theory and Practice (Part II). Mahwah, New Jersey: Lawrence Erlbaum Associates, Publishers, 2003, s. 193-197. ISBN 0-8058-4931-9.

[7] *Klima M., Mikovec Z., Slavik P.*: **Mobile Multimedia Collaboration and Annotation within the EU Project Mummy** 2003, In: Proceedings of Workshop 2003 [CD-ROM]. Prague: CTU, vol. A, s. 276-277. ISBN 80-01-02708-2.

[8] **MPEG-7**, ISO/IEC JTC1/SC29/WG11 2000 [WWW Document] `http://mpeg-7.com/; http://www.cselt.it/mpeg/`

[9] *W3C Consortium*: **The Extensible Markup Language (XML).** [WWW Document] `http://www.w3.org/XML/`

[10] *W3C Consortium*: **Scalable Vector Graphics (SVG).** [WWW Document] `http://www.w3.org/TR/SVG/`

[11] *Web3D Consortium*: **The Virtual Reality Modelling Language (VRML).** [WWW Document] `http://www.web3d.org/Specifications/VRML97/`

[12] *Stanford University School of Medicine*: **Protégé 2000.** [WWW Document] `http://protege.stanford.edu/index.html`

[13] *ONTO Knowledge*: **OntoEdit.** [WWW Document] `http://www.ontoprise.de/products/ontoedit_en`

[14] *W3C Consortium*: **IsaViz.** [WWW Document] `http://www.w3.org/2001/11/IsaViz/`
Stanford University: **Ontolingua.** [WWW Document]