# 3D Metaphoric Information Visualization

Cristina Russo dos Santos

Télécom Paris · Institut Eurécom

**Abstract**

In the Information Age the difficult part is not producing or even having access to information. Our society has become increasingly rich in information and new analysis tools that enable users to make sense of it are demanded. Visualization goes beyond using computer graphics to communicate information; visualization enables us to think. This thesis studies using three-dimensional (3D) metaphoric representations for visualizing large volumes of abstract data. Specifically, it focuses four main challenges in 3D information visualization: mapping the information, visualizing it, navigating the representations, and managing dynamics.

A key research problem in the visualization of abstract information, since it has no natural and obvious physical representation, is to discover visual mappings for representing the information and to understand what analysis tasks they support. We present a framework and a strategy for automatically mapping abstract information onto 3D metaphoric virtual worlds. We give several examples of implemented visual mappings that illustrate and validate our strategy.

The fact that the information mapping is done automatically allows us to create different visualizations on the fly. Furthermore, since our mapping strategy is based on metadata, our mapping engine is virtually application domain independent. We took advantage of both features to develop multiple views for different domain applications and research their usefulness in information visualization.

Now that available graphics computing power is not a hindering factor anymore, navigation is frequently seen as the weakest link in 3D virtual worlds. We introduce a novel concept of navigation – metaphor-aware navigation – and build on that concept to design and implement an integrated helped-navigation system that extends the traditional navigation systems present in most 3D viewers. We provide examples to illustrate our helped-navigation mechanisms, namely the metaphor-aware navigation.

The visualization of dynamic data is still a seldom explored issue in information visualization. In this thesis, we investigate the impact of real-time visualization of dynamic data and we propose strategies to cope with data dynamics across the entire pipeline of canonical information-visualization systems. The impact of data dynamics on the visualizations is attacked with more detail and we devise constraint propagation strategies to minimize the disruptive effects the dynamics may have on the stability of the virtual world.

**Résumé**

En cette ère des Technologies de l'Information, la difficulté principale n'est pas tant de produire ou d'accéder à l'information elle-même. Notre société est devenue de plus en plus riche en information et de nouveaux outils d'analyse qui permettent aux utilisateurs d'en extraire la pertinence sont exigés. La visualisation dépasse le simple fait d'employer l'infographie pour communiquer l'information; la visualisation nous permet de penser.

Cette dissertation étudie l'utilisation de représentations métaphoriques tridimensionnelles (3D) pour visualiser de grands volumes de données abstraites. En particulier, elle se focalise sur quatre défis principaux dans la visualisation 3D de l'information : le mappage de l'information, la visualisation, la navigation dans les représentations, et la gestion des données dynamiques.

Puisque l'information abstraite n'a par définition aucune représentation physique naturelle et évidente, un des problèmes de recherche principaux dans le domaine de la visualisation de l'information abstraite est de découvrir des mappages visuels pour représenter l'information et de comprendre quelles tâches d'analyse elle supporte. Nous présentons un cadre et une stratégie pour mapper automatiquement l'information abstraite sur des mondes 3D virtuels métaphoriques. Nous donnons plusieurs exemples de mise en oeuvre de mappages visuels qui illustrent et valident notre stratégie.

Le fait que le mappage de l'information soit fait automatiquement nous permet de créer différentes visualisations à la volée. En outre, puisque notre stratégie de mappage est basée sur des méta-données, notre moteur de mappage est pratiquement indépendant du domaine d'application. Nous avons tiré profit de ces deux qualités pour développer des vues multiples pour différents domaines d'application et pour rechercher leur utilité dans la visualisation de l'information.

Maintenant que la puissance de calcul graphique disponible n'est plus un facteur restrictif, la navigation est fréquemment vue comme le maillon faible des mondes 3D virtuels. Nous présentons un nouveau concept de navigation – "la navigation metaforiquo-consciente" – et nous nous fondons sur ce concept pour concevoir et mettre en oeuvre un système intégré d'aide à la navigation qui étend les systèmes traditionnels de navigation existants dans la plupart des visualisateurs 3D. Nous fournissons des exemples pour illustrer nos mécanismes d'aide à la navigation, en particulier la navigation metaforiquo-consciente.

La visualisation des données dynamiques est encore un sujet peu exploré dans le domaine de la visualisation de l'information. Dans cette dissertation, nous étudions l'impact de la visualisation en temps réel des données dynamiques et nous proposons des stratégies pour faire face à la nature dynamique des données pour l'ensemble du processus de traitement des systèmes canoniques de visualisation de l'information. L'impact de la dynamique des données sur les visualisations est traité avec plus de détails et nous concevons des stratégies de propagation de contraintes pour réduire au minimum les effets perturbateurs que la dynamique peut avoir sur la stabilité du monde virtuel.

**Acknowledgments**

I first of all wish to thank Pascal Gros, my research advisor, for having provided me with excellent advice and expertise during the duration of my thesis. I am greatly indebted to him. I also want to express my thanks to Christian Wellekens, my thesis supervisor, for having supported me as his Ph.D. student.

I wish to acknowledge my thesis committee for their insightful comments. I am especially indebted to Don Brutzman for his careful reading of the manuscript. His numerous comments, suggestions, and corrections have substantially improved the quality of the text.

Furthermore, I wish to acknowledge the financial support of the Portuguese program for advanced education PRODEP. I am grateful to João Paulo Baptista for his help and support during my leave of absence from the Polytechnic Institute of Porto.

I gratefully recognize everybody at Eurécom for always making me feel privileged to work there. I am especially grateful to Alain for his friendship and his endless patience during the final months of my work. I also wish to thank Jussi for his technical advice and friendship.

I am thankful to Arnd, Carine, Despi, Giuseppe, Isa, Jan, Karim, Max, Paula, Pedro, Pierre, Sérgio, and Ulla for their friendship and for making my life in France better.

Special thanks to Bernhard for his also special support.

Finally, I wish to extend my deepest gratitude to my family, especially to my parents, for providing me with enduring love and support.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

> As we move in fits and starts from a world of words to a world of images, there are many ironies and paradoxes. One paradox is that visual thinkers – who (...) often have difficulties with word-based educational systems – may find themselves to be among those best able to lead others in solving the increasingly pressing problem of our age – that is, quickly and effectively navigating oceans and oceans of information. (West, 1998)

## 1.1 Motivation

In the Information Age the difficult part is not producing or even having access to information; finding and retrieving relevant information amidst the tons of information available is the hard part. Lots of efforts are being taken to tackle this problem. More powerful and intelligent search systems are developed in an attempt to simplify the problem of data mining. But that is not the sole hardship and it leads directly to the following question: once we have access to the data, how can we take advantage of it?

### Visualization

Visualization is one of the best answers. Thomas Erickson argues in (Erickson, 1993) that it is no coincidence that visual terms are used as a pervasive metaphor for indicating understanding, giving as example "I *see* what you mean" and "*insight*", amongst others.

*Visualization* is defined in (McCormick et al., 1987) as "the study of mechanisms in computers and humans which allow them in concert to perceive, use and communicate visual information". In (Card et al., 1999) visualization

is defined as "the use of computer-supported, interactive, visual representations of data to amplify cognition", and some pages further we come across yet another definition: "visualization can be described as the mapping of data to visual form that supports interaction in a workspace for visual sense making". Robert Spence, in (Spence, 2001), states that (information) visualization is about "forming a mental model of data, thereby gaining insight into that data".

The common line of all the above definitions is the idea of perception, cognition, making sense of data, or gaining insight into data. The aim of visualization is thus to represent and communicate data in a way that makes it (more) perceptible. That is why perceptualization is probably a better word, as argued in (Erickson, 1993), although the term visualization has been already coined and its use is widespread.

At this point there is a distinction that should be made between information visualization and scientific visualization. Most of the early research in the field of visualization has dealt with representing scientific data – *scientific visualization*. In scientific visualization "what is seen primarily relates to, and represents visually, something *physical*" (Spence, 2001). That is, in scientific visualization the data frequently has a real world visual representation or some kind of inherent spatial concept, that makes its visual representation a natural one. Common examples of scientific visualization are brain scans, fluid visualization, earth models, or DNA visualization.

### Information visualization

*Information visualization* is a more abstract concept. The data to be visualized may have no natural physical representation and its complexity is increased due to the enormous amount of data generally available. Card et al., define information visualization by extending the definition given for visualization; hence, information visualization is defined as "the use of computer-supported, interactive, visual representations of *abstract* data to amplify cognition" (Card et al., 1999).

The fact that information visualization deals mostly with abstract data does not imply that it has no real meaning. It only means that the visualization will not forcefully be enhanced by the representation of the physical entity the data relates to. As Robert Spence argues in (Spence, 2001) a currency trader knows exactly what a dollar bill looks like and does not need to see an image of a bill while trading.

There exist obvious overlapping between information and scientific visualization. Furthermore, some of the techniques used in scientific visual-

ization may also be used in information visualization and as successfully. Nonetheless, we would like to stress that this thesis relates to information visualization and does not address scientific visualization specific issues.

In information visualization, the abstract data to be represented can be structured – in hierarchies or networks, for instance – or even possess no structure at all. Abstract information is thus not inherently spatial. Yet, since we live in a physical world, it is easier to convey information if it is displayed in the relatively familiar physical space.

### 3D Metaphoric information visualization

Visualization is a cognitive activity (Ware, 2000), and the human conceptual system is fundamentally metaphoric in nature (Lakoff and Johnson, 1980). It seems thus natural to ally visualization with metaphoric representation. Moreover, since in information visualization the data has usually no natural real world representation, a visual metaphor provides a way to map the abstract information onto a structured visual representation.

*Metaphoric information visualization* deals thus with visualizing abstract data with the support of visual metaphors. The use of metaphors in information visualization allows for facilitating data comprehension by drawing from a user's a priori experience from the real-world (Lakoff and Johnson, 1980). The almost omnipresent desktop metaphor in our computer screens is an example of this, though its behavior does not exactly match the real-world counterpart, and it has been mapped onto a two-dimensional interface.

Two-dimensional information visualization makes use of visual parameters (e.g., color) to represent data characteristics and properties. Three-dimensional (3D) information visualization adds another dimension to the data representation, thus making it possible to make a more efficient use of the limited real estate available – the computer screen. This is critically important when we are dealing with large quantities of data to be displayed.

Furthermore, three-dimensional visualization also facilitates our understanding of the data by making use of new visual parameters (e.g., transparency) and interaction techniques (e.g., navigate through the data) that invite users to explore and manipulate large complex information systems. These information systems can range from library catalogs to the market stock quotations of the day or archives of airline flights.

The rooms metaphor (Henderson, Jr. and Card, 1986) was proposed as the natural three-dimensional evolution for the two-dimensional desktop metaphor. The rooms metaphor intended to give users a sense of space they could perform their tasks in. Other spatial and physical metaphors, such as a

cityscape (Gershon and Eick, 1995) or a garden (Crossley et al., 1997), have also been proposed for visualizing large collections of abstract information.

Thus *3D metaphoric information visualization* can be described as the visualization of abstract data with the help of spatial metaphors. Therefore, this thesis positions itself in the field of three-dimensional information visualization.

The motivation for this dissertation may be formulated as a question: can 3D metaphoric information visualization be a helpful instrument for understanding and analyzing large data volumes of abstract information? Furthermore, how can we foster the construction of rich mental models that allow the user to effectively gain insight into the data?

In fact, we can see the broad motivation of this thesis as providing an answer to a variant of the question in the very first paragraph: how can we take advantage of 3D metaphoric information visualization to better comprehend and analyze large volumes of dynamic abstract data? With this broad goal in mind we attack several open issues in this research topic, as will be described in the next section. We intend to prove that yes, 3D metaphoric information visualization helps gaining insight into abstract data.

## 1.2 Problem statement

Information visualization is pervasive in our lives. We cross it daily in the traffic lights. We can see it in the television in a more or less sophisticated environment when the weather report appears. And we can use it as an aid to solve an apparently prosaic problem: the monk on the mountain (Erickson, 1993):

> One morning a monk awake and decided to make a pilgrimage to the top of a nearby mountain. At 6 A.M. he began climbing a path that led from the foot of the mountain to its peak. After spending the night on the mountain top, he arose at 6 A.M. and began retracing his steps, following the path back to its beginning. The question: was there any point on the path where the monk was at the same time on each day?

This same problem appears also with a slightly different phrasing at (The monk on the mountain, 2000).

Many people have difficulty finding a quick answer if they consider the problem mathematically, for instance. But transform it into a graphical problem and the answer is immediate: yes, there is a place when the monk was at the same time each day (Figure 1.1).

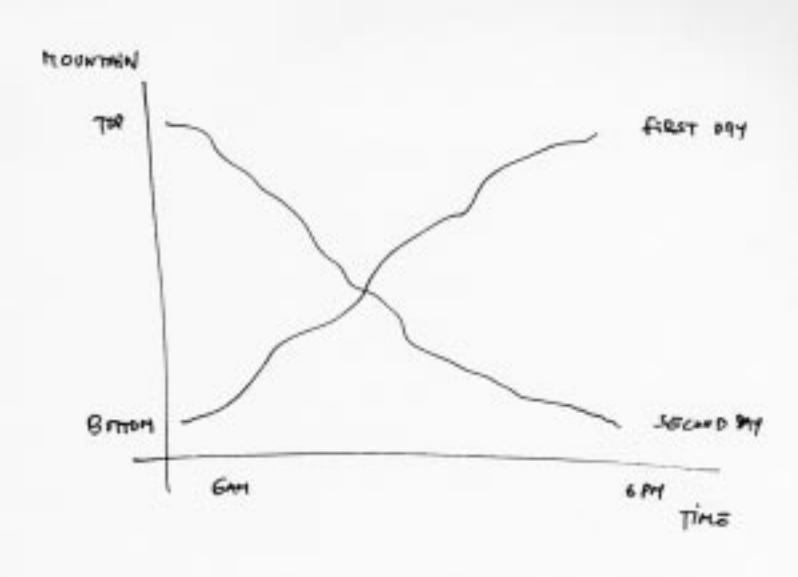



Figure 1.1: The monk on the mountain problem.

In our view, this problem epitomizes in a way the goal of information visualization: gaining insight from one (or more) image/visual representation. Clearly, the same problem could be solved analytically. Also, if we were to analyze an actual path and were trying to find the exact location at which the monk was at the same time both days, the graphs would have to be exact and quantitative details had to be plotted. Nonetheless, a simple quite naïve sketch as the one in Figure 1.1 allows us to immediately answer the question above: yes, there was!

So, we see how even non-experts can use information visualization to solve more or less complex problems instantly; we have seen how information visualization is pervasive in our lives. However, information visualization as a computer science research subject is still fairly recent. Although seminal work in the area, that is still constantly cited, dates from some decades ago, such as the book from Jacques Bertin (Bertin, 1967), most of the work done in the field is quite recent – one look through the references for this thesis and we notice that more than three quarters have dates since the nineties. There still are lots of open issues in the field of information visualization. In this thesis we try to shed some light on some of those issues.

The thesis approach relates a lot to the monk on the mountain problem in the sense that we study how 3D metaphoric information visualization may help us gaining immediate insight into analyzing large volumes of abstract

data. As in the monk on the mountain problem, sometimes that insight has a more qualitative than quantitative nature; and likewise, further analysis is required for a more quantitative evaluation. But the clarification is usually immediate. The kind of enlightenment that makes you emphatically exclaim "ah, now I *see* it!"

This work focuses four main axes in the research domain of information visualization.

### Mapping

One of the main issues studied in this thesis is the visual mapping of abstract information. Contrary to scientific visualization where the data has an intrinsic natural representation, information visualization deals with abstract data, which as its name indicates does not necessarily possess a real-world representation – we deal with abstractions that have some data values that can be visualized. For instance, how do we represent a process on a computer? The process itself has no physical representation, but there are some values relating to the process, such as the process owner, CPU usage, memory usage, and so on, that can be represented visually. How we devise this mapping, and with what criteria, is one of the major subjects tackled in this work. We further investigate ways of automating the mapping process.

### Multiple views

Even in our daily discourse the idea of different perspectives and different points of view is frequent; moreover, it is seen as desirable, especially when there is a decision to be taken or an evaluation to be made. In information visualization, multiple views and different perspectives are also broadly seen as an attractive feature. Still, this is a recent issue (due to the latest computing developments that made multiple views feasible?) and there is a demand for guidelines that pave the way for their usage (Baldonado et al., 2000). We tackle the multiple views issue using an empirical approach, by proposing different solutions and empirically evaluating the added value of each.

### Navigation

Now that available graphics computing power is not a hindering factor anymore, navigation is frequently seen as the weakest link in 3D information visualization (West, 1998). Navigation in electronic media has reportedly an history of difficulty – hence the term "lost in hyperspace" (Conklin, 1987).

Nevertheless for 3D virtual worlds, even power users report sometimes getting lost in the 3D virtual space. Furthermore, there is no standard way of navigating virtual worlds. In this thesis we attack the subject of navigation in 3D metaphoric worlds. We further investigate how the metaphor influences and drives user navigation in virtual worlds. We also attack the problem of combining semantic and physical navigation in the same world.

#### Dynamics

The visualization of dynamic data is still a seldom-explored issue in information visualization. Frequently, in related work, when dynamic visualization is referred is *a posteriori* analysis of dynamic data – i.e., data that has been recorded over time and is then played back with some kind of animation. In this thesis, we investigate the visualization of dynamic data in real-time. Some of the related issues attacked are: how to map dynamic data in real-time; how to update dynamic worlds on the fly; how to propagate data changes in order that the consistency of the virtual world is not endangered; and what kind of constraints dynamic data imposes for navigation and interaction within virtual worlds.

## 1.3 Contributions

This dissertation makes several contributions, which may be synthesized in the exploration of a new field: using 3D metaphoric worlds for real-time monitoring of large dynamic abstract data sets.

In relation to the four research axes introduced in Section 1.2 – respectively mapping, multiple views, navigation, and dynamics – this dissertation's main contributions may be summarily cited as follows:

- we propose a dynamic mapping automation engine for the visualization of abstract information in 3D metaphoric worlds; we elaborate a characterization of the information to be visualized, the user tasks, and the virtual worlds used for presenting the information (Chapter 4)
- we present a detailed analysis and experimentation of the use of multiple views, validated through several visualization tools for different applications, namely in the field of network monitoring (Chapter 5)
- we introduce the novel concept of metaphor-aware navigation and we design and develop an assisted-navigation system for user navigation in 3D metaphoric worlds that builds on that concept (Chapter 6)

- we elaborate different strategies to cope with data dynamics in real time, across the entire pipeline of canonical information-visualization systems (Chapter 7)

A detailed description of this dissertation contributions is further presented in the concluding chapter, Chapter 8.

## 1.4 Structure

This dissertation consists of eight chapters, including this introductory chapter.

Chapter 2 introduces some background information for the rest of the thesis. We define the research scope of this thesis and we provide justification for some of the *a priori* research options.

In Chapter 3 we present the research context in which this work was carried out. We describe a development framework for the construction of visualization tools.

Chapter 4 attacks the challenge of visually mapping abstract information. We use metaphoric worlds for the information representation and we present a complete strategy for the mapping process, starting with the data characterization and ending with the actual display of the information.

In Chapter 5 we discuss the utility of multiple views within the same visualization tool. We give numerous examples of the utilization of multiple views and present the motivation for each one of them. We also study some interaction mechanisms, with a particular focus on the selection mechanism.

Chapter 6 deals exclusively with user navigation in 3D metaphoric worlds. We introduce a novel concept of metaphor-aware navigation and devise an integrated helped-navigation system that extends the traditional navigation systems present in most 3D viewers.

In Chapter 7 we focus on our last research axis: visualizing dynamic information in real-time. The visualization of dynamic data in real-time has a traversal impact on the whole visualization system, from the data collection, through the information structuring, to the visualization of the information. We devise specific dynamics handling strategies for each one of the impacted areas.

Each one of these five central chapters presents the motivation and problem statement for the research contained in it, and extensive related work and conclusions. Finally, Chapter 8 summarizes the results and contributions of this dissertation and concludes with a description of potential areas for further study.

# Chapter 2

# Information Visualization Overview

> **in·for·ma·tion:** the communication or reception of knowledge or intelligence (Merriam-Webster OnLine, 2002)
>
> **vi·su·al·i·za·tion:** formation of mental visual images (Merriam-Webster OnLine, 2002)

## 2.1 Introduction

Definitions for information visualization are abundant in related literature (Card et al., 1999) (Spence, 2001) (Ware, 2000) (McCormick et al., 1987) (Foley and Ribarsky, 1994) (Mackinlay, 2000). Still, we used the traditional method of looking up the words' meaning in the dictionary and obtained interesting results. According to the Merriam-Webster's dictionary (Merriam-Webster OnLine, 2002) information visualization would then be the active transmission of knowledge via the formation of mental visual images. This agrees with visualization being a cognitive or perceptual activity (Ware, 2000), and information visualization's goal being augmenting the comprehension or amplifying the cognition (Card et al., 1999).

Information visualization understood as a computer science research field is still fairly recent though, which might explain why the dictionary definition lacks a mention of the computer. This research field studies how we can take advantage of computer-generated, interactive visual representations of abstract data to facilitate and augment users' comprehension of the data. This chapter briefly introduces the research field of information visualization. We then use this chapter to justify some our research options regarding information visualization. We discuss the utilization of visual metaphors.

We also argue why use three-dimensional information visualization. The fact that we have opted for non-immersive environments will also be dissected. Some examples of commonly disseminated visualizations are also given.

### 2.1.1 Related bibliography

This chapter, however, is not intended as a tutorial in information visualization. Nor as an exhaustive state of the art report in the field of information visualization. As the research in this area gains some maturity, more and more related bibliography is available.

For an introduction to the field of information visualization refer, for instance, to (Ware, 2000) (Card et al., 1999) (Spence, 2001) (Chen, 1999) (Keller and Keller, 1993). Some seminal work that should not be overlooked, independently of their publications' date are the Jacques Bertin's book (Bertin, 1967) (also published later in English (Bertin, 1983)) or the beautiful books from Edward Tufte (Tufte, 1983) (Tufte, 1990) and (Tufte, 1997). Some tutorials or course notes on (information) visualization or are available on-line; for instance (Young, 1996) (Domik, 1996) (Andrews, 2000) (North, 2001). In addition, existing position papers give a more or less detailed overview to the field, such as (Keim, 2002) (Mackinlay, 2000) (Gershon and Eick, 1997) (Card, 1996) (Gershon and Eick, 1995), among others.

### 2.1.2 Chapter organization

The remainder of this chapter is organized as follows: Section 2.2 defines visualization as a computer sciences research field. Section 2.3 introduces scientific visualization and Section 2.4 is dedicated to information visualization. In Section 2.5, Section 2.6, and Section 2.7 we argue and justify our use of visual metaphors, three-dimensional visualization, and non-immersive virtual worlds, respectively. Section 2.8 gives some examples of information visualization techniques and applications. In the last section, Section 2.9, we present some conclusions.

## 2.2 Visualization

Visualization, in computer science, is described as using computer graphics to gain insight into information. Visualization should not be confounded with the discipline of graphics presentation. The latter aims essentially to communicate information in a fast and easy way. Information visualization main objective is to represent data in a way that helps the user understand its meaning.

Presentation is primarily concerned with communicating through images, while in visualization the images are there to make us *think*. That is the reason why information visualization usually presupposes some user interaction, whereas in the case of graphics presentation this is usually not the case. Mackinlay (Mackinlay, 2000) further associates the power of visualization with quantity: visualization allows us to think about more cases, more variables, more relations.

The research field of visualization has greatly evolved in the nineties and at present state can be divided, *grosso modo*, into two research subjects: scientific visualization and information visualization. Scientific visualization focuses primarily on physical data, whereas information visualization focuses on abstract, nonphysical data.

## 2.3 Scientific visualization

Scientific visualization is thus chiefly associated with physical data. Typical applications of scientific visualization may be found in medicine (e.g., brain scans), in physics (e.g., fluid visualization), in biology (e.g., molecular visualization), or in geophysics (e.g., seismic wave propagation over a earth model).

Surely both fields of scientific and information visualization overlap and many visualization techniques may be used in both cases. The main difference to retain is that scientific visualization involves the display of a physical thing, while in information visualization, even if the data relates to physical objects, their representation is usually not necessary and many times is completely irrelevant. Robert Spence gives a clear example in (Spence, 2001): a study of baseball statistics does not require an image of a baseball, whereas flow in a pipe is usually best represented in the context of the pipe itself.

This thesis does not address issues related specifically to scientific visualization; nevertheless, some of the issues attacked in the context of information visualization are also valid in the case of scientific visualization.

## 2.4 Information visualization

Information visualization, as opposed to scientific visualization, deals essentially with abstract, nonphysical data that has, in general, no natural visual representation, such as text, hierarchies and statistical data. As we have seen in Chapter 1, independently of the multiple definitions, the emphasis in information visualization is mainly over two axis: it is a cognitive activity

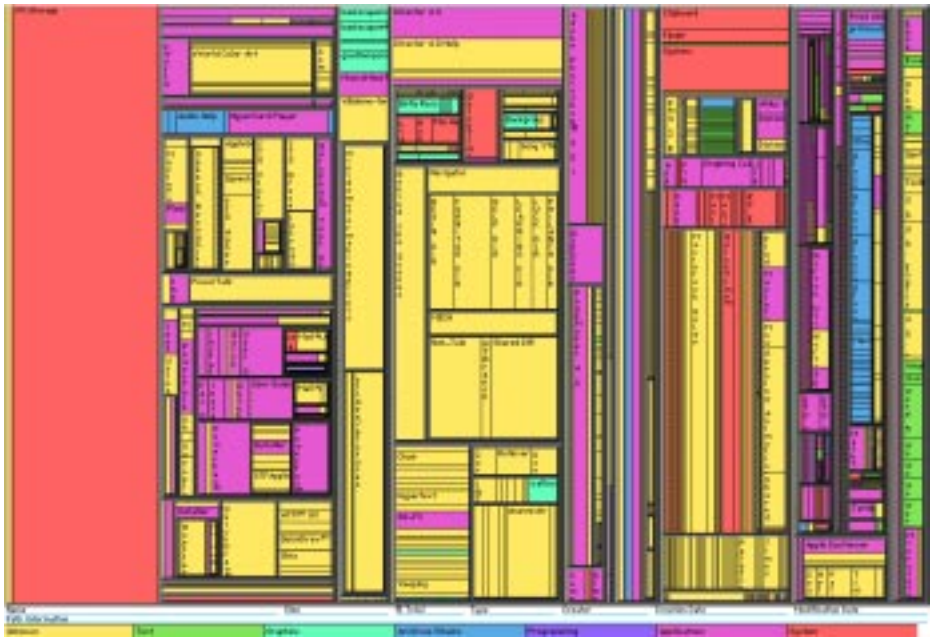


Figure 2.1: Treemap for visualizing disk occupancy. Source: (Treemaps, 2002).

(images that make us think!) and it aims to give us insight into the data.

One main challenge for information visualization, as it focuses mainly in abstract, nonphysical data is to design a visual representation. There are essentially two approaches to this challenge (Mackinlay, 2000).

One approach is to map the nonphysical data onto the visual parameters of points, lines, and areas. Figure 2.1 shows an example of this approach: a treemap used for visualizing disk occupancy. This is a visualization invented by Ben Shneiderman and first published in (Johnson and Shneiderman, 1991) and is a technique for visualizing hierarchies in a constrained space. In the first implementation shown in Figure 2.1 the screen is split into rectangles in alternating horizontal and vertical directions as the disk is traversed down. Since then, the treemaps have been used for various different applications, ranging from visualizing a tennis match to financial data analysis, and its algorithms have been refined over the time. A very recent version is available for download from the treemaps' website (Treemaps, 2002).

The other approach is to map the abstract data onto a virtual object or collection of objects. These objects can then be explored as if they were physical objects. Figure 2.2 shows an example of this approach. Figure 2.2 depicts an example of the information cube, a visualization technique in-

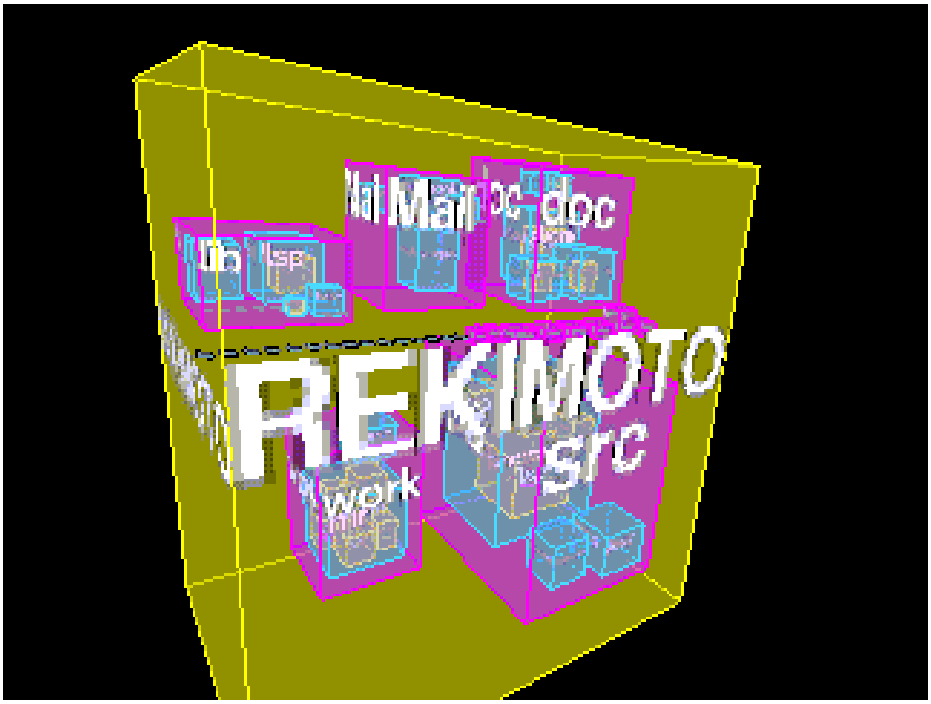



Figure 2.2: Information cube displaying a Unix directory structure. Source: (Rekimoto and Green, 1993).

vented by Jun Rekimoto (Rekimoto and Green, 1993), also targeting the display of hierarchical information. The idea is that users are already familiar with the concept of nested boxes and will be able to recognize and inspect the information structure intuitively. The semi-transparency controls the complexity of the information presented to the user in a way that is also easily recognizable.

The latter approach for mapping abstract information is thus using a *metaphor* – something that the user is able to relate to due to *a priori* knowledge. This is the approach that we have adopted for our work.

## 2.5 Metaphors

The primary goal of information visualization is thus to facilitate user understanding of vast amounts of abstract data. As we have seen, and as opposed to scientific visualization, information visualization aims to create visual representations of data that usually has no natural visual representation. Consequently, and in order to derive meaning from the data, informa-

tion visualization requires an additional step in the application architecture to create a comprehensible visual mapping. In our approach, this mapping makes use of a visual *metaphor* to represent the information.

A metaphor is defined as a rhetorical figure whose essence is understanding and experiencing one kind of thing in terms of another (Lakoff and Johnson, 1980). The use of metaphors in information visualization is important due to the possibility of letting users draw from their previous knowledge to better understand the metaphoric representation. Furthermore, the use of metaphors allows the introduction of new concepts by building on previously acquired knowledge (Newman and Lamming, 1995) (Lakoff and Johnson, 1980). Hence, a metaphor makes it possible for the user to move from familiar concepts to unknown ones, in this way incorporating new knowledge into known concepts.

Metaphors can sometimes be a mixed blessing though (Halasz and Moran, 1982). The user may sometimes expect that his real-world knowledge will fully transfer to the virtual world. This is usually not the case as the virtual world representation most frequently does not implement all the affordances of the metaphor. A good metaphoric representation should hence foster the maximum transfer of the user's real-world knowledge to the virtual environment. This should be done in a way that clearly shows what are the affordances that are transferable from real-world experience, so that the user's expectations are not frustrated. Section 4.6 in Chapter 4 delves further into the use of visual metaphor in information visualization and provides numerous examples of possible metaphors.

