Using Virtual Reality for Network Management: Automated Construction of Dynamic 3D Metaphoric Worlds

C. Russo Dos Santos, P. Gros, P. Abel, D. Loisel
Eurécom Institute
2229, Route des Crêtes
F-06904 Sophia Antipolis - France
+33 (0)4 93 00 26 26
{russo,gros,abel,loisel}@eurecom.fr

J-P. Paris
CNET / France Télécom
905, rue A. Einstein
F-06921 Sophia Antipolis - France
+33 (0)4 92 94 53 00
jeanpierre.paris@cnet.francetelecom.fr

ABSTRACT

In this paper we will present CyberNet, an interactive 3D visualisation tool designed to study how virtual reality may enhance network management. The main focus of the paper is to point out the problems raised by the dynamic nature of the data and to present the solutions that were adopted to cope with these problems.

Keywords

Information Visualisation, Interactive Network Management

1. INTRODUCTION

Network management is a difficult and attention demanding task; large volumes of information must be analysed and relevant features must be extracted in order to make judicious decisions. The task is even more aggravated since we are dealing with highly dynamic information. We believe that virtual reality can bring a synthetic understanding of the states and tendencies of complex network systems.

The aim of the CyberNet project is to automatically build virtual worlds that represent some network services. As the network behaviour is constantly changing, the 3D virtual world will continuously evolve in order to represent the modifications that take place. Since we want to present all the information in a 3D virtual world, and as we want that world to be updated without requiring user intervention, the CyberNet project uses a distributed object framework composed of three distinctive parts:

- *The collecting layer* is used to gather the raw network data from the monitored devices.
- *The structuring layer* is the kernel of the system. It structures the raw information according to the service being monitored, and maps the structured information onto graphical components according to the metaphor chosen.
- *The presentation layer* is used to present the 3D world and provides the user interaction.

2. DATA COLLECTION

The basic components of the collecting layer are called *informators*. Informators are distributed agents that are responsible for collecting the raw network data. These agents are designed so that they "push" information to the upper layer of the system according to predefined policies. These policies optimise information transferred by the informators agents to the structuring layer and thus minimise the amount of information traded between layers.

When a user monitors a service, the system automatically locates (via a CORBA trading service) the distributed informators responsible for collecting the necessary data to monitor that particular service. After being located, each informator is configured to forward information to the structuring layer according to the defined policies. Once the configuration is completed, the agents start "pushing" every network modification that falls into these policies criteria to the structuring layer.

3. STRUCTURING INFORMATION

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, and dynamically update the graph according to real world modifications.

We have designed the structuring layer so that it produces a structured tree that may be easily mapped onto a 3D scene hierarchy. A *service* constitutes the main component of this layer. In order to construct the service graph, the raw data is structured into building blocks, called *entities*. Entities can further be arranged into groups or linked together by using another component of the structuring layer: *relations*.

Entities are the atomic building blocks used to model the managed services; they may represent physical devices or they may be conceptual. Once created, an entity contacts the informator(s) that is (are) able to provide the necessary data for its attributes values and subscribes to this data according to a given policy. The policy allows the entity to specify how, when and how often, it should be notified of an attribute's (new) value. The entity stores the values of its attributes and updates these values upon being notified by an informator of a value change.

Since the network status data is continuously changing, entities may have a longer or shorter life span. They may appear or disappear and the system has to reflect these changes by reacting whenever an entity is created or deleted. To cope with this high level of data dynamics the solution we have chosen is to use a

broker. Every entity has to register itself with the broker and every time an entity is deleted the broker has to be notified of its disappearance, thus keeping track of all existing entities.

Relations are used to group entities or model some common properties shared by entities. They provide the "glue" to construct the service graph, either by grouping entities together or by making visible a relation between entities or groups of entities. Relations may be structural (e.g., a process is related to the workstation it is running on) or functional (e.g., an NFS client/server relation between two workstations). A relation may involve any number of entities and an entity may be involved in any number of relations.

In order to cope with the life cycle of entities, when created a relation makes a query to the broker with the relevant parameters that describe the relation. The broker acts upon the query by searching all the registered entities that correspond to the query and returns this information to the relation. Along this procedure, the broker also maintains a record of the query. This way, if a new entity is created (or if an entity dies) that is pertinent to that query, the broker will forward this information to the relation. Thus, along with entities, the broker also keeps track of all the queries.

