Network management and virtual reality

P. Abel(*), P. Gros(*), D. Loisel(*), J.P. Paris (**)
(*) Institut Eurécom, Multimedia Communications Dpt.
(**) CNET France Télécom.
{abel,gros,loisel}@eurecom.fr, jeanpierre.paris@cnet.francetelecom.fr

Abstract

This paper presents CyberNet, an interactive 3D visualization tool designed in order to enhance network management. The key consideration of the CyberNet design is to be able to collect, process and visualize large amount of dynamical information. But the aim of the project is to study how 3D may help the user to analyze this information, notably thanks to the use of metaphors and specificity of real-time 3D computer graphics. The collected information is structured and mapped on visual parameters (size, shape, color, etc.) of 3D objects in order to built dynamical metaphoric world where the user may navigate and interact.

1. Introduction

The network management process requires human operators to monitor and interact with a huge amount of dynamical information coming from all over the network. We believe that 3D visualization could help the user to have a global perception of the states and tendencies of the system and to detect potential problems before they happen. The CyberNet system can also be used in the context of performance evaluation for network planning and prototyping.

In order to be able to experiment with 3D representation we had to work on three different issues that are related to each other. The first point to is to study how to gather the information according to the way the administrator analyzes the service he supervises with a minimal impact on the system behavior. The second point is to organize and structure this information. The third point is how to visualize this information in a virtual world. This paper will focus on the third point but some highlight on the information structuring will also be given.

Section 2 presents the information model that we are using. Section 3 focuses on the visualization model. Section 4 briefly presents the implementation and section 5 concludes this paper and introduces future work.

2. The CyberNet information model

The trend is now to develop and use distributed applications running on a large number of workstations interconnected by LANs and WANs. In order to manage a system, the operator has to monitor lots of network devices and workstations. To cope with these complex systems, the operator has to think in terms of *services* rather than in terms of devices. For a service to run, a lot of devices are required, and nowadays, operators just gather information and manually (or visually) correlate them (in general

after the fact). In this section we will first describe a software architecture that allows us to organize and structure information to build services, then explain how we characterize information.

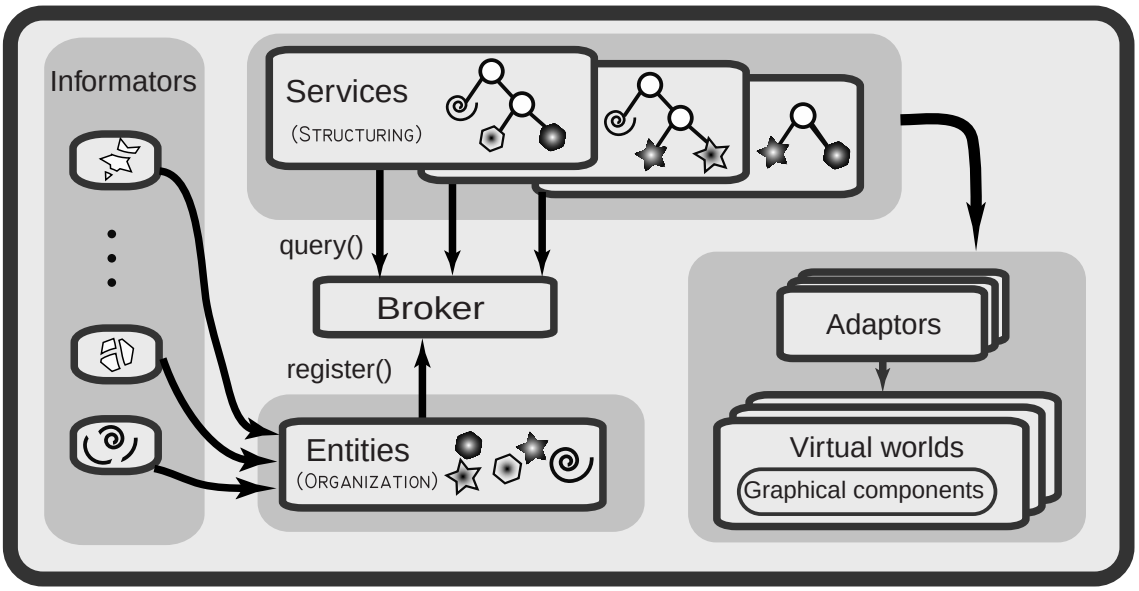

Fig1: Software architecture.