## 2.6 Three-dimensional (3D) visualization

Now that latest developments both in the hardware and software industry have made three-dimensional (3D) graphics ubiquitous at everyone's desktop, there is the risk of seeing 3D as the solution for everything in information visualization. 3D information visualization takes advantage of another dimension to represent information. But is three-dimensional visualization always useful?

Scientific visualization often uses 3D graphics. This is only natural since the scientific data has a natural physical representation that is usually reproduced in the computer screen. However, in information visualization, since the data is mostly abstract, it could eventually be shown in either 2D or 3D. There should be some added value to be gained from a 3D representation that justifies the higher developments costs, more complex navigation, and the increase in computing resources.

George Robertson divides the advantages of interactive systems in two broad categories of enhanced human capabilities (Mullet et al., 1995): the extension of perceptually oriented capabilities and the enhanced support of human activity. Since 3D contributes to these effects it is a benefit. Adding to the motivation for using 3D, Kim Fairchild states (Fairchild, 1993) that if objects are placed in a 3D display as opposed to a 2D display, the perceived complexity of the information is reduced, for large collections of complex objects. Also in (Ware and Franck, 1994) the same statement of complexity reduction is made for viewing graphs.

3D visualization also provides an additional dimension for showing data, which is especially important in the case of information visualization, since it usually deals with large volumes of multidimensional data. Furthermore, the visual vocabulary available for representing the data in 3D is augmented (e.g., transparency, depth, superposition, etc). For instance, the cube visualization, in Figure 2.2, takes full advantage of transparency.

An added value of using 3D representation is the fact that the use of three dimensions allows for a natural zooming and level of detail mechanism. Depth conveys distance. Thus, things in the foreground are in the user's attention focus and the perspective view makes objects nearer to the user's viewpoint appear larger. This helps the user analyzing close objects more effectively and allows for putting more or less emphasis on data based on it being mapped on the foreground or on the background (see examples in Section 2.8.1).

The use of 3D also promotes real world resemblance; in other words, real world entities can be shown (almost) as they are (e.g, the New York Stock Exchange virtual trading floor in Figure 2.10). This is important for constructing real-world based visual metaphors, since it helps the user recognize the metaphor and its components. Besides, animation, made more realistic and possible in more directions than in a 2D display, helps conveying dynamic information or sifting through large volumes of information.

But there also drawbacks to the use of 3D. For one, the development costs are bigger. Additionally, it often requires specialized graphics hardware, though these are becoming more and more popular on everyone's desktop. Another drawback of using 3D is that the perspective and texture mappings usually degrade font quality – which is often a key aspect of information visualization (Mackinlay, 2000). Finally, there is also the risk that the technical virtuosity of the 3D representation may obscure the underlying structure of the information.

In (Mullet et al., 1995) Kevin Mullet suggests that any visualization whose third view dimension does not carry a data dimension would be better

served by an optimized 2D display. We follow this recommendation.

Our approach is to use 3D metaphoric representation for visualizing large volumes of abstract data. This option is justified in the reasons given above, especially for two particular reasons: the extra dimension allows for the representation of larger volumes of information that would be unfeasible in 2D; and 3D allows us to use real-world based metaphors to depict information in a way that the user may easily relate to.

We believe that the use of 3D allows for better immediate insight into the data, as well as a global overview of the whole information. However, we also acknowledge that this insight has very frequently an inherent qualitative property. In view of this, we provide quantitative details on demand, in the form of a 2D pop-up interface. This agrees with the above suggestion of not using a third dimension unless it serves a purpose, such as mapping additional information.

Furthermore, we live and act in a 3D environment and 3D may indeed enhance the representation of information. Users seem to prefer 3D to 2D interfaces even when there is no evidence of gained usability (Cockburn and McKenzie, 2001). In fact, 3D per se is not a warranty of usability. According to Don Norman, the spatial organization of information is only effective when, amongst other factors (Norman, 1993): there is a natural mapping between the mapped data and the spatial location; desired data can be located in a minimum of attempts and the amount of time and work required to try a location, inspect its content and then try another location is small.

We try to answer the first requirement by using metaphors for information mapping, as is described in Chapter 4 – this allows to attach meaning to shapes and locations in a way that is comprehensible for the user. The second requirement is addressed by our navigation scheme – described in Chapter 6 – which fosters rapid location of items by providing different ways for target selecting, and provides a powerful mechanism of helped navigation to traverse the whole virtual world.

As a closing remark on the use of 3D: although we are more used to 2D user interfaces it is reasonable to imagine that 3D interfaces may develop and bring new capabilities to enhance user performance. The first user interfaces were two-dimensional and text based. They were later superseded by 2D windowing systems, based on the desktop metaphor. This new desktop systems blossomed later into new ways of interacting and arranging information in space, such as the multiple virtual screens. In all likelihood then, 3D user interfaces will also blossom into new interaction paradigms. Such growth and normalization of diverse 3D metaphors will be encouraged by the widespread availability of 3D graphics on the Web.

## 2.7 Non-immersive virtual worlds

Virtual reality is usually related to full immersion: the user is placed in a three-dimensional environment and wears special equipment that lets him directly manipulate the environment. Non-immersive virtual reality also places the user in a 3D environment but uses a more conventional set-up: a computer screen, a keyboard, and a mouse.

The advocates of full immersion often argue that its advantage is obvious: only through it can the user effectively experience the full paradigm of a virtual worlds. However, a quick visit to any game arcade shop around the corner would make us think otherwise. There we see players completely immersed in the game's virtual world, even if their interaction devices are sometimes nothing more than a couple of buttons. And there are several advantages of non-immersive interfaces over immersive virtual worlds. (Robertson et al., 1993b) evaluates these advantages in three different types: evolutionary, technical limits, and use.

Evolutionary advantages relate essentially to the use of familiar tools – display, keyboard, and mouse – and to lower initial costs. Another advantage of non-immersive virtual has to do with limitations of immersive technology, that sometimes deviates some of the development effort that should focus on interaction to solving technical shortcomings. The final advantage relates to ease of use. Users will probably be reluctant to wear special (and frequently cumbersome) equipment, specially if it shuts down the exterior world. Non-immersive virtual worlds do not require any special equipment or gadgets and do not prevent users from seeing what is around them.

Our approach was to use what is sometimes referred as traditional tools – computer screen, keyboard, and mouse – to explore our metaphoric virtual worlds. Our reasons encompass the advantages of non-immersive over immersive virtual environments described in the last paragraph. Additionally, our visualizations tools are intended for an audience that has other tasks to attend to and wants a smooth switching mechanism (e.g., pop up a monitoring window in the same screen). The use of immersive virtual worlds presupposes exclusiveness to other tasks not executed from within the immersive environment.

## 2.8 Examples

This section presents some examples of information visualization applications. The examples are not intended as an overview to what exists in the field. Their purpose here is either to show a point (e.g., how to take ad-

vantage from the natural perspective of 3D), to illustrate a concept (e.g., metaphoric visualizations) or to give an idea of existing commercial applications. For a detailed overview to information visualization, please refer to the bibliography cited in Section 2.1.

### 2.8.1 Natural zoom in 3D

As we have referred in Section 2.6 one of the motivations for using 3D is the natural zooming and level of detail effect available. There has been some research work that attempts to solve the difficulty of giving a detailed view while still maintaining context in 2D visualization of large volumes of information.

Namely, the work of Sarkar and Brown who implemented the concept of fisheye views introduced by Furnas (Furnas, 1986) to the visualization of 2D graphs (Sarkar and Brown, 1992). Sarkar further developed the rubber sheet technique (Sarkar et al., 1993) which allows for multiple foci and lets the user control the amount of space allocated to each focus.

The techniques mentioned above attempt to solve the problem of visualizing vast amounts of data, giving simultaneously a detailed view while keeping the context. They are known as focus + context visualization techniques. In 3D this problem of providing focus + context is eased naturally, since perspective allows for a natural mechanism of providing detail (for objects closer to the user's viewpoint) while still showing the remaining data on the background. The next two examples, the perspective wall and the sphere visualization, illustrate this point.

#### The perspective wall

The perspective wall is a technique developed by (Mackinlay et al., 1991) for viewing large information volumes, ordered along a single dimension, such as directory entries ordered alphabetically. Typically, this a case where the width of the information structure is much smaller than its length. This technique wraps a 2D layout around a 3D wall and is an example of the use of 3D to enhance a traditional 2D interface.

The perspective wall is constituted by a panel in the center to provide a detailed view of the region of interest, and one perspective panel on each side to provide context for the overall data set. The region of interest remains in the center, so that the observer can always have a detailed view. This technique allows for the display of an entire layout in a single view. It integrates a detailed view while retaining information context. A shortcoming of the


Figure 2.3: Perspective wall displaying files ordered by last modified time. Source: (Mukherjea and Foley, 1995)

perspective wall is that it can only handle information ordered along a single dimension. Figure 2.3 shows an example of a perspective wall.

### Sphere visualization

The sphere visualization technique is part of the VisNet (Fairchild et al., 1993) visualization tool and is used to represent associative relationships in a graphical manner. There is an object of interest and all the relationships associated with it are displayed. Figure 2.4 shows an example of a sphere visualization. The structure of the sphere resembles that of an onion: spheres are embedded within spheres. This allows for the representation of different levels of information, as objects that are closed to the object of interest are represented in the same sphere (i.e., the outermost one) and objects not so closely related to the object of interest are represented in the inner spheres. The fact that the visual representation is a sphere allows for a natural fisheye view effect: objects close to the object of interest are represented with more detail.

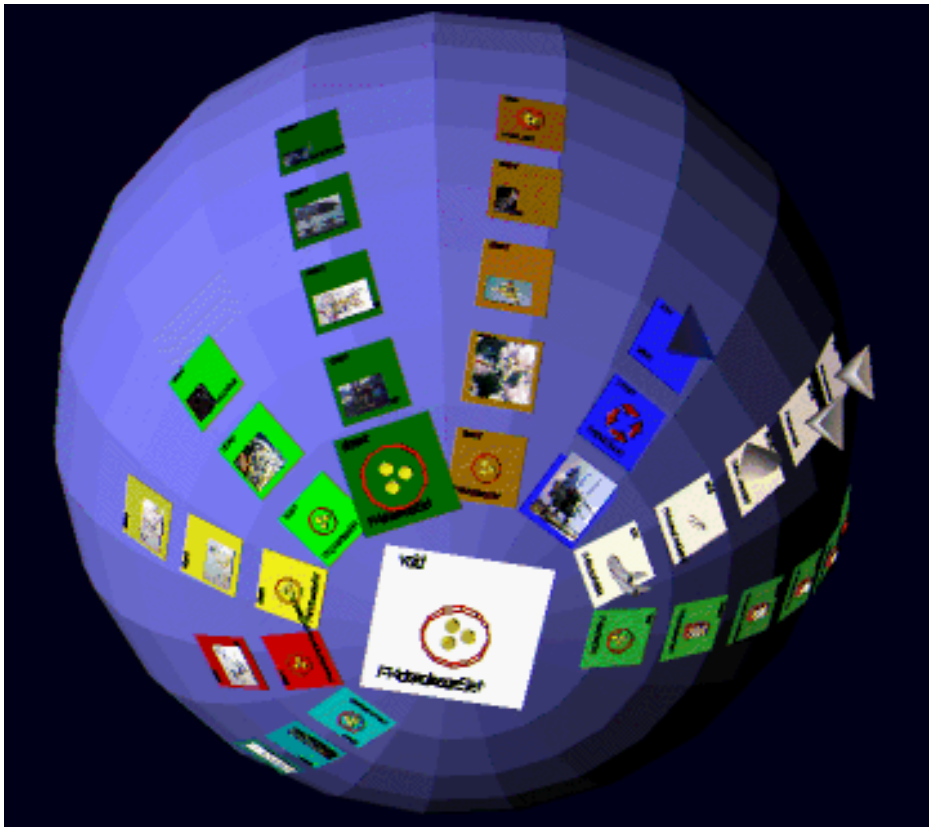


Figure 2.4: Sphere visualization of the links associated to a set of images related to an aircraft. Source: (Fairchild, 1993).

### 2.8.2 Metaphoric visualization

In this section we give some examples of visualizations using metaphors for representing information. All of them have been implement and used by us for different visualization tools, as we will see further in Chapter 5.

#### Conetrees

Conetrees, first introduced by (Robertson et al., 1991) may be described as a three-dimensional representation of a tree of cones. Each tree node is either the apex of a cone or a terminus leaf node for its parent cone. The children of each node are displayed around the basis of the associated cone.

This technique is primarily aimed at the display of hierarchical information, as the conetree itself is a hierarchy of cones. For highly dimensional data its interest is somewhat reduced since the number of dimensions that

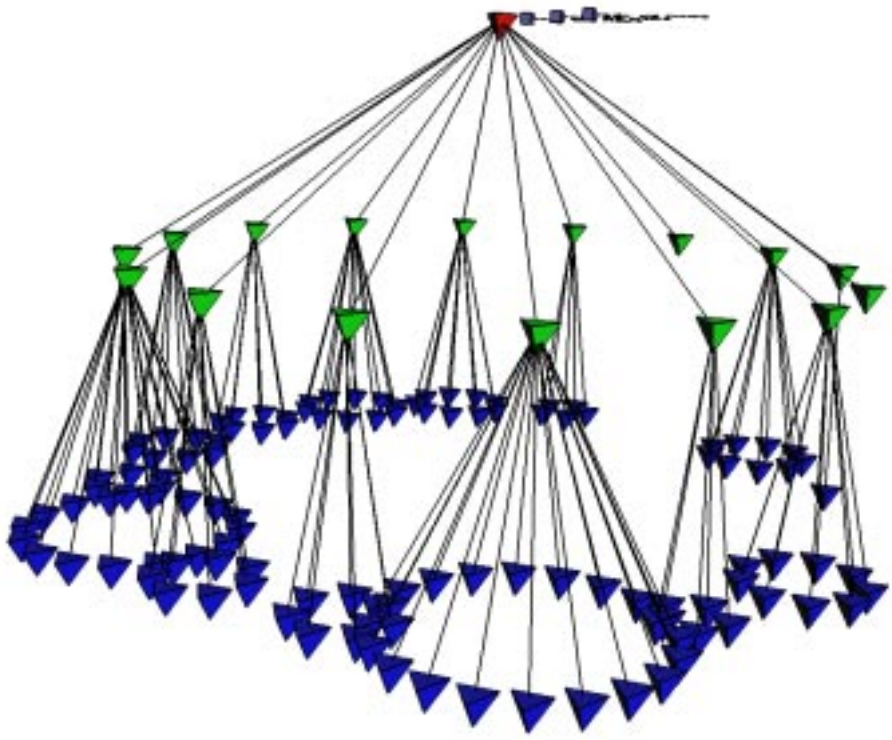


Figure 2.5: Conetree representing a web directory. Source: (Munzner and Burchard, 1995)

can be represented is quite limited. Figure 2.5 shows an example of a conetree from (Munzner and Burchard, 1995). There is a variant to the conetree, called camtree, which is basically a horizontal conetree. Camtrees support better labelling of nodes.

### Information Pyramids

Information pyramids are a quite recent approach to visualize large hierarchies in 3D (Andrews et al., 1997) (Andrews, 1998). The pyramid's base represents the node of the hierarchy and other smaller pyramids are arranged on top of it to represent the subtrees. Separate nodes are represented by separate icons. Figure 2.6 depicts a directory visualization using information


Figure 2.6: Information pyramids visualization showing a directory and its subdirectories. Source: (Andrews et al., 1997)

pyramids (Andrews et al., 1997).

The pyramid metaphor allows for the display of large hierarchies in a compact manner. The user is allowed to freely navigate in the pyramid metaphor. A global overview of the whole hierarchical structure is provided by a bird's eye view of the pyramid metaphor.

### The information landscape

The information landscape is a paradigm for representing large amounts of multidimensional data as if placed in a landscape. The most famous implementation of an information landscape is eventually the File System Navigation (Tesler and Strasnick, 1992), illustrated in Figure 2.7, after its appearance in the movie Jurassic Park catapulted it to a hall of fame. It lays out the directories in a hierarchy with each directory represented by a pedestal. The height of the pedestal is proportional to the size of the files in the directory. The directories are connected by wires, on which it is possible to travel. On top of each directory are boxes representing individual files. The height of the box represents the size of the file, while the color represents

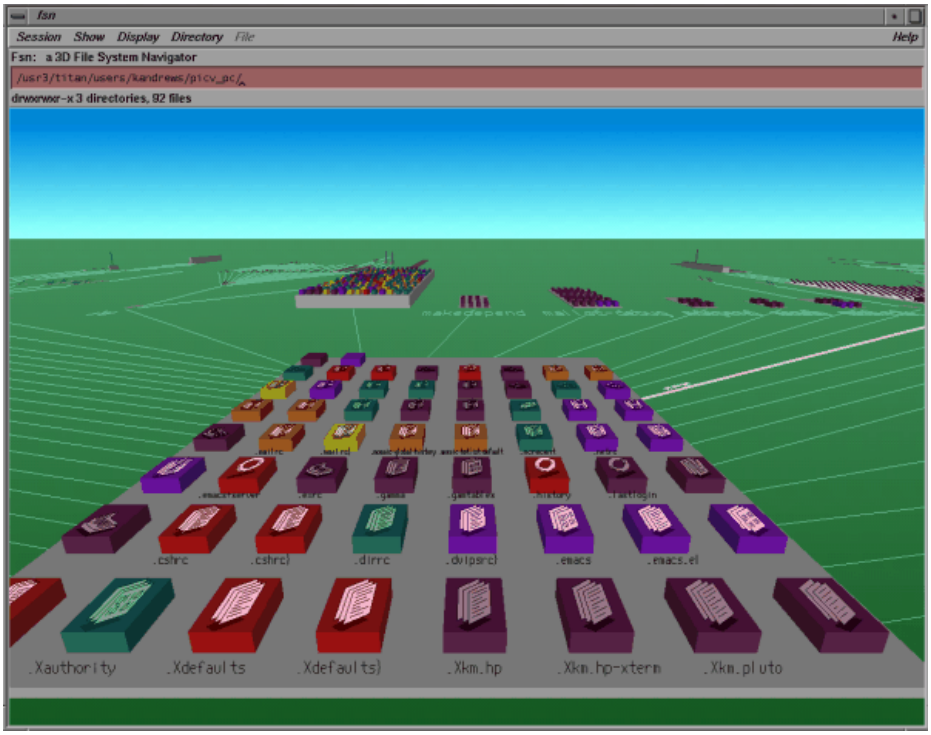



Figure 2.7: File System Navigator (FSN): visualization of a file system. Source: (Tesler and Strasnick, 1992)

the age. The information landscape has also been used for other types of data, such as documents in an information retrieval application (Chalmers, 1993).

The information landscape metaphor combines some of the benefits of 2D and 3D. The 3D allows for the display of more information than would be possible in 2D. At the same time, since the layout is limited to a simple 2D plain, the number of degrees of freedom for navigation can be reduced to make it resemble our day to day navigation.

### 2.8.3 Commercial applications

Even though information visualization is a quite recent research field per se, there are already numerous commercial applications. This section presents three examples of commercial applications of information visualization.

The examples presented here were a subjective choice. We chose to refer a fairly disseminated technique – the start tree – that may be applied to any content, and two very distinct financial applications: the Market map


Figure 2.8: Star tree viewer. Source: (Startree, 2002)

– a very simple stock capitalization visualization in 2D – and the 3DTF – a high-tech visualization for the NYSE's trading floor. Both examples are a demonstration of the power of visualization, even though they are miles away in sophistication.

### Inxight star tree

Star Trees (Startree, 2002) are a technology developed at Inxight (Inxight, 2002). They are interactive maps that help users easily navigate large amounts of information. Star trees are based on a technique called hyperbolic trees (Lamping et al., 1995) developed at Xerox Parc. Figure 2.8 shows a star tree for visualizing and navigating the Inxight website (Inxight, 2002).

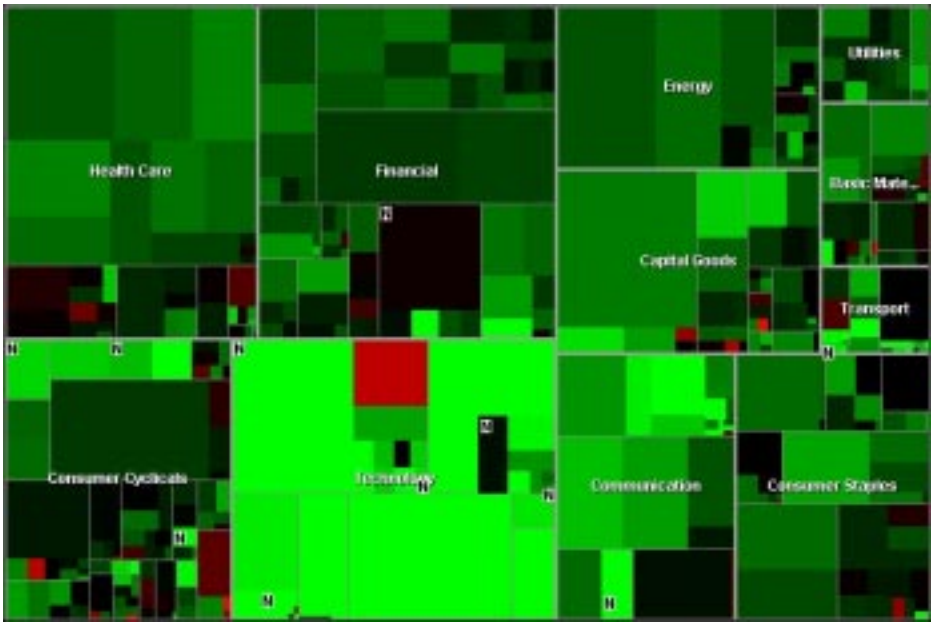



Figure 2.9: Market map. Source: (Marketmap, 2002)

### Market map

The Market map (Marketmap, 2002) is a technology created by Martin Wattenberg (Wattenberg, 1999) for SmartMoney.com for visualizing the capitalizations of companies on the stock market. It is based on the tree maps technology (Treemaps, 2002) developed at the University of Maryland and already cited in Section 2.4.

Figure 2.9 shows an example of the Market map, for the 3rd of March, 2002. The rectangles area represents the capitalization of the respective group of companies. Green (red) shows positive (negative) changes in price.

### 3D Trading Floor

The 3-Dimensional Trading Floor (3DTF), is a multidimensional real-time virtual reality environment developed by Asymptote (Asymptote, 2002) for the New York Stock Exchange (NYSE).

The 3DTF is a real-time model that displays stock exchange activity and events to users in a fully interactive navigable space. Particular attention was paid to qualitative aspects such as lighting, color and texture and the various methods of navigation, movement, viewing and the overall graphic interface. Figure 2.10 shows an example of the NYSE's 3D trading floor.

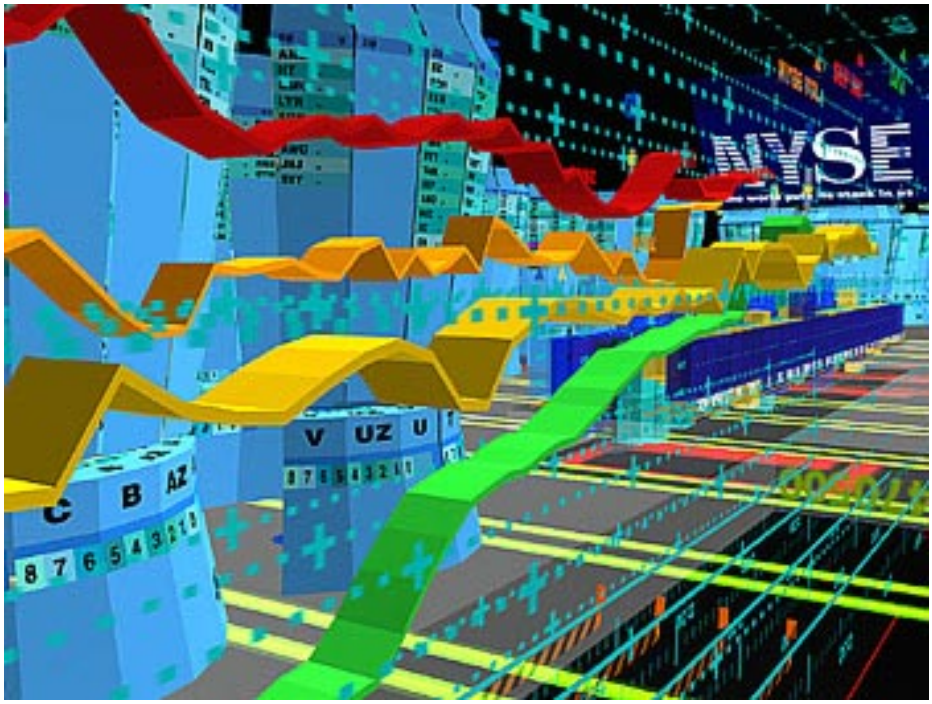



Figure 2.10: 3-Dimensional Trading Floor (3DTF). Source: (3DTF, 2002)

## 2.9 Conclusion

In this chapter we introduce the field of information visualization and we show how it differs from scientific visualization, even though both fields may overlap. We also provide an extensive bibliography for further reading.

We argue for and justify our options for using visual metaphors, three dimensions, and non-immersive environments in our visualizations. Additionally, we provide information visualization examples to illustrate points made before and to give a very succinct idea of some of the commercial applications already available in the domain of information visualization.

As a concluding remark, Graham Walker (Walker, 1995) identifies four major challenges in information visualization: a growing volume of data with declining information content; more complex data analysis tools and models; increasingly abstract data and issues; and a wider, less specialist audience. It is our belief that 3D metaphoric information visualization may help in dealing with all of these challenges.

The use of 3D allows for attacking the first two challenges, by permitting the display of larger volumes of data and the development of more sophisti-

cated analysis tools. The use of visual metaphors to represent the information allows for dealing with more abstract data and a wider less specialist audience, by attaching a meaning to the spatial mapping of information that the user can relate to.

# Chapter 3

# The CyberNet Project

> What is needed is a visualization engine that takes any kind of abstract information and creates a visualization of it. (Fairchild, 1998)

## 3.1 Introduction

Larger and larger volumes of information are being generated everyday. Most of this data is available in a digital format and does not have a natural physical representation. Furthermore, large volumes of data are usually not concentrated in one place – the growing tendency is to have data distributed over several places and available through a network. Additionally, this distributed data may also be dynamic.

Information visualization uses visual representations to help gain insight into abstract data. The CyberNet project is a research project taking place in the Multimedia Communications Department at the Eurécom Institute that builds up on the field of information visualization. The broad goal of the CyberNet project is to study how three-dimensional metaphoric information visualization may help in gaining insight into large volumes of distributed dynamic information. The CyberNet project worked as a test-bed for this thesis work.

This chapter only addresses a brief description of the CyberNet framework. Specific issues related to 3D information visualization and that have been attacked in this work, such as the visual mapping of data, multiple visualizations, user navigation, and real-time visualization of dynamic data, are further developed in the remaining chapters and constitute the main part of this thesis work. For further information on the system's architecture please refer to (Abel et al., 2000), or (Abel, 2001) for a complete description.

### 3.1.1 Motivation and problem statement

We believe that 3D metaphoric information visualization may bring some added value to the visualization of large amounts of multidimensional data. The third dimension allows for larger quantities of data to be displayed and fosters the real-world resemblance of the visual metaphors. Besides, visual metaphors promote the introduction of new concepts in a easy way, give a meaning to the representation of abstract data, and generally provide multiple opportunities for information mapping, which is especially interesting for the case of data with high-dimensionality.

A main challenge in information visualization is how to represent the abstract information. We feel that there is no universal solution for the visual representation of abstract data. Different visualization tools, different data sets, or different tasks, may require a different visual representation. This was the main motivation to design a general framework that facilitates the development of specific visualization tools. 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 was designed to be application-domain independent. The initial application domain for which it was tested was network management and monitoring. This way we took advantage from in-house know-how in the networking domain and it also made it possible for the actual data collection to be done within our own network facilities. Furthermore, network data was compliant with the requirements regarding quantity, distribution, dynamics, and multidimensionality.

We have subsequently used CyberNet's framework to visualize Web server data, for file-systems analysis and others. Virtually any kind of application domain that requires the visualization of large volumes of abstract dynamic data may be targeted. Further examples, completely unrelated to the computing or networking field, are financial data (e.g., stock market bonds), or flight-reservation data visualization.

Independently of the broad goal of the CyberNet research project and that the development framework was designed to be application domain independent, the bulk of the work done so far in the context of CyberNet is related to the networking field. Moreover, some concepts that arose from the network management and monitoring application, such as the concept of a *service*, were extended as way for structuring data to other applications and are part of the generic framework.

In this chapter, we present CyberNet's framework with a high level of abstraction, so as to emphasize its data domain independence. Nonetheless,

whenever examples are to be given for the sake of presentation clearness, they are usually related to the network-monitoring application as it was the very first application domain and has stayed as the privileged one so far. With this in mind, and in order to provide some background information to the rest of the chapter, the following will briefly discuss network monitoring. Also, this chapter's related work, in Section 3.2, targets mostly networking related applications.

Network management and monitoring are difficult and attention demanding tasks 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 then makes a 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 do not think in terms of *devices* anymore, they think in terms of *services*. In our terminology, the term service covers, for example: topology, connectivity, mail, routing, NFS (Network File System), printing, and so on. The status of a service may 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 the services, in order to comply with the service-based network management paradigm. The service concept was extended to provide a structure for any kind of data to be visualized, as we will see further in this chapter.

### 3.1.2 Chapter organization

The remainder of this chapter is organized as follows: Section 3.2 presents some related work in the particularly in the field of network management and monitoring. In Section 3.3 we describe our approach. The next section, Section 3.4 presents CyberNet's distributed system architecture. In Section 3.5 we describe the data collection strategy and in Section 3.6 and Section 3.7 we present the data model and the visualization model, respectively. Finally, we give some implementation details in Section 3.8 and we draw some conclusion in the last section, Section 3.9.

## 3.2 Prior work

The interest in 3D interfaces for network management started less than ten years ago. The first experiments in this 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 (Crutcher and Waters, 1992) (Crutcher et al., 1993) (Crutcher et al., 1995). (Kahani and Beadle, 1996) and (Cubeta et al., 1998) presents a network management tool that supports collaborations between users and is accessible from a web browser. In the work presented in (Cox et al., 1996) the traditional two-dimensional maps used to display geographic networks are enhanced with 3D technologies. This allows to address the scalability limitation of 2D displays, while retaining their advantages.

In parallel, researchers started to focus on visualizing the network data rather than to limit the visualization to the network structure and connectivity. (Becker et al., 1995) presents some work done on this topic and introduces studies about services: the mail service is studied in (Eick and Wills, 1993) and highlights the interest for the visualization to exploit the hierarchical structure of the data (e.g., cluster, sum up).

More recently, (Wei et al., 2000) (Koutsofios et al., 1999b) (Koutsofios et al., 1999a) reports AT&T laboratories experiments concerning the management of large dynamic telecommunication data sets ($\approx$ 250 million events per day). The results of the studies seem promising since the presented tool improves network management and marketing of network services. Also in (Becker et al., 1991) the use of computer graphics was studied for the visualization of the AT&T long-distance network in the United States of America.

(Shaffer et al., 1999) 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 Co-Pilot (Performance Co-Pilot, 2002). The tool can be used for monitoring the hardware and the operating system as well as be used in the context of layered services (e.g., domain name servers, Web servers, and e-mail servers) monitoring. 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. (Atlas, 2002) 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 (Keller and Keller, 1993) (Chen, 1999) (Card et al., 1999) (Ware, 2000) (Spence, 2001). 3D information visualization has lots of aspects and applications. In a special report (Gershon and Brown, 1996) 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. (Robertson et al., 1993a) reports an experiment done by Xerox in order to go beyond the usual desktop metaphor.

