CyberNet: A framework for managing networks using 3D metaphoric worlds

CyberNet : Utilisation de mondes 3D métaphoriques pour l'administration de réseaux.

P. Abel, P. Gros, D. Loisel, C. Russo Dos Santos
Eurecom Institute - B.P. 193 - 06904 Sophia Antipolis, France
{abel,gros,loisel,russo}@eurecom.fr

J. P. Paris
CNET France Telecom
06904 Sophia Antipolis, France
jeanpierre.paris@cnet.francetelecom.fr

ABSTRACT

In this paper we will present CyberNet, an interactive 3D dynamic information visualization tool designed to evaluate the added value this technology brings to network management. This paper explains how 3D metaphoric worlds may help the user and points out the problems raised by the dynamic nature of the information. It focuses on the solutions that were adopted to cope with these problems. We first present some prototypes of network administration tools implemented using the CyberNet framework. Secondly we explain how information is collected and structured, and then we show how this structured information may be mapped onto 3D graphical components in order to automatically build and update metaphoric 3D worlds. The last part of the paper presents the user interaction mechanisms provided by the system.

Keywords

Dynamic Information Visualization, Web Visualization, Metaphors, VRML, 3D Navigation, Virtual Worlds, Network Management, TupleSpaces, Distributed Systems.

RESUME

Ce papier présente CyberNet, un outil de visualisation d'information dynamique en 3D construit afin d'étudier les apports de cette technologie pour le management de réseaux. Nous montrons comment l'utilisation de mondes 3D métaphoriques peut être utile à l'utilisateur, et nous détaillons les problèmes soulevés par la nature dynamique et distribuée des informations. Nous étudions les solutions adoptées pour faire face à ces problèmes. Nous présentons dans un premier temps quelques prototypes d'outils de gestion de réseaux construits à l'aide du système CyberNet. Nous exposons alors comment les informations sont collectées et structurées, pour ensuite présenter la méthode d'association de ces informations structurées à des composants graphiques 3D afin d'automatiser la construction de mondes métaphorique 3D. La dernière partie de ce papier présente les mécanismes d'interaction avec l'utilisateur supportés par le système.

Mots clés

Visualisation d'information à caractère dynamique, Métaphores, VRML, Navigation 3D, Management de réseaux, TupleSpaces, Système distribué.

1 - INTRODUCTION

Nowadays, network management is a difficult and attention-demanding task since a network manager has to understand the behavior of very complex networks constituted by thousands of manageable *components*. Traditionally, these components are network devices (such as hubs, switches, and routers). The network information is usually presented to the user using tables and/or graphs. When there is a change in the network status, the interface signals it to the user, usually using some kind of dialog box. The user shall then make a new request and, acting upon it, the system updates the user interface content.

Nowadays the trend is to manage computers (configuration, system, and activity) as well as software (distributed systems, client/server applications). Network managers cannot think in terms of devices, they have to think in terms of *services*. In our terminology, the term service covers Topology, Connectivity, Routing, DBMS, NFS, Mailing, etc. The status of a service can generally be understood by analyzing large amounts of data distributed on the network devices. In order to improve data presentation, we believe that information needs to be logically structured according to services. A service has its own set of involved components and should be presented using the most suited graphical interface

The objective of the CyberNet project is to study the usability and effectiveness of 3D techniques and virtual reality interfaces for system and network management. We intend to prove that visualization of distributed dynamic data can be made more efficient with 3D technology. There is no reason to use the same representation to visualize the topology of a network and the behavior of a complex client/server application. For this purpose, we are designing a general framework that makes the development of specific visualization tools easy. This framework is designed to dynamically build and update 3D worlds according to the real world data. It also features some specific support for user interaction and navigation.

The CyberNet framework has been used in order to develop and test several network management tools. Each tool offers specific characteristics that can be exploited in order to answer to a specific network management task. Our visual structure is based upon the concept of 3D metaphors. Metaphors are used because their underlying structure is familiar to the user [1]. We have designed real world metaphors such as a building, a city or a solar system. Nevertheless it is sometimes useful to design more abstract metaphors such as a cone-tree or a landscape. A metaphor is not designed for a specific service. In fact, a service can be visualized using any metaphor provided it offers sufficient capabilities for viewing the service information. This allows the user to select the metaphor best suited to his needs. Because of the dynamic nature of the system, every time there is a change in a service it is reflected in the metaphoric 3D world.

2 - RELATED WORK

The interest in 3D interfaces for network management started less than ten years ago. The first experiments in that field were devoted to topology and device administration. One of the first experiments began in 1992 at Columbia University. The COMET group has studied how virtual reality may be used within the context of ATM network management [2]. [3] and [4] presents a network management tool that supports collaborations between users and is accessible from a web browser. In parallel, researchers started to focus on visualizing the network data rather than to limit the visualization to the network structure and connectivity. [5] presents some work done on this topic and introduces studies about services: the mail service is studied in [6] and highlights the interest for the visualization to exploit (cluster, sum up) the hierarchical structure of the data.

More recently, [7] reports AT&T laboratories experiments concerning the management of large dynamic telecommunication data sets (~250 million events per day). The results of the studies seem promising since the presented tool improves network management and marketing of network services. [8] presents a performance analysis tool called Virtue. The virtue tool adapts itself to the dynamic behavior of the studied application. It gathers information from multiple sources involved in the monitoring of complex distributed parallel applications. Related to our field, SGI has developed a 3D computer-monitoring tool called Performance-Copilot [9]. The tool can be used for monitoring computer arrays and clusters as well as in the context of Web server and Oracle DBMS management. The major benefit of this tool is to help the users to quickly isolate, drill down and understand performance behavior, resource utilization, activity levels and performance bottlenecks. [10] presents a set of network visualization tools that gives 3D graphic representations of the Internet from a geographical point of view.

This research domain is part of the Information Visualization field [11] [12]. 3D Information Visualization has lots of aspects and applications. In a special report [13] shows how it may help in the context of the "Global Information Infrastructure" to retrieve information, interact with databases, manage networks or browse the web. [14] reports an experiment done by Xerox in order to go beyond the usual desktop metaphor. An active research area is Business data 3D visualization [15]. The main idea is to provide 3D graphical representations of rows and columns of numbers in order to improve perception and support decision-making. [16] presents a data mining visualization software developed by SGI called MineSet. The force of this tool is the integration of several 3D visualizations that can be used simultaneously by the user. Extensive work has been done towards web based information visualization [17].

3 - MANAGING NETWORKS USING CYBERNET

The CyberNet Framework has been developed to study how 3D interfaces may enhance the network management tasks. Our goal is to demonstrate that 3D visualization allows the user to have a synthetic understanding of services and have faster reaction. For this purpose we have designed and experimented several metaphoric worlds and have applied them to several classes of services. This chapter presents experiments based on the CyberNet framework. These examples highlight the use of 3D worlds to visualize network information. All these 3D worlds have been generated automatically from information collected on real networks. Any 3D-world metaphor may be used to visualize any data set as long as a mapping between the real world data and the graphical components of the metaphor has been defined. At present time, the mapping is hard coded but we are working toward an automatic mapping scheme. You may experience static demonstrations of most of these examples as either VRML files or real-video streams in our web site [18].