2.1. Software architecture

The software architecture is shown in figure 1. It is an enhanced version of the one we presented in [1]. Let's look at the information flow to see how information is organized and structured.

1. Raw information is collected by an *informator* agent which pushes the data to its subscribers according to subscribing policies.
2. The information sent by the informator is logically gathered as *entities*. For instance, a hub entity is described by its name, its IP address, its number of ports, etc. A Unix process entity is described by the name of the owner, the memory and the CPU time it is using, etc. The entity is the first step in organizing information. All these entities are registered in a broker with their types and attributes. The broker is informed every time a new entity is created or deleted.
3. The structuring layer allows us to build *services* : it queries the broker for specific entities related to the service we want to visualize, and then structures them hierarchically in trees. For example, a simple service of workstation supervision is composed of the following entities : the workstation, its users, and the processes running on it. The structuring layer is designed so that when new entities are created or deleted, the structure is updated accordingly. It is the task of the broker to propagate structural modifications. In this structuring phase, we describe the service by making effective some of the implicit relationships that are contained in the entities and we build a hierarchy out of these relationships.
4. At this point, information is organized in entities and some entities are structured in trees to build services. The final step is to visualize structured information. For that purpose we use *adaptors* which map entity information onto the visual parameters –

e.g. size, shape, position, color, etc. - of graphical components. We will focus on this stage in the next section.

2.2. Information characterization

We can define different kinds of information in order to help us map data on visual parameters. In our current model, we categorized information in two main classes : basic information and relationships information.

Basic information

This information is that contained in the different fields of an entity. This is the raw information collected by informators (except for tendencies, see below.)

- **Qualitative information**: there are two subclasses, **identification** and **classification**. For instance, a workstation entity will be of class "workstation" and will have a name and an IP address as identification fields.
- **Quantitative information**:
  - **Values** are used to quantify and may be absolute values, percentages, dates, etc. These values are directly collected from the real world (for example the load of a workstation, the number of collisions in the case of an Ethernet hub, etc.)
  - **Tendencies** or history traces can be generated from the above values data that varies dynamically with time using derivatives or statistics.
- **Events** (notification information): they are used for notification purposes, for example to indicate that a new user is logging on a workstation.

Relationship information

This kind of information describes relationships between entities. The relationships are based upon the analysis of fields (i.e. basic information) and/or classes of entities. For example, there is a relationship between all the user entities bound to the same machine because their "workstation" fields contain the same information (i.e. the same name or IP address). Thus, relationship information is not collected directly but may be deduced from the information field of each entity. It is important to note that most of the time, one entity is referenced by several relationships. We have defined two classes of relationships:

- **Groups** are collections of entities that belong to the same class and/or have one or more fields containing the same information. For example, all the users entities form a group because they belong to the same class and all user entities that are logged on the same workstation form another group of users since their "workstation" fields contain the same information.
- **Dependencies** are introduced in order to structure information hierarchically. A dependency may contain entities, groups or other dependencies. For example, a simple workstation dependency contains the workstation entity and the previous group of user entities.

- **Relation:** some relations between entities may not be expressed as group or dependencies, and may relate two items in the hierarchy. The relation is used to model them and is conserved down to the visualization domain**.**

3. The CyberNet visualization model

Most traditional network management tools represent information as tables, graphs and simple schemes. How may an operator be able to monitor a complex system of thousand of nodes with such basic user interfaces? To solve this problem, some specific visualization tools should be designed in order to represent huge amounts of dynamical information. Our guess is that 3D could help for that purpose.

A lot of work has already been done about 2D graphical representations of information [2] [3], but how could 3D be used to enhance data representation is still a research topic of interest. Our approach is not to add a third dimension to usual 2D charts nor to copy the real world with realistic rendering, but to experiment new visualization strategies. Our strategy is to build 3D virtual worlds based on the use of metaphors in order to enhance the legibility of information. We want to make the most of the specificity of 3D computer graphics such as 3D scene hierarchical description, real-time animation and visual parameters. Most of these specificities are related to the fact that the whole screen appearance is recomputed for every frame, and thus that the object's location, shape, color, size, etc. may naturally vary in time.