Another active research area is *bizviz* – the visualization of business data (Wright, 1997). The main idea is to provide 3D graphical representations of rows and columns of numbers in order to improve perception and support decision-making. (Becker, 1997) presents a data mining visualization software tool developed by SGI (SGI, 2002) called MineSet. The force of this tool is the integration of several 3D visualizations that can be used simultaneously by the user. Finally, extensive work has been done towards the visualization of the Web (Munzner and Burchard, 1995) (Mukherjea and Foley, 1995) (Andrews et al., 1996) and also towards Web-based information visualization (Rohrer and Swing, 1997) (Wood et al., 1996).

## 3.3 Our approach

We consider that the CyberNet framework is specific in its approach because of the following features – all of them are further developed in the remaining chapters:

1. The visualization is based on 3D metaphors. The traditional way to represent, for instance network information in 3D is to extend the usual two-dimensional (2D) graphical business representations such as 3D bar charts, 3D graphs, and so on. These representations are well suited for displaying small amounts of information but they generally fall short when large amounts of information are involved and when it is important to show relationships between different data subsets. In order to represent information CyberNet uses 3D metaphors based on real-world structured systems such as towns, solar systems, and buildings, for instance.

2. The translation between the real-world data and their visual counterparts is handled automatically by the system. We call this translation the mapping process. This feature allows for the automatic construc-

tion of the virtual worlds and for updating the visualizations on the fly. Chapter 4 will be entirely dedicated to the mapping process.

3. From our point of view there is no single visual metaphor that is "the solution". Different *data sets*, different *data structures* and different *tasks*, may require different *metaphors*. Globally speaking, we think that different visualizations tools usually demand different visual metaphors. Furthermore, we believe that one of the strengths of 3D metaphoric representation will come from the combined used of several interacting tools. We examine this subject further in Chapter 5.

4. User navigation is also a particular concern in our system. We have created the concept of *metaphor-aware navigation* which states that the way a user navigates in a virtual metaphoric world is dependent on the metaphor used. Chapter 6 assembles all issues related to user navigation in our visualization system.

5. Last but not least, the processed data is dynamic. The visualizations are designed to cope with this dynamic nature so that the 3D worlds are permanently updated. New objects are added to the 3D scene, existing objects disappear, and the visual appearance of the displayed objects is continuously modified in order to reflect the state of the system's data. The management of data dynamics will be dealt with more thoroughly in Chapter 7.

## 3.4 System architecture

As we have seen from Section 1.1 the first application of the CyberNet project was to automatically build three-dimensional virtual worlds to visualize network data and analyze network services (Abel et al., 2000) (Abel, 2001). The CyberNet system uses a distributed object framework that handles all the tasks from collecting the data to the 3D visualization. The use of distributed object technology allows us to address two information management problems that we were faced with: the data is distributed and is dynamic. It also makes it possible to separate the 3D user interface (which is very computing intensive) from the rest of the system.

CyberNet's framework is composed of three distinctive parts:

- the *collecting layer* is used to gather the raw data – it is the only layer that is partly application domain dependent

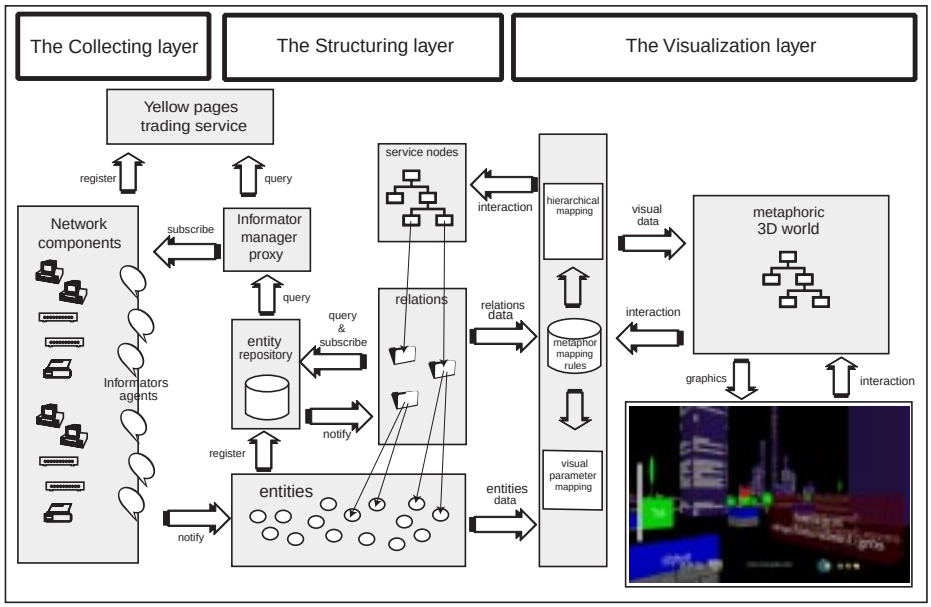



Figure 3.1: CyberNet's system distributed architecture.

- the *structuring layer* is the kernel of the system – it structures the raw information according to the service model
- the *visualization layer* translates the service model into a 3D metaphoric world – it also manages the visualization and provides the user interaction and navigation

Figure 3.1 shows a representation of the CyberNet's system architecture and its distributed nature over the three layers. As it is visible from the figure, only the collecting layer is partly dependent on the application domain (Figure 3.1 depicts the case for network management and monitoring, with the corresponding network components).

### 3.4.1 Collecting layer

In CyberNet the collecting process is implemented using distributed agents – designated *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 the structuring layer (more exactly a service node) is interested in a given data, it should contact the corresponding informator and subscribe to the 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. 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 system 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.

### 3.4.2 Structuring layer

The structuring layer, though it does not constitute the main focus of this thesis's work, plays a very important role in the context of the CyberNet system. In order to be efficient, the representation of information requires the data to be logically structured into a service model. Each data set (i.e., service) has its own structure. This structure is dependent on the application domain and requires expertise knowledge. Basically, structuring the data requires to group data according to common properties and to identify relationships.

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 visualization layer. In other words, the structuring layer is designed to produce at its output a structured tree. This structured tree is called the *service model*. This service model is mapped onto the virtual world scene hierarchy (see Chapter 4 for the description of the mapping process); hence, the structuring layer plays a determinant role in the world creation process.

### 3.4.3 Visualization layer

The visualization layer is the last layer in our distributed framework. It is responsible for displaying the virtual worlds that are used to visualize the information. Besides the actual display of the information, it also provides the user navigation and interaction in and with the virtual world. It should be noted that the translation of the delivery of the preceding structuring layer (the service tree) into the 3D-scene hierarchy that describes the virtual world – the *mapping process* – is also under the responsibility of the visualization layer. The CyberNet system uses *visual metaphors* to define the appearance of the virtual world.

## 3.5 Data collection

The collecting layer, more exactly the collecting agents – informators, are the only part of the framework that is data-dependent. According to the type of service and data, the agents may use any method, such as for instance SNMP – Simple Network Management Protocol (SNMP, 2002), HTTP – HyperText Transfer Protocol (HTTP, 2000), or scripts for retrieving the information.

As an example, and as we will see later in Chapter 5, SNMP requests were used in order to obtain network topology information and network bandwidth and error ratio – network topology visualization tool (Section 5.5.3); systems calls were used to retrieve NFS and other operating system data – NFS service visualization tool (Section 5.5.2); and database accesses were used to retrieve building information – Eurécom virtual building tool (Section 5.5.5), just to refer some examples. In the following we will describe the collecting data process for the network management application.

For a given network-management service, the network data that has to be monitored is spread all over the network. In traditional systems, SNMP (Simple Network Management Protocol) 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 are continuously updated without user intervention. 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 collecting strategy – we use a combination of data pushing with collecting policies. The informators push information to the structuring layer according to data subscription requests issued from service components called service nodes and described further in Section 3.6.2. The subscription request contains the data identification and a collecting policy specification for that data. For example, the structuring layer may be interested in receiving all the data of type "user" on a computer named "abyss".

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* (e.g., every five seconds), *freshness criteria* (e.g., every five seconds but only if modified), or *percentage of alteration* since last notify (e.g., every fives seconds and only if new value differs by more than 10 percent from the previous).

When a user wants to monitor a service (i.e., launches the corresponding visualization tool for that 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 the policies criteria to the structuring layer.

## 3.6 Data model

The visualization of abstract data involves translating the data into visual elements. This translation requires several steps. The first step towards this process is to logically structure the data in order to obtain a data model – this step has been referred to as *data transformations* (Card et al., 1999). CyberNet's information structure is based on the concept of services.

### 3.6.1 Services

The concept of a *service* arose from the need of structuring the raw data in order to be visualized in a metaphoric representation. A service is hence just a way of structuring information in a manner that is more easily exploitable for its future visual representation.

A service is described by a *service tree*, composed of *entities* and *service nodes*; *relations* are implicit in the service model tree, since they gather entities. The service also possesses a certain number of attributes (e.g., recursive) that are used for mapping the service information. The service attributes and the role they play in the mapping process will be described in Chapter 4.

### 3.6.2 Service components

The service structure is represented by a tree which is composed of entities, relations and service nodes:

- *entities* are the smallest building blocks that are used to construct the service tree – they are the leaf nodes
- *relations* act as the "glue" used to build the service model tree – they provide for the connections between the nodes.

- *service nodes* use the service elements (entities and relations) to build the service tree – they are the internal nodes of the service tree

### Entities

*Entities* are the atomic building blocks used to model the service which information is being displayed. They gather related information concerning a logical element of the service. For instance, in the context of network management, entities may represent physical devices (such as hubs, switches, routers, workstations, users, and so on) or they may be conceptual (such as connections, processes, groups, and so on).

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 and all the entities must be registered there. The entity repository is extensively used by the service nodes and is responsible for managing the life-cycle of entities. The role of the repository is thus to keep track of all the existing entities and to be able to answer to queries concerning entities.

### Relations

*Relations* are used to group entities according to some common properties. Conceptually, a relation may be seen 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 the user *bernhard* on the computer *abyss*. That would result in a query to the entity repository that asks for all entities of type process with attribute *user=bernhard* and attribute *computer=abyss*. The relation will then group references to all the entities that match the query. As a consequence, a relation is a reference to one or more entities and an entity can be involved in one or more relations. Relations are created by the service nodes.

### Service nodes

*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.

Thus, a service node is an object that knows how to interpret a small part of the data model. The way a service node works is the following: the service node asks, using SQL (Structured Query Language)-like queries, for the entities that have some specific properties and analyzes the retrieved entities in order to create the children service nodes. For example, upon receiving a workstation entity, a service node may create a child service node in charge of building a new relation containing all the users of that particular workstation.

### 3.6.3 Service creation

The way the service tree is constructed is as follows: the service node issues a persistent 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 repository returns what we call a relation, which groups references to all the entities that satisfy the query. The service node adds that relation as one of its children, hence constructing a tree-like structure. Figure 3.2 depicts a simplified version of this mechanism.

It should be noted that the entity repository permanently forwards to the interested relations references of entities that match the requests it has received. This allows for the dynamic data management, as we will see in Chapter 7.

### 3.6.4 Service description file

In the CyberNet system services are described by a service tree. This tree is created according to a service description file. The *service file* defines the structure of the service. A service model is always represented by a tree structure – thus a service file must always have a root service node. Therefore, a service file always starts with the directive `services{}`.

The `services{}` directive has two parameters, `class` and `logicalclass`, and another directive: `children{}`. The `children{}` directive allows for the creation of a parent-child relation. There are three types of children:

- the `launchservicechild{}`
- the `defaultservicechild{}`
- and the `entitychild{}`

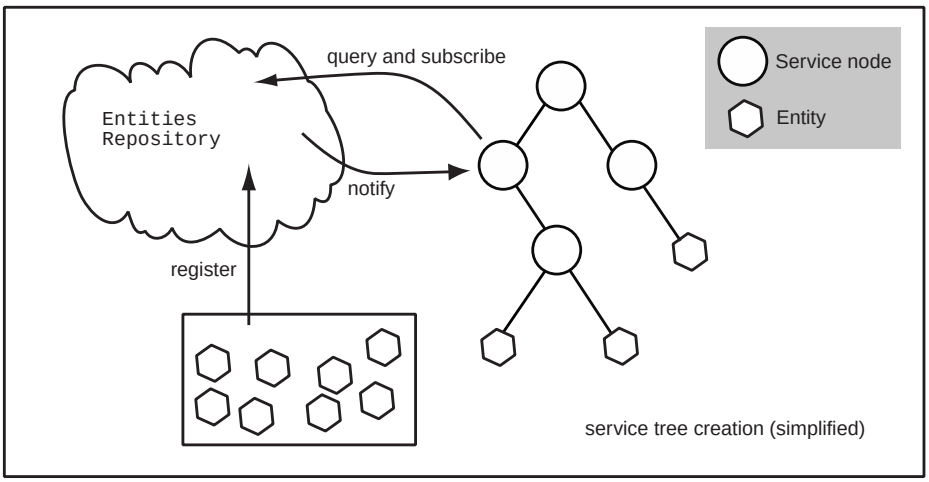



Figure 3.2: Building the service model tree.

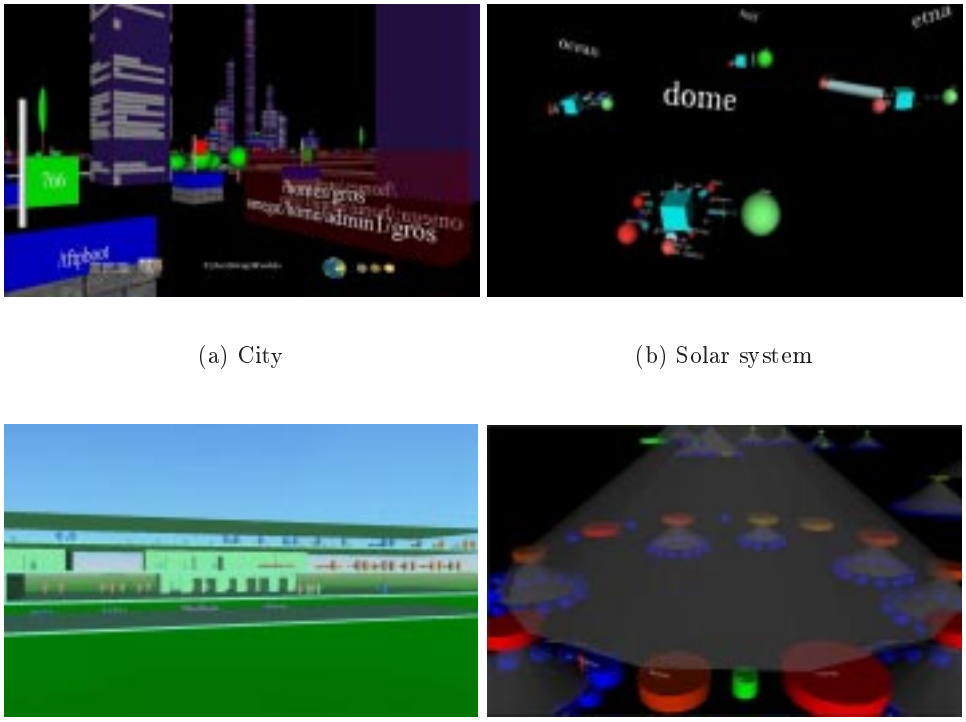
The first two types generate new service nodes, whereas the latter references an entity. The three directive types generate, respectively, *set of* relations, *group* relations, and *dependency* relations. Further information on the relations' types is given in Chapter 4, namely regarding how they interfere in the mapping process (Section 4.4.2).

## 3.7 Visualization model

CyberNet's visual representation is supported on the use of visual metaphors. Our conceptual system is fundamentally metaphorical in nature (Lakoff and Johnson, 1980). The use of visual metaphors makes appeal to underlying concepts the user is already familiar with and therefore the user is able to relate to the relationships existing between different objects of the metaphoric world (Newman and Lamming, 1995). This facilitates the comprehension of the information's visual representation. For example, in a city metaphor it is straightforward that houses in the same district share some common properties. The use of visual metaphors is argued with more detail in Section 4.6.

### 3.7.1 Metaphors

Some of the previous work done in the field of information visualization using 3D metaphors includes a cityscape (Gershon and Eick, 1995), a solar system (Crossley et al., 1997) or a garden (Crossley et al., 1997), or a cone-tree (Robertson et al., 1991).




(a) City (b) Solar system

(c) Building (d) Conetree

Figure 3.3: Examples of metaphors – CyberNet project.

CyberNet virtual worlds use mostly real-world based metaphors that allow the user to better and more rapidly comprehend the visualizations. We have designed real-world based metaphors such as a city (Figure 3.3 a), a solar system (Figure 3.3 b) or a building (Figure 3.3 c). Nevertheless, it is sometimes more useful to use abstract metaphors. For instance, for depicting hierarchical information, a conetree (Figure 3.3 d) may be a better option. The choice of the metaphor depends essentially on the information to be displayed and the user's task.

It is important to note that a metaphor is not designed for a specific service model. In fact, a service model can be visualized using any metaphor provided the metaphor offers sufficient visual and structural capabilities for displaying the service information. How we choose a metaphor for a given service is discussed in Chapter 4 and in Chapter 5 we discuss all the metaphors we have implemented, as well as for which service(s)/visualization tool(s) they have been used.

### 3.7.2 Metaphoric worlds

A metaphor may be viewed as a world construction pattern. The metaphor is a hierarchical structure that will drive the world construction and fits the structure of the scene description graphs found in most 3D libraries.

A 3D metaphoric world is a set of *metaphoric components* (MCs) organized hierarchically. Metaphoric components are *graphical components* (GCs) with add-on capabilities. CyberNet supports two classes of graphical components: *3D glyphs* and *layout managers*. In the next section we will clarify the distinction we make between metaphoric components and graphical components in the CyberNet system.

### 3.7.3 Metaphoric components

*Metaphoric components* are the graphical elements of a metaphor, i.e., 3D glyphs and layout managers, with embedded interaction and navigation capabilities. Thus, according to our terminology, metaphoric components and graphical elements of a same metaphor do not exactly mean the same thing, from a theoretical point of view.

As an example, a rectangular box may be a graphical element (3D glyph) for constructing a given metaphoric world (e.g. a virtual building). However that rectangular box when instantiated in the same metaphor to play the role of a floor with navigation features (e.g. moving the user along the floor's corridor) is then perceived by our system as a metaphoric component.

This latent ambiguity of the metaphoric components goes to the argument that contrary to Graphical User Interface (GUI) conventions, 3D worlds contain objects that function both as navigational and destinational elements (Benedikt, 1991). This kind of multipurpose property of virtual-world elements has also been advocated in other research (Alexander, 1982) (Furnas, 1997). It should be noted that metaphoric components also provide further functionality, not related to the navigation – e.g., built-in interaction mechanisms.

Specific graphical components must be defined for each metaphor. For example, a solar-system metaphor might be composed of star and planet glyphs, and an orbital layout manager. As more numerous graphical components are used in a metaphor, also more numerous are the ways of visualizing data. Even so, some metaphors cannot be used to visualize a service because they lack sufficient graphical components to show all the service information. This issue is discussed in Chapter 4.

The next sections describe the graphical components.

### Layout managers

The presentation of large volumes of information needs to be organized in space in order to make its interpretation easier (Fairchild et al., 1988). Layout managers (LMs) are a step towards organizing information spatially: the primary function of the layout managers is to arrange their children in space. Layout managers can also add some visual elements like semi-transparent bounding boxes to enhance visual representation effectiveness, as suggested in (Zhai et al., 1996). These additional visual elements can also be used to map information, just as 3D glyphs do.

Layout Managers are thus responsible for one of the most important visual choices: the use of space. They are used as containers and may contain 3D glyphs or other other layout managers. Layout managers organize their children in space according to a built-in policy. For instance, a `orbit` LM organizes elements in orbit around a center, and a `chess` LM on a plane in rows/columns.

We have developed several layout managers that are used in a number of metaphors. For instance, the `chess` LM is used in the metaphor city (Figure 3.3 a) amongst others, and the `orbit` LM is used in the solar system metaphor (Figure 3.3 b) or the conetree metaphor (Figure 3.3 d). In Chapter 4, layout managers are further characterized according to several criteria (e.g. placement policy).

### 3D glyphs

3D glyphs (3DGs) are the atomic building blocks of the virtual world. They are 3D graphical objects whose appearance may be controlled in real time through visual parameters. They are used to construct the world and their visual parameters are used to visually map the information.

3D glyphs are thus 3D graphical objects that represent data through visual parameters; these may be either retinal (e.g. color and size), or temporal (Chuah and Eick, 1998). The level of complexity of a 3D glyph can be related to the number of visual parameters it offers for modification and hence to the dimension of data that can be displayed. Some examples of complex 3D glyphs can be found in (Chuah and Eick, 1998) and (Rohrer and Swing, 1997).

For CyberNet we have developed several different types of 3D glyphs. In the city metaphor (Figure 3.3 a), for instance, there are trees, buildings and houses (made from boxes), among others. Chapter 4 further describes the 3D glyphs that have been used to build CyberNet metaphors, and also characterizes them according to the visual parameters they offer.

### 3.7.4 Mapping the data model

Mapping is the process that associates a visual representation to the service information. In the CyberNet system, special objects called *adaptors* handle the mapping. 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. The adaptors implement the mapping association rules.

We distinguish between two types of mapping: *structural* and *visual-parameters* mapping. We call *structural mapping* the process of defining which service element (entity or service node) will be visualized using which MC. For instance, for the workstation supervision service using a solar system metaphor like in Figure 3.3 (b), the model mapping rules states that *computers*≡*stars*, *users*≡*planets*, and *processes*≡*satellites*.

Besides mapping the structural elements onto the graphical components, the attributes of each entity must also 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 (e.g., position, orientation, size, color, etc).

How we map the data values on these visual parameters is the responsibility of the visual-parameters mapping. A simple example is the mapping of a disk entity on a box glyph. In this example, the size of the disk could be mapped on the size of the box while the percentage of use of the disk could be mapped on the color of the box (green corresponds to 0%, yellow to 50% and red to 100%; the values in-between are mapped to the color transition between the limits). When defining the mapping rules, care must be taken to preserve metaphor coherency. For example, if one rule uses color for identification purposes, other rules cannot use this visual parameter to represent a data value.

Chapter 4 explains the mapping process in detail.

### 3.7.5 Building and updating the worlds

There are two different operations, *creation* and *update*, which can be used to update the metaphoric world. Creation, as the name implies, involves creating one or more new graphical components – glyph or layout manager – and update involves bringing up to date the visual parameters of one or more GC (e.g., updating the size of a box to reflect the new value of the mapped data). LMs can receive orders to add new children (3DGs or LMs) to themselves and any GC can update itself upon receiving up-to-date data.

Chapter 7 will describe in more detail the update of the virtual worlds and how the world dynamics are managed.

## 3.8 Implementation details

The CyberNet platform is chiefly written in Java (Java, 2002), and the CORBA (CORBA, 2000) distributed object technology is used to support communications between the different parts of the system.

The collecting layer is composed of a large number of distributed objects. AdventNet (AdventNet, 2002) Java/SNMP, operating system calls and perl scripts are used for collecting information. In fact, the informators may use any method (e.g., 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 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 these purposes the CyberNet platform uses IBM Tspaces (Wyckoff et al., 1998).

The CyberNet user interface is accessible from any Web browser. In the latest implementation, the visualization layer drives the Java3D virtual world through the use of Java and Javascript (JavaScript, 1997). In a former implementation, where VRML was used for the visualization, the VRML – Virtual Reality Modeling Language (VRML, 1997) plug-in was controlled through the EAI – External Authoring Interface of VRML (EAI, 1999).

## 3.9 Conclusion

In this chapter we present a complete and flexible framework for a visualization tool that uses metaphoric virtual worlds to display large volumes of multidimensional information. We have succinctly presented the whole process, from collecting the raw information, structuring it, mapping the structured information onto the visualization and displaying the information. We have introduced important concepts used by the CyberNet system such as services and visual metaphors for structuring and visualizing the information, respectively.

Importantly, we present a distributed framework that allows us to separate the three main parts of the visualization process: collecting the information, structuring the data, and presenting it visually.

Regarding the collection of the information, we also describe our data-collecting policy and the data-collecting agents – the informators. The fact that we use a push policy in combination with a subscribe-publish design pattern for the raw-data collection allows us to save bandwidth while still being able to continuously update the visualizations.

For structuring the raw information we use the concept of service model, which is a hierarchical information structure that organizes the raw data to be exploited by the visualization layer. The nucleus of the structuring layer is an entity repository that allows us to keep track of all the data entities and manage their life-cycles. The repository is also able to answer queries from the service nodes asking for a certain type of entities. This queries are persistent, thus allowing the system to always keep updated information.

For the visualization we use two kinds of metaphoric components – 3D glyphs and layout managers – to build the visualizations. The 3D metaphoric virtual worlds are built directly from the exploitation of the service model tree, by mapping agents that we have named adaptors, and are accessible via a web browser.

Although the work presented in this chapter focuses primarily on the visualization of network management and monitoring data, the framework is essentially application domain independent and so other domains may be targeted. The only part of the system that is naturally dependent on the data are the collecting agents, but they are straightforward to implement for other data domains. As an example, we have already tested the system for Web server data visualization. Virtually any domain that needs to visualize large volumes of multidimensional dynamic data may utilize this approach – for instance, stock exchange or bank trading.

# Chapter 4

# Mapping

> At the core of all good science and engineering is the respectful treatment of data (...). Visualization should help make sense of the flood of output data. When applied without some insight into visual perception, however, it can introduce errors in understanding as surely as if a wrong analysis algorithm were used. In short, a picture can tell a thousand lies. (Rogowitz and Treinish, 1998)

## 4.1 Introduction

The core of information visualization is finding a way of visually representing information in a manner that is most effective and pleasing for the user comprehension. This involves mapping data values onto visual parameters. One of the fundamental issues in information visualization is how to visually represent data that has no natural visual representation. In other words, how do we map abstract data onto the graphical elements that are used to convey the information?

Metaphors provide a way of conveying information visually in a representation that the user is familiar with or can more easily understand (Newman and Lamming, 1995). Metaphors can be based in the real word or based on fairly abstract concepts. To illustrate, Figure 4.1 shows an example of the former, depicting the information as a virtual city; Figure 4.2 is an example of the latter, using the conetree metaphor to visualize information.

Our goal is to automatically provide the best mapping, given a certain data set and a number of different visual metaphors. In this chapter we present our strategy for automatically mapping time-varying abstract data on the metaphoric virtual worlds that are used to display the information.

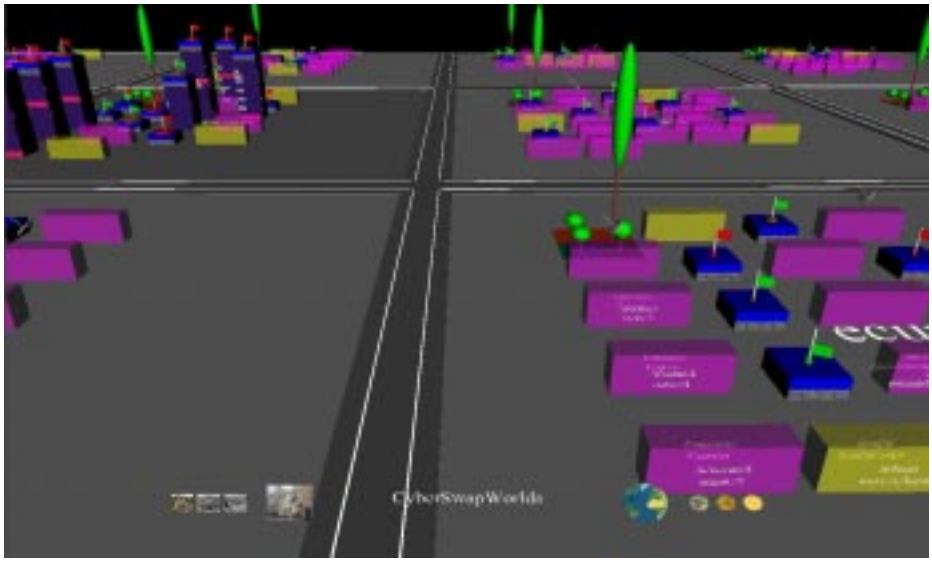


Figure 4.1: City metaphor for NFS data visualization – CyberNet project.

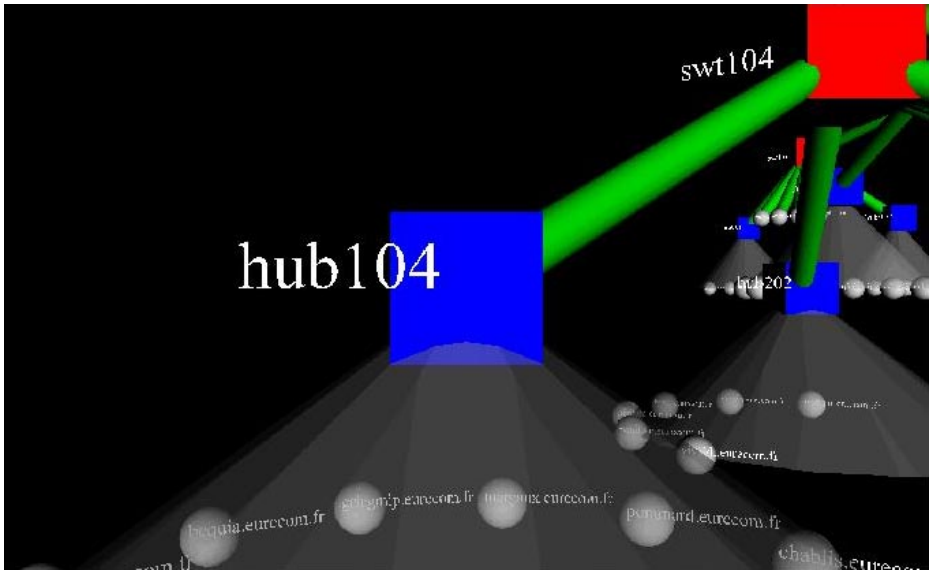



Figure 4.2: Conetree metaphor for topological visualization – CyberNet project.

We have devised a mapping engine that is based on a metadata characterization of the information, on the characteristics of the virtual worlds used to display the information, and also on the user task. We present the criteria

that are used to find the best mapping and the strategy taken to implement those criteria. We describe our utilization of visual metaphors and the graphical components that we use, and we give some examples of different mappings.

The work presented here is done in the context of researching how 3D information visualization may help when trying to visualize large volumes of dynamic data. This work, as done in the context of the CyberNet project (Chapter 3), has been mostly applied to the construction of visualization tools that convey network data for network monitoring and management. Nonetheless, the framework, as described in Chapter 3, is application-domain independent and so is the mapping strategy.

### 4.1.1 Motivation and problem statement

Scientific visualization deals with visualizing data that usually has an intrinsic real-world representation. On the other hand, in the case of information visualization, the data to be represented has, in general, no natural physical representation. This leads us to one of the most important matters in the field of information visualization: the visual mapping of abstract data.

How to visualize abstract data is a most relevant problem. For instance, regarding network monitoring, how do we visualize a open file-handle? Moreover, how do we decide on a certain visual representation, on a certain metaphor to represent the service information? Is there a way of making sure that a particular visual metaphor is better than another for a given data set? Furthermore, within a metaphor, are we also able to determine the best visual mapping for the data entities? If we can determine the criteria that make one mapping better than the other, is there a possibility of automating the mapping process? These are all open questions that relate to and justify our work regarding automatic visual mapping of data.

The visual mapping of abstract data is undoubtedly a very important part of the visualization building process. Since there is no established way of visually represent the information, an ineffective mapping can bias the user interpretation of the actual data (Rogowitz and Treinish, 1996), reduce the rate of absorption and understanding or practically prevent the user from getting the information (Gershon, 1998). These factors are the motivation behind trying to find the best mapping – i.e., a visual mapping that truly helps the user understand the data and does not distort its meaning.

In this chapter we present a strategy for mapping large volumes of dynamic data onto metaphoric virtual worlds. The starting application domain, as part of the CyberNet project was network monitoring and management

in order to benefit from available in-house know-how. Network monitoring data does fulfill the required attributes of quantity and time-variance. Besides, we could take advantage from the fact that the data collection could be done from our own local network, at Eurécom institute. Later we have also developed visualization tools for web-server analysis, and for file systems and directories visualization, for instance. All these visualization tools are presented in Chapter 5.