3.1 - Geographic Administration tool

Geographic administration tools are useful for tele-maintenance as well as for physical link problem detection. In the latter case, the network manager may notice that some physically grouped devices have a transmission error ratio abnormally high. The scope of the geographic visualization depends upon the application; for instance, WANs are usually based upon earth or country maps [5] [10]. We developed an example of a LAN geographic administration tool based on the building metaphor (Figure 1). The building metaphor developed by us is very much oriented toward network administration needs. The building is schematic and is viewed as a container for classifying network devices. The precise size and aspect of the building is not rendered but the relative locations of objects are. Each network device (e.g.: computers, hubs, switches and routers) is placed at its real world location. Using this tool, network managers can analyze their network according to the actual location of devices and identify problems related to geographic proximity. The user's navigation is automated: the user interface provides predefined navigation paths that follow the building logical structure. The user is able to select a destination point in the interaction interface provided on the left side of the tool. Using this interface the user can also interactively change the visualization; for instance, by selecting offices/users/workstations/devices filtered by some criterion (e.g., belonging to a specific department) or by rendering walls and floors transparent.

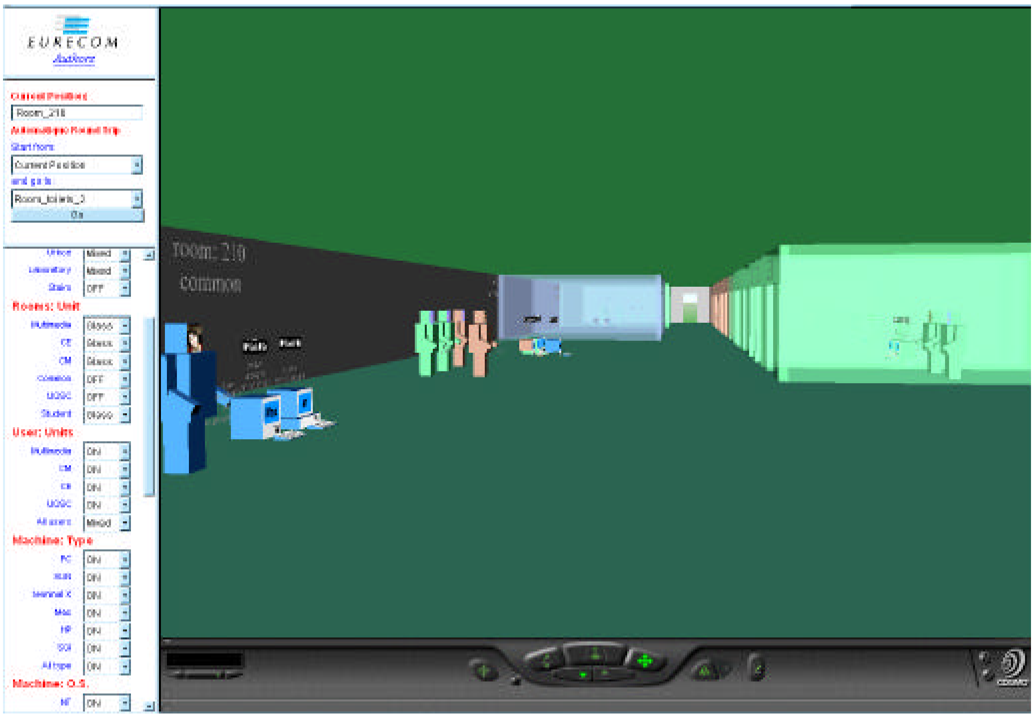

Figure 1. Geographic Administration tool based on the building metaphor
Figure 1. Outil d'Administration Géographique fondé sur une métaphore de bâtiment.

3.2 - Topology Administration tool

Topology administration tools are useful for detecting network element failures or problems concerning load balancing. We develop an example of LAN topology administration tool based on the cone-tree metaphor (Figure 2). The cone-tree [19] is well adapted to visualize hierarchical data structure and most LANs (or at least parts of LANs) may be conceptually modeled as trees with leaf nodes being computers and intermediate nodes being routers, hubs and switches. Computers that have several network links appear several times in the hierarchy. In this example, a box (which identifies the hub) and a cone (that represents the collision domain of that hub) represent hubs. All the devices that are connected to that hub are connected to its cone. Switches are represented by a box and a set of cylindrical links that connect that box to the devices that are connected to the switch. The bandwidth usage and error ratio measured (using SNMP) on each device affects, in real-time, the color and size of these cones and links. Using such a representation, the user may rapidly understand the overall traffic and notice congestion problems. Of course, other mappings could be defined in order to monitor other network values.

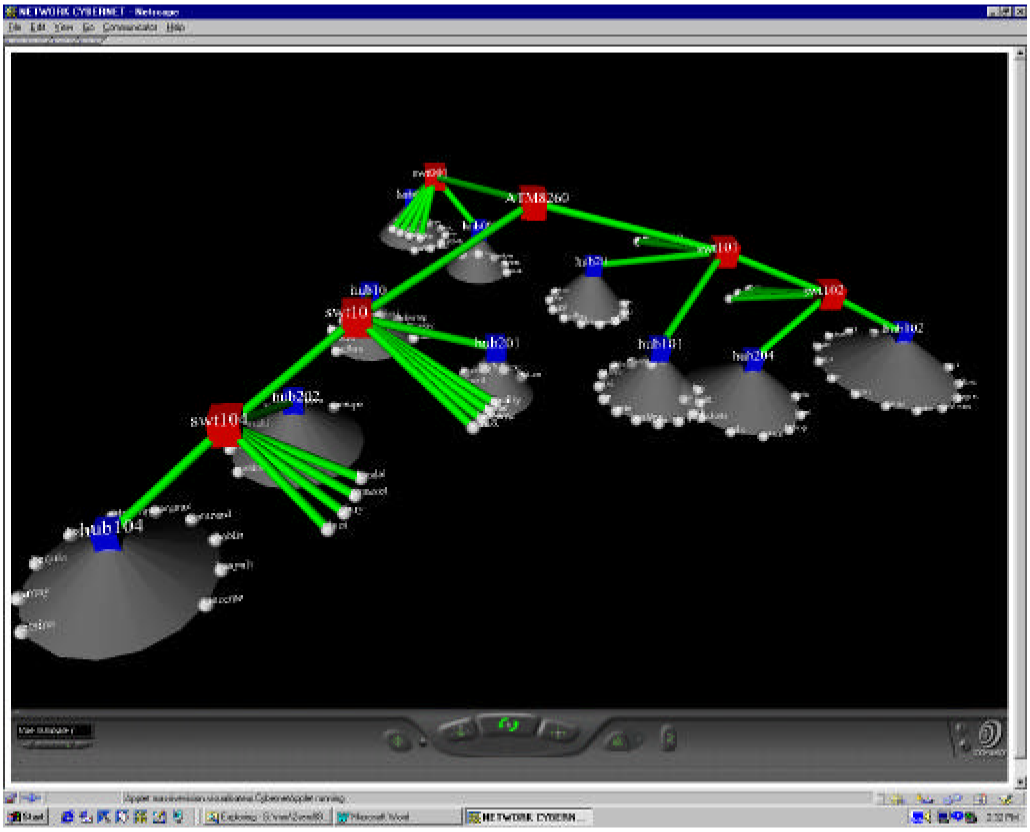

Figure 2. Topology Administration tool based on the cone-tree metaphor.
Figure 2. Outil d'Administration Topologique fondé sur une métaphore d' « arbre de cônes ».

3.3 - Distributed Services Administration tool

The previous examples are quite traditional in the way they represent network data. We want to address more complex problems related to the management of a complete service that involves lots of network components. Client/Server applications (like mail, DBMS etc) are typical services for which we believe 3D metaphoric worlds can help. An example of such a service is the NFS (Network File System) client/server application. Unix computers to export their file systems to other computers use NFS. Using NFS a remote disk is viewed in the same way as a local disk. But from the network administrator and system manager's point of view, each access to a remote disk generates network traffic.