In this section, we will describe our graphical components and analyze how different kinds of information can be mapped on the visual parameters of graphical components. Then, we will present other features of our model: adaptive representation, the use of metaphors, views and user interaction.

3.1. Graphical components

There are two graphical components: 3D glyphs and layout managers. The former are a kind of 3D animated icons which in most cases represent information contained in an entity, the latter arrange the 3D glyphs in space, most of the time to show the relationships between them.

3D glyphs

Our basic visualization building blocks are called 3D glyphs. 3D glyphs are predefined 3D objects with visual parameters that can be dynamically modified in order to show information. The glyph concept was first used in 2D [4]. 3D glyphs range from extremely simple (such as bar charts) to really complex objects. The level of complexity of a 3D glyph can be measured by the number of *visual parameters* it offers and thus to the amount of information that can be displayed. Examples of visual parameters are the shape, the color, the location, the size of the 3D glyph or of one of its graphical parts. 3D glyphs may be any object that our metaphoric visualization needs, it can be related to real world objects (for instance, a colored map of the United States in which each color is assigned to a state) or be some abstract data representation such as a simple pie chart or more elaborate ones like the one presented in [5] [6].

Layout managers

Large amounts of information need to be organized in order to make their interpretation easier [7]. 3D glyphs are a first step towards organization since they group information. Layout managers are a second step in organizing 3D glyphs in space. The task of the layout manager is to arrange its children elements in space. The children may be only 3D glyphs or both 3D glyphs and layout managers. The visual parameters of a layout manager are the position and the orientation it gives to its children. Layout managers can also add some visual elements like semitransparent bounding boxes (see [8] for the use of semitransparency) around their children to enhance the visualization. They define the geometric location and orientation of their elements according to a set of policies. For instance, a layout manager may organize elements in orbit around its center, on a plane in rows/columns, in Russian puppet boxes, or use a cone tree model [9] (see figure 2).