Our motivation to devise an automatic mapping strategy arose from the fact that it soon became clear that hard-coding the actual mappings was too cumbersome. Additionally, with a mapping strategy that basically works over metadata, we could implement a mapping policy independent of the actual data. Since CyberNet's framework is application-domain independent (with, naturally, the exception of the raw-data collecting agents), a way of mapping abstract information onto visual parameters in a manner that is application-field independent was also sought.

Moreover, the fact that the visual mapping of the data can be done automatically, via one mapping engine, without the need to hard-code the individual data mappings one by one, and without requiring user intervention, allows us to change the visual mappings on the fly and to experiment and test different kinds of mappings and different visualization metaphors, in an easy and transparent way. This is a powerful means to construct and experiment with different virtual worlds on the fly.

In view of the foregoing, our goal regarding the mapping process can be defined as follows: given a certain data set (service) and a certain visual set (metaphor), the goal is to be able to automatically map the service on the metaphor, taking into account the task the user wants to accomplish. Furthermore, we want to be able to determine the best possible mapping for each triplet service–metaphor–task, based on predetermined criteria.

This chapter presents thus the requisites and the mapping process we have devised to automatically map abstract information onto visual parameters.

### 4.1.2 Chapter organization

This chapter is organized as follows: in the next section, Section 4.2, related work in the field of data and task characterization for visual mapping is presented. In Section 4.3 we describe our approach regarding the visual mapping of abstract information. Section 4.4 and Section 4.5 are dedicated to the services and to the tasks characterization, respectively. In Section 4.6 the use of visual metaphors for information visualization is debated and in

Section 4.7 we present the characterization of the metaphors regarding the mapping. Section 4.8 is dedicated to the presentation of the mapping engine, and in Section 4.9 some examples of different mappings are given and briefly discussed. Finally, in Section 4.9 we draw some conclusions.

## 4.2 Prior work

Some prior work in the field of automatic presentation of information has already been done, namely by (Mackinlay, 1986), (Roth and Mattis, 1990), (Casner, 1991), and (Zhou and Feiner, 1996). This work generally involves some kind of data characterization and visual parameters/techniques characterization also. Some of the prior work also took into account the user's task when designing the visualizations. In this section, we will describe some of the more relevant related work.

### 4.2.1 Data characterization

Data characterization is usually the first step to understand a phenomenon or system. Developing a taxonomy helps making sense of information. Some research has already been done on data characterization for automatic presentations of information, in an effort of making automatic decisions of different presentations, depending on data characteristics.

Although many of the previous work done in this area, only considered static two-dimensional visualizations, and we are interested in 3D visualization for dynamic data, most of the data characterization work is still interesting for our purposes.

The SAGE system work on data characterization for automatic presentations (Roth and Mattis, 1990) extends and generalizes the work previously done by (Mackinlay, 1986).

Four criteria are identified for evaluation of the relevance of data characteristics:

- characteristics necessary to distinguish graphical techniques for information expressiveness
- characteristics to help ordering graphical techniques based on effectiveness at conveying information
- characteristics useful for determining how quantities of diverse or similar information can be integrated within a display
- and characteristics that users can easily apply

Based on these criteria (Roth and Mattis, 1990) several dimensions along which data can be characterized are defined:

- data types
- properties of relational-structure
- expressing relations among relations
- distinguishing unary from binary and N-ary relations
- user information seeking goals

Within the data types dimension the following categories are identified:

- sets ordering (possible values being *quantitative*, *ordinal*, and *nominal*)
- coordinates versus amounts
- domain of membership

Another related work, the Vista system (Senay and Ignatius, 1994) identifies sequences that are generally present in a visualization system for converting a data set into a displayable image: data manipulation, visualization mapping, and rendering. This same sequence model, with some additions such as the introduction of the user's task, is later cited in relevant bibliographies such as (Card et al., 1999).

For the visualization mapping – mapping data variables onto visual parameters – it is necessary to characterize the data and the visual parameters/techniques. Knowing the data characteristics relevant to visualization mapping is the first important step in designing an effective visualization. As shown in (Mackinlay, 1986) (Roth and Mattis, 1990), usually data can be divided into two major categories, *qualitative* and *quantitative*, which are then subdivided as follows:

- qualitative
  - nominal
  - ordinal
- quantitative
  - scalar
  - vector

– tensor

Quantitative data is more common than qualitative data for problems in scientific domains .

The IMPROVISE system (Zhou and Feiner, 1996) advances further in characterizing data for automatic presentations, since the prior work done was mostly concerned with 2D graphs visualization and did not take into account highly dynamic presentations of information. This system advances a little regarding time-variant data by distinguishing between static and dynamic information. IMPROVISE uses six dimensions to describe those characteristics:

- *data type* which relates to the atomicity of the data
- *data domain* that categorizes information semantically
- *data attributes* that distinguishes data properties
- *data relations* which specifies relations between data
- *data role* for characterizing data based on user information seeking goals
- *data sense* that differentiates data based on user visual interpretation preferences

The IMPROVISE system makes use of an object-oriented approach: each piece of information is an object. Each object belongs to a data domain and has a data type, properties (data attributes) and connections to other objects (data relations).

A problem-oriented classification for visualization techniques by Wehrend and Lewis (Wehrend and Lewis, 1990) positions visualization problems and their corresponding techniques in a matrix formed by two crossed classifications: *objects* and *operations*. This classification falls back into data characterization and task characterization, respectively.

In (Wehrend and Lewis, 1990) objects are basically an attempt of classifying the nature of the things to be visualized, in the target domain. The interest of this classification is that different representations are more appropriate for different object classes. Objects may be such as a scalar or a position, for instance, and the visual representation for a scalar does not necessarily fit that for a position. For example, a bar length is appropriate for representing a scalar, but is not suitable for representing a position. Possible values for objects are:

- scalar
- scalar field
- nominal
- direction
- direction field
- shape
- position
- spatially extended region or object
- structure

Operations are an attempt to distinguish between different user goals (or user tasks, in our terminology). For instance, if the user wants to read a value of a scalar, that is an *identify* operation; on the other hand, if he needs to determine the larger value between two or more scalars, that is a *compare* operation. Possible values for operations in (Wehrend and Lewis, 1990) are: *identify*, *locate*, *distinguish*, *categorize*, *cluster distribution*, *rank*, *compare within and between relations*, *associate*, and *correlate*.

(Wehrend and Lewis, 1990) applied both the objects and the operations classification to problems, obtaining as a result a matrix with problems populating the cells. Then, this matrix was populated with visualization techniques abstracted from representation problems found in related literature. This resulted in a catalog of visualization techniques, where for each cell with a given problem, one might find appropriate techniques.

### 4.2.2 User's tasks

Related work has also been done in an attempt to classify user tasks. We have already cited the work of (Wehrend and Lewis, 1990) where the classification terminology used is operations. Also in (Roth and Mattis, 1990) another possible classification for user's tasks, dating from roughly the same time, is given. The work of (Wehrend and Lewis, 1990) was further used by (Keller and Keller, 1993).

In (Zhou and Feiner, 1998) the authors distinguish between high-level presentation intents and the visual tasks that achieve them. Different presentation intents share many visual tasks in different ways. For the visual tasks, (Zhou and Feiner, 1998) built on previous work done on user's tasks

classification, (Wehrend and Lewis, 1990) (Keller and Keller, 1993) among others, and extended the classification.

Table 4.1 shows the different classifications and how they relate to each other.

| User's Task Classification | | |
|---|---|---|
| from† | from‡ | from⋆ |
| identify | lookup value | identify |
| locate | – | locate |
| distinguish | – | distinguish |
| categorize | – | categorize |
| cluster | determine | cluster |
| rank | – | rank |
| compare | compare | compare |
| associate | – | associate |
| correlate | correlate | correlate |
| – | index a structure | – |
| – | – | background |
| – | – | emphasize |
| – | – | generalize |
| – | – | reveal |

Table 4.1: User's task classification according to different references: †(Wehrend and Lewis, 1990), ‡(Roth and Mattis, 1990), and ⋆(Zhou and Feiner, 1998).

In yet another related work, (Casner, 1991) developed an automated graphic design tool that designs graphics according to the task the graphic is intended to support. The tool analyses a logical description of the task to be performed and tries to design an equivalent perceptual task. It was used to design graphic presentations of airline schedule information to support five different airline reservation tasks. Reaction-time studies indicated that the tool reduced users' performance time for the tasks (Casner, 1991).

More recently, Shneiderman (Shneiderman, 1996) proposed a Type by Task Taxonomy (TTT) of information visualizations to guide researchers that advanced a classification of high-level tasks:

- *overview* – gain an overview of an entire collection

- *zoom* – zoom in on items of interest
- *filter* – filter out uninteresting items
- *details-on-demand* – select an item or group and get details when needed
- *relate* – view relationships among items
- *history* – keep a history of actions to support undo, replay, and progressive refinement
- *extract* – allow extraction of sub-collections and of query parameters

## 4.3 Our approach

Contrary to other systems where a specific visual representation was designed for a given application, our goal is to provide different ways to visualize different sets of information and, sometimes, even the same data set, within the same application domain. We believe that the data set and the task are both important for determining the visual representation. Furthermore, we want the mapping process to be automatic and as domain independent as possible.

In our system there are four major components in the mapping process:

- the *services* that contain the data to be visualized
- the *tasks* that the user wants to accomplish
- the *adaptors* that implement the mapping rules
- the *metaphors* that are used to visualize the information

Regarding the services and the mapping, we need to characterize the service information so that we are able to map it. This involves characterizing the service data, as well as characterizing the service itself globally – for instance, whether the service is recursive or not. This subject is further developed in Section 4.4.

Although sometimes overlooked in visualization applications, we think that the task the user wants to accomplish plays an important role in how to display the information. Therefore, we take the task into account when deciding on the visual mapping for a given data set. The importance of the user task in our mapping process is further explained in Section 4.5.

The visual metaphors are the substance of our virtual worlds. They provide consistency to the visualization, and support former knowledge of the user when they are real-world based. With respect to the mapping process, it is important to have the visual metaphors fully characterized, both regarding their properties and their components – especially the visual parameters that the metaphor makes available for the information to be mapped on. The relevance of the visual metaphors in the mapping process is detailed in Section 4.7 and Section 4.7.5, and the visual parameters in Section 4.7.4.

As referred previously (see Section 3.7.4), the adaptors are non-graphical elements of the visualization layer that are responsible for implementing the mapping rules. They do the translation between a service element and a metaphoric component. The mapping rules decide what is going to be mapped on what and how. Section 4.8 further explains the role of the adaptors in the mapping process and how they work.

Instead of viewing the mapping process as a global and unique procedure, directly mapping the service onto the metaphoric world, we distinguish between two different phases. We first consider the mapping of the service structure on the metaphor structure. Contrary to previous systems (Zhou and Feiner, 1998) (Crutcher et al., 1995) (Senay and Ignatius, 1994) (Mackinlay, 1986) where the visualizations' structure could be tailored to the information, the fact that we use a metaphor to display the information imposes some constraints on the structure (Section 4.7.1). This led us to distinguish a first level of mapping that we have designated *structural mapping*.

After the first level of structural mapping, we have devised a second level of mapping that accounts for the correlation between the data and the visual parameters of the metaphor. We have named this mapping step *visual-parameters mapping*. Both mapping steps as well as the criteria for mapping are presented in Section 4.8.

## 4.4 Service characterization

In our terminology a *service* defines the time-varying information set that we want to visualize. The concept of a service arose from the need of structuring the raw data in order to be visualized in a metaphoric representation (Section 3.6.1). A service, thus, is just a way of structuring information in a manner that is more easily exploitable for its future visual representation. A service can be represented by a direct acyclic graph that comprises entities and relations. Figure 4.3 shows an example for the Network File System (NFS) service.

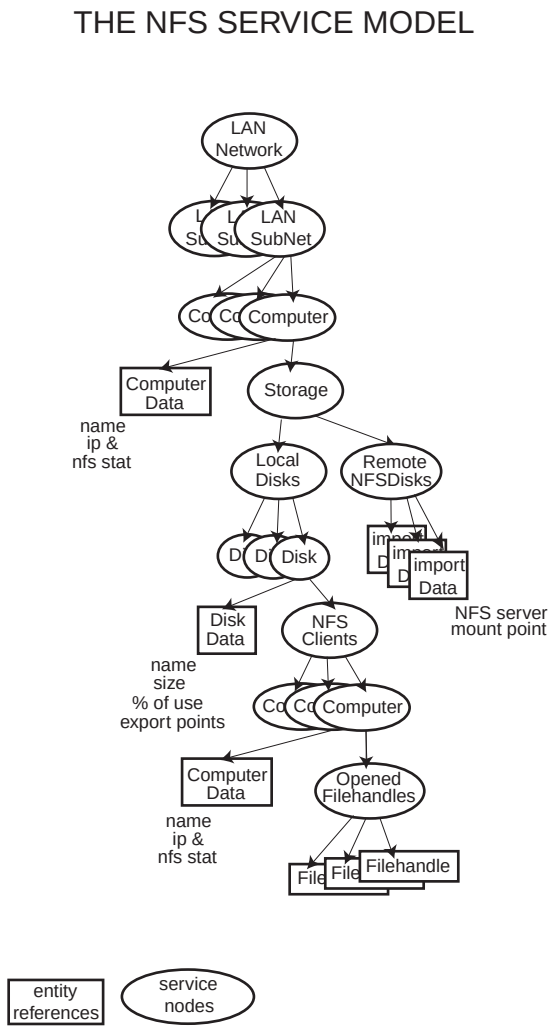



Figure 4.3: Example of the Network File System service model.

The characterization of a service is done by characterizing its elements, entities and relations. The entities are characterized by the type of data they possess (Section 4.4.1). Relations are characterized according to their basic types (Section 4.4.2).

The service also possesses a certain number of generic attributes that are useful for mapping the service information. For example, whether it is recursive or not – it is not possible to map a recursive service onto a non-recursive metaphor, without incurring on the risk of not being able to visualize all the information. These service attributes are useful for the first level of mapping – structural mapping – as they are relate to the structure

of the service and not to the actual service data.

The generic attributes of a service thus relate to its recursiveness, the nature of its hierarchy, and the number of levels that the service hierarchy may possess. The service attributes are then:

**recursive** describes whether the service is recursive or not. Possible values are yes and no. This characteristic, as we have already referred, is important since it conditions the metaphor choice: a recursive service implies a recursive metaphor.

**hierarchy** relates to the fact that some services are hierarchical by nature – for instance, a hierarchical list ('`ls`') of the current directory and respective subdirectories. We call this an explicit hierarchy. Other services are hierarchical just because of the way that we specified them – note that, in our system, a service is always described by a tree. For instance, in the NFS service we can consider a hierarchy of machines → users → processes or the other way around, processes → users → machines. The hierarchy in this case is implicit, not explicit. The possible values for the hierarchy property are thus explicit and implicit.

This characteristic is important for the mapping since there are visual metaphors that are better suited for displaying explicit hierarchical information (e.g., the conetree metaphor, the pyramid metaphor), as the actual visual representation conveys an immediate notion of hierarchy.

**number of levels** defines the maximum number of hierarchical levels (N) a service may have. A hierarchical level is counted for each service node level (see Figure 4.3). Naturally, in the case of recursive services, N is theoretically unbounded. Cycles are not permitted.

### 4.4.1 Data characteristics

We think that the best approach is to keep the data characterization as general as possible, so that it can be easily applied to other application domains not necessarily related to network monitoring and management. An excessively complex and detailed data characterization will be application domain dependent (at least, partly). This concern is relevant since the CyberNet system's framework was designed so that it can be used in other application domains for 3D information visualization of dynamic data, such as stock exchange visualization, for instance.

In our system, data is characterized along three basic dimensions:

- type
- time
- semantic

### Type dimension

The *type* dimension refers to the basic types of data, comprising the main division between quantitative and qualitative data, that can be found in numerous previous works of data characterization, such as (Mackinlay, 1986), (Roth and Mattis, 1990) or (Zhou and Feiner, 1996).

Basically, information can be divided into two major types: *quantitative* and *qualitative*; the latter can be further divided into *nominal* and *ordinal* information.

- *quantitative* information: is defined by a scalar value (e.g., size of a storage disk).
- *qualitative* information:
  - *nominal* information: is an unordered set of nouns and does not possess units (e.g., names of the machines).
  - *ordinal* information: is an ordered set but the ordering gives no information about the difference in magnitude between any different values belonging to the set (e.g., the disk size may be defined as small, medium or large, but it gives no insight regarding the actual values for small, medium, or large).

These two data types are only a first division of the data types. We further found the need to distinguish, for quantitative data whether it is a *ratio* or *percentage*, or an *absolute value*; and whether the data value is *binary* or not.

For the moment, although this data characterization may appear quite naïve, it seems sufficient for a general classification. From an analysis of the services created so far, every data piece seems to fall in these basic categories. In fact, most of the data encountered in network management, network monitoring or web-server analysis is quantitative data, either absolute values or percentages, and nominal. Nevertheless, to achieve a finer mapping, this data characterization may eventually need finer detail, allowing for more complex data types such as structures.

### Time dimension

The *time* dimension classifies data according to its time-dependent behavior, which is particularly important since our framework has to deal with highly dynamic data.

The time dimension does not usually appear in previous systems of automatic construction of visualizations since most of these systems only deal with static visualizations of information. For our system, it is of major importance to characterize information according to its time-dependent behavior.

We want to visualize information that is highly dependent on time and we want the displayed information to be dynamically updated in real time. The information is thus classified regarding whether it is time dependent or not (i.e., static or dynamic) and also according to more specific behavior (e.g., continuous or discrete, and the frequency of change).

According to the time dimension, we thus distinguish *static* from *dynamic* information. For dynamic information we further divide its time-dependent behavior according to whether it is *discrete* or *continuous*, and the *frequency of change* (possible values being frequency *high*, *medium*, *low* or *zero*; zero is for taking into account static data, the other values are tuned according to the service).

Whether the data is dynamic or static influences greatly the mapping of the data values. For instance, highly dynamic data is usually not mapped onto position, otherwise we would create virtual worlds where the location of their elements frequently changes. This approach would surely confuse the user and render any attempt to build a cognitive map of the virtual world very difficult. The problems and constraints related to the visualization of dynamic data are further developed in Chapter 7.

### Semantic dimension

The *semantic* dimension provides the context in which the data is going to be used (useful for the task-dependent mapping) and allows for the encoding of more important data more effectively. It should be stated that one piece of data (i.e., one entity) may belong to more than one service. The importance of the data may also vary according to the service. The semantic dimension allows for specifying the relative importance of the service data, and also adds context-information regarding the task (operation) in which the data is going to be used. This provides additional valuable information for the visual encoding.

For instance, a single set of information regarding a NFS disk (e.g., name,

size, percentage of use and export points) may have different degrees of importance according to the service. In a service monitoring the usage of the resources, the disk size and percentage of use may be more important than the export points. On the other hand, in a service regarding network security, the information concerning the export points is probably more important than data like disk size and usage. We thus specify, for the former service, a task regarding disk size and percentage of use (e.g., compare); for the latter case, we specify a task for the information that interests us most (e.g., identify the export points). Furthermore, we order the visual priority of the data according to its task priority – more important data is treated more prominently. This is useful because when, for the same data type, decisions must be made regarding the visual encoding, the mapping engine can encode more important data more effectively, while still taking into account the user's task (see Section 4.8 for details regarding visual-encoding effectiveness according to data type).

### 4.4.2 Relation types

The type of relationship among entities also influences the mapping. For instance, a dependency relationship is preferably mapped onto a visual dependency also. For example, open file handles on a mounted disk are mapped on windows of a floor representing that computer (Figure 4.6); both the data and the visual elements can not exist without their parents – i.e., file handles do not exist on their own, and the same for the corresponding windows and floors.

In our system, relations can belong to one of the following types:

- *set-of* relations define relationships among elements that share some common property and are of the same type. So, set-of relations define the relations existing among identical (or closely related) siblings.
- *group* relations define a relationship among elements that share some common property (e.g., NFS hard disks, either local or remote), but are not identical.
- *dependency* relations define a relationship of belonging between two or more elements (e.g., parent directory and its children).

The relation type intervenes in the mapping in the following manner: relations of type set-of are preferably mapped onto visual elements of the same type, contrary to group relations that usually oppose this rule; dependency relations, as said above, are mapped onto visual dependencies. For example,

the three cases are present in Figure 4.5 as computers, local disks and remote disks, and file-handles, respectively.

## 4.5 Tasks

After some initial experiments without taking user's tasks into consideration, we came to the conclusion that the tasks that are to be performed on the information also need to play a role regarding the visual mapping of information. Although the user's tasks are intimately related to the service, we feel that there are cases where it is worth to look for a better information-visualization encoding depending on the user task, even though the service and the service data remain the same.

To comply with the above, we also have a characterization of the tasks. For mapping purposes, the mapping rules thus take the tasks into account. For instance, color is useful for correlation tasks. For comparison purposes, heights must be positioned all on the same plane – it is very hard to accurately compare height on different plane levels.

So far, the tasks that we have globally identified from the services already developed are the following:

- *identify* relates to lookup a value (e.g., identify server)
- *compare* is used for comparing values (e.g., compare disk sizes)
- *detect* relates to being able to identify outliers or extremes (e.g., detect the biggest disk)
- *sort* intends to order according to a specific parameter (e.g., sort files according to number of positive results of the Unix `grep` command)
- *correlate* identifies related objects (e.g., correlate server to respective clients)
- *cluster* relates to grouping a set of objects (e.g., aggregate all machines belonging to the multimedia communications dept.)
- *filter* has the purpose of filtering out according to a data value (e.g., show all disks with CPU usage over 90 percent)

## 4.6 Why use visual metaphors?

The human conceptual system is fundamentally metaphoric in nature (Lakoff and Johnson, 1980). There are studies reporting that the knowledge of

a large-scale space is isomorphic to the information stored in a graphical map – this is known as the "Map in the Head" metaphor (Kuipers, 1982). This metaphor states that information is stored in, and retrieved from, the cognitive map using the same operations by which information is added to or retrieved from a graphical map. In other words, the functional behavior is the same in both contexts: the mind's eye when inspecting the "map in the head" is functionally identical to the physical eye inspecting a graphical map. On the other hand, maps are not mirrors of reality; they are a means of understanding it (Wurman, 1989).

Our motivation to use visual metaphors to depict the information derives from the fact that the use of metaphors allows for the presentation of information in a way that the user is already familiarized with. Furthermore, since metaphors let the user draw from previous knowledge to understand new concepts (Lakoff and Johnson, 1980) (Newman and Lamming, 1995), we have a means of presenting unknown information to the user in a way that he is able to understand.

In a seminal work by Lakoff (Lakoff and Johnson, 1980), a metaphor is defined as a rhetorical figure whose essence is understanding and experiencing one kind of thing in terms of another. A metaphor thus allows the user to move from familiar concepts to unknown ones, in this way incorporating new knowledge by utilizing patterns of known relationships. This property gains even more relevancy when we consider that most of network data has no natural physical representation. We can use a virtual representation of a workstation that is a visual facsimile of a real world machine, but how do we represent a process, for example?

Additionally, the use of metaphors based on the real world (e.g., city metaphor – Figure 4.1, solar system metaphor – Figure 4.9) or otherwise rather abstract metaphors (e.g., cone-tree – Figure 4.2), provides consistency to the information presentation. A metaphoric world gives a consistent look and feel to the information presentation. This is important since it makes it easier for the user to construct and then maintain a mental model of the visual representation. Moreover, metaphors also provide for a consistent way of navigating in a virtual world – as we will further explore in Chapter 6.

There are numerous examples of different metaphors that can be used for information visualization. Some particular examples, based on the real world and referred in (Benyon and Höök, 1997), are: a countryside metaphor that could include different types of terrain (e.g., wilderness, desert) for different types of information; the open sea, which encourages a distinction between the surface and depth, thus relating to information being hidden beneath the surface and only available if the user dives down; or the night sky, which may

represent objects, clusters, and patterns, and provides a very large digital canvas for the display of information.

Nonetheless, designers need to be careful with the use of visual metaphors. The user utilizes the metaphor to form a cognitive model of the system and establish expectations regarding the system's functionalities. If the system fails to meet the user's expectations, the user can then be discouraged and even disoriented by the system's behavior.

One example of inconsistency that is fairly omnipresent in literature is the use of the trash can icon in the old Macintosh systems (before System 7) to eject a floppy disk or to unmount the hard disk as opposed to deleting files and directories. (Erickson, 1990) contains a detailed description of the process that led to this solution. On the other hand, care must be taken with the metaphor in order that accidental properties are not confused with the real properties of the phenomenon being studied (Benyon and Höök, 1997). This relates to the problem of "lying" with visualizations (Rogowitz and Treinish, 1996), already tackled in Section 4.1.1.

Visual metaphors provide new ways of depicting, interacting with and navigating in the virtual world. For instance, in a solar system based metaphor, we might use a virtual spaceship to navigate in the world, and we might interact with the data by creating new galaxies with different data attributes. We examine metaphor-dependent navigation in Chapter 6. We have already designed and implemented several different metaphors, such as a virtual city, a solar system, a library, or a pyramid, for different visualization purposes, as discussed in Chapter 5.

## 4.7 Metaphors

As we have seen in Chapter 3, a metaphor is described by an acyclic graph, composed of the metaphor's elements: *layout managers* and *glyphs* – see Figure 4.4 for an example of the city metaphor description graph. The fact that the metaphor is described by an acyclic graph is of great use. In fact, since the service is also described by an acyclic graph (see Figure 4.3), the mapping of the service onto the metaphor becomes easier, as it may be reduced to a one-to-one mapping.

In order to map information onto the visual metaphors, we need to know which visual parameters are available in the metaphor for the information mapping. In other words, we need to characterize the metaphor and its visual components. Since a visual metaphor is composed of layout managers and 3D glyphs, in order to characterize a metaphor we need to classify also those components. However, the components classification is not enough; there

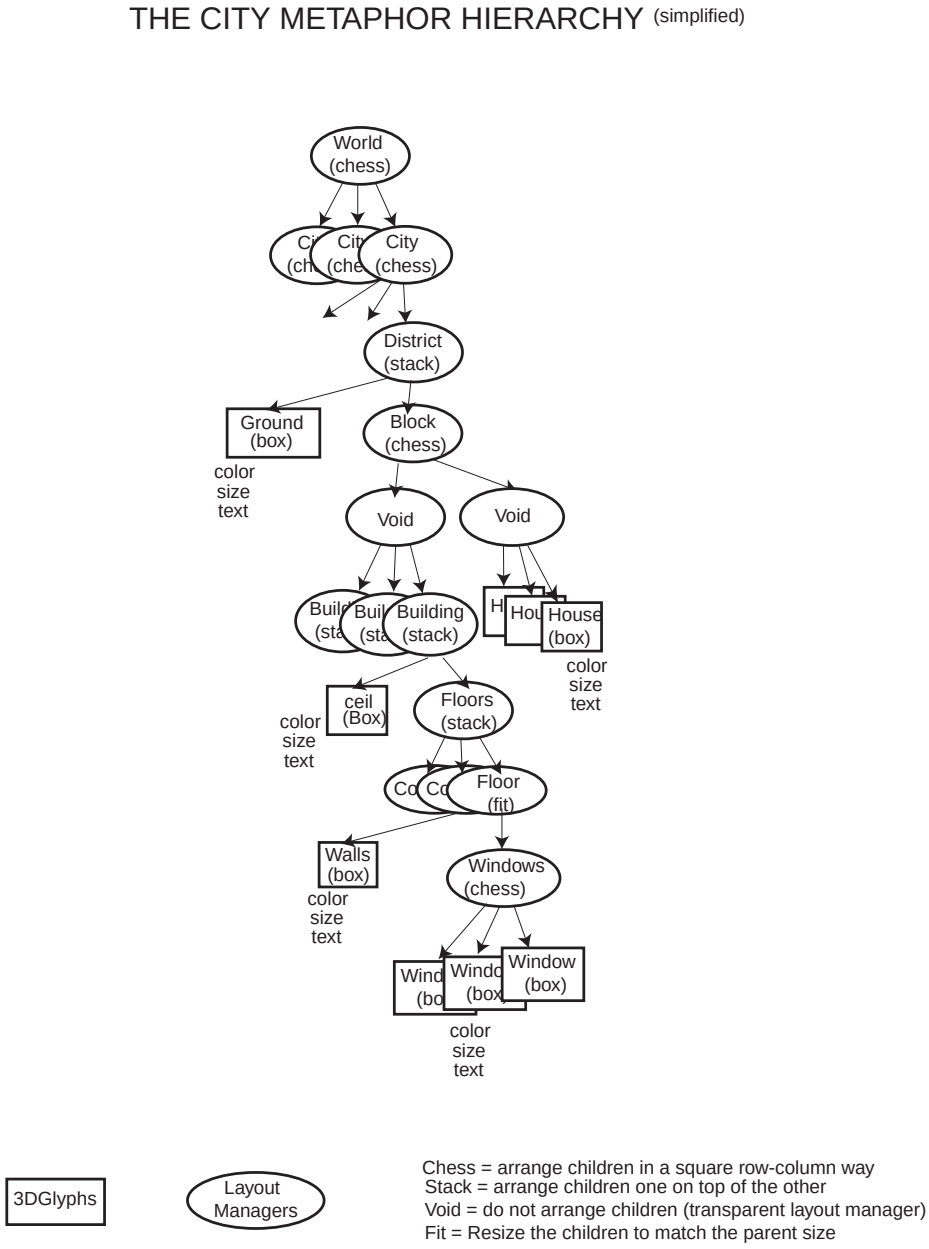



Figure 4.4: Example of the city metaphor model.

are some constraints, either in the capacity of displaying the information or relating to the efficacy of that presentation, that do not arise from the metaphors components, but from the metaphor structure (e.g., whether the metaphor is recursive or not).

In the next sections we are going to present the constraints to which a graphical metaphor must obey in order to consistently display visual information, and the layout managers and 3D glyphs characterization. We also present the visual parameters that, so far, we use in our system to map the information. Finally, we summarize all this information to achieve a visual metaphor's characterization for our mapping system.

### 4.7.1 Metaphor constraints

There are some constraints imposed in the virtual world that derive from the metaphor itself and not from its components abilities for displaying information. Nonetheless, these constraints have an impact on the visual presentation and thus condition the ability of the metaphor to visually display the information.

The added value from using metaphors based on the real world for visualizing information comes from the fact that the user can relate to the way the information is being presented, since it uses the same underlying rules that are present in the real world (Lakoff and Johnson, 1980). For instance, in the city metaphor (where information is mapped onto districts, blocks, streets, buildings and so on) there is a natural hierarchy between the different elements (e.g., buildings belong to blocks, blocks are part of districts and districts belong to a city) as depicted in Figure 4.4.

On the other hand, a service is also defined, via the service graph, in a hierarchical manner, with several levels of hierarchy. In the first step of mapping – structural mapping: mapping a service on a metaphor – the number of hierarchical levels present in the service should be, at most, equal (or fewer) than the number of hierarchical levels of the metaphor. Otherwise, we risk losing the service-hierarchy identification, as there would be some different service levels that would have to be mapped on the same hierarchical level of the metaphor.

There is also another type of constraint that may restrict the metaphor ability, taken as a whole, for visually presenting information. We already have referred that the added value from the use of metaphors based on the real world is the user's inherent capacity of relating to the way the information is presented. This is due to the fact that the user can make use of prior knowledge of how the elements of the metaphor are spatially arranged in the real world, and then build on that prior knowledge to understand the relation between the different elements (Newman and Lamming, 1995).

The quality of preserving the same spatial relations and underlying notions between the real world and its virtual counterpart is called consistency.

Thus, a virtual world must be created and retain a certain degree of consistency, even if the data that is being displayed is constantly changing. This obviously poses restrictions on the information mapping.

Such restrictions may be related to the comprehension of the metaphor. For instance, in a city, buildings are usually tall and thin. Even when they are wider than taller, this ratio is never a big one. If, for instance in a cityscape metaphor (Figure 4.6), choices are made so that a data value with a wide range is mapped onto the width of a building and a lower range data value is mapped to its height, we risk losing the resemblance of the visual element with an actual building – it will look more like a wide flat platform and hence the metaphor usefulness is impaired. This is why we give the size constraints special attention when defining the visual metaphor (Section 4.7.5).