A typical NFS site configuration involves a high number of computers on a typical Unix/NT workstation network. We chose NFS as an example of the administration of a complex networked client/server service. The first design principle was to avoid the use of graphical links to represent relations between a client and a server. This kind of representation easily leads to an incomprehensible spaghetti graph that is of no help to the user. The underlying idea was to represent both the server and the client as separated objects, and to identify the existence of a client on the server representation as well as the existence of a server on the representation of the client.

We developed the city metaphor [20] to show both the configuration and the runtime characteristics of such a client/server applications (Figure 3). Each town is a sub network, each district of the town is a computer, and each building is a disk resource. There are several types of buildings, mainly buildings that represent local disks, and buildings that represent remote disks. On the server side, local disks may be exported; in this case each client (computer that import that disk) is represented on the server side as an additional floor in the local disk building. Furthermore, the number of windows at each floor monitors the activity of the client (number of filehandles opened). Colors are used to identify configuration information. Other information is also accessible such as the traffic generated by the NFS server or the error ratio. By using the city metaphor it is very easy to locate servers and potential problems. It gives a good understanding of the loads of the servers. But the crucial point is that all the data that are required to understand the service are available in one view.

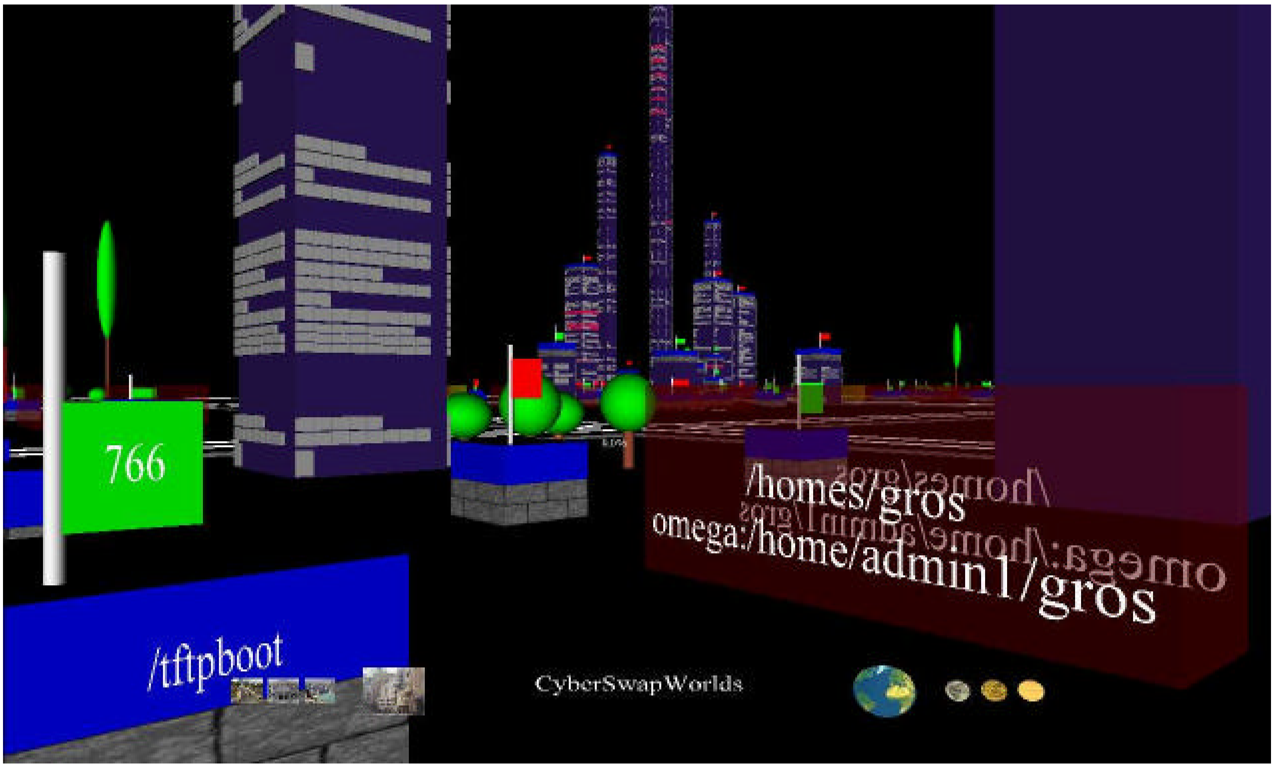

Figure 3. Distributed Systems Administration tool based on the city metaphor.
Figure 3. Outil d'Administration de système distribué fondé sur une métaphore de ville.

3.4 - Computer Administration tool

Computer administration tools [9] are useful to track computers and user's activities on a network in order to prevent problems and easily administrate networked systems. Monitoring a network of workstations requires lots of information to be accessible at the same time (e.g. users, processes, memory and processor usage). Our goal was to display in one world the dynamic result of the "top" Unix command ran simultaneously on all the computers of the domain.

For that purpose we developed an example of a Computer monitoring tool based on the solar system metaphor (Figure 4). In this metaphor, computers are stars, users are planets in orbit around stars and processes are satellites floating in-between the star and the planet. The visual parameters (e.g. orbit radius, size, color) are used to represent CPU usage and virtual memory consumption. The system administrator can observe the load of computers, and detect when some limit is reached.

In the context of networked workstations, the tool can be used to monitor the same set of data using a different mapping. An example of such is to monitor user's activity by representing users as stars and computers as planets with processes floating between. In this case, we can easily identify the computers a user is logged on and the activity of the user on each computer. Another example is the process centered view for which processes are stars, computers are planets and users are satellites. In that case the user may easily identify extensively used applications and detect potential problems concerning a specific instance of an application by visualizing the same data set with the focus on processes.

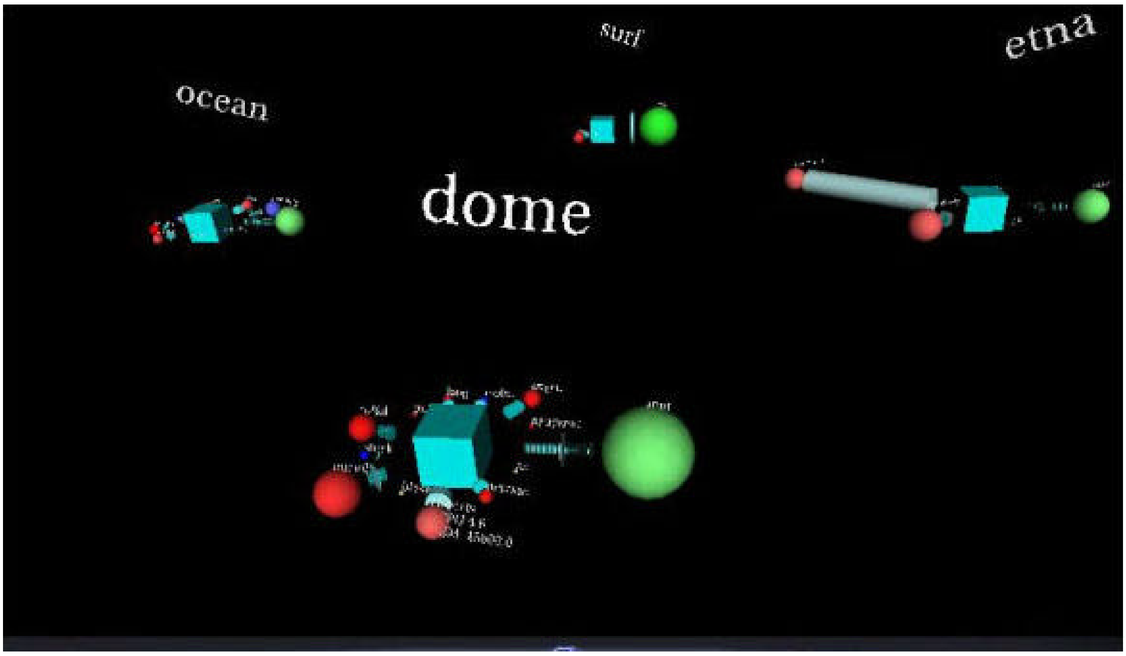

Figure 4. Computer Administration tool based on the solar system metaphor.
Figure 4. Outil d’Administration d’Ordinateurs fondé sur une métaphore de système solaire.