The role of a *service* is to build and maintain an acyclic graph (a tree) whose leaves are entities and intermediate nodes are relations. A service graph structures the information needed to manage a specific network service (e.g., NFS, printer service, etc.). The structure of the service graph evolves in time according to the entities and relations life cycles, hence reflecting the service status evolution in time. For this purpose the service issues queries to the broker demanding a specific type of entities. When the broker returns a matching entity, the service applies modifications to the current state of the structuring tree.

4. PRESENTATION LAYER

The presentation layer is responsible for displaying the virtual world and for providing navigation and interaction capabilities to the user. It is also the presentation layer that maps data values to visual parameters, via components that we have named *adaptors*. The virtual world is automatically built from the service graph. Each node/leaf of the graph is associated to a graphical element according to its type and to its position in the graph. There are two kinds of graphical components in the presentation layer: *3D glyphs* and *layout managers*.

The basic visualisation building blocks in our system are *3D glyphs*. In the CyberNet system, 3D glyphs are predefined 3D objects needed to construct the metaphoric virtual world and are associated with entities or relations. The system automatically maps the network data values onto the visual parameters of the 3D glyphs. In this mapping, the network data is translated from its real world range to the value range supported by the visual parameter. The 3D glyphs that build the virtual world are necessarily dependent on the metaphor used.

The presentation of large volumes of information needs to be organised in space in order to make its interpretation easier [1]. 3D glyphs are a first step towards organisation since they group multidimensional data. Layout managers are a step further in organising information spatially; the task of a layout manager is to arrange its graphical children in space. In CyberNet, layout managers are typically associated to relations and their children are 3D glyphs or both 3D glyphs and other layout managers. The visual parameters of a layout manager are the position and orientation in space in which it places its children. It can also use other visual elements, such as semitransparent bounding boxes, to enhance visual representation effectiveness, as suggested in [3].

In order to map information onto visual parameters, we designed another type of components: the *adaptors*. Adaptors implement a set of mapping rules between the structured information and the visual parameters of 3D glyphs and layout managers. The adaptors act hence as a kind of translators between the structured information and the graphical elements. The set of mapping rules implemented by adaptors are dependent on the metaphor used and thus on the visual parameters available. For each metaphor, there is a proprietary set of adaptors and a new set of mapping rules.

The user can interact in several ways with the system. Mechanisms for data clustering [2] are provided, as well as mechanisms for visualising data with different levels of detail, dependent on the distance between the user and the data being observed. Quantitative values, presented in tables, are available on user demand. The user can also create a new view to display a service from scratch or from some specific information encountered in the virtual world that is being visited. The CyberNet system also allows the user to change the visual metaphor used to display the information in the 3D world (e.g., from a solar system metaphor to a cityscape metaphor). This 3D world swapping may apply to a service being displayed at the moment, or to a new service that the user wants to visualise.

5. IMPLEMENTATION

The CyberNet project is implemented in Java, VRML and CORBA. The components of the collecting and structuring layers are written in Java and communications between the distributed elements are based on CORBA. 3D glyphs and layout managers run on the Java Virtual Machine of a VRML enabled browser and the worlds are dynamically updated using the External Authoring Interface (EAI). The end user may access the system from anywhere since it only requires a VRML enabled WWW browser.

6. CONCLUSIONS

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 the virtual worlds for information display and that takes into account the fact that the data being visualised is highly dynamic. The system is able to update data modifications in real time and does so without requiring user intervention.

7. REFERENCES

[1] Fairchild, K., Poltrock, S., and Furnas, G. SemNet: Three-dimensional graphic representation of large knowledge bases. In Cognitive Science and its Applications for Human-Computer Interaction, ed., Guindon, R., May 1988, 201-233.

[2] Hendley, R., Drew, N., Wood, A., and Beale, R. Narcissus: Visualising information. In Proc. IEEE Symp. Information Visualisation, Oct. 95, 90-96.

[3] Zhai, S., Buxton, W., and Milgram, P. The partial-occlusion effect: utilizing semitransparency in 3D human-computer interaction. ACM Trans. on Computer-Human Interaction, Sep. 96, 3(3):254-284.