There are other constraints that are also related to the consistency of the metaphor and are aimed at avoiding disturbing the user's mental model of the visual world; for instance, again for the case of the cityscape metaphor, we impose a constraint that makes sure that we only map static data on the position of the skyscrapers in the virtual city – otherwise, if we did map dynamic data on the building's actual position, we would see buildings moving all over the city! This kind of constraint is thus taken into account by our system and is a part of the characterization of the visual metaphors.

### 4.7.2 Layout managers' mapping desiderata

A 3D metaphoric virtual world is built using two types of graphical components: the 3D Glyphs (3DGs) and the Layout Managers (LMs). As referred in Section 3.7.2, layout managers are metaphoric components that arrange their children – other layout managers or glyphs – in the 3D space; in other words, they intervene by placing their children in the virtual world according to the metaphor rules. For instance, a solar-system metaphor uses an orbital layout manager for placing elements in an orbit around a specified center, in the way the satellites orbit around the planets. In the metaphor model tree (Figure 4.4), layout managers correspond to the intermediate nodes.

Layout managers are characterized according to their skills for positioning their children in the virtual space. A stack LM, which places children on top of each other along one dimension, is characterized according to its mapping parameters (e.g., position along one dimension, two dimensions fixed), its placing order (e.g., place by order of creation), and the task it may be used for (e.g., sorting), among other characteristics. On the other hand, a chess LM, that places children in a grid-like manner on a plane, positions children

along two dimensions and may be used comparison purposes (specially in the case where it places its children on a horizontal plane).

We have also found that there is a need for layout managers that have different skills than those of spatially placing children. In particular, we have felt the need for the existence of layout managers that are able to scale visual elements, thus doing other tasks than placing elements since they interfere in their dimensions. For instance, they may intervene in order that the metaphor's consistency is not lost due to the dynamic character of the information being presented. An example of this kind of layout manager is the `fit` layout manager employed in the city metaphor (Figure 4.6, for instance) in order to fit the windows on the walls of a floor, as is visible in the simplified model of the metaphor depicted in Figure 4.4.

Examples of layout managers skills are: fit, stack, orbit, chess, among others. For example, in the city metaphor, the `chess` layout manager is used to place the buildings in space and the `stack` layout manager is used to pile up the floors to construct the buildings (see Figure 4.4).

In this section we examine the requirements the mapping of visual information imposes on the characterization of the layout managers. That is, we define the layout managers' desiderata for visual mapping from the point of view of the information regarding the capabilities of the layout managers necessary to accomplish the mapping process.

#### Mappable visual parameters (position)

Regarding the visual mapping of information, layout managers have essentially one mapping parameter: the position. Hence, for each layout manager, we are interested in knowing what dimensions regarding the position are still available for mapping. This must be done taking into account the constraints imposed by the metaphor, as we show further (Section 4.7.1). It should be noted that, to date, we have not yet used orientation as a mappable parameter.

#### Placing strategy

Another important characteristic of a layout manager, one that actually defines its skills, is its placing strategy. By placing strategy we mean how does the layout manager place its children in space (e.g., in an orbit, in a chess-like board, in a stack, and so on). This positioning of the children in space, besides determining the nature of the layout manager, is also important regarding the mapping criteria. For instance, the layout managers' placing strategy influences the task-driven mapping: if a layout manager places its

children in a checkerboard horizontal plan, then its very useful for comparing heights; on the other hand, a layout manager that places its children in a vertical stack, is not interesting for the same height comparison task.

### Placing order

Related to the placing strategy, the placing order is also an important characteristic of the layout managers. By imposing a specific order for the placing of its children we can control the position of the children in space and use that order as a mapping parameter, if we so wish. As an example, the placing order may be just a reflection of the creation order (i.e., children are placed in space by the same order as they are created) or it may be dependent on another different parameter (e.g., by size). This is specially important for the layout managers that are most suited for ordering children, like the stack layout manager.

### Placing policy

It is also important to know if the layout manager has an adaptive or fixed placing policy. In other words, if the space allocated for each children is fixed (e.g. takes the biggest child and allocates the same space for all the other children) or if it is adaptive (computes every children size and places children according to an efficient use of space). The latter is more interesting from the point of view of screen real-estate usage, but is considerably heavier from the computing point of view.

### Task

We have also found it relevant to classify the layout managers regarding the user's tasks they are more adequate to perform. For instance, as we have referred before, a chess layout manager placing its children on a horizontal plan is useful for comparison and detection tasks; a stack layout manager is suited for sorting objects.

It is important to notice that the same layout manager may have different capabilities regarding the users' tasks according to the dimensions in space it uses. For example, the cited chess layout manager is useful for comparing heights of children if it is a `chess(X,Z)` LM – i.e., if it places children in a horizontal plan; in the case of a `chess(Y,Z)` LM, where the plan is vertical, the same reasoning does not apply.

This characterization of the layout managers according to task parameters, allows us to effectively take the user's task into consideration when

doing the mapping, in a transparent manner. With this in mind, in the characterization of the layout managers, we have a field that contains information regarding the task(s) the layout manager is better suited to, in an ordered manner (i.e., most suitable task in the first place and so on).

#### Visual parameters constraints

It is also important to know what constraints are applicable and where. In addition to size constraints are constraints related to the visual parameters of the layout managers' children. For instance, it is interesting that a LM has the possibility to impose a color for all its children – thus, in practice, issuing a `color.constraint` to all its children (`color = lock`). There is also the possibility of one of the child elements imposing its own visual parameter (`color.constraint = self`) independently of the restriction imposed by the parent.

Eventual constraint management conflicts are dealt with in the same manner as the constraints propagation strategy we have developed for dealing with dynamics – which relates mainly to size constraints propagation – that is examined in detail in Chapter 7.

#### Layout manager example

For example, the chess layout manager used in the city metaphor hierarchy for placing the districts in a city as depicted in Figure 4.4 is essentially:

```
LM chessCity{
   structural{strategy = chess(X,Z)
              policy = fixed
              placing = orderByParam(orderOfCreation)}
   visual{pos.X = free
          pos.Y = lock
          pos.Z = free}
   task{compare,identify,correlate,detect}
}
```

As explained further in Section 4.7.5, the keywords `free`/`lock` mean that the visual parameters are available/unavailable for mapping data.

### 4.7.3 3D Glyphs' mapping desiderata

3D Glyphs are graphical elements that offer a number of visual parameters to map information on. They are the visible 3D components of the metaphor and the number of visual parameters that they provide for information

mapping is directly related to their complexity. In the acyclic graph that describes the hierarchy of the metaphor, the 3D glyphs correspond to the leaf nodes of the tree (Figure 4.4).

### Type

The 3D glyphs in our system, so far, are fairly basic and primarily defined by a type field. The glyph type basically defines its visual aspect (e.g., box). Regardless of the apparent simplicity of the 3D glyphs, we are able to construct complex worlds by using a combination of multiple simple elements.

### Visual parameters

The 3DGs are characterized according to the visual parameters they offer for information mapping. For instance, a box 3DG is defined by such parameters as geometric (e.g., size, shape), surface (e.g., texture, color – hue and saturation), textual (e.g., text label) and so on.

Regarding the mapping process, essentially we have to be able to determine what visual parameters are available for mapping information. This kind of knowledge must be available for every leaf node (glyph) of the metaphor tree.

### Constraints

Another important information that must be available relating to the glyphs is whether there are constraints imposed or not. These constraints are related to the visual parameters of the glyph and typically affect the size, though they may affect other visual parameters such as color.

### 3D Glyph example

For example, and always relating to the metaphor model in Figure 4.4, for the 3D glyph used for representing a house in the residential district, we have (simplified):

```
child opt mul 3DG house{
   structural{size.*.constraint = top-down}
   visual{type = box
          size.* = lock
          color = free
          label = free
          transparency = free}
}
```

Note that, as referred before (Section 4.7.1) the size of the house is imposed from higher up in the hierarchy (`size.*.constraint = top-down`) and it is also not available for mapping (`size.* = lock`).

### 4.7.4 Visual parameters

The visual parameters we have so far identified as relevant to our system's needs are the following:

- geometric
  - shape
  - size
  - position
  - orientation
- photometric
  - color hue
  - color saturation
  - texture
  - transparency
- textual
  - label

Undoubtedly this is a work in progress and the classification may be debatable. However, and so far, they have met our requirements. In fact, our virtual worlds, though they seem fairly complex when visualized, are built using quite simple objects and using only a reduced number of visual parameters. It is the combination of the different elements and the design of the metaphor itself that accounts for the final (apparent) complexity of the world.

In the example of the city metaphor, for instance, for the visualization of the NFS service, in which the resulting world is rather impressive (Figures 4.6 and 4.7, for instance), we have only used "simple" visual parameters for the information mapping. From Table 4.6, which describes the mapping for one of the city worlds, we can observe that we use mainly color, size, and text for the mapping of the actual data values.

It should also be noted that position and orientation constitute a special case – both parameters are the responsible of the layout managers. Additionally, orientation has not been used at present state as a mappable visual parameter; so far, it has only been used to provide a better understanding of the metaphor (e.g., for the conetree metaphor).

### 4.7.5 Metaphor characterization

From the information presented so far regarding the metaphor's constraints (Section 4.7.1), the desiderata for mapping regarding the layout managers (Section 4.7.2) and the 3D glyphs (Section 4.7.3), we are now able to characterize the metaphor.

We divide the characterization of the metaphor in three major categories:

- information regarding the metaphor structure – which relates essentially to the metaphor seen as a whole
- the structural constraints – concern mostly information regarding the layout managers, as they affect the structure of the world
- the visual constraints – relate more to the 3D glyphs, since they refer to constraints imposed on the visual parameters

We use keywords to cope with the metaphor characterization.

#### Metaphor structure

Regarding the metaphor structure, we characterize:

**required/optional elements** refers to whether an element is mandatory (`req`) or not (`opt`). E.g., in a city metaphor, buildings are mandatory but trees are optional – note that this is for a *virtual* city!

**multiplicity** describes whether an element is unique (`uni`) or there are multiple (`mul`) elements (`N`).

**relation type** specifies whether a relation is a group (`group`), a set-of (`setof`), or a dependency (`dep`) relation. The definition is the same as for the service tree relations (see Section 4.4.2).

**recursiveness** refers to whether a part of the metaphor hierarchy is recursive (`REC`) or not.

### Structural constraints

For the structural constraints we consider the following:

**placing** regards the order in which a layout manager places its children (other LMs or 3DGs) in the space. Possible values are the following:

```
placing(LM) ::= orderByParam(param)|orderByList(children)
param ::= orderOfCreation|name|size
```

where `param` refers to the parameter of placing and `children` is a ordered list of children to be placed in the order specified.

**policy** refers to the layout managers' placement policy, either fixed or adaptive:

```
policy(LM) ::= fixed|adaptive
```

where `fixed` means that the layout manager allocates a fixed space for each child (e.g., maximum size of child) and `adaptive` corresponds to a layout manager that allocates variable space for each child (e.g., based on individual sizes) in order to use space more effectively.

**size constraint** deals with the constraints imposed on the size of the metaphoric components.

```
size.?.constraint ::= top-down|bottom-up|self
```

where `?` represents the coordinate (`X`,`Y`, or `Z`) along which the size is constrained (`*` for all directions). `Top-down` means that the constraint is imposed from above (parent), `bottom-up` the constraint is imposed from below (children), and `self` means that the component has no external constraint imposed, rather it defines its own size.

The size constraint is very important since it relates directly to the use of space – that is why it has a special visibility in the structural constraints, that other visual parameters do not have. The size constraint is, namely, used to cope with the dynamics of the information (e.g., when all the children of a layout manager must be resized down to make room for a new entity/glyph).

More on the size constraints propagation and on different constraints propagations policies according to direction in Chapter 7.

### Visual constraints

By visual constraints we refer to the constraints imposed on the visual parameters available for mapping. These constraints refer to constraints imposed, prior to mapping, on the values of the visual parameters and on their availability for mapping information:

**values** may be fixed (`fix`) or variable (`var`). A fixed value means the visual parameters has some constraint imposed that fixes its value (e.g., `var Y, X=Y` for a square box). A fixed value may be used for mapping provided the visual constraint is respected. A variable value means the value may be changed without constraints.

**availability** for mapping is determined by the keyword `free` (visual parameter available for mapping) or `lock` (visual parameter non-available for mapping). A visual parameters may be locked, i.e., non-available for mapping information, due to a number of reasons.

As an illustrative example, take the city metaphor used for the visualization of the NFS service (Figure 4.1). While designing the metaphor, we have imposed that the houses, in the residential blocks, all have the same size. Hence, the size of the houses is constant, and their appearance can not be mistaken for a building – we wanted to make a clear distinction between houses and buildings, so that we are able to map distinct things on either. Note that houses encode imported NFS disks and the buildings encode local disks. The distinction between remote and local disks is fundamental for the analysis of a NFS system.

### Metaphor template

From the information presented above, we can introduce the metaphor template used for mapping (simplified):

```
metaphor <metaphor-name>{
         req|opt uni|mul dep|group|setof  LM|3DG|REC <name>{ }

case: LM

   req|opt uni|mul dep|group|setof LM <LM-name>{
       structural{strategy
                  policy
                  placing
                  size.?.constraint}
       visual{pos.?}
```

```
        task{<task-name>,<task-name>,...}
        child req|opt uni|mul dep|group|setof LM|3DG|REC <name>{ }
        ...
    }


case: 3DG

    req|opt uni|mul dep|group|setof 3DG <3DG-name>{
        structural{size.?.constraint}
        visual{type
               <visual-param>.?.constraint
               <visual-param>}
    }

case: REC

    child req|opt uni|mul dep|group|setof REC <name>{metaphor|LM|3DG}

}
```

## 4.8 The mapping engine

In this section we describe the complete mapping process: the types of mapping we distinguish, the mapping criteria, the steps of the mapping process, and the mapping rules.

### 4.8.1 Types of mapping

We distinguish between two different types of mapping, *structural* (or *functional*) and *visual parameters* mapping:

- *structural mapping* accounts for the mapping of the service structure on the visual world structure. This kind of mapping deals mostly with mapping relations on layout managers.

- *visual parameters mapping* accounts for the mapping of the service data onto the visual parameters provided by the graphical components of the virtual world. This kind of mapping deals mostly with mapping entities on the visual parameters of the glyphs that constitute the virtual world.

### 4.8.2 Criteria for mapping

As mapping criteria we use the *expressiveness* criterion and the *effectiveness* criterion, in a manner similar as they are used in (Mackinlay, 1986):

- the *expressiveness* criterion makes sure that the chosen metaphor is capable of representing all the service information, and encodes only the service information and nothing more. The result of the application of this criterion can be more than one metaphor, meaning that there are several metaphors capable of displaying all the service information. A step further in the quest for the best representation is the application of the effectiveness criterion.

  Expressiveness refers thus to the capability of the metaphor to represent all the information we desire to visualize. For instance, if the number of visual parameters available in the metaphor for displaying information is fewer than the number of data values we wish to visualize, the metaphor will not be able to meet the expressiveness criterion. This is due to the fact that we cannot map more than one data value onto one visual parameter.

  The relationship between data values and visual parameters has to be a univocal relationship; otherwise, if more than one data value is mapped onto the same visual parameter, then it will be impossible to distinguish one value's influence from the other. On the other hand, there can always be visual parameters that are not used to map information, as long as there is no need for them to be utilized.

  Moreover, not only the number of visual parameters has to be sufficient to map all the data, but also, they must be able to map the right data. I.e., there are visual parameters that are not able to map a specific category of data; for instance, shape is not useful for mapping quantitative data.

  Additionally, to obey to the expressiveness criterion, the visual parameters should only encode the data and not add nonexistent information. For instance, size should not be used to map nominal information, as it may foster the erroneous assumption that the information is ordered.

- the *effectiveness* criterion searches for the most effective mapping after having passed the expressiveness criterion. The most effective mapping is decided based on the adequacy of the different visual parameters to represent different types of information. It is also dependent on the task the user wants to accomplish.

The effectiveness criterion relates thus to the efficacy of the metaphor as a means of representing the information. Along the effectiveness dimension we can further distinguish several criteria: effectiveness regarding visual perception, regarding visual impact, regarding optimization (e.g., number of polygons need to render the world). In our system, at present state, we only studied effectiveness regarding accurate interpretation of the data by the user (visual perception).

This criterion implies the categorization of the visual parameters according to its capabilities of encoding the different types of information. Moreover, this also implies categorizing the information according to its importance so that more important information can be encoded more efficiently when options must be taken (semantic dimension, Section 4.4.1).

For the application of the expressiveness criterion the mapping engine only knows whether the metaphor and its visual elements are capable of encoding the information or not. There is no order of preference or judgment on the efficacy of the encoding capabilities.

For the application of the effectiveness criterion, the metaphors and the metaphoric components are judged according to their capability of best encoding the information. E.g., for the glyphs, the visual parameters are ranked according to their capacity for encoding the various types of data (Table 4.5).

### 4.8.3 The mapping process steps

The two types of mapping and the two criteria for mapping are not the same thing, even though expressiveness does indeed relate more to the structural mapping; and, similarly, the criterion of effectiveness is more related to the mapping of the visual parameters.

In the mapping process, both criteria should be applied to both steps, taking into account that the criteria of expressiveness always comes first. There is no point in searching for efficiency when it is not certain that the information can actually be mapped. The actual strategy for the mapping process is the result sequence that follows from applying (in order) the above criteria to the two steps of mapping:

1. The *first* step in the mapping process is verifying that the metaphor is able to structurally map the service information. For this, we have to compare the general attributes of the service and the metaphor, and evaluate whether they are compatible or not. For instance, a recursive service cannot be mapped onto a non-recursive metaphor. Also, the

mapping capabilities of the layout managers have to be evaluated in order to see if they are able to map the relations existent in the service.

The result of this step is a set of metaphors that are able to structurally map the service information. At this point, there is still no information regarding the efficacy of each one of them.

2. The *second* step is the visual parameters mapping from the point of view of the expressiveness criterion. For this we evaluate the number of visual parameters and their capabilities, in order to ascertain if the service data (entities) can actually be mapped.

   This kind of mapping is done for all the metaphors that have surpassed the first stage; i.e., the metaphors that are capable of mapping the service structure. The result of this step is a set of metaphors that can completely map the service structure and data.

3. The *third* step is then to apply the criterion of effectiveness to the result of the structural mapping, obtained from the first step, that have also succeeded in surpassing the second step. This criterion uses the information regarding the encoding capabilities of the layout managers with respect to relations, combined with the knowledge of the users tasks, to find the most effective structural mapping.

   The outcome of this mapping process is a classification of the available metaphors for a given service, capable of functionally mapping the service information. The classification is ordered in terms of effectiveness for encoding the structure of the information.

4. The *last* stage of the mapping process is evaluating the efficacy of the metaphor with respect to the data encoding. For this we have to apply the criterion of effectiveness to the visual parameters mapping. This criterion is supported on the encoding capabilities of the visual parameters for the different types of data and on the importance of the data, as defined in the service description.

   The result of this last step of mapping is the best metaphor and visual parameters mapping for encoding a given service, taking into account the users tasks and providing the best encoding for the most important data, within the context of the service.

The complete process ensures that the mapping obeys the criteria of expressiveness and the criterion of effectiveness, both for the structural mapping and for the visual parameters mapping. Our mapping experiments so far have focused on the expressiveness criterion.

### 4.8.4 Mapping rules

The *mapping rules* are implemented by the adaptors (Section 4.8.6). The mapping rules specify everything that relates to the association of the different elements that play a role in the mapping process. Most of the visual perception rules are derived from previous studies for automatic mapping for two-dimensional visualization, such as (Mackinlay, 1986), or knowledge-based visualizations, such as (Senay and Ignatius, 1994), and extrapolated for 3D visualization where the case applies.

The concept of best mapping is a not a static one. We can have different types of "best" mapping. The definition of best mapping results from the application of the effectiveness criterion; nonetheless, the effectiveness criterion can be based primarily on the visual perception rules, or on the task, or even, eventually, on optimization (e.g., number of polygons needed for the rendering). For the moment, though, only visual perception concerns and the users tasks were into account when trying to define the best mapping.

The mapping rules are supported on the visual perceptions presented in the next section.

### 4.8.5 Visual perceptions

Visual perceptions relate to how efficiently the different visualizations and visual parameters are perceived by the users. The visual parameters must then be characterized according to their capability of displaying the different data types (expressiveness) and to their efficacy in doing so (effectiveness).

The objective of a visual parameters characterization is then to classify 3D visual parameters (e.g., size, texture, transparency) according to their ability for displaying data values of different types. Some previous work has already been done for 2D visual parameters characterization, namely the seminal work done by Mackinlay (Mackinlay, 1986). This work was based on a previous study dealing with quantitative data only (Cleveland and McGill, 1984).

#### Effectiveness

The work in (Cleveland and McGill, 1984) ranked visual parameters according to their effectiveness for displaying quantitative data. Although it only examines one type of data (quantitative) this work is quite relevant since it has the support of experimental evidence, which is seldom found in related work. Table 4.2 gives the ranking with respect to the way users perceive different visual parameters when they encode quantitative information.

| | |
|---|---|
| More accurate | Position |
| ⇑ | Length |
| | Angle / Slope |
| | Area |
| ⇓ | Volume |
| Less accurate | Color / Density |

Table 4.2: Accuracy ranking of visual parameters for quantitative data, according to (Cleveland and McGill, 1984).

Since the work behind the ranking shown in Table 4.2 does not address qualitative information, Mackinlay (Mackinlay, 1986) built on this work to extend it to other types of information and additional visual parameters that are not involved in the encoding of quantitative data. The resulting ranking for the three basic data types – quantitative, nominal, and ordinal – is shown in Table 4.3. This extended ranking was built using existing psychophysical results and by analyzing different perceptual tasks, but was not empirically validated.

| **Quantitative** | **Ordinal** | **Nominal** |
|---|---|---|
| Position | Position | Position |
| Length | Density | Color Hue |
| Angle | Color Saturation | Texture |
| Slope | Color Hue | Connection |
| Area | Texture | Containment |
| Volume | Connection | Density |
| Density | Containment | Color Saturation |
| Color Saturation | Length | Shape |
| Color Hue | Angle | Length |
| Texture | Slope | Angle |
| Connection | Area | Slope |
| Containment | Volume | Area |
| Shape | Shape | Volume |

Table 4.3: Mackinlay's ranking of perceptual tasks (Mackinlay, 1986). Tasks grayed out are not relevant for the corresponding type of data.

### Expressiveness

The work above only relates to effectiveness – i.e., how effective are different visual encodings for different data types. In order to choose between different encodings we have to first examine whether the visual parameters are in fact able to encode the information (and nothing more) – i.e. expressiveness.

In (Mackinlay, 1986) the visual techniques introduced by Bertin (Bertin, 1983) were used to construct a table of visual techniques expressiveness, again for the three major types of data – quantitative, ordinal, and nominal. The resulting ordering is shown in Table 4.4.

| | **Quantitative** | **Ordinal** | **Nominal** |
|---|---|---|---|
| **Size** | • | • | – |
| **Saturation** | • | • | – |
| **Texture** | | • | • |
| **Color** | | * | • |
| **Orientation** | | | • |
| **Shape** | | | • |

Table 4.4: Expressiveness of retinal techniques according to Mackinlay (Mackinlay, 1986). The * indicates that although the full color spectrum is not ordered, parts of it are ordinally perceived. The – indicates that size and saturation should not be used for nominal data, because they risk being ordinally perceived.

### Combining both

We expected that Mackinlay's work can be effectively used in our system since 2D visual parameters are a subset of the 3D set, and we feel that the perception of the user regarding information mapped in a 3D world will not substantially change. We should point out, however, that this is a conjectural evaluation as insofar it was not empirically tested.

In the case of visual parameters that exist only in 3D virtual worlds, we believe that for most of them their ability to display information can, with a certain degree of confidence, be inferred by the ability of the 2D parameters that are related to them. For instance, color saturation is not very useful for displaying quantitative data (Mackinlay, 1986); we feel that we can safely infer from this that transparency, as a related parameter to color saturation, also should not be used for mapping quantitative data in 3D. Nonetheless,

we think that usability studies with actual users are desirable to validate our classification.

The resulting ordering of visual parameters according to their effectiveness for encoding our three major data types (as described in Section 4.4.1) is shown in Table 4.5. It must be stressed that, in the absence of actual tests with users regarding visual perception of the different encodings, this ordering is a conjectural one and may eventually be further refined. It was based, as already said, mainly in Mackinlay's work (Mackinlay, 1986), but also on other related work (Senay and Ignatius, 1994) (Senay and Ignatius, 1996), and on what we could define as "common sense extrapolations" from previous work regarding 2D visual parameters to 3D visual parameters.

| **Quantitative** | **Ordinal** | **Nominal** |
|---|---|---|
| Label | Label | Label |
| Size | Transparency | Color Hue |
| Transparency | Color Saturation | Texture |
| Color Saturation | Color Hue | Shape |
| Color Hue | Texture | Transparency (–) |
| Texture | Size | Color Saturation (–) |
| Shape | Shape | Size (–) |

Table 4.5: CyberNet's ranking of visual parameters according to combined expressiveness and effectiveness. Parameters in gray are not effective for the corresponding type of data (binary data is a special case). Parameters marked with (–) defy the expressiveness criterion for nominal data because they may be ordinally perceived.

As is visible from Table 4.5, we have only ranked the visual parameters according to the main data types. For the subtypes of quantitative data, we have chosen a encoding order priority: we first encode scalars, then percentages, and finally binary data. This option is based in the fact that it is easier to encode binary data than percentages, and similarly it is easier to encode percentages than scalars. In other words, for binary data we only have to distinguish between two values, for instance – this means that even if we use a less efficient visual encoding, the possibility of mistaking the encoded data is diminished. Even shape, which appears at the end of the expressiveness ranking, can be used to distinguish between two different values, given that some legend is provided. Therefore, when there is an expressiveness choice to be made, we give priority to encoding more-difficult data.

It should be noted that in Table 4.5 there are two visual parameters,

previously referred in Section 4.7.4, missing. They are position and orientation, and they are not included in Table 4.5 because we have not considered position or orientation as parameters to encode actual entities data values. Position is used by the layout managers to place their children in space, thus encoding structural information. Orientation, another visual parameter reserved for the layout managers, is not used at present state for encoding information.

It might also be argued that there are additional encoding schemes that need to be taken into account, such as brilliance, granularity, bitmaps, or even sound and video. However, so far, the visual parameters ranked in Table 4.5 were able to meet our mapping need. Furthermore, much more complex encodings are not useful – humans are not cognitively well-suited to easily or intuitively understand them (Miller, 1956).

### 4.8.6 Adaptors

Adaptors are the non-graphical elements of the visualization layer and are responsible for the mapping. They implement the mapping rules and both types of mapping: structural mapping by defining the associations between service nodes $\rightleftharpoons$ layout managers and visual parameters mapping by visually encoding the data values with the visual parameters. For the visual-parameters mapping the adaptors also do some data filtering when needed – a number of data values can not be directly mapped and have to be processed (e.g., by applying a logarithmic function).

The adaptors are typed: an adaptor is created for each node of the service tree and may be reused for the same element type. For instance, in the case of dynamic data, if another service element of the same type appear, the adaptor for that service element is reused to provide the mapping of the newly created element. For each service, there is a metaphor mapping file that specifies the adaptors for each and every element.

## 4.9 Examples

We have already developed several examples of different metaphor mappings and different visual parameters mappings within the same metaphor. The choice of the metaphor is intimately related to the task the user wants to accomplish.

We begin by giving a detailed example of the NFS service and its visualization employing different structural and visual parameters mappings. Regarding different structural mappings, we use three different metaphors to

represent the same NFS service: a city metaphor, a solar system metaphor, and a conetree metaphor. We further describe and present different visual parameters mappings for the city metaphor. A more detailed explanation on the different mappings is further given in Chapter 5 in Section 5.5.1.

Regarding the other two metaphors, and in order to demonstrate the use of the same metaphor for different services, we present a network topology service and a workstation monitoring service visualized with a conetree metaphor and with a solar system metaphor, respectively.

Further explanation on the mappings and more examples of different metaphors for different tasks, i.e., different visualization tools, are given in Chapter 5.

### 4.9.1 Network File System service

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. This application allows for computer disks to be shared in a user transparent way. NFS servers export their file systems to other computers and NFS clients import file systems from servers. Using NFS a remote disk is viewed in the same way as a local disk.

Nonetheless, from a 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. For this purpose, we developed data-collecting agents that are able to get the NFS information from all over the Local Area Network (LAN) at Eurécom Institute (see Section 3.4.1).

The NFS service model designed by us is hierarchical as shown in Figure 4.3. The root service node of the hierarchy is the LAN backbone, the second level comprises the different workgroups of the institute (split according to subnets). Each subnet service node groups a set of computers, each computer service node features a set of local disks and a set of remote disks (accessed through NFS). The local disk may be shared using NFS. In that case, the computers that remotely access the local disk are added as children of the local disk service node. Each client computer service node also groups all the filehandles opened on that file system.

The NFS service model is composed of the previously described service nodes hierarchy. In addition to the service nodes, the hierarchy includes as leaf nodes the references to the entities retrieved from the repository. For

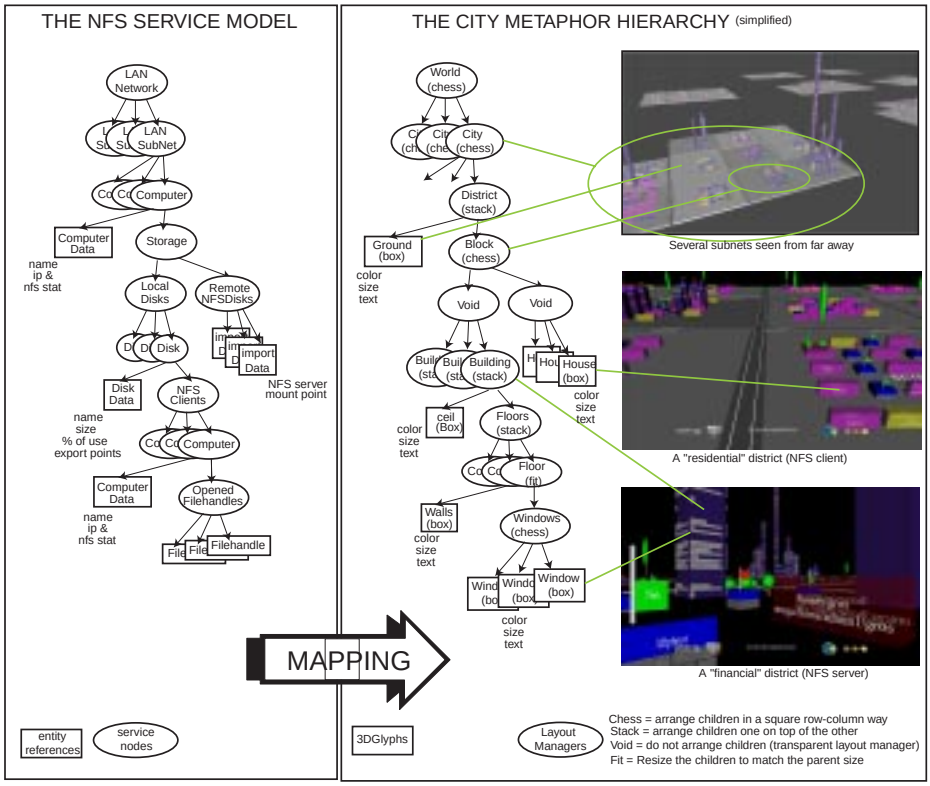



Figure 4.5: The NFS service model and its mapping onto the city metaphor.

instance, a computer service node has the following children: one reference to the actual entity that describes all the characteristics of the computer (as retrieved by the collecting agents) plus a storage service node. This storage service node has two children service nodes: one for managing local disks and one for managing remotely accessed disks. The former has as many children service nodes as actual local disks; the latter has leaf nodes that are direct references to the entities that represent each remotely mounted disk on that computer.