3.5 - Network Traffic Characterization tool

Network traffic characterization tools are important in the context of network planning. The idea is to be able to analyze real traffic data in order to identify potential architecture problems. The network traffic is captured at a specific point of the network using a network sniffer. The result is a very large set of network packets. The packets are analyzed in order to extract some typical values such as round trip time (rtt) or congestion window size (cws). Finding interesting information in such high flow of data is very difficult. Here again, 3D can help. The network traffic characterization tool can help the user to rapidly identify interesting data. For this purpose, the raw data and the derived values (rtt, cws) are displayed in 3D. Information is classified according to source/destination addresses and ports.

We have developed a tool based on a 3D-landscape representation (Figure 5). The tool is used in different situations. Firstly it is used to identify and select connections of interest. In that case the landscape’s x and z are the source and destination addresses and the height and color of the bars represent some computed values like the bandwidth and the variance of the cws. Secondly, it may be used to do a time based analysis of a specific connection (or set of connections) in order to identify when a problem occurred and what were the source, destination and port. The network traffic characterization tool is also of great help for the user to quickly identify relevant connections according to predefined characteristics based on rtt, cws, etc. Once detected, these connections will be further studied using standards network traffic analysis methodology.

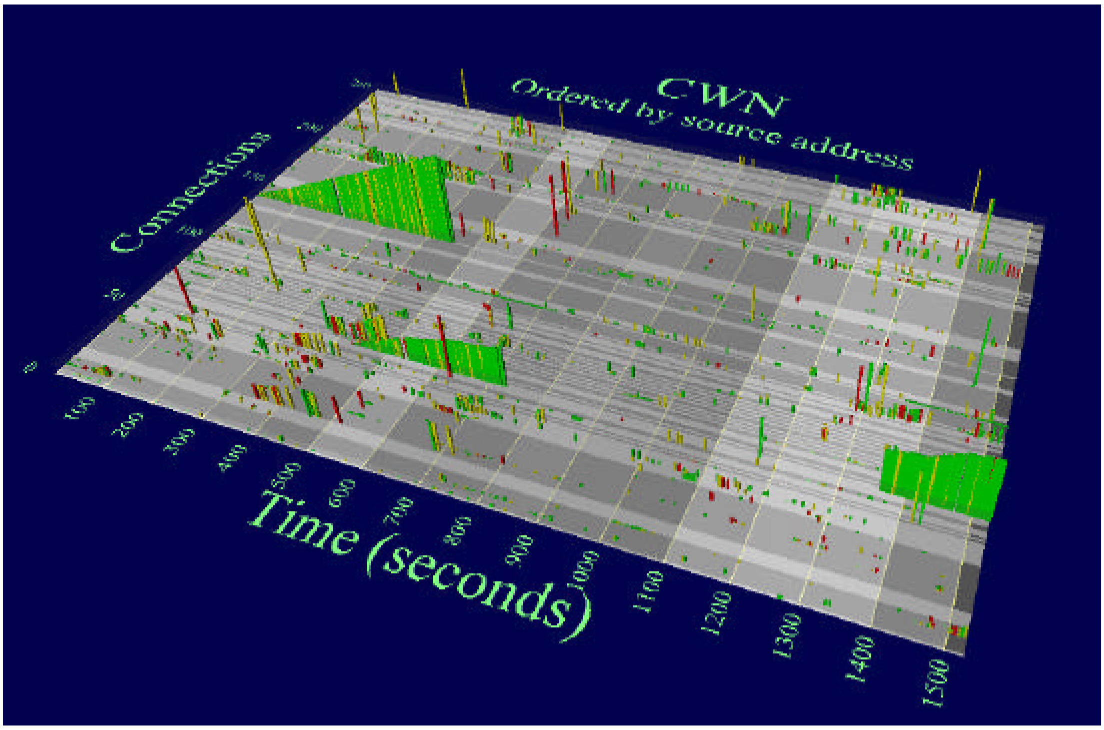

Figure 5. Network Traffic Characterization tool based on a landscape metaphor.
Figure 5. Outil de caractérisation de trafic fondé sur une métaphore de terrain.

3.6 - Combining the tools

We believe that one of the strengths of 3D metaphoric representation will arise from the combined use of all the tools that have been presented so far. This conclusion comes from several experiments done within the context of the previously presented tools. An experiment has been done within the context of the geographic metaphor: the user was able to select all the computers connected to a specific hub or switch by simply clicking on its visual representation. This selection mechanism was not efficient since it was difficult for the user to do the selection in the geographic administration tool. It seems preferable to have two views, one with the topology (which really facilitates the selection) and one with the geography and allow actions done within the context of one view to be reflected on the other view.

The underlying design rule is that when a problem occurs, the first task of the network manager is to identify the nature of the problem. The problem can be caused by a lot of factors: Is there a physical link or a power supply problem somewhere? Is some network component overloaded or out of order? Is there a software crash problem? Etc. Lets present a typical scenario: a network manager detects a problem on a specific service (lets say the mail) using a service administration tool. He notices that the problem seems to come from a server that is not running. He switches to the computer administration tool but he cannot get information on that computer. So he activates the topology administration tool and notice that all the computers connected to the same switch are inaccessible. He concludes that the switch seems to have a problem.

4 - THE CYBERNET FRAMEWORK

The CyberNet system is a distributed object framework that handles all the tasks from collecting the network data to the 3D visualization. The use of distributed object technology answers one of the problems that we were faced with: the data are distributed over the network, and are dynamic. It also allows us to separate the 3D user interface (which is very computing intensive) from the other part of the system. This framework is composed of three distinctive parts (Figure 6):

- *The collecting layer* is used to gather raw network data from the monitored devices.
- *The structuring layer* is the kernel of the system. It structures raw information according to the service model.
- *The visualization layer* translates the service model into a 3D metaphoric world; it manages both the visualization and the user interaction.