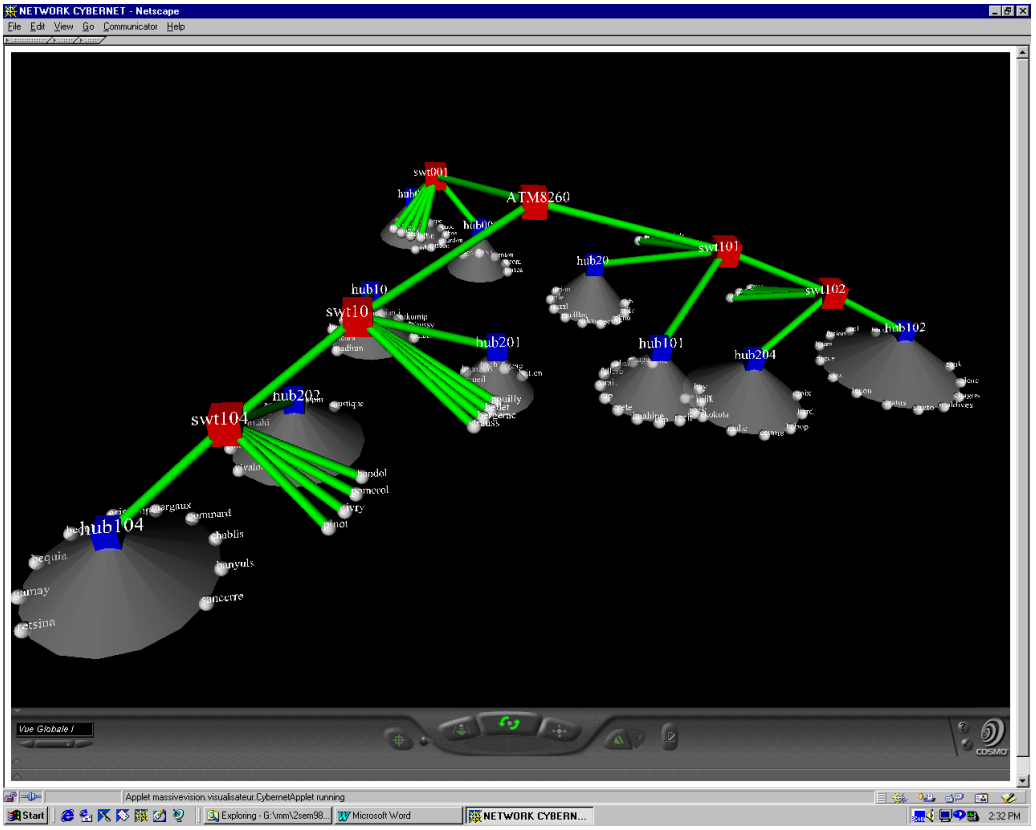

Fig 2 : Visualization of the topology of the Eurecom Institute Network using a cone tree model [18]. The size of each link is relative its output. The transparency of a cone corresponds to the collision rate of the hub it represents

3.2. Mapping information onto visual parameters

Information must be mapped on visual parameters in order to be visualized. In most cases, an entity is visualized through 3D glyphs, which means that each of its information fields (i.e. basic information) is mapped on a visual parameter (color, shape, size, etc.). But basic information can also be visualized through the visual parameters of

a layout manager: relative orientation and location (a value can be represented by a distance between two glyphs). These last two visual parameters can also be exploited to show relationships: a cone tree model is a clear visualization of dependency information for example. Relationship information can also be visualized through other visual parameters. For example, the shape of a 3D glyph can represent the membership in a group. In the software architecture, it is the role of the adaptation layer to do this mapping. This layer allows visualization to be changed without affecting the structuring layer.

Virtually any information could be mapped onto any visual parameter. However, some visual parameters are likely to be more suited to a class of information than others. For example it seems that the clearest way (and the most used) to represent dependencies is based on geometric proximity. Placing 3D glyphs near to each other in the virtual world gives the user the feeling that they share some common properties [10] [11]. The following table gives some examples of possible mapping:

Visual Parameter	**Example of use**
Color/Material	Value, classification, identification, tendency
Size	Value, classification
Shape	Classification, tendency, group
Animation (movement, deformation)	Tendency, event
Lightning	Event, identification, group
Text	Identification, value
Relative Position & Orientation	Group, dependency, value,
Bounding volumes/transparency	Group, dependency

The idea behind that table is to find rules of mapping that follow logical guidelines in order not to confuse the users. We experimented that it's important that one class of visual parameters should not be mapped onto several different classes of atomic data since it can confuse the users and may lead to loss of information. For instance, once the color has been associated with identification information, it should not be used for other purposes (imagine a world where the color encodes identification and value in the same time). One can find a lot of different sets of coherent rules, but inside one virtual world only one set of rules should be applied. These rules are meant to maintain a high level of consistency between glyphs so that users can analyze and compare glyphs easily.

At present, the mapping is hard-coded but one of our goals is to automatically map information onto visual parameter according to the class of the information and to the adopted metaphor. It requires to characterize all information of an entity as well as the visual parameters of 3D glyphs.

3.3. Adaptive representation

It is important to provide an adaptive representation of information in order to help the user analyze information. For instance, the computer representation should not be the same when the user monitors the whole domain or a specific workstation (see figure 3 and 4). In the first case, the computer representation should be very synthetic since tens or hundreds of computers could be represented on the screen, while in the second

case the representation should be very detailed since the user only requires information about that specific workstation. Clustering [12] is one way to do so, the use of different levels of detail [13] is another one. Clustering consists in replacing several glyphs with a new glyph that is a synthetic representation of all the information contained in the hidden glyphs. Complex 3D glyphs that display a lot of information may be difficult to interpret when they are far away from the user in the 3D world. Different levels of detail are used to have several graphical representations of the same glyph according to user attention (in this case we assume that user attention can be measured by the distance between the user and the observed 3D glyph).