One interesting characteristic of this example is that each local disk service node has descendents that reference the computer entities that are its NFS clients. Thus, the computer entity may be referenced several times in the same hierarchy.

Figure 4.5 presents an example of the mapping between the NFS (Network File System) service and the city metaphor. On the leftmost side, there is the NFS service-model tree, and the graph on the right describes the hierarchy of the city metaphor. On the rightmost side, we can see some

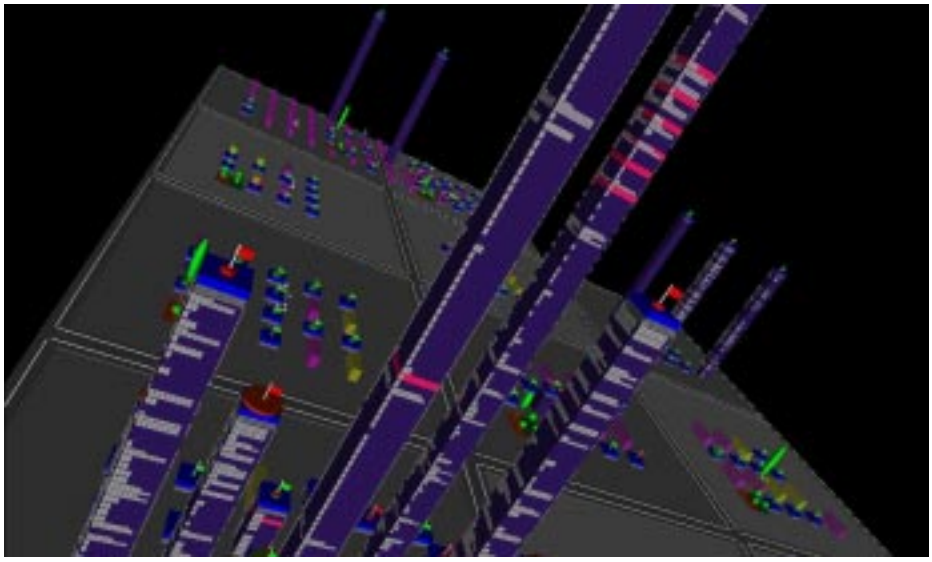


Figure 4.6: City metaphor for NFS data visualization:overview – CyberNet project.

screen-shots of the virtual world, based on the city metaphor, used to visualize the NFS service data.

As we can observe, it is very easy to distinguish between NFS clients, depicted as "residential districts" and NFS servers, depicted as "financial districts" (a district corresponds to a computer). Although they are not very visible, the icons (spheres and cubes) that are located in a bar at the bottom and appear superimposed on top of the visualizations, allow the user to change the structural (metaphor) and visual parameters mapping.

Further detail on the different structural and visual parameters mappings is given in the next sections.

### City metaphor

All the NFS information may be available in one virtual world (e.g., Figure 4.6). In the virtual world, there are several cities (one city per subnet). Each city comprises several districts or blocks (a district is a computer) and it is easy to identify servers (districts with buildings) and server loads (tall buildings). NFS clients are residential districts made of houses.

The city metaphor (see Figure 4.6, Figure 4.7, and Figure 4.8) is used thus to visualize the NFS service. Although all the visualizations presented use the same visual metaphor – a virtual city – each one uses different data


Figure 4.7: City metaphor for NFS data visualization: disks' size and partitions – CyberNet project.

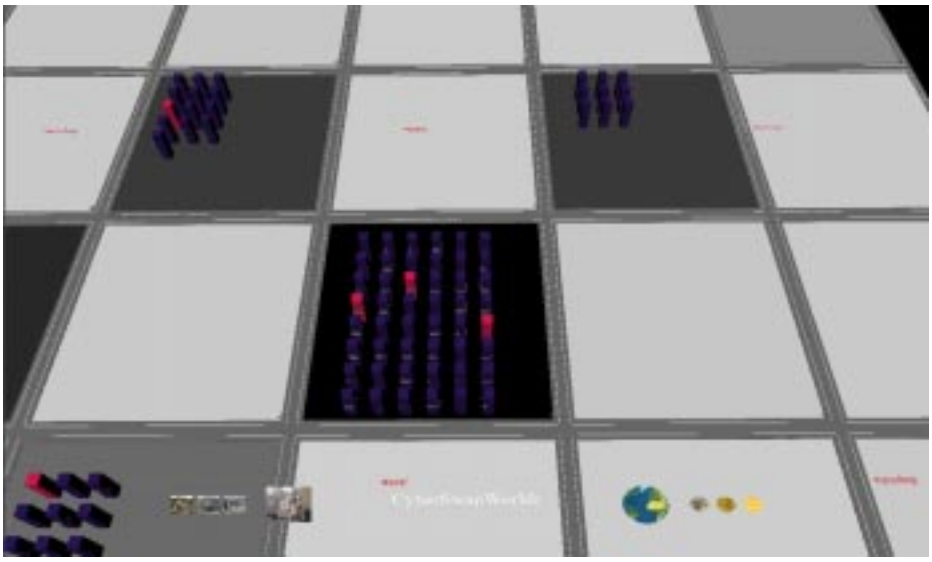


Figure 4.8: City metaphor for NFS data visualization: disks' status – CyberNet project.

encoding for the visual parameters. Since these visualizations encode a large number of data, we are not going to detail here all the individual data mappings. As an example, Table 4.6 depicts the mapping used for constructing the world in Figure 4.6.

| Data | Mapping |
|---|---|
| Computer | Block |
| Server/Client | Color of the block (dark/light) |
| Name of the Computer | Text on the block sides |
| Local disk | Building |
| Name of the local disk | Text on the roof |
| Size of the disk (relative) | Diameter of cylinder on the roof |
| Memory used (percentage) | Roof cylinder color and flag height |
| Megabytes left on the disk | Text on the flag |
| Client mounting the disk | Building floor |
| Name of the client | Label on the floor |
| Open filehandles | Windows on corresponding floor |
| Imported disk | House (semi-transparent) |
| Type of import (hard/automatic) | Color (yellow/pink) of the house |
| Belongs to the group (yes/no) | Antenna on top of the house (no) |
| Identification of the partition | Label on the side of the house |

Table 4.6: Mapping example (simplified) of the NFS service on the city metaphor.

However, we should refer that the different mappings foster the use of the representation in Figure 4.6 to get a general overview of the system; Figure 4.7 is more useful to view the size of the machines disks and which disks are partitions; Figure 4.8 is more valuable for an overview of the status of the physical disks and to know how many users are mounting a disk and how many file-handles they have opened. Further explanation on the different mappings is given in Section 5.5.1.

### Solar system metaphor

For the NFS service we have experimented other metaphors, besides the city metaphor. One experiment used a solar system metaphor to depict the NFS service. The solar system metaphor is also based on the real-world and conveys the notion of hierarchy through the use of different orbits.

Figure 4.9 shows a visualization of the NFS service using a solar system


Figure 4.9: Solar system metaphor for NFS data visualization – CyberNet project.

metaphor. Not all the information depicted in Figure 4.6 is available with the solar system metaphor, but the visual mapping is roughly the same (cf. Table 4.6), with planets taking the place of the buildings and orbits representing the blocks in the city.

#### Conetree metaphor

Figure 4.10 shows yet another visualization for the NFS service, this time using a conetree metaphor. Again, not all the information available for the NFS service is depicted, but this visualization is very useful to clearly perceive the service hierarchy.

### 4.9.2 Network topology service

For network topology analysis, the network topology is visualized with the help of a cone-tree metaphor (see Figure 4.11) where the user can easily grasp the entire network topology and see what elements are connected to where. Since the conetree metaphor is specially useful for depicting hierarchies (Robertson et al., 1991) and the network topology service is fundamentally hierarchical in nature, the conetree metaphor is perfectly suited for depicting the hierarchy existent between the network elements, while at the same time, showing information such as traffic and packet losses.

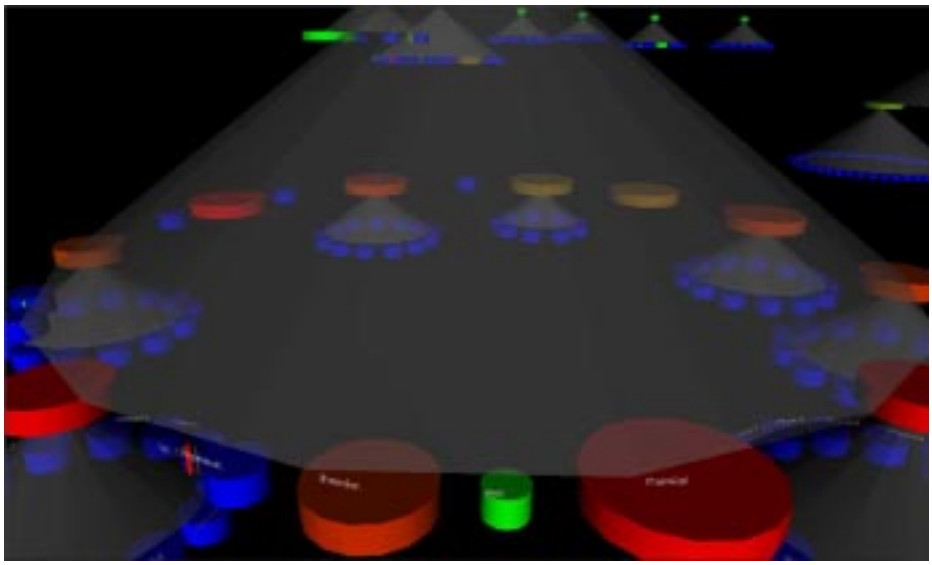


Figure 4.10: Conetree metaphor for NFS data visualization – CyberNet project.

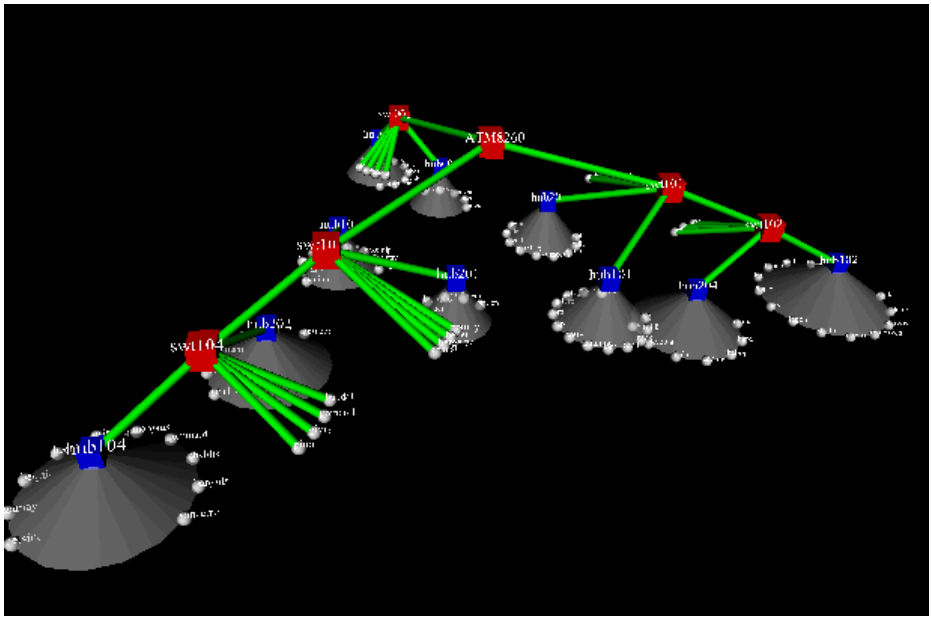



Figure 4.11: Conetree metaphor for topological information – CyberNet project.

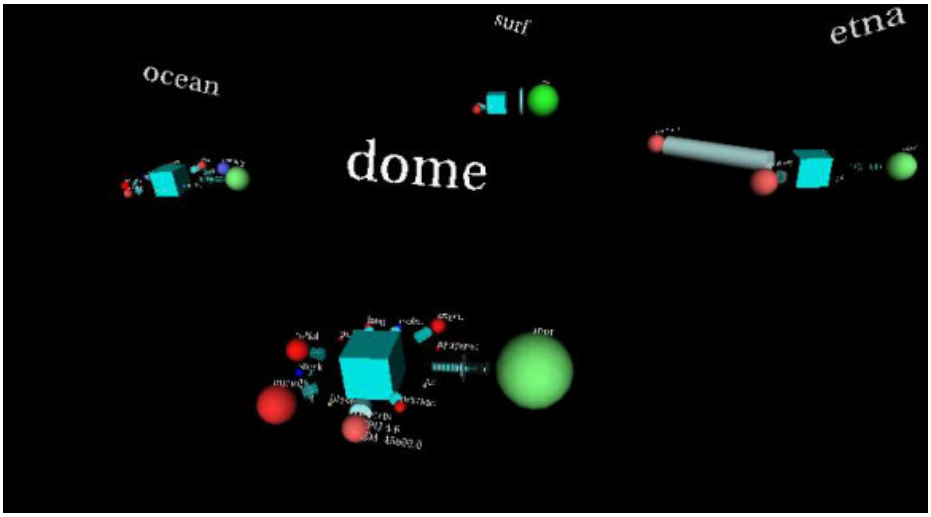



Figure 4.12: Solar system metaphor for workstation monitoring – CyberNet project.

In this visualization, switches are displayed as the red boxes, hubs as cones, with a blue box on top and the workstations that are connected to that hub presented as spheres at the base of the cone. Links between the switches and hubs are represented by cylinders, whose diameter and shade correspond, respectively, to the bit-rate loss of the link and to the packets loss for that link. The shade of the cone corresponds to packet loss for the respective hub.

### 4.9.3 Workstation monitoring service

Figure 4.12 represents another visualization using yet another visual metaphor based on a solar system. This tool is used for workstation supervision. This tool allows visualizing a workstation with all the users that are currently logged on it. Workstations are represented as planets (the cubes in the center) while the users are represented as satellites (the spheres orbiting around the center). Further mappings are the sphere's color that encodes the Unix group of the user, the size encodes the memory and the color saturation encodes the CPU time. Between the workstation and the users, the corresponding user processes are displayed as cylinders. The cylinder's visual parameters use the same encoding (i.e., size for memory and color for CPU).

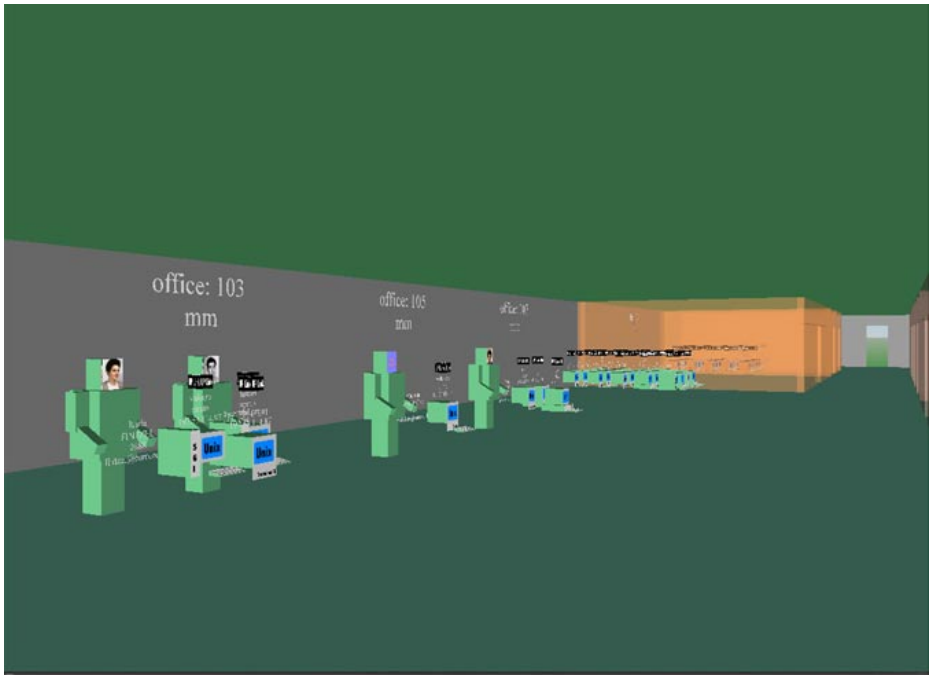



Figure 4.13: Building metaphor for geographical information – CyberNet project.

### 4.9.4 Staff and devices location service

Figure 4.13 shows an example view of the geographical tool that is used to visualize information about the Eurécom Institute building. This tool allows the user to get information on real world offices' location, staff, physical place of network static elements and so on. In this mapping, the color of the elements encodes the dept. to which the elements belong. Other mappings are straightforward as they follow the real world locations of the objects.

This visualization uses a metaphor based on the real world, a virtual building, since it is very easy for the user to relate to the actual building. Nonetheless, the virtual building does not faithfully reproduce the limitations of reality. The user can interact with the virtual building for different purposes: make the walls transparent to have a global view from the outside, choose to display only the offices belonging to a certain dept. and even redesigning the whole building by choosing to view one dept. per floor, among other possibilities.

### 4.9.5 Discussion

We have presented several different examples of both structural and visual parameters mappings that illustrate the capabilities of our mapping engine. Some additional comments are worth mentioning. The mapping engine fully complies with the expressiveness criterion. As we can observe from the visualizations regarding the NFS service, if we want to visualize the service in its entirety, only the city metaphor is capable of displaying all the information – thus obeying to the expressiveness criterion – as this metaphor possesses a large number of mappable visual parameters. The use of the solar-system metaphor and the conetree metaphor for visualizing the NFS service clearly implies that the information to be visualized is a subset of the initial NFS data.

Furthermore, we show how our mapping engine targets as successfully services with essentially abstract data (such as the NFS service) and services that contain information regarding entities with an actual physical representation (such as the staff and devices location service). Additionally, the mapping engine is capable of both employing metaphors that are inspired from the real world (such as the city metaphor) and abstract metaphors (such as the conetree).

Related work done so far in the field of automatic mapping of information, targeted mostly 2D presentations (Mackinlay, 1986) (Roth and Mattis, 1990) (Casner, 1991) (Senay and Ignatius, 1994). Besides dealing with 3D visual representations of data, our mapping engine uses metaphoric virtual worlds for the data representation which, as referred in Section 4.7.5, poses additional constraints to the information mapping.

Moreover, the automatic mapping engine we have designed deals with real-time visualization of dynamic data – contrary to prior work in the domain that only dealt with either static data (Mackinlay, 1986) (Casner, 1991) (Roth et al., 1994) (Goldstein et al., 1994) (Senay and Ignatius, 1994) or replayed animations of dynamic data (Zhou and Feiner, 1996) (Zhou and Feiner, 1998). Nonetheless, we believe there still is further work to be done in order to improve our automatic mapping engine, namely regarding extensive experimentation and validation of the effectiveness criterion.

## 4.10 Conclusion

We presented a framework and a strategy for automatically mapping abstract information onto metaphorical virtual worlds. Our mapping process distinguishes between two different types of mapping – structural and visual

parameters mapping – and uses information regarding the service (data set and structure), the user's tasks, and the visualization metaphors. The role that these components play in the mapping process was detailedly described.

A services characterization was presented, involving both generic characteristics and the characterization of data and relations. Our mapping process also takes into account the user's tasks, and consonantly we present a user's tasks classification to be used by our mapping engine.

The use of visual metaphors for information mapping was argued in some detail. We further presented a thorough metaphor characterization based on the metaphoric components – layout managers and 3D glyphs – mapping desiderata, and on constraints imposed by the structure and the visual aspect of the metaphor.

In the possession of the characterization of all the mapping players, we described our mapping engine. Our mapping engine attacks both types of mapping (structural and visual parameters) based on two criteria: those of expressiveness and effectiveness, although so far our work has mostly targeted the first criterion. The actual mappings are implemented by special elements that we have named adaptors.

The fact that we are able to do the visual mapping automatically allows us to change the mappings on the fly. This makes it easier to have different metaphoric representations of the same data, as we have shown in the examples given. The capacity of multiple mappings is further exploited in the next chapter to create multiple visualizations.

Our mapping engine is based on metadata that characterizes the actual data – thus making the mapping engine truly application-domain independent, since it makes use of a higher abstraction level. This independence from the data domain is in accordance with the underlying visualization framework, described in Chapter 3.

# Chapter 5

# Multiple Views

> Multiple images reveal repetition and change, pattern and surprise – the defining elements in the idea of *information*. Multiples amplify, intensify, and reinforce the meaning of images. (Tufte, 1997)

## 5.1 Introduction

In exploring large volumes of information, more often than not, one single visualization is not sufficient for grasping the whole data set. This may be due to different reasons: the screen real-estate is limited, the user's capacity of comprehending a large data set also has limits, different views or perspectives of the same data may be needed to grasp all the details, and so on.

Multiple visualizations allow the user to explore large amounts of complex information more easily and rapidly. We believe that one of the strengths of 3D metaphoric information visualization will emerge from the combined use of several interacting tools.

### 5.1.1 Motivation and problem statement

Challenges regarding how we visualize information are being posed not only by the sheer quantity of data and the limited space for visualizing it, but also by the diversity of tasks the user wants to perform. Besides, the user does not always have a specific task, such as a question that needs a concrete answer. Frequently, the task is more exploratory than targeted – in what is commonly designated as data exploration (Goldstein et al., 1994) – and may change as new data is apprehended. The user then needs different visualizations, eventually simultaneous, to accomplish different goals.

Additionally, and contrary to scientific visualization where the information has an intrinsic natural representation, in the case of abstract data there is no "correct" way to display the data that is valid for every application scenario.

Our motivation to develop multiple visualizations was a natural consequence of the fact that we wanted to experiment different metaphoric mappings for the same service (data set) and for different services. Furthermore, we wanted to, within the same hierarchical (metaphor) mapping, experiment different mappings for the actual data values – visual parameters mapping. (For further explanation regarding the different types of visual mappings please refer back to Chapter 4.)

The fact that we have designed and implemented a framework to construct metaphoric worlds in an automatic manner (see Chapter 3) provided us with a means for rapid prototyping and experimenting with multiple views.

### 5.1.2 Chapter organization

The remainder of this chapter is organized as follows: in Section 5.2 we present related work in the field of multiple views and in Section 5.3 we describe our approach. Section 5.4 is dedicated to a brief description of all the metaphors already implemented by. In the next section, Section 5.5, we demonstrate how we can use those different metaphors to create multiple views for several visualization tools, and we discuss our use of multiple views in view of prior related work. In Section 5.6 we tackle the subject of user interaction with 3D metaphoric worlds. Section 5.7 completes this chapter by drawing some conclusions.

## 5.2 Prior work

Related work in the field of multiple views has increased substantially in the last few years. This may be due to an increase in the amounts of information available to be visualized, and to the realization that one view is no longer sufficient. Also, the computing overhead need to implement and display multiple visualizations is less and less relevant as the computing power increases exponentially – especially for the graphics, where the improvements have really made a noticeable difference.

In a work dealing with information retrieval applied to provide tourist information to the ancient city of Nara in Japan – WING (Whole Interactive Nara Guide) – (Masui et al., 1995) presents a multiple-view system with four

different views. The four different views are called Map, Content, Category, and Index View.

The Map View displays geographical information and allows for operations such as rotate, move, and zoom. The Content View shows content information (e.g., pictures) for items close to the current center of the Map View. The Category View shows all the information arranged in semantic categories (e.g., "temple" is a subclass of "sightseeing"). Finally, the Index View allows for searching any item by a name, either complete or partial – its functional behavior is the same as the "grep" command in Unix – and consists of a permuted index. All the views are linked together so that a name chosen in the Index View is located and displayed in the Map view, for instance. The system integrates smoothly a visualization technique, a keyword-search technique, and a category-search technique.

(Roth et al., 1997) also presents an integrated information visualization system that allows for multiple views. It comprises an information-centric workspace (Visage), a tool for creating integrative visualizations for the workspace (SAGE), and a tool for dynamic user interaction (SDM – Selective Dynamic Manipulation). The combination of these three systems enable the user to communicate with the workspace in multiple complimentary ways.

More recently, (Roberts, 2000) distinguishes between multiple views and multiform visualizations: multiple views describe multiplicity in visualization (various realizations of the data depicted in separate windows); multiform visualizations describe a change in the visual representation method (the visualization is depicted in a different form). It also presents five groups of reasons to use multiple views, how to generate multiple views from different stages of the visualization data flow model, and a multiple views checklist to help developers. Throughout this chapter we will use the term multiple views to express multiplicity, even when there is a change in the visual representation method.

In a related work, (Roberts, 2001) analyses issues of data-flow and presentation in multiple views visualization. The issues analyzed regarding data flow are: choosing between fan-in and fan-out, when to replicate, how to control the view explosion, hierarchical exploration and information refinement, whether to push or to pull the data, and whether replication means more work for the user. The issues tackled regarding the structure of the presentation are the effective screen management and understanding window, parameter and session relationship.

(North and Schneiderman, 1997) introduces a taxonomy for multiple-window coordination. It proposes a generalized multiple-window coordina-

tion capability so that the user can create custom environments for information seeking. It uses two basic user actions, selecting items and navigating views, to arrange coordination in a 2x3 matrix, identifying three possible combinations for coordinations: selecting items $\longleftrightarrow$ selecting items, navigating views $\longleftrightarrow$ navigating views, and selecting items $\longleftrightarrow$ navigating views. The putative applications are demonstrated with mockups.

The Snap-Together Visualization, presented in (North and Shneiderman, 2000), provides a user interface for users to "snap" visualizations together. It enables users to specify their own coordinations between different visualizations in order to construct custom data-exploring interfaces. Snap does not require any programming per se, since it interconnects visualizations tools developed by other researchers in the field, and is flexible in data, views, and coordinations. This allows for constructing coordinated browsers to rapidly explore and navigate data and relationships.

After ascertaining that, although multiple views systems are common and helpful, little specific guidance was available for the persons designing them, Baldonado and Kuchinsky (Baldonado et al., 2000) established a set of guidelines to help designers. These guidelines are divided in two sets: *when* to use multiple views and *how* to use multiple views. For each guideline, a justification is given, and the pros and cons of using it are assessed. We have used this guidelines to validate our use of multiple views.

## 5.3 Our approach

All the related work cited above (e.g., (Roth et al., 1997) (North and Shneiderman, 2000) (Roberts, 2000)), as well as some other earlier work not described here (e.g., (Perlin and Fox, 1993) (Shneiderman, 1994) (Mukherjea et al., 1995)), may use different approaches to multiple views, but all agree on one thing: multiple views can help making sense of information – the ultimate goal of information visualization. However, multiple views must each bring some added value. They must shed new light on the representation, in order to be useful.

According to Baldonado's et al. work on guidelines for using multiple views (Baldonado et al., 2000), there are four rules to help deciding when multiple views should be used:

- the *Rule of Diversity* states that multiple views should be used when there is diversity of attributes, models, user profiles, levels of abstraction, or genres.

- the *Rule of Complementarity* states that multiple views should be used when different views bring out correlations and/or disparities.
- the *Rule of Decomposition* states that complex data should be partitioned into multiple views to create manageable chunks and to provide insight into the interaction among different dimensions.
- the *Rule of Parsimony* states that multiple views should be used minimally.

Our tools comply with the guidelines above. For instance:

- we have created multiple views when the target audience is diverse, as is the case for the web server analysis tool (Section 5.5.6), or when the attributes changes, as is the case of the `top` command visualization according to different parameters (Section 5.5.7). This is in accordance with the *diversity* rule.
- we have divided a large volume of data, in order to concentrate on a smaller set, so that the data that interests most the user is made to stand out clearly (NFS service visualization tool, Section 5.5.1). This complies with the rule of *decomposition*. We also provide a global overview of all the data, so that the user is able to get the whole picture – we find that having a general overview is always useful to acquire context knowledge.

These are only two illustrative examples (more examples can be provided for the other rules of complementarity and parsimony) that stress our approach to multiple views. In general, they are created when the data changes, the target audience changes, the data under the spotlight changes, or the change in the visual representation brings out a detail that otherwise went unseen.

We give numerous examples of multiple views in the next sections, and try to always refer the added value of using yet another view. First, though, we briefly describe the metaphors used in the construction of the different views.

## 5.4 Metaphors

Our motivation to use metaphoric worlds to visualize mostly abstract information has already been discussed in Section 4.7. We have developed a

substantial number of visual metaphors for our system, several of which have already been introduced in previous chapters.

Some of the metaphors we have designed are almost entirely real-world based (e.g., the building metaphor, the city metaphor) and some are more abstract from a conceptual point of view (e.g., the conetree metaphor, the pyramid metaphor). The next sections contain a brief description of all the metaphors that have so far been developed in the context of the CyberNet system (Chapter 3).

### 5.4.1 City metaphor

In the city metaphor, information is visualized using the structure of a real-world city. In our implementation there are districts, residential blocks with low height houses and also financial blocks with tall office-resembling buildings. There are also roads and trees.

The metaphor is quite easy to grasp as the hierarchical relations are evident from the real world counterpart: cities contain districts, that contain blocks, that contain houses and buildings, and so on. The visual effect is quite impressive and provides lots of different elements and visual parameters to map information on. On the other hand, the city metaphor is not recursive – thus, it is not a valuable option for visualizing recursive services.

### 5.4.2 Conetree metaphor

The conetree metaphor is fairly omnipresent in information visualization reference bibliography (Spence, 2001) (Card et al., 1999) (Chen, 1999). As the name indicates, it uses a tree of cones to depict information.

This metaphor is generally associated with displaying hierarchical information due to the immediately visible hierarchy appearance. It is a recursive metaphor so it is able to map recursive data. However, the number of different visual parameters available for mapping information is quite limited. We feel that for large volumes of data, its performance degrades more rapidly than for the pyramid metaphor.

### 5.4.3 Pyramid metaphor

Like the conetree, the pyramid metaphor is also recursive. It uses the concept of nested pyramids to display information. It is also usually associated with the visualization of large hierarchies of information.

Conceptually, the pyramid metaphor is similar to the conetree metaphor, using a hierarchy of pyramids instead of cones. Also the number of different

elements and visual parameters it provides for information mapping is quite similar to the conetree metaphor. However, we feel that for very large hierarchies, the pyramid metaphor has better performance than the conetree metaphor: the visual clutter is reduced in the latter case. This might be explained by the bottom-up construction of the pyramid metaphor that allows for an uncluttered view from above, in opposition to the conetree metaphor that uses a top-down construction.

### 5.4.4 Rooms metaphor

In this metaphor, information is displayed with the aid of interconnected rooms. The rooms themselves and their position encode structural information. The space inside the rooms is also used to display information, namely containment relations, as well as the walls that are used as yet another virtual screen.

For the time being, our implementation of the rooms metaphor is still quite naïve, though. We can easily foresee extended features to add to the current implementation. For instance, the "paintings" on the wall that we use to display additional information, could also be used as "magical features" (Dieberger and Frank, 1998) – anchors that would automatically take the user to other parts of the metaphor or to another view.

### 5.4.5 Solar-system metaphor

The solar-system metaphor, in our current implementation, is fairly simple. It uses suns, planets, and satellites to map information. The structure is reflected by the orbits of the various elements.

The hierarchical organization is provided by the different orbital relationships: planets are attached to a sun and satellites are attached to a planet. The metaphor allows for the representation of a large volume of information since there are no restrictions (except for the orbits) regarding the position of the elements in the virtual space – contrary, for instance, to the city metaphor, where the city elements are placed on a horizontal plane. On the other hand, with a relevant number of elements the metaphor quickly tends to appear as lacking structure.

### 5.4.6 Building metaphor

The building metaphor is a special case in our system. Contrary to the other metaphors, it was developed and implemented for a specific service: show network and staff information for the Eurécom institute.

The building metaphor hence represents a virtual building for Eurécom, with a structure that somewhat follows the actual physical structure: there is the same number of floors, offices, labs, stairs, and so on. In the default visualization, the rooms' locations are the same as for the real-world counterpart – however the user may interact with the visualization and change this (Section 5.6.3).

### 5.4.7 Landscape metaphor

The landscape metaphor uses a information landscape to visualize data. The information is placed in a virtual landscape, usually using the shape of a vertical bar or 3D spike. It was inspired by the File System Navigator (Tesler and Strasnick, 1992).