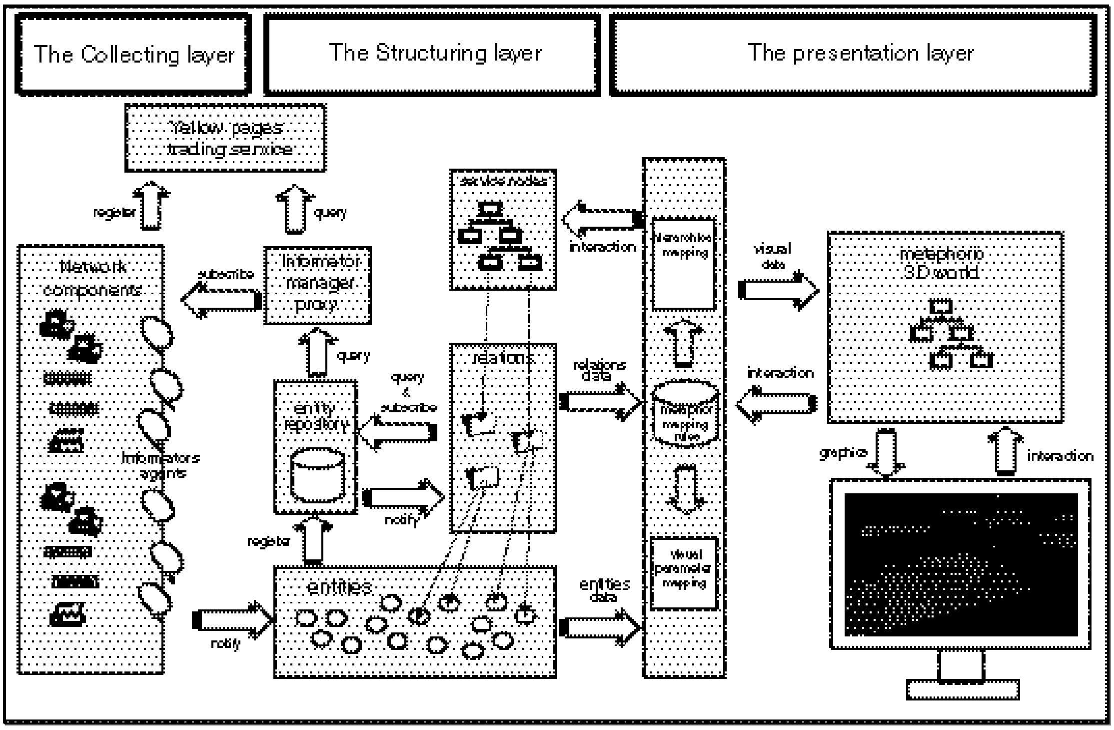

Figure 6. The CyberNet Framework Architecture.
Figure 6. Architecture de la plate-forme CyberNet.

4.1 - THE COLLECTING LAYER

For a given service, the network data that has to be monitored is spread all over the network. In traditional systems, SNMP agents that run on every network device supply this information. The network manager should “pull” the data by using SNMP requests to the agents. This is a cumbersome task if the user wants lots of information. Furthermore, it is not suited to our system since we want to automatically build 3D worlds that continuously display the behavior of a service. Using the traditional pull mechanism would largely increase the network bandwidth, because it would require the system to pull the data at extremely short intervals to present a quasi real-time visualization to the user. In order to reduce the generated traffic, we had to develop our own collection strategy.

The collecting process is implemented using distributed agents (called *informators*) located on or near the observed devices. Informators are designed so that they are able to “push” information to the structuring layer. The mechanism is based on a subscribe/notify design pattern. When it is interested in a data, the structuring layer should contact an informator and subscribe to that data. This subscription is made through a *subscription request*. The subscription request requires the structuring layer to identify the data it is interested in together with a *collecting policy* for that data. For example, the structuring layer may be interested in receiving all the data of type “user” on a computer called “ocean”. The collecting policy is a set of criteria defined by the structuring layer and used by the informator to describe when a notification should occur.

Policies include criteria such as periodic notification (every 5 seconds), freshness criteria (every 5 second but only if modified), or percentage of modifications since last notify (every five second and only if new value differs by more than 10% from the previous). The collecting policies have been introduced to optimize information transferred by the informators agents to the structuring layer and thus minimize the amount of information traded between layers. The systems benefits from this strategy since it does not need to query all the monitored network devices in order to collect relevant data; instead, as soon as an interesting (as defined by the policies) modification occurs, the system is notified and pertaining information is "pushed" to the upper layer.

When a user wants to monitor a service, the structuring layer locates the informators responsible for collecting the required data using a *"Yellow Pages" trading service*. It subscribes to these informators and specifies policies. At that level, an *informator manager proxy* is used in order to avoid duplicate subscriptions: each subscription is first sent to the informator manager proxy who analyzes the subscription and determines if a previous subscription with compatible criteria has already been done for that data. Once the configuration is completed, informators start "pushing" every network modification falling into these policies criteria to the structuring layer. These agents may use any method (e. g., SNMP, HTTP, scripts) for retrieving information. In the previous examples, SNMP requests were used in order to obtain topology information and network bandwidth and error ratio; Systems calls were used to retrieve NFS and other operating system data; Database accesses were used to retrieve building information.

4.2 - THE STRUCTURING LAYER

Structuring information is of prime importance since the 3D representation will be automatically built from the structured data. The goal of the structuring layer is to construct a service graph that will be directly exploited by the presentation layer. There are two main problems to solve:

- define how the graph will be structured according to the collected information,
- dynamically update the graph according to real world modifications.

4.2.1 - Structuring elements

The structuring layer is designed in order to produce at its output a structured tree. This structured tree is called the *service model*. This service model will automatically be mapped onto a 3D scene hierarchy thus the structuring layer plays a most relevant role in the world creation process. The service model is based on three classes of elements: *entities*, *relations* and *service nodes*. We will first describe these element classes in a static environment. We will then introduce the additional requirements of a dynamic information environment.

Entities are the atomic building blocks used to model the managed services. Entities may represent physical devices (such as hubs, switches, routers, workstations, users, etc.) or they may be conceptual (such as connections, processes, groups, etc.). Every entity has a type. Each type of entity requires a specific set of *attributes*, whose values are collected from the network in the form of raw data, in order to precisely identify and describe the entity. For instance, a workstation is identified by its name, its IP address, its DNS domain, its hub port and it is described by its CPU power and usage, its total and available memory and disk space, its network traffic, etc. Entities are created and updated by informators according to the collecting policies. Each time an entity is created, it is registered in a centralized entity repository. The entity repository is the heart of the structuring system. It is extensively used by the relation elements.

Relations are used to group entities according to some common properties. Conceptually, a relation may be defined as the result of a query to the entity repository. This query is defined by a set of selection criteria regarding the attributes of an entity. For example, it is possible to define a relation that groups all the processes of user "marcel" on the computer "ocean". This would result in a query to the entity repository that asks for "all entities of type process which have their attributes user=marcel and computer=ocean". The relation will then group references to all the entities that match the query. As a consequence, relations may involve any number of entities (sometimes only one) and an entity may be involved in any number of relations. Service nodes create relations.

Services nodes use entities and relations to build a tree, and organize the information to be visualized. Every service node is related to, at least, one relation and therefore is aware of the entities involved in that relation. By analyzing the entities contained in that relation, a service node may decide to add a new service node to the tree. For example, upon receiving a workstation entity, a service may create a child service node in charge of building a new relation containing all the users of that particular workstation. The relationship between a node and one of its children may be structural (e.g. a process is related to the workstation it is running on) or functional (e.g. a client/server relationship between two workstations). The service nodes are the main components of the structuring layer. The design of a service model should involve an application domain expert since the service is responsible for all the structuring of the data, a task that requires domain expertise.

A service model is described in terms of service nodes. Service nodes create relations. Each relation sends a query to the entity repository and a subscription request to the informator agent proxy. The informator agent proxy analyzes the request and if it has not been already processed, it contacts the yellow pages trading service. This service returns a reference to the informator that is designed to collect that information. The informator agent proxy subscribes to that informator. Concurrently, the entity repository permanently forwards to relations references of entities that match the requests it has received.