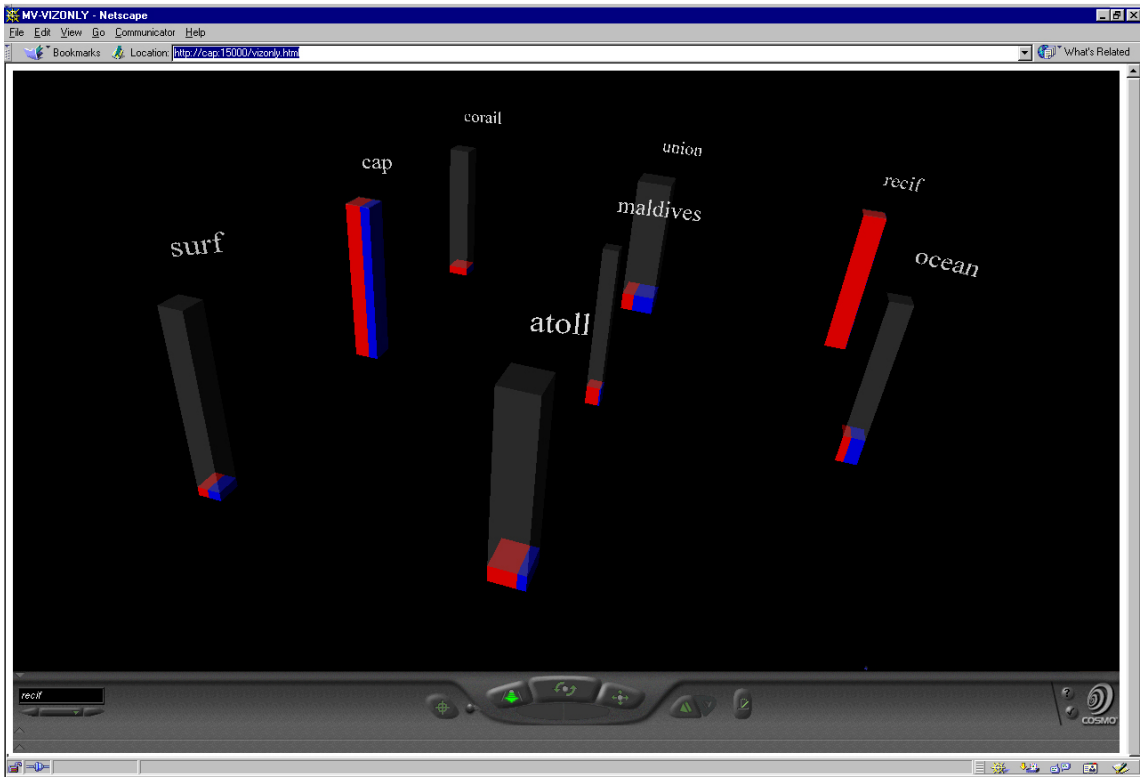

Fig 3: This image presents a synthetic visualization of several workstations [19]. Each one is visualized using a transparent column that contains an indicator. The height of this indicator corresponds to the CPU load and its coloring to the swap memory used. Figure 4 shows a non-synthetic view of a workstation.

3.4. Metaphors

Metaphors have been used several times in user interface design with good results. The desktop metaphor that we use on most computers is one of the best-known examples. As we can read in [14], “metaphors provides ways of introducing concepts to users with the aid of analogies with familiar real-world objects.” Thus, using metaphors the user is already familiar with can help enhance the legibility of information. An interesting use of the metaphor of a city has been done in [15].

Our goal is to design several metaphors which can be chosen by the user regardless of the service visualized. It means that a metaphor should not be bound to a specific

service. Thus, the users can try different metaphors and choose the way to visualize information that best suits their needs.

To construct a metaphor, we must define a set of 3D glyphs and layout managers which reflect the properties of the chosen metaphor. For example, let's say we use a city metaphor. We need glyphs – for buildings, houses, roads, etc. - and layout managers which place them in the various districts. If we want to use a solar system metaphor [1], we need glyphs -for stars, planets, satellites and rockets - and an orbital layout manager.

3.5. Views

A virtual world can contain as much information as we want but it can be useful to have two different worlds in separate windows in order to show two different kinds of services. Each virtual world is called a *view*. In the case of network management, the user may require to have, simultaneously, a topological view of the network [16][17], and a view of a particular service (for instance the NFS or printer services). Moreover, visualizing the same data from different points of view can sometimes enhance the understanding (for example, to analyze a complex network). In addition, the user can choose a metaphor for each view so that he can take advantage of each one.