Although there is basically only one element (the vertical bar) to map information on, this metaphor is interesting because position can be effectively used to encode information – as an illustrative example see the network traffic visualization in Figure 5.8.

### 5.4.8 Library metaphor

Finally, we have also implemented a metaphor inspired by the usual elements one finds in libraries: thematic corridors, bookshelves, shelves, and books. The hierarchical structure is provided by the fact that corridors contain bookshelves that contain shelves and shelves contain books.

The number of hierarchical levels is somewhat reduced if limited to the hierarchy corridor $\rightarrow$ bookshelf $\rightarrow$ shelves $\rightarrow$ books. Nevertheless we can make books act as an entrance for a further hierarchy, in a manner similar to opening a door to another library (or another part of the same library).

## 5.5 Tools

In the next section we will describe several visualization tools designed and implement in the context of the CyberNet project. We will also describe some of the different views available for the user to inspect data from different perspectives.

### 5.5.1 NFS service visualization

We have developed a visualization tool to represent information regarding the Network File System service (NFS service). The NFS service has already been introduced in Chapter 3 and was described to some extent in

Section 4.9.1. For further information regarding the data and the mappings, please refer to the latter.

This service is characterized by an enormous amount of data. We have developed several views for the NFS service. One to visualize all the data set and others views that only display a subset of the NFS data. These other views, representing a reduced amount of data, allow for the emphasis to be put on the relevant data under scrutiny for that particular visualization. Furthermore, the visualization of smaller subsets reduces the risk of overwhelming the user with a large volume of complex data, some of it not relevant for the particular task in question. This is a clear application of the decomposition rule – a guideline for multiple views applications (Baldonado et al., 2000).

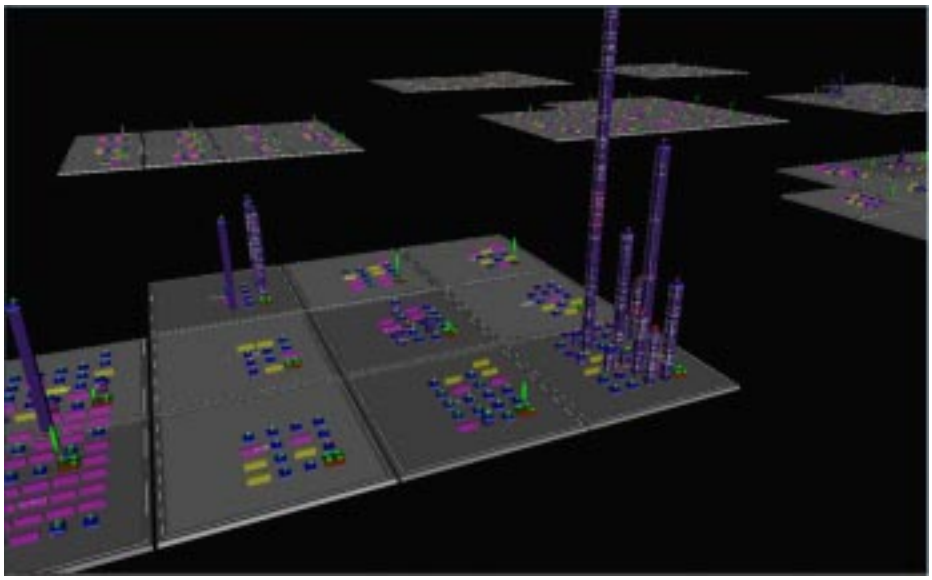
For the NFS service visualization tool we have used primarily the city metaphor. This metaphor revealed itself an appropriate choice due to the large number of mapping parameters available to map the NFS data, which is also very large. Nonetheless, we have also implemented other visualization using two other metaphors: a conetree and a solar-system metaphor. These views, though less useful than the city metaphor, especially when we want to visualize the entire data set, allow for bringing important details under the spotlight.

In the next sections we present some of the multiple views available for the NFS service visualization, using the city metaphor. We also present examples of views using the conetree metaphor and the solar-system metaphor for visualizing NFS data. The different views and metaphors may be selected from a menu at the bottom of the visualizations – the views are chosen from the buttons on the left and the metaphors are changed with the buttons displayed on the right.

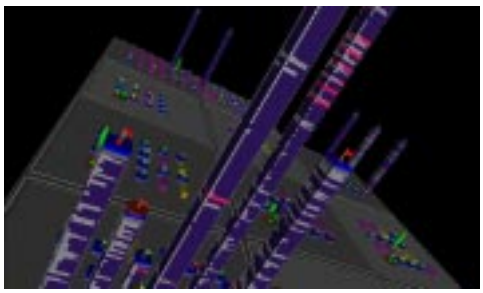
### Overview

Figure 5.1 shows several views for the whole data set describing the NFS service. The mapping has already been described in Table 4.6. The user can have a global overview of the entire data set.

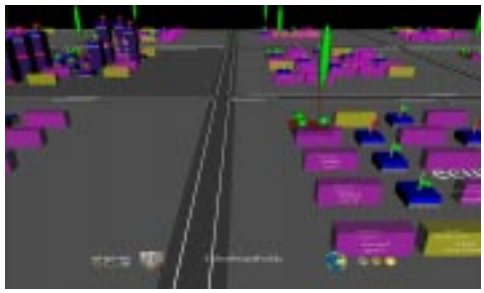
This general overview allows, nonetheless, for immediate knowledge regarding the NFS service. For instance, the user can easily identify servers and clients. A computer is mapped on a district and each disk is represented by a building; additionally, each client mounting that disk is a floor on the building. On the other hand, each imported disk is mapped on a house. In this way, "financial" districts, i.e., districts with tall buildings, identify unambiguously servers. Mutatis mutandis, "residential" districts, i.e., dis-



(a) Overview



(b) Zoom on a financial district



(c) Zoom on a residential district

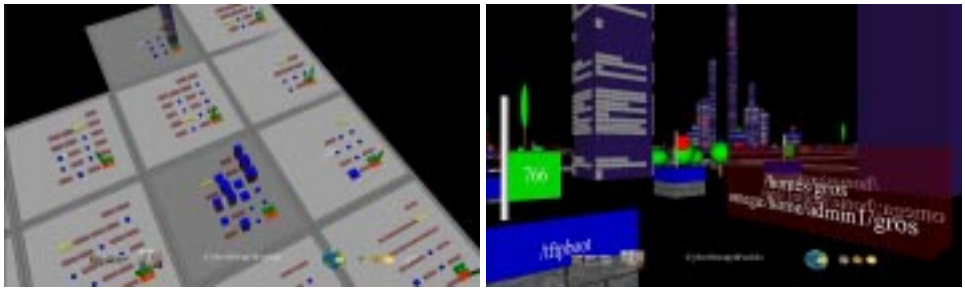
Figure 5.1: NFS service visualization – global view of all data.

tricts with low houses, clearly identify the clients. This is a powerful visual perception immediately evident upon first inspection.

### Disk size

Figure 5.2 depicts two views of information regarding the NFS service. The data represented in a subset of the data represented in Figure 5.1. The emphasis is put on the size of the disks and on identifying disks that are system's partitions.

In view of the foregoing, in Figure 5.2 the size of the disks is mapped



(a) Overview (b) Zoom on a computer

Figure 5.2: NFS service visualization with emphasis on information regarding disks' size and system's partitions.

to the height of the buildings. For the houses representing the imported disks, the color hue maps whether the disk is a system disk or not. The user can thus easily identify large disks and which of their imported disks are non-system disks.

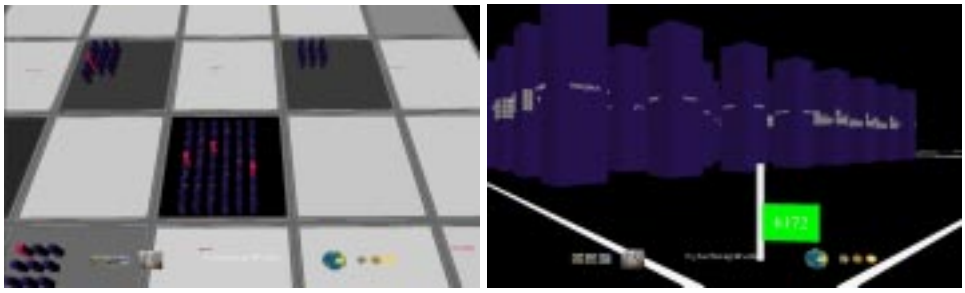
### Disk status

The data displayed is a subset of the NFS set. The focus is on disk status: number of users mounting the disk and number of open filehandles they have. The information regarding the users and the filehandles is immediately perceived as is, uncluttered by other non-relevant data. To achieve this, each disk is represented by a square, and each client mounting that disk is represented by a building (Figure 5.3). The number of open filehandles is mapped on the number of windows present in each building.

### Using other metaphors

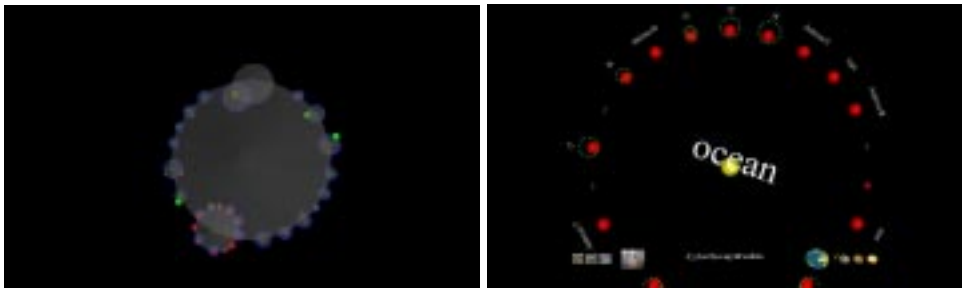
As referred before, we have also experimented different metaphor mappings for the NFS service data. These views primarily had an experimental nature, in order to validate our mapping process, and the city metaphor proved to be the most efficient one as expected. Nonetheless, we feel that there are scenarios where these views implemented with different metaphors may provide valuable insight.

Figure 5.4 (a) depicts a visualization of the NFS service hierarchy using a conetree metaphor. Figure 5.4 (b) puts the focus on the visualization of a machine and its disks, using a solar-system metaphor.



(a) Overview (b) Zoom on a disk

Figure 5.3: NFS service visualization with emphasis on information regarding users and open filehandles.



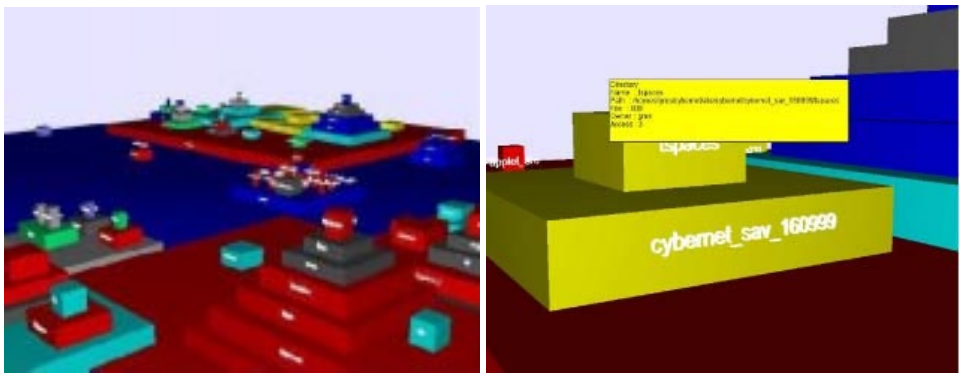

(a) Overview with conetree (b) Zoom on a computer in a solar-system metaphor

Figure 5.4: NFS service visualization – using other metaphors.

### 5.5.2 File system visualization

In order to visualize the contents of large file systems, a file systems visualization tool has been developed. The visual metaphor that revealed itself more adequate to display large volumes of hierarchical information, as is the case with file systems, is the pyramid metaphor.

Multiple views have been developed to be able to exploit (visualize) the file system according to different parameters. Essentially, the file system's structure is always encoded in the same manner – hierarchically with nested pyramids – and the data under scrutiny is color coded (color hue). In the multiple views, the color hue can thus encode the file type, the file owner, the file date, or the file security details. Figure 5.5 shown an example of color hue encoding file type.




(a) Overview

(b) Zoom on a directory and details-on-demand

Figure 5.5: File systems visualization – color hue maps the file type.

We have also done other experimental mappings with different metaphors for the file systems visualization tool. Figure 5.6 shows an example for the visualization with the focus on the files' owners. In Figure 5.6 (a) we can see an example using the cone tree metaphor, and in Figure 5.6 (b) an example using the rooms metaphor. On both, the files' owner is mapped using a texture with the photo.

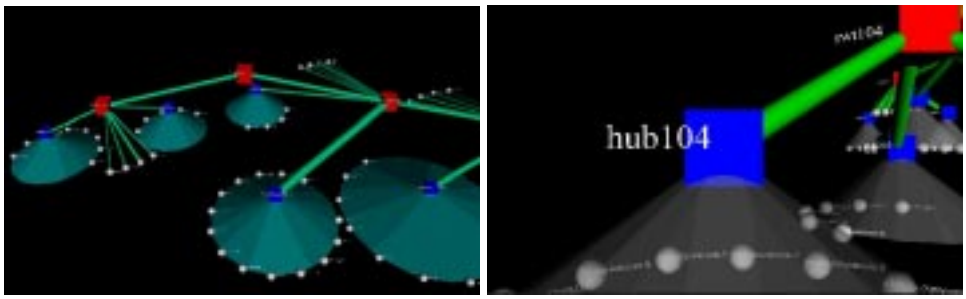
(a) Overview

(b) Zoom on a directory and details-on-demand

Figure 5.6: File systems visualization – texture (photo) maps the file owner.

### 5.5.3 Network topology visualization

We have developed a network topology visualization tool for depicting the topology of a computers' network – we have used Eurécom's intranet as an example. Due to the strong hierarchical characteristic of the network topology service and the fact that the number of different entities to be visualized was relatively small, the conetree metaphor was a manifest choice. Figure 5.7 depicts the network topology visualization tool.




(a) Overview (b) Zoom on a hub

Figure 5.7: Network topology visualization.

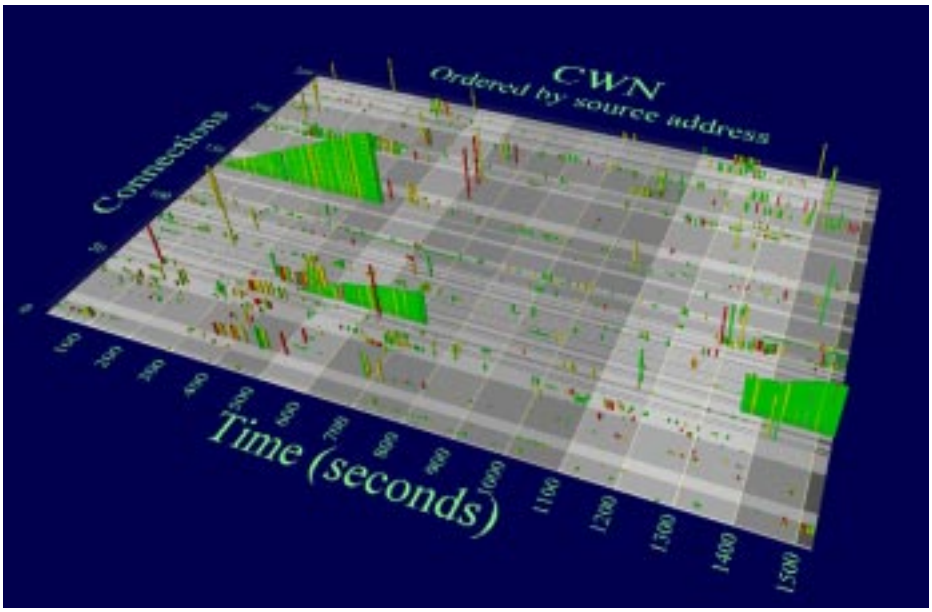
A conetree metaphor is thus used to display the network topology and some additional information regarding the network's performance. The hubs are cones with a blue box at the top and the switches are visualized as red boxes. The machines connected to a given hub are represented as spheres placed at the base of the corresponding cone.

The connections between the different elements – switches, hubs, and machines – are depicted as cylinders. Additional data is mapped on the cylinders: the size corresponds to the transmission bit-rate of the connection and the color saturation to the packets' loss rate. The hub's rate of lost packets is mapped on the color saturation of the cone.

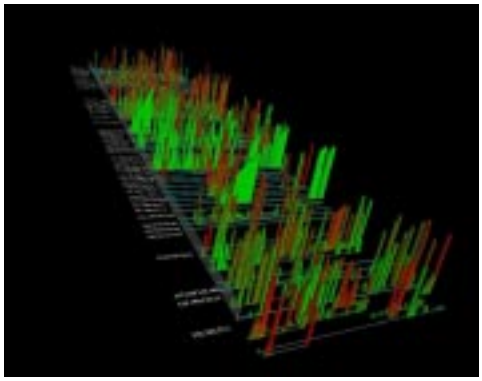
### 5.5.4 Network traffic visualization

A tool for visualizing network traffic data was also developed in the context of the CyberNet project. The network traffic data is captured at a specific point of the network using a sniffer for planning purposes. The result is a very large set of network packets. The packets are automatically analyzed in order to classify them according to the source/destination addresses and ports, and specific network data values – such as congestion window (cwn) size – are displayed.

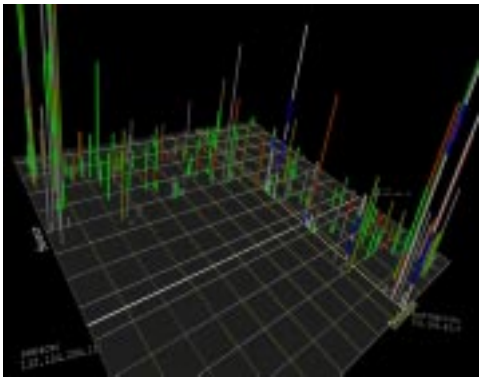
The metaphor used to visualize the network traffic data is an information landscape (Figure 5.8). The 3D landscape representation is a great help to the user for quickly identifying connections of interest. However, the traffic information displayed is essentially qualitative information. In order to have a quantitative estimate, the connections will have to be further evaluated using other network traffic analysis methods.




(a) Global overview



(b) Time overview



(c) Connections overview

Figure 5.8: Network traffic visualization tool.

Regarding the mapping, color hue is used as a quality criteria: green ≡ good, yellow ≡ average, and red ≡ bad. Other relevant information is displayed on the ground of the information landscape – such as time or the IP addresses – according to the respective view, as we show next.

#### Global overview

The overview gives a general idea of traffic patterns in the network and points out potential problems. The data is displayed qualitatively regarding the number of connections, the congestion window size (cwn), and the time. Figure 5.8 (a) depicts the global overview.

#### Time overview

A second view, called the time overview is proposed to represent the temporal evolution. Here, the connections are ordered according to the source IP address on the `Z` axis and the time increases along the `X` axis. An example of the time overview is represented in Figure 5.8 (b).

#### Connections overview

The connections overview is organized in the form of a matrix, with source IP addresses on the `Z` axis and destination IP addresses on the `X` axis. At a source/destination pair intersection, the connections between the two addresses are represented by cylinders. Selecting a particular connection (cylinder) displays the source and destination IP addresses on the corresponding sides of the matrix. Figure 5.8 (c) shows a representation of the source/destination connections overview.

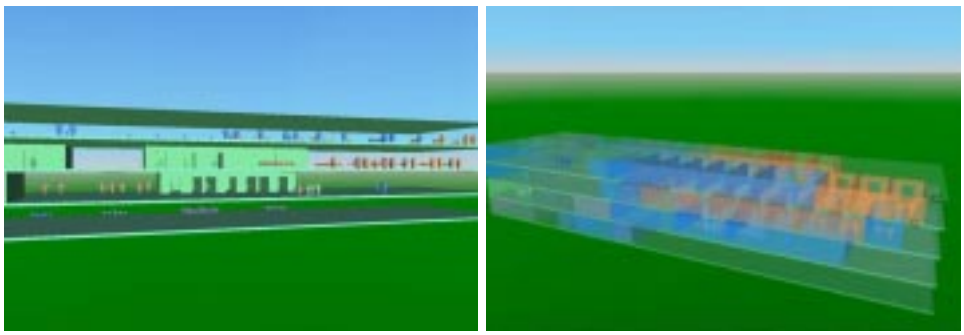
### 5.5.5 Staff and devices visualization

The goal of this tool was to build a symbolic 3D model of the Eurécom building that could be exploited to acquire knowledge regarding the staff and devices. The staff and devices information is represented according to their real-world locations. For this, the default representation of the virtual building shows the offices and other infrastructure with the relative locations they have in the physical building.

The mapping is quite simple: staff and devices are represented in a similar manner to their real-world counterparts, respectively by avatars and computing devices' facsimiles. Color encodes the department which the element – staff, device, room – belongs to.

By interacting with the world, the user is able to create his own views. There is one main interaction feature: selection. The user may select to visualize only the objects he wishes so. This selection traverses all the objects and all the different combinations are possible (e.g., show the rooms of type office *and* that belong to the multimedia dept. or show all the users *and* all the floors *and* the exterior walls). This is a powerful mechanism for creating typed user-defined views (e.g., an office overview, or a staff overview).

It is worth noting that all the objects that are not part of the user's selection are rendered transparent or semi-transparent. The latter allows for a selective display of relevant information and for still keeping the global context in a non-invasive way. Figures 5.9, 5.10, and 5.11 depict some multiple views for the staff and devices visualization tool. Figure 5.9 shows a building overview, Figure 5.10 is more interesting to staff visualization, and Figure 5.11 depicts essentially office information.



(a) Building with transparent walls (b) Semi-transparent offices and floors

Figure 5.9: Staff and devices visualization – building overview.

### 5.5.6 Web server data visualization

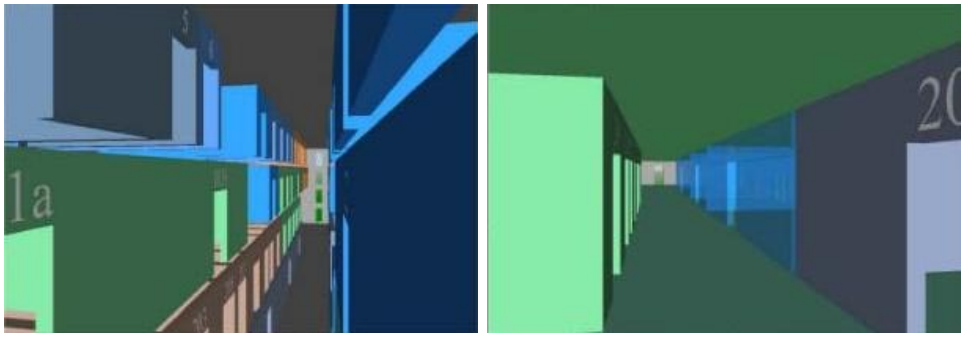
The analysis of Web server logs is a hot topic nowadays because of the e-business explosion. Our objective was to design a Web server logs visualization tool that uses 3D technology to present the information to the user. The information is displayed in different 3D metaphoric worlds, customized according to the needs (e.g., temporal, geographic representation), or the target audience (e.g., web master, sales personnel).

The implementation is fairly simple at present. There are three data types that we visualize: the number of hits, the traffic generated, and the number of hosts. This information is represented in three different views:

(a) Staff information with transparent offices and floors

(b) and zoom in a floor with semi-transparent offices

Figure 5.10: Staff and devices visualization – staff overview.




(a) Office information with transparent floors

(b) and semi-transparent (blue) offices belonging to a dept.

Figure 5.11: Staff and devices visualization – offices overview.

geographic view, site view, and temporal view. Both the data and the view are selected from a selector interface presented in the next section.

### World and data selector

The interface depicted in Figure 5.12 allows the user to select the view type – geographic, temporal, or site view – and the type of data – number of hits, traffic, or number of hosts. The choices are unambiguously displayed by translucent spheres.

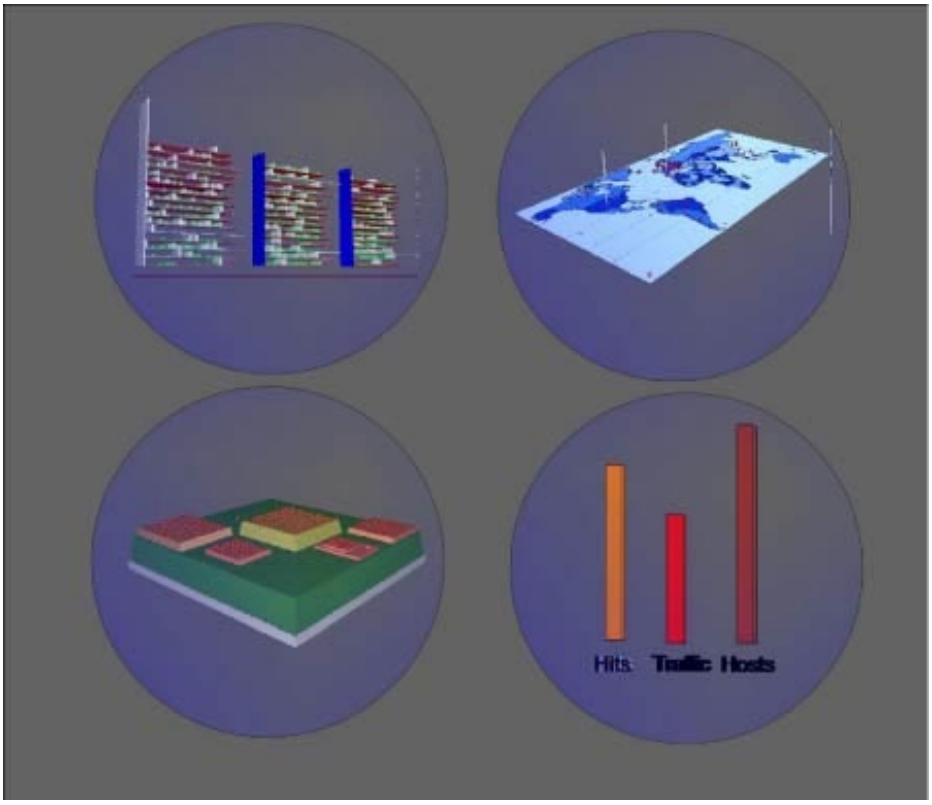


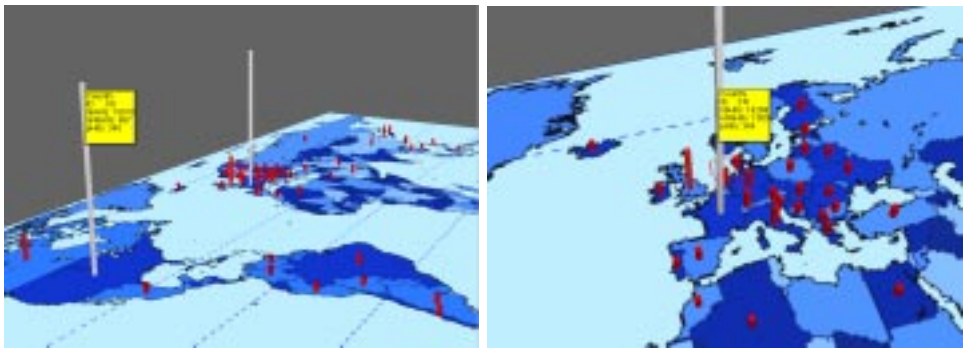

Figure 5.12: View selector for the web server logs visualization tool.

### Geographic view

The geographical diagram allows the user to situate and compare data from a geographic perspective. The metaphor used is an information landscape with a world map top encode position. Details-on-demand are available by selection. Figure 5.13 depicts an example of the geographic view.

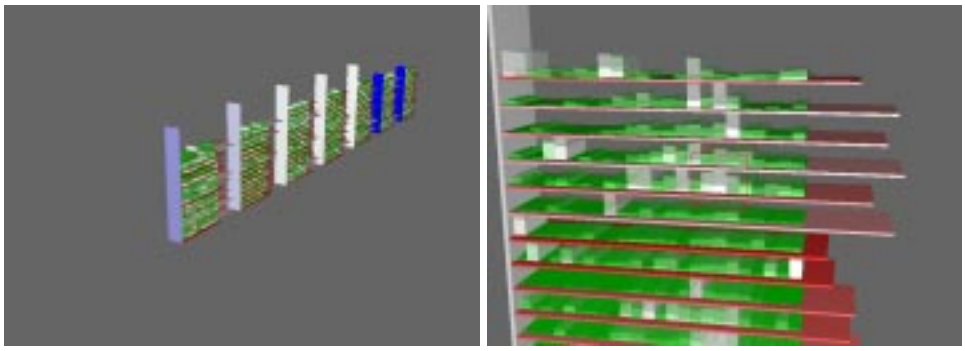
### Temporal view

The temporal view allows the user to compare data in a temporal manner. The data is represented in a library containing several bookshelves; each bookshelf is subdivided into shelves upholding series of books. There are three levels of temporal segmentation: bookshelves, shelves and books – where data is sorted respectively by week, by day, and by hour (Figure 5.14).



(a) World view (b) Zoom in Europe

Figure 5.13: Web server logs visualization tool – geographic view.



(a) Week overview (b) and day overview

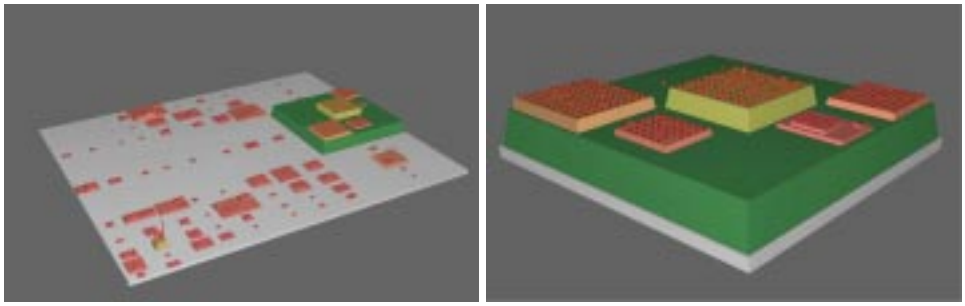
Figure 5.14: Web server logs visualization tool – temporal view.

#### Site view

The site view is a representation of the actual hierarchy of the website – directories and pages. This view allows for visualizing the most popular pages, for instance, or the files that are more frequently downloaded. Figure 5.15 shows an example of the site view using an information-pyramids metaphor.

### 5.5.7 Workstation data visualization

In the context of the CyberNet project we have also developed a group of visualization tools to visualize workstation information. We have used different metaphors and different views according to the most relevant data value.



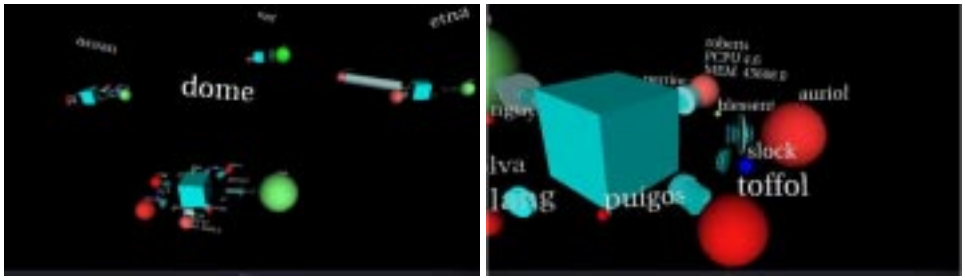
(a) Web site hierarchy (b) Focus on a particular directory

Figure 5.15: Web server logs visualization tool – hierarchical view.

A number of this visualizations were inspired by the Unix `top` command – multiple views were created to cater for different information priorities.

### Global view

Figure 5.16 shows information regarding workstations. The metaphor used is a solar system metaphor. Workstations are depicted as stars (cubes) with planets, representing users, orbiting around. The users are mapped on spheres – color hue corresponds to the Unix group, size to memory usage, and color saturation to CPU time. Between the workstation and the users, satellites represent user processes and are mapped as cylinders. The mapping on the cylinders visual parameters is coherent with that of the spheres: size ≡ memory and color saturation ≡ CPU.