4.2.2 - Managing dynamics

In a dynamic information environment, raw network data are constantly modified. Informators track these modifications and according to the collecting policy they create or kill entities or update their internal attributes. This has an impact on each element of the structuring layer. New entities are created and existing entities die. In addition, dynamic attributes of an entity are evolving with time. The evolution of the entities (creation, death) results in changed relations. Services depend on relations and therefore the service's tree evolves according to the current state of relations.

Entity dynamics are handled by the informators that automatically push information to the entity repository according to the subscription request. For example, if an informator was supposed to report all the users logged on a computer then, each time a new user logs in, the informator will notify the structuring layer and a new user entity is created in the entity repository. In the same way, any modification of an attribute is transmitted by the informator and the related entity is updated. Relation dynamics are handled using the subscribe/notify design pattern. In addition to the query to the entity repository, a relation also subscribes to it in order to be sent every new entity that meets the query selection criteria. Likewise, each service node is notified when a relation receives a notification and the service node reacts accordingly (e.g. by creating new service nodes). Furthermore, service nodes may directly subscribe to the entities of its relation(s) so as to be notified when their dynamic attributes change or when they die.

4.3 - THE PRESENTATION LAYER

The presentation layer is responsible for displaying the virtual world and for providing navigation and interaction capabilities to the user. The presentation layer is also responsible for translating the service model into a metaphoric 3D world. The CyberNet system automates the construction and update of the 3D world through a process called mapping. In the CyberNet system, the visual structure of the 3D world is dependent upon the concept of metaphor.

4.3.1 - Metaphor and graphical components

A metaphor is a set of *metaphoric components* (MCs) organized in a hierarchy. Metaphoric components are based on *graphical components* with interaction and navigation add-on capabilities related to the metaphor itself. We will present the interaction and navigation capabilities in the next chapter, focusing here on the graphical part. Each graphical component has visual parameters that can be used in order to visualize information. CyberNet supports two classes of graphical components: *3D glyphs* and *layout managers*.

- *3D Glyphs* (3DG) are 3D objects whose appearance may be controlled in real time through visual parameters. 3DGs represents data through visual parameters that are either retinal (e.g., color and size), or temporal [21]. For example, a box is a 3DG that offers three retinal parameters (color, width and height) and one temporal (rotation around itself). The level of complexity of a 3DG can be related to the number of visual parameters it offers for modification and hence to the dimension of data that can be displayed.
- *Layout managers* (LM) are responsible for one of the most important visual choice: the use of space [22]. They are used as containers and may contain 3DGs or other LMs. They organize their children in space (position and orientation) according to some built-in policy and may also animate these positions along paths. For instance, a LM may organize elements in orbit around its center, on a plane in rows/columns, in Russian puppet boxes, or use a cone tree model [19]. Layout managers can also add some visual elements like semitransparent bounding boxes (see [23] for the use of semi-transparency) around their children to enhance the visualization. These additional visual elements can also be used to show information as 3D glyphs do.

Using these two kinds of graphical components, one can build a 3D-scene hierarchy. Each internal node of this hierarchy is a LM and each leaf is a 3DG. Specific graphical components must be defined for each metaphor. For example, in a city metaphor the minimal requirement is 3DGs of houses, buildings or roads and a LM that positions these elements in different districts. A solar system metaphor could be composed of star and planet glyphs, and an orbital layout manager. The more GCs are used in a metaphor, the more numerous the are ways of visualizing data. Some metaphors cannot be used to visualize a service because of a lack of graphical components to show information.

4.3.2 - Mapping

Mapping is the process that automatically constructs a 3D metaphoric world from the information contained in a service model. It results in a graphical hierarchy where internal nodes are layout managers and leaves are 3D glyphs. In the CyberNet system, special objets called *Adaptors* handle the mapping. Adaptors are dependent on the type of *metaphor* used for displaying the information and provide the necessary mechanism to automatically construct the virtual worlds.

4.3.2.1 - Hierarchical Mapping

In our current implementation, each metaphor is defined as hierarchy of MCs (i.e.: a town metaphor is based on MCs called Town, Districts, Streets, Buildings, Floors etc). We call *hierarchical mapping* the process of defining which service element (entity or relation) will be visualized using which MC. The main idea behind the mapping process is to define a set of association rules for each service based on the type of each service element and its position in the service tree. In particular, since an entity may be a part of several relations (thus being located at more than one position in the service model tree), it may have several visual counterparts in the presentation domain.

So far, the mapping process is hard-coded but we are currently developing an automatic mapping process based on information and visual property characterization. For instance, in the previous NFS example, the model mapping rules state that computers=districts, disks=buildings, NFSimportpoints=floors, FileHandles=windows, and so on.

4.3.2.2 - Visual Parameters Mapping

The attributes of each entity must be translated into visual information. This is the purpose of the *visual parameters mapping*. Each MC has a number of visual parameters that may be dynamically modified in order to display information (position, orientation, size, color etc…). A trivial example is the mapping of a process entity on a box glyph: the CPU used could be mapped on the color hue of the box and the memory used on the size of the box. When defining the mapping rules, care must be taken to preserve metaphor coherency. For example, if one rule uses the color for identification purposes, other rules can not use this visual parameter to represent a value.

In general, a relation is associated with a LM and an entity is associated with a 3D glyph. Thus, the graphical hierarchy follows the structure of the service. Nevertheless, we have found that it is sometimes useful to add special nodes in the graphical hierarchy in order to add functional capabilities to the graphical level (e.g., a 'fit' node that constrains the size of its children).

4.3.3 - Managing dynamics

It should be noted that data dynamic, which has been discussed for the structuring layer, has also an impact on the presentation layer. Any modification in the service model must be presented to the user and the virtual world should be modified accordingly [24]. In particular, if the number or dimension of the components managed by a layout manager change substantially, this layout manager must recompute the new position and orientation of those components. It may also have to propagate the changes to its parent (if existing) and to its other children. Depending on the metaphor, layout managers may adopt several strategies. They may propagate any changes thus modifying the world structure or they may try to reduce the impact of the modifications on the whole world. When a modification occurs, the implementation should take into account the two following design principles. On one hand, special care must be taken to make changes perceptible (using animation for example), on the other hand, substantial changes that have a great impact on the virtual world must be presented in such a way that they do not confuse the user. One of the techniques to overcome this problem is to make major impact changes appear smoothly in the visual representation, in order to make the visual representation evolve in time in a way that is not disruptive to the user [19].

4.4 - USER INTERACTION

In order for the user to easily interact with the entire CyberNet presentation layer, several topics have been studied. Firstly, all the tools should be accessible from the general framework. They should be able to share information (mainly entities) and the user should be able to interact (select and manipulate objects) and navigate easily within the 3D worlds.

4.4.1 - Navigation

Anyone that has ever experienced 3D interfaces will agree that navigating in a 3D world is not a trivial task. The user interface of traditional 3D browsers provides navigation mechanisms that allow the user to modify the virtual camera parameters. Using these tools, it is common that after some movements, the user gets lost in space and tries to restart from the beginning. It is obvious that when the user navigates from office to office in a virtual building he does not use the same navigation mechanisms than when he is exploring a landscape of data or studying the topological structure of a cone tree.

In the CyberNet visualization framework, all those metaphoric worlds are constructed using standard well-defined graphical components: Layout managers are used to arrange 3Dglyphs in space. For example the same "stack" layout manager is used to align offices along a corridor as well as to stack floors one on top of the others in order to construct a building. But how to navigate along a corridor is obviously not the same as how to navigate from one floor to the other. Our guess is that navigation mechanisms should be associated to graphical components. This association will lead to the design of what we call *metaphoric components*. The goal is that the user navigates into the world with the mechanism the most suited to the metaphor itself. We call this principle "*metaphor-aware navigation*".