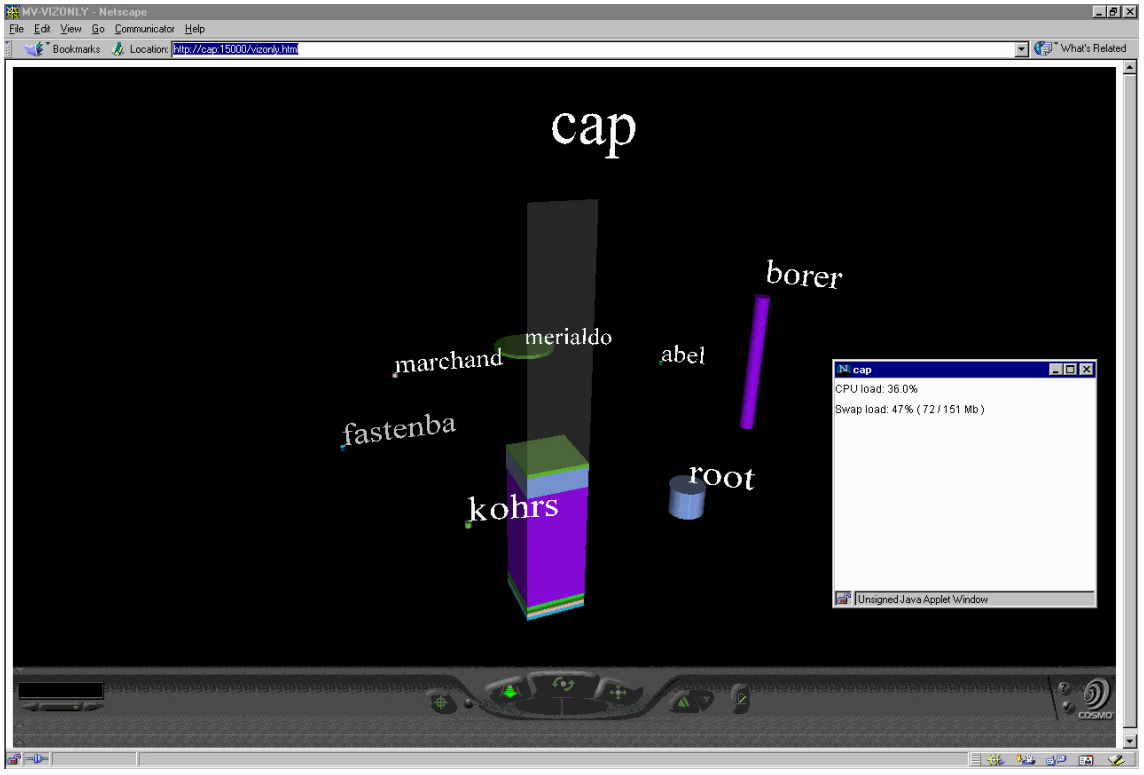

Fig 4: This snapshot comes from the same virtual world as in figure 3. The "cap" workstation has been opened to obtain details about users logging on it. Each user is visualized as a column the width of which indicates the memory he's using and the height the CPU he's using. The little opened window contains quantitative details (in digits) that the user has asked about the "cap" workstation.

3.6. User interaction

The user can interact in several ways with the system. He can navigate in the virtual world, but 3D navigation is a difficult task since it is very easy to get lost. Thus we implemented viewpoints in order for the user to be able to jump from place to place. We are also studying how to use “rampart paths”, which can be either predefined paths or a sequence of movement made by the user and automatically reproduced. The navigation may be also part of the metaphor itself (for instance a public transport inside a city metaphor). The user can exploit one feature offered by the adaptive representation: he can cluster or open some parts of the world to efficiently manage visualization. Quantitative values in digits related to an entity can also be obtained on demand. The user also has the power to create views to display new services from scratch or from specific information found in the virtual world he’s visiting. Finally, he can change the metaphors of each virtual world in order to find the best visualization.

4. Implementation

In order for the system to be portable, all the components of the system are written in JAVA, and all communications between the distributed components of the system are based on CORBA. 3D glyphs and layout managers run on the Java virtual machine of a VRML enabled WWW browser and they dynamically update a VRML world using the External Authoring Interface (EAI). We have also begun to study how to use Java3D instead of VRML. The end user may access the system from anywhere since it only requires a VRML enabled WWW browser (that support hardware shading and texture mapping).