In order to be able assist the user in its navigation task, the CyberNet system maintains a *user's current location* in the 3D world, a *current node of interest* (the node that is supposed to have its current attention) and *a new node of interest*. When the user is simply moving around into the world, the user's current node of interest and its current location are identical. But some navigation tasks require the user to be located in a place and to have its node of interest on another object (an example is the "look at" navigation mode).

At a first level CyberNet proposes to support two schemes for the user to identify his node of interest: *Absolute navigation* requires the user to select a new node of interest using a selection mechanism similar to the one supported by most VRML browser for viewpoint selection. This selection can be done, either by using a 3D embedded interface (by clicking on an object for example), or using an external menu. This menu can be hierarchical and context sensitive. *Relative navigation* requires the user to select a new node of interest relatively to the underlying metaphor hierarchy using traditional browsing operators such as up/down (in the hierarchy), or next/previous (element at that level). For example, when the user is in an office, choosing "next" will automatically bring the user to the next office in the corridor, in the direction the user is moving or facing.

The system has to determine the set of movements that should be done within the context of the current metaphor in order to go from the user's current node of interest to that new node of interest. These movements are dependent upon several user navigation modes. Basically the idea is to be able to specify what the user wants to do according to that new node of interest and how this should be done. This is done by the combined use of a *user mode ("go to and "look at")* and a *movement mode ("point to point", "Interpolated" and "path")*. The movement possibilities obtained by logically combining these two modes are explained in (table 1).

Movement mode **User mode**	**Point to point**	**Interpolated**	**Path**
Go to	The user jumps to the new node of interest and gets attached to it. (VRML)	The user flies in a straight line from his current location to the new node of interest and his orientation is modified. (VRML)	The user travels from his current location to the new node of interest according to a metaphor-based path.
Look at	The user stays at his current position and look toward the direction of the new node of interest. If the object is moving then it is tracked.	The user stays at his current position, his direction of view is animated from its current value to the direction of the new node of interest.	The user stays at his current location, but his direction of view follows a metaphor-based path from the current node of interest to the new node of interest.

Table 1. movement obtained by combing navigation modes (movements marked VRML are supported by most VRML browsers).

Table 1. Mouvements obtenus par la combinaisons des modes de navigation (les mouvements marqués VRML sont déjà supportés par la plupart des navigateurs VRML).

The notion of path based navigation is central to our navigation. The main interest of the path-based navigation is to give the user knowledge of the relative locations of objects within the context of the metaphor. We already referred that it is of prime importance for the paths to be metaphor-aware. Navigating in a building (see figure 7) is a good example of such a metaphor-aware navigation mechanism.

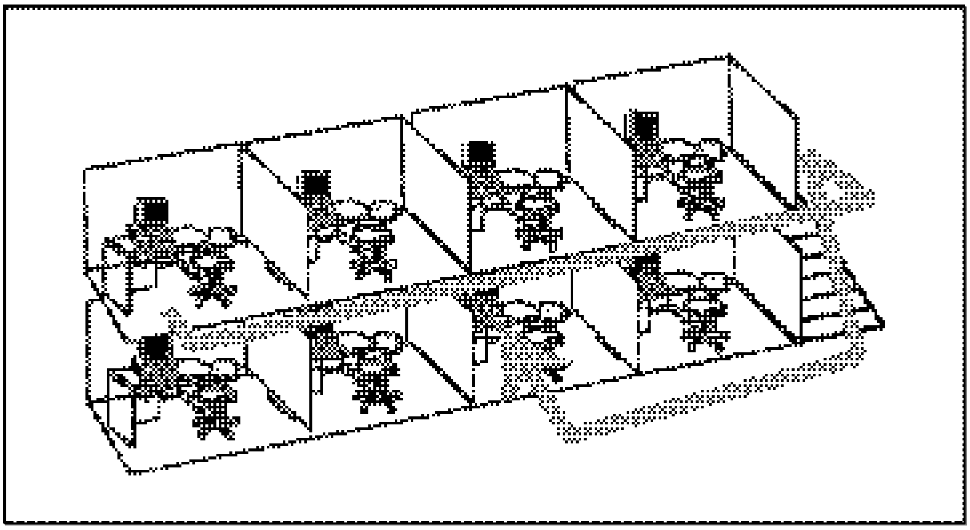

Figure 7: The metaphor-aware navigation path from one office to another.
Figure 7. Chemin de navigation adapté à la métaphore pour se déplacer d'un bureau à un autre.

4.4.2 - Selection

The main point to note about selection is that the selection mechanism should be coherent between all the tools. When the user is using the topology administration tool and he selects all the computers connected to a specific switch, he should be able to switch to another tool (say the geographic administration tool) and see the locations of these computers in the building. This requires the selection mechanism to work directly on the entities at the service model level and not to be limited to the scope of the current metaphoric world.

4.4.3 - Visual parameter modifications

We found important to be able to interact with the world in order to be able to modify its appearance. An example of such a study has been done within the context of the building metaphor. Our idea was that having a model of the building and being able to navigate and interact with objects according to the metaphor is important but sometimes frustrating. Virtual worlds should bring more possibilities than just mimic the real world. With this idea in mind, we develop mechanisms that allow the user to have control over the world appearance itself. Here are some examples: in the building metaphor, the user may modify walls and grounds transparency. The user may render all the ground and office walls transparent in order to directly see the content of all the building; he may render some offices partially transparent according to criteria such as the department they belong to, etc. In the Network traffic characterization tool, the user may select some data to be transparent and thus not visible. This is frequently used to remove aberrant values that disturb the comprehension of the valid data.

5 - IMPLEMENTATION

The CyberNet platform (Figure 8) is written in JAVA and the CORBA distributed object technology is used to support communications between the distributed parts of the system.

The collecting layer is mainly composed of a large number of distributed objects. AdventNet JAVA/SNMP, Operating System calls and perl scripts are used for collecting information. Technically speaking, these informators may use any method (SNMP, HTTP, scripts...) for retrieving the information. They use Visibroker CORBA for communicating with the structuring layer. The Yellow Pages mechanism is a trading service supported by Visibroker.

The heart of the structuring layer is the entity repository. Its implementation is based on the use of a Tuplespace middleware. Tuplespaces provide communication, synchronization and notification primitives for building distributed system data repositories. The most advanced tuplespaces [26] support template queries and event notification. An entity is registered as a tuple and a relation is implemented as a template, which is transmitted to the tuplespace as both a query and a registration to the event notifier. For that purpose the CyberNet platform uses IBM Tspaces[4].

The CyberNet user interface is accessible from any VRML-enabled WWW browser. In the current implementation, the Presentation layer drives the VRML plugin through the EAI (External Authoring Interface) of VRML. We are investigating the porting of the user interface onto JAVA3D.

Aside this dynamic environment, we have developed the same framework in a static environment (i.e. the raw data do not evolve in time) in order to do some fast prototyping. This system is implemented in Perl and generates a VRML file with embedded interactions coded as javascripts VRML nodes.