5. Conclusion & future work

The first phases of the CyberNet project are about to be completed. We are now upgrading the software architecture from the one presented in [1] to the one presented above (i.e. adding a structuring layer, a broker and optimizing data flow). With this new framework, we will be able to build services easily. Several topics are being further investigated. So far, few implementation has been done on graphical components and metaphors (some primitive glyphs and layout managers, including a cone-tree model). We mean to develop new glyphs and layout managers with special focus on their adaptive representation capabilities. We plan to work on the automation of metaphoric world design. To achieve this, the visualization model must be conceptualized in order to allow automatic mapping of atomic data onto visual parameters according to the class of the data and to the adopted metaphor. Further work on the enhancement of user interaction in relation with the metaphors will also be conducted. This is specially true about navigation which still requires some skill and habit from the user. We plan to apply visualization to other applications that require the interactive monitoring of large sets of dynamical data (for example Stock Exchange).

This research was supported by France Telecom and the Eurecom Institute.

5.1. References

[1] P.Abel, P.Gros, D.Loisel, J.P.Paris : “CyberNet : utilisation de métaphores et des techniques de réalité virtuelle pour la représentation de données. Proceeding des 6ème journées du GT Réalité Virtuelle. pp 26-31

[2] Tufte, E., The visual display of quantitative information, Graphics Press, Cheshire, CT (1983).

[3] Bertin, J., Graphics and graphic information processing, Walter de Gruyter, Berlin ,N.Y., 1981

[4] Chernoff H., The use of faces to represent points in k-dimensional space graphically. Journal of American Statistic Association, Vol. 68, 1973, Pages 331-368

[5] Chuah, M. C., and Eick, S.G. “Information rich glyphs for software management Data.” IEEE Computer Graphics and Applications. July/August 1998. Pages 24-29.

[6] Swing, E., Flodar : Flow visualization of network traffic. IEEE Computer Graphics and Applications. September/October 1998. Pages 6-8.

[7] K.M. Fairchild, S.E. Poltrock and G.W. Furnas, Semnet: “Three-dimensional Graphic Representations of Large Knowledge Bases, in "Cognitive Science and Its Applications for Human-Computer Interaction"”, Fairlawn, NJ: Lawrence Erlbaum Associates, 1988

[8] Zhai, S., and Buxton, W., and Milgram, P. The partial occlusion effect : Utilizing semitransparency in 3D human interaction. ACM Transactions on Computer-Human Interaction, Vol 3, No. 3, September 1996, Pages 254-284

[9] Robertson, C.G., and Mackinlay, J.D., and Card, S.K. Cones trees:Animated 3D visualizations of hierarchical information. Proceedings CHI’91 Human Factors in Computing Systems, ACM, Pages 173-179

[10] Benford, S., Snowdon, D., Greenhalgh, C., Ingram, R., Knox, I., and Brown, C. “VR-VIBE: A Virtual Environment for Co-operative Information Retrieval” Proceedings Eurographics '95, pp 349-360

[11] Chalmers, M. and Chitson, P., Bead: Explorations in Information Visualisation, Proceedings of SIGIR’92, ACM Press, pp. 330-337.

[12] Hendley, R.J., and Drew, N.S., and Wood, A.M., Beale, R. , Narcissus: Visualising information. Proceedings IEEE Information Visualization 95. Pages 90-96.

[13] K.M. Fairchild, S.E. Poltrock and G.W. Furnas, Semnet: “Three-dimensional Graphic Representations of Large Knowledge Bases, in "Cognitive Science and Its Applications for Human-Computer

[14] Newman W.M., Lamming M.G., Interactive system design, Addison Wesley,1995.

[15] Ingram, R. and Benford, S., Legibility Enhancement for Information Visualisation, Proceedings of Visualization 1995, Atlanta, Georgia, October 30 - November 3, 1995

[16] Eick, S.G., Aspects of network visualization. IEEE Computer Graphics and Applications. March 1996. Pages 69-72.

[17] Cox, K.C., and Eick, S.G.,and He,T., 3D geographic network displays. Sigmod Record, Volume 24,Number 4, December 1996

[18] Oria L. and Girardot M., Network Cybernet, Internal Report, Eurécom, 1998

[19] Borer J.M. and Zapf C, System Cybernet, Internal Report, Eurécom, 1998