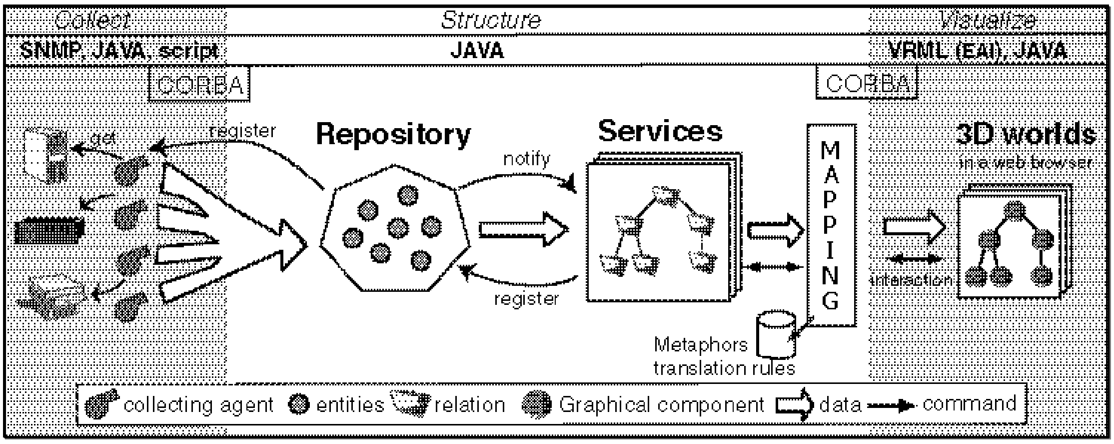

Figure 8 . The CyberNet framework Implementation.
Figure 8. Réalisation de la plate-forme CyberNet.

6 - CONCLUSION AND PERSPECTIVES

We have presented a tool for network management that takes advantage of virtual worlds interaction capabilities and the possibility of displaying large volumes of multidimensional information. We have designed a system that is able to automatically build virtual worlds for information display and is capable of handling highly dynamic data. The system is able to update in real time data modifications and does so without requiring user intervention.

A mechanism for collecting data in real time was implemented and the structuring of the collected information was presented. From this structured information, we have showed how we are able to dynamically build virtual worlds in order to display it.

Preliminary feedback from users showed that 3D metaphoric visualization seems promising. It eases problem detection and understanding. However, we are still looking for new visualizations that make the most of the opportunities of virtual environments. New interaction mechanisms also need to be studied. In particular, we want to provide ways to interact with the real world (e.g. kill a process on a workstation).

We have presented a flexible framework adaptable to other application domains. Although this work has been done within the context of network management, every application that involves the user to interact with huge amounts of dynamically structured information may be targeted (such as stock exchange, bank trading or web administration).

Several topics are currently being object of further study. Data characterization is one of them. Up till now, data values mappings onto visual parameters are hard-coded. We are developing a visual parameters taxonomy that, along with a coherent data characterization, will allow us to implement adaptors, and thus automatically map real world data values onto visual parameters, according to the current metaphor. A lot of work still has to be done in the field of user interaction and combined use of different tools. We also have to study how to add persistency to the system in order to be able to playback and analyze offline some crucial sequence of events. We also want to provide ways to interact with the real world (e.g. kill a process on a workstation).

Another topic of interest is the experimentation of our 3D based administration tools using virtual reality devices such as a head mounted display, gloves or a large barco screen although we feel skeptical about users acceptance. Finally, we envisage developing new metaphors and associated 3D glyphs and layout managers with special focus on their adaptive representation and navigation capabilities.

The CyberNet project is supported by France Telecom and the Eurecom Institute.

7 - REFERENCES

[1] Newman W. M. and Lamming M. G..*Interactive System Design*. Addison-Wesley Publishing, Wokingham, England, 1995.

[2] Crutcher L., Lazar A., Feiner S., and Zhou M.. Managing networks through a virtual world. *IEEE Parallel and Distributed Technology*, 3(2), Summer 1995, 4-13

[3] Kahani, M., Beadle, H. : "WWW-based 3D Distributed, Collaborative Virtual Environment for Telecommunication Network Management", Proc. Australian Telecommunication Networks and Applications Conference (ATNAC'96), December 1996, pp. 483-488.

[4] Cubeta, J., Egts, D. : VENoM - Virtual Environment for Network Monitoring" http://www.nrl.navy.mil/CCS/people/cubeta/venom/paper.html

[5] Becker, R., Eick, S., and Wilks, A. : "Visualizing Network Data", IEEE Transactions on Visualization and Graphics, 1:1 (March 1995), pp. 16-28.

[6] Eick, S., Wills, G. : "Navigating large networks with hierarchies", In *Visualization '93 Conference Proceedings*, pages 204-210, San Jose, California, 25-29 October 1993.

[7] Koutsofios, E., North, S., Keim, D. : "Visualizing Large Telecommunication Data Sets" IEEE Computer Graphics and Applications, Mai/June 1999 pp 16-19.

[8] Schaffer, E.; Reed, D. A.; Whitmore, S.; Schaeffer, B. Virtue: Performance Visualization of Parallel and Distributed Applications. Computer. December 1999. 44-51.

[9] Performance Co-Pilot Features http://www.sgi.com/software/co-pilot/features.html

[10] Atlas webpage http://www.geog.ucl.ac.uk/casa/martin/atlas/atlas.html

[11] Chen, C. Information Visualization and Virtual Environments. Springer-Verlag. 1999.

[12] Card, S. K., Mackinlay J. D., Schneiderman B. Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann Publishers. 1999.

[13] Gershon, N., Brown, J. : "Computer Graphics and Visualization in the Global Information Infrastructure", IEEE Computer Graphics and Applications, March 1996 pp 60-75.

[14] Robertson, G., Card, S. Mackinlay, J. : "Information Visualization Using 3D Interactive Animation" Communications of the ACM 36(4) 1993 pp 57-71

[15] Wright, W. : "Business Visualization Applications" " IEEE Computer Graphics and Applications, July/August 1997 pp 66-70

[16] Becker, B. : "Using MineSet for knowledge discovery" IEEE Computer Graphics and Applications, July/August 1997, pp 75-78.

[17] Rohrer R. M. and Swing E.. Web-based information visualization. *IEEE Computer Graphics and Applications*, 17(4):52–59, July/August 1997.

[18] CyberNet Project webpage http://www.eurecom.fr/~abel/cybernet

[19] Robertson G. G., Mackinlay J. D., and Card S. K. Cone trees: Animated 3D visualizations of hierarchical information. In *Proc. ACM Conf. Human Factors in Computing Systems*, CHI, pages 189–194. ACM Press, 28 April–2 May 1991.

[20] Gershon N. and Eick S. G.. Visualization's new tack: Making sense of information. *IEEE Spectrum*, pages 38–56, November 1995.

[21] Chuah, M., C., and Eick (S. G.). Information rich glyphs for software management data. *IEEE Computer Graphics & Applications*, 18(4), July – August 1998. ISSN 0272-1716.

[22] Fairchild K. M., Poltrock S. E., and Furnas G. W.. SemNet: Three-dimensional graphic representation of large knowledge bases. In Raymonde Guindon, editor, *Cognitive Science and its Applications for Human-Computer Interaction*, pages 201–233. Lawrence Erlbaum Associates, Hillsdale, New Jeresy, U.S.A., May 1988.

[23] Zhai S., Buxton W., and Milgram P.. The partial-occlusion effect: utilizing semitransparency in 3D human-computer interaction. *ACM Transactions on Computer-Human Interaction*, 3(3):254–284, September 1996.

[24] Hendley R. J., Drew N. S., Wood A. M., and Beale R. . Narcissus: Visualising information. In *Proc. IEEE Symp. On Information Visualization, InfoVis*, pages 90–96. IEEE Computer Soc. Press, 30–31 October 1995.

[25] P. Wyckoff, S. McLaughry, T. Lehman, and D. Ford. "T spaces". *IBM Systems Journal*, 37(3):454–474, 1998.