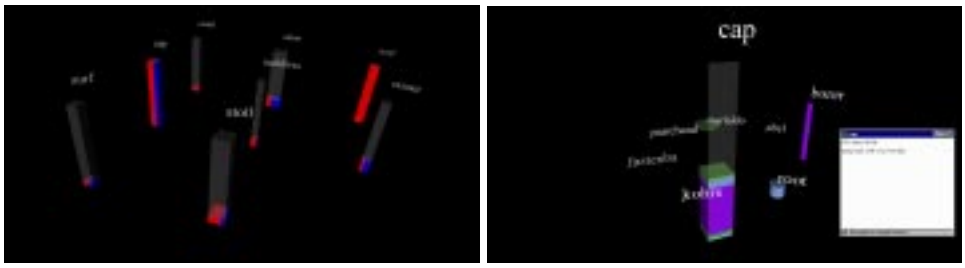

(a) Cluster view (b) Zoom in one machine

Figure 5.16: Workstation monitoring – global view.

### Synthetic view

Regarding workstation monitoring, we have developed another visualization tool where quantitative and qualitative information concerning the CPU load and the swap memory used are easily perceived. There is a synthetic view, shown in Figure 5.17 (a), where each workstation is depicted as a transparent vertical bar, with an indicator inside that maps CPU load and swap memory usage.

The user may then ask for details regarding a specific machine by selecting it. Figure 5.17 (b) shows the details for the workstation named *cap*. The information regarding the users logged on that machine are depicted as cylinders around the machine (height ≡ CPU usage and radius ≡ memory usage) and quantitative details are given in a small 2D window.



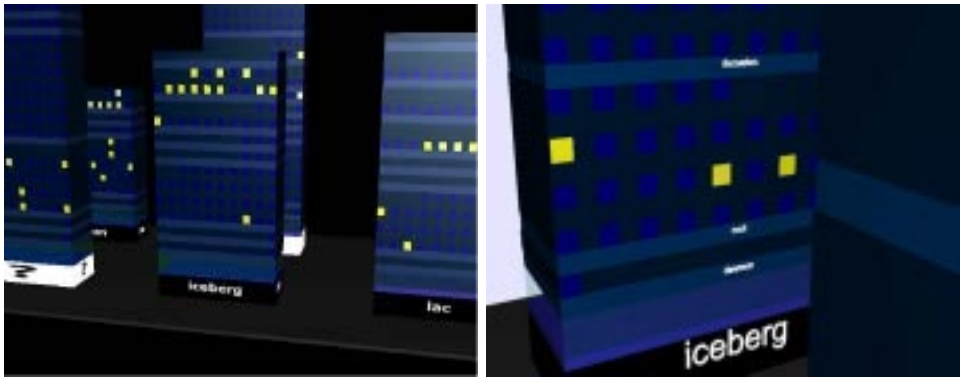

(a) Cluster view (b) and machine view

Figure 5.17: Workstation monitoring – synthetic view.

Still regarding workstation data visualization, yet another tool was developed in the CyberNet project's context. This tool uses different metaphoric representations for displaying data that may be obtained with the Unix `top` command for a workstation (e.g., users logged on, number of processes, and percentage of CPU used). The multiple representations are dependent on what kind of data is given priority – users, process, or machines.

### `top` command view

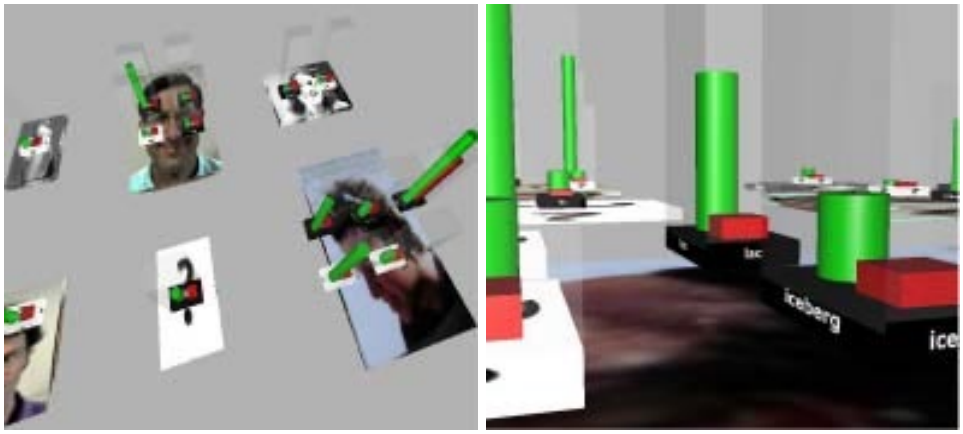
Figure 5.18 displays workstation monitoring data, arranged by workstation. The metaphor used is a simplistic city metaphor, where each building represents a workstation. Each user logged on that machine is represented by one or more floors in the building – his processes running on that machine are represented by windows (lighted windows represent user processes with a high CPU consumption).




(a) Cluster view (b) Zoom in one machine

Figure 5.18: Workstation monitoring – `top` command results organized by workstation view.

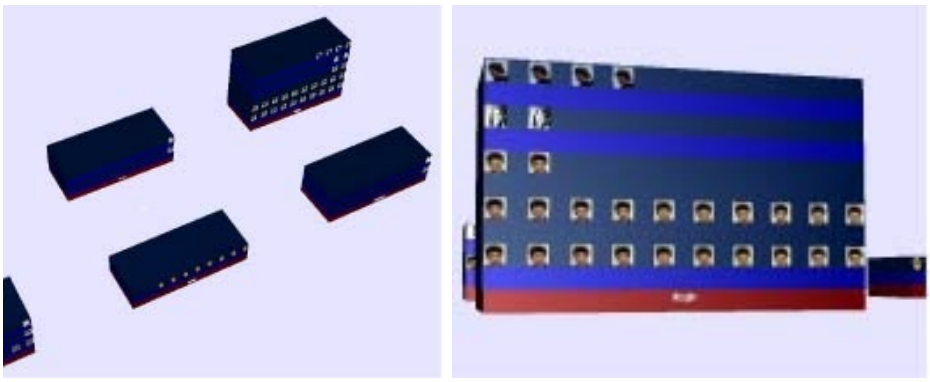
Figure 5.19 displays also information regarding workstation monitoring, with the emphasis put on the user. The main idea is to see who are the users consuming more resources. The metaphor used is an information landscape and the user's identity is mapped on the ground with a photo. Further information, such as the machines a specific user is logged on, plus CPU and memory usage, is displayed on top of the user's identity.




(a) Cluster view (b) Zoom in one user

Figure 5.19: Workstation monitoring – `top` command results organized by user view.

In the representation shown in Figure 5.20 the emphasis is put on the processes currently being executed in the network (independently of the machine) and by whom. The metaphor used is again a quite primitive city metaphor, like the one used for the `top` by workstation visualization. The processes are displayed in buildings representing command types and each user process is represented by a photo of the corresponding user. This visualization, though somewhat naïve, may be used for programs that have a limited number of licenses, thus allowing for immediately locating the users currently running it.



(a) Cluster view (b) Zoom in one machine

Figure 5.20: Workstation monitoring – `top` command results organized by process view.

Finally, in Figure 5.21 we show how these different views may be combined. In the example, two visualizations are combined: `top` by user and `top` by workstation. The illustrative mockup example shows how we can select a user in the `top` by user view, and have information regarding that same user highlighted in the `top` by workstation view.

### 5.5.8 Discussion

We have presented a number of different visualization tools for different application domains that explore the concept of multiple views. These multiple views may imply different structural mappings, or different visual-parameters mappings keeping the same underlying structure – i.e., metaphor. Note that we use the term multiple views even when there is a change in the visual metaphor used. This is in contrast to other related work (Roberts, 2000) that use the designation of multiform visualizations for that case.

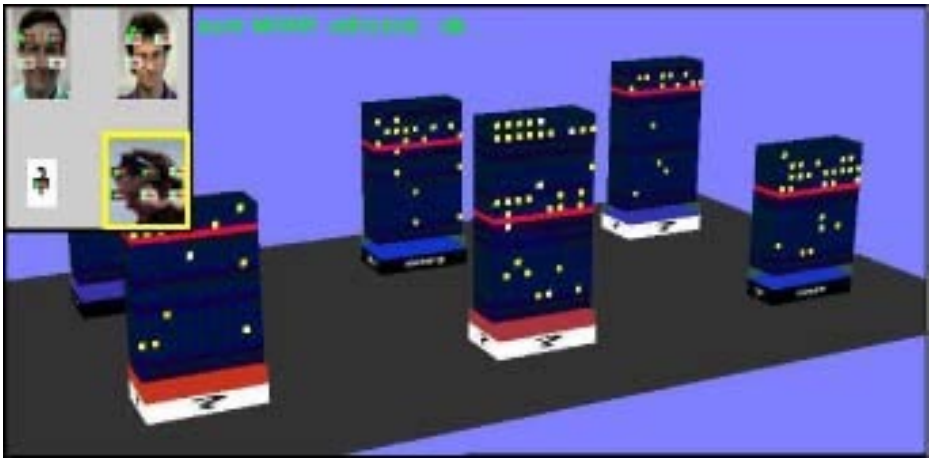


Figure 5.21: Workstation monitoring – combining views: `top` command by user and `top` command by workstation view.

The multiple views concept, although not novel per se (Mukherjea et al., 1995) (Masui et al., 1995) (North and Schneiderman, 1997) (Roth et al., 1997) (Baldonado et al., 2000) (Roberts, 2001), is used by us essentially as a means for validating our work, namely regarding the automatic visual mapping of information. Furthermore, it is our automatic mapping engine that allows for using multiple views as a proof of concept since it substantially reduces the development time of yet another visualization.

The examples shown illustrate the need for multiple views. However, the desirable integration of the different tools and views in a single information visualization workspace, similar to the one referred in (Roth et al., 1997) is still missing. As referred in Section 5.6 our development framework is already prepared to deal with this integration of multiple worlds and this further step is thus a logical direction for future work.

As a final note, it is worth stressing the fact that in our system we deal with 3D real-time visualization of dynamic data – which naturally poses additional challenges compared to combining static or off-line 2D visualizations such as in (Roth et al., 1997).

## 5.6 Interacting with 3D metaphoric worlds

This section presents some interaction mechanisms already implemented in the CyberNet project, namely the selection mechanism. We describe how the user can select objects inside a virtual world and across different worlds.

We also present briefly other interaction mechanisms, such as the horizontal slide used in the network traffic visualization tool (Section 5.5.4) or the selective transparency used in the Eurécom's building application (Section 5.5.5).

### 5.6.1 Selecting objects inside a world

Selection is one of the most important interaction mechanisms. The user needs to be able to select a node in the metaphor hierarchy using either direct manipulation or external menus. The user also needs the choice to extend the selection to all the descendant nodes of the selected nodes.

In the building metaphor (Section 5.5.5), these kinds of mechanisms can be used, for instance, to select all the computers that are located at a given floor. This kind of selection only allows to select groups of objects that belong to the same metaphor hierarchy sub-tree.

For instance, selecting a building in the city metaphor (e.g., Figure 5.1) automatically propagates the selection to all the floors, walls and windows of that building. We call this selection mechanism a *metaphor-based selection* (Figure 5.22 on the left) since the selection criteria are based on the metaphor structure.

However, the user often wants to select objects according to service model criteria, which may not be directly available from the hierarchical structure of the 3D world. In the NFS service visualization tool, for instance in the city metaphor (Figure 5.1), when the user selects a building he may want to visualize which elements are related to that building. This is dependent upon the service model. We name this mechanism *service-model-based selection* (Figure 5.22 on the right).

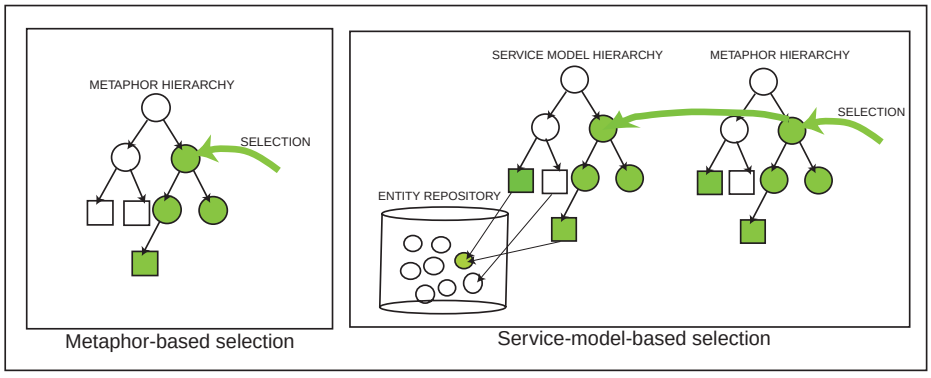



Figure 5.22: Selection mechanisms.

In our previous NFS service visualization example (Figure 5.1), each floor of the building represents a client computer. NFS client computers are also represented as districts (clients are residential districts). When selecting a building (disk) the user may want to select all the districts (computers) in all towns (subnets) that are related to that building (disk).

For that selection to be effective, the selection must be propagated from the metaphor to the service model down to the entity level. When a building is selected, the selection is transmitted to the corresponding local disk service node and all its descendant service nodes and related entities are also selected. When an entity is selected, the selection is retro-propagated to the service nodes that reference that entity and from these service nodes up to their graphical representation in the metaphoric world. This mechanism can be used to help the user identifying the clients of a NFS server and, specially, the clients that are not located in the same subnet of the server. This is useful for performance tuning or for security purposes.

### 5.6.2 Selecting objects across different worlds

Being able to propagate the selection down to the service model and entities is a very powerful mechanism that can be exploited to extend the scope of the selection mechanism to several metaphoric worlds. The main idea is to allow the user to interact with several metaphors that represent different service models connected simultaneously to the same entity repository. We believe that one of the strengths of 3D metaphoric representation will emerge from the combined use of several interacting tools.

The multi-world selection mechanism (Figure 5.23) can be used to allow for specifying free selection criteria; i.e., selection criteria that are neither metaphor based nor service model based. The user may view the same network according to several views, each view being based on the metaphor that is most suited to represent each service model (Figure 5.24). In one view he may visualize the building and the location of computers and network devices. In another view, he can visualize the topology of the network. In a third view, the user can monitor the NFS service using the city metaphor.

With the multi-world selection mechanism, it is possible to select a network device (e.g., a hub) in the cone-tree view where they are more easily identified, and all the computers connected to that hub will be selected in all the three views. The user can also select the clients of the a NFS server in the city view and locate them in the building and in the topology views. The selection mechanism can be further extended in order to allow the user to combine several criteria (by ANDing or ORing successive selection actions).

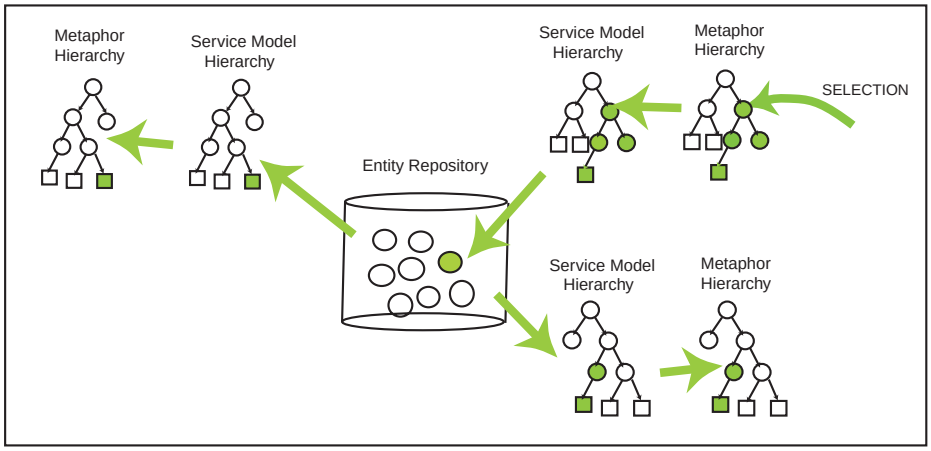



Figure 5.23: Propagating selection between views.

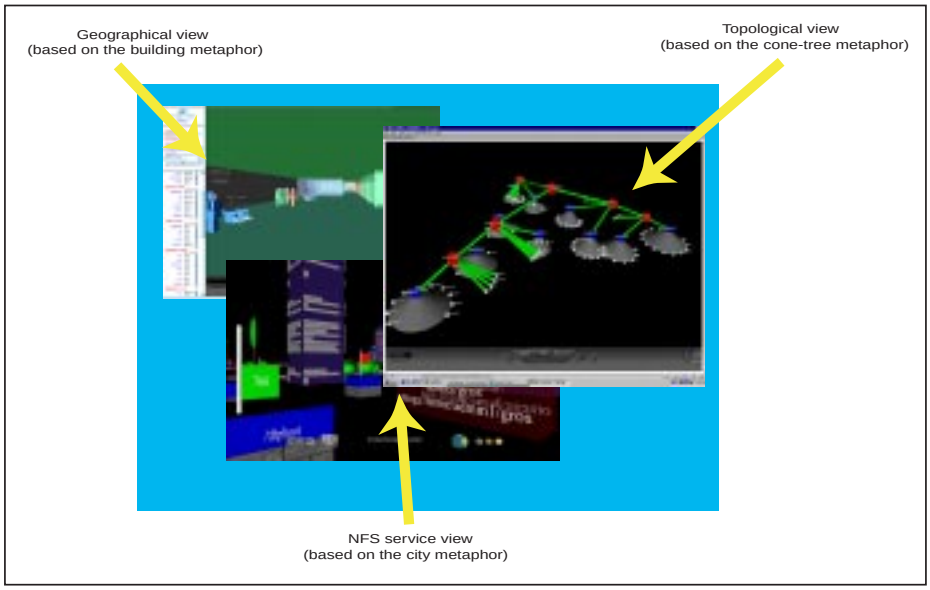



Figure 5.24: Working with several metaphoric worlds.

### 5.6.3 Other interaction mechanisms

We found that being able to interact with the world in order to modify its appearance was important. An example of such a study has been done within the context of the building metaphor. We felt that having a model of the building and being able to navigate and interact with the objects according to the actual building is important but sometimes restrictive. Virtual worlds

should bring more possibilities than just mimic the real world. For this purpose, we have developed mechanisms that allow the user to have control over the world appearance itself.

For instance, in the building metaphor, the user may modify the transparency of walls and grounds. The user may render all the ground and office walls transparent in order to easily observe the contents of all the building. He may also render some offices partially transparent according to criteria such as the department they belong to. Further stretching the interaction, the user may even decide to "rebuild" the building by placing a department per floor (in real life, the departments at Eurécom Institute are spread over the different floors). Figure 5.25 shows some examples of the above interactions.

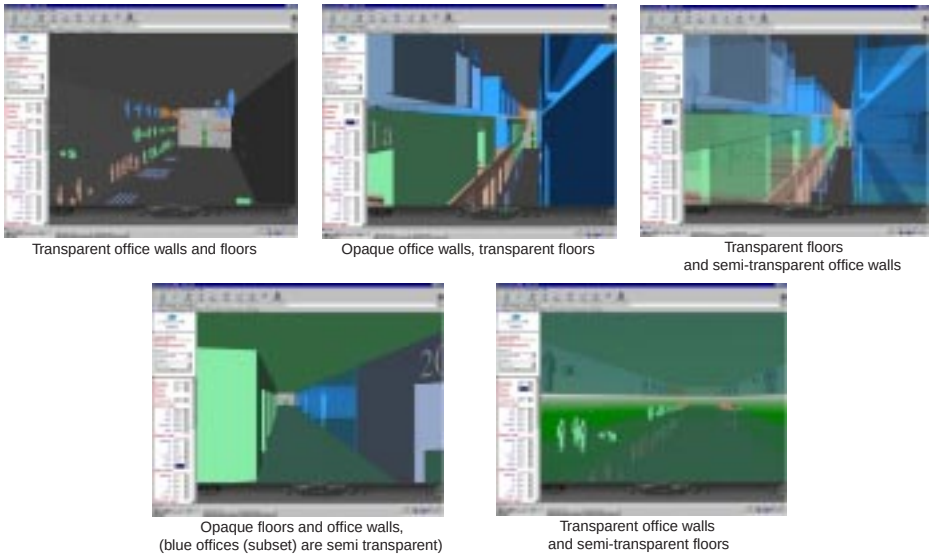
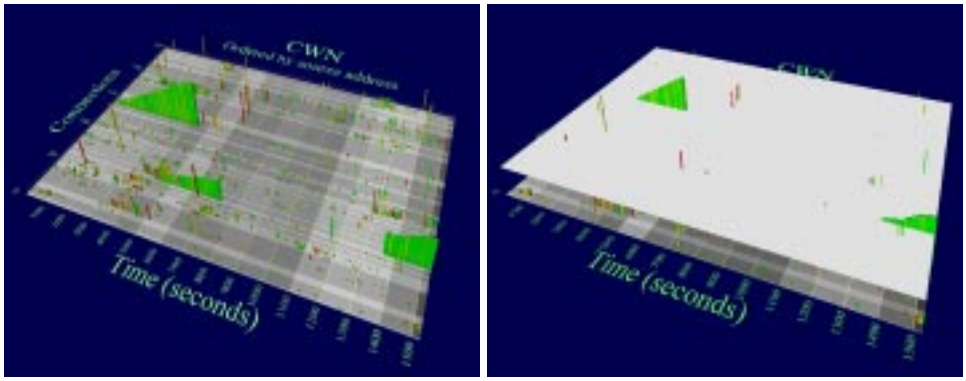



Figure 5.25: Modifying the appearance of the virtual representation of the Eurécom Institute building.

Another example of interaction with metaphoric worlds in order to better analyze interesting information may be shown in the network-traffic visualization tool (Section 5.5.4). Here the information is displayed in a information landscape metaphor, where the height of the spikes, as well as their color hue, encodes important information. In order to compare different heights, namely to see values above a certain threshold, the user may utilize a horizontal slide that "cuts" information below that threshold. Figure 5.26 shows an illustrative example of the horizontal slide application.




(a) All data is visible

(b) Values below a certain threshold are hidden

Figure 5.26: Network traffic visualization – application of the horizontal slide interaction tool.

## 5.7 Conclusion

In this chapter we have presented our work on multiple views, illustrated by many examples taken from the visualization tools that we have already implemented. For this, we have briefly introduced the visual metaphors we have used so far and the tools in which they were utilized. We have also presented some related interaction techniques.

Regarding the multiple views, in the numerous examples that we have given and that have been implemented, we have tried to always present the motivation behind each different view. Multiple views per se are not a guarantee of a useful visualization tool. Their usage must be justified and they must bring some added value to the tool. This added value priority is to ease user's task – this entails a better user visual perception and comprehension of the data displayed.

The multiple views have been validated in a number of different tools for different applications, ranging from network topology visualization, to staff information visualization, or to web server data analysis. This multidisciplinary trait agrees with the application-domain independence characteristic of our development platform, and serves also as a validation for our approach.

We have presented several different visual metaphors, and how we have used them in different visualization tools. Some of them are still quite naïve though. Their inclusion in our examples is useful to illustrate their potential, but they must be refined and extended. Furthermore, we are still looking for

new and inventive visualizations that make the most of 3D metaphoric representation. All the visual metaphors presented in this chapter are adapted from prior work in the field, and all show the potential for further refinement.

Some user-interaction techniques have also been dealt with, namely the selection mechanism. We have described our concepts of metaphor-based selection and service-model-based selection, and how we are able to propagate selection between different views, in order to implement a multi-world selection mechanism. We have also presented interaction mechanisms based on different levels of transparency and a horizontal slider to compare values (heights) above a given threshold.

Further work has to be done in the field of user interaction and the combined use of different tools. The ultimate goal would be to create a global information workspace where all the services were integrated. Up until now, this integration is not transparent to the user – each tool constitutes a different application. We also envisage using such interfaces to provide ways to interact with the real world (e.g., kill a process on a workstation).

# Chapter 6

# Navigation

> In hypermedia, one can become "lost in hyperspace"; in spatial worlds, one can simply get lost. (Modjeska, 1997)

## 6.1 Introduction

We have seen on previous chapters how to structure, map, and visualize data using virtual metaphoric worlds. Nevertheless the visualizations per se are only truly useful if the user is provided a means of exploring the information. A common way of data exploration is navigation. In the case of 3D information visualization, navigation as a means of information exploration attains even more importance due to the extra exploitable dimension.

Navigation has been defined in various ways. It has been defined as "the process whereby people determine where they are, where everything else is, and how to get to particular objects and places" (Furnas and Jul, 1997). Or, as Rudy Darken puts it: *navigation* = *wayfinding* + *locomotion* (Furnas and Jul, 1997), wayfinding being "knowing where to go" and locomotion "how to get there". However, Jock Mackinlay, when asked to give a definition for navigation just put it as "navigation is getting lost" (Furnas and Jul, 1997). In fact, navigation in large three-dimensional worlds is by all accounts afar from a trivial task. Thomas West goes as far as portraying it as "the increasingly pressing problem of our age – that is, quickly and effectively navigating oceans and oceans of information" (West, 1998).

Navigation in large virtual worlds is thus still perceived as a difficult task and not only for naïve users; there is anecdotal evidence that electronic navigation is considered difficult even by the builders of these virtual worlds. Wayfinding, "knowing where to go", is sometimes perceived as the hardest part; other times, it is the locomotion, "getting there", that is found difficult.

Independently of the definition of navigation in electronic worlds, or whether it is the locomotion or knowing where to go that the user finds more difficult, the fact remains that navigation in virtual worlds is still perceived as a very difficult task.

Unfortunately, without navigation one of the strongest means of data exploration and comprehension is lost. Action and perception are tightly coupled: "a moving point of observation is necessary for any adequate acquaintance with the environment" (Gibson, 1979). Hence, if the user is given the possibility of navigating in the virtual world, among the data that is currently displayed, he possesses a powerful mechanism of data exploration and comprehension. For instance, the user may look at interesting data more closely. He may look at important data from all the perspectives he wishes to. Also the value of serendipity needs to be emphasized; if the user is allowed to "stroll" around and inside the data, he may discover some detail or relation that would have been otherwise overlooked. This unsupervised navigation is related to the concept in (Gibson, 1979) of discovering affordances in the environment in an exploratory mode.

Furthermore, without adequate navigation we speculate that it does not make sense to use 3D virtual worlds for data visualization. As we have seen in Section 2.6, one of the strongest pros to use 3D information visualization, as opposed to 2D, is the capability of the former to display larger volumes of data, when compared to the latter. Without navigation, the 3D visualizations are just 2D visualizations with perspective. The absence of navigation render the depth dimension almost useless. The navigation in large virtual worlds thus constitutes one of our major axes of research, as it constitutes a fundamental asset for data exploration.

### 6.1.1 Motivation and problem statement

Anyone who 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 simple navigation tools that allow the user to modify the camera parameters such as orientation, position and focal. Using these tools, it is frequent that, after some movements, the user is lost in the virtual 3D space and tries to restart from the beginning. Our ultimate goal is to ease user navigation in electronic metaphoric worlds.

When interacting with a 3D virtual world, one of the first requirements is being able to navigate in the world in order to easily access and explore information that allows for judicious decision making and for solving eventual problems. Basic navigation requires being able to modify the viewpoint

parameters (position, orientation and focal). For the user's movements to be efficient, it is important for the user to have a spatial knowledge of the environment and a clear understanding of his location. In order to enhance the user's navigation, navigation tools have to take into account the user's goals and provide tools that help the user accomplish specific tasks.

This chapter presents how we have addressed the 3D navigation problem in the context of the CyberNet project (Chapter 3). Our underlying principle is to help the user navigate by adapting the navigation tool to the virtual world. We feel that the navigation schemes provided by the 3D browsers are too generic for specific 3D visualization tools and we have developed adaptive navigation features that are dependent on the 3D metaphor used for visualizing the information and on the user's task.

In this chapter we introduce our concept of *metaphor-aware navigation* and describe how, in this context, the system takes care of the locomotion navigation part for the user, by providing different automatic methods for the user to move around in the world. Our navigation system also addresses the wayfinding part and, in this context, we present the *CyberNet metaphor wizard*. It provides means for the user to know rapidly what information is available and how to choose that information as target destination. We further show that we give the possibility for the user to navigate semantically and physically/metaphorically based due to a tight binding between the semantical and the physical structure.

This work focuses on the navigation in 3D virtual worlds using standard user interface tools (i.e., mouse and keyboard). In other words, we only address desktop navigation – we do not address specific problems and solutions related to immersive navigation.

### 6.1.2 Chapter organization

The remainder of this chapter is organized as follows: Section 6.2 is dedicated to the presentation of related work regarding user navigation, and in Section 6.3 we present our approach. Section 6.4 describes our novel concept of metaphor-aware navigation. In Section 6.5 we discuss task-driven navigation and in Section 6.6 we describe our system's assisted navigation mechanisms as well as three types of assisted navigation available in our system: physical, semantic, and historical navigation. Section 6.7, Section 6.8, and Section 6.9 describe, respectively, the navigation modes, the user modes, and the movement modes existent in our navigation system. Section 6.10 is dedicated to implementation details and Section 6.11 to navigation examples. Section 6.12 closes this chapter with some conclusions.

## 6.2 Prior work

Some previous work done in the field of 3D navigation focused mainly in viewpoint manipulation. Spatial knowledge is another subject that has also been addressed in several research work dealing with electronic navigation. Other related research work has primarily tackled the issue of constrained navigation. This section presents previous work regarding these three subjects, as they are closely related to the navigation research done in the context of CyberNet; respectively, the navigation user's modes (Section 6.8), survey knowledge (Section 6.4), and helped navigation (Section 6.6), for instance. Finally, we refer previous work addressing semantical and physical structures and navigation, as it also relates to our navigation implementation options and to our goal of allowing the user to navigate both semantically and physically.

### 6.2.1 Viewpoint manipulation

Hand (Hand, 1997) reports most of the work already done on viewpoint manipulation. Navigation tools can be classified as being egocentric – moving a viewpoint through the world, or exocentric – moving the world in front of a viewpoint. They are also classified (Mackinlay et al., 1990) according to the viewpoint movement type: general movement – exploratory, targeted movement – with respect to a specific target, specified coordinate movement – exact position and orientation are specified, and specified trajectory movement – path of the viewpoint is specified.

Most of the navigation tools implemented by VRML (Virtual Reality Modeling Language) browsers fall in the egocentric category (Mackinlay et al., 1990) and the movements allowed have names such as fly, pan, or walk. General movements require to fix all the viewpoint parameters but one, and to let the user modify the value of that specific parameter using the mouse or the keyboard. Some targeted movements (such as 'fly to' with direct selection of the target, or 'jump to' with a list of viewpoints for target selection) are already supported. Although they may exploit the 3D world to simulate gravity or collision, these navigation mechanisms are completely independent of the virtual environment itself.

### 6.2.2 Spatial knowledge

Spatial knowledge has been classified in three classes (Thorndyke and Goldin, 1983): landmark knowledge (being able to identify positions using visual cues), route knowledge (having a knowledge of spatial relationship between

visual cues) and survey knowledge (having a global spatial understanding of the environment). Or, according to research work on navigating large virtual worlds (Darken and Sibert, 1996a), spatial knowledge can be described in three hierarchies of information:

- landmark knowledge gives visual information about specific locations (landmarks)
- procedural knowledge (or route knowledge) is information about the sequences of steps needed for following a particular route
- survey knowledge is topological information – there is a global frame of reference encoding object locations and distances between objects

This work was done based on the fact that what is already known about real-world navigation and wayfinding also applies to electronic environments, since such principles appear to be independent of the type of space. The two basic principles investigated were spatial knowledge and environmental design.

Environmental design tries to put in place the foundations on which to build spatial knowledge. The basic urban-design elements, as cited in (Darken and Sibert, 1996a), include:

- paths are channels of movement (e.g., streets, walkways, etc)
- edges are linear but do not facilitate movement, and often constitute a boundary or a break (e.g., rivers, railroad tracks, etc)
- districts are mid-sized sections of a city, having some common characteristics such as a particular architectural style
- nodes are strategic spots in the city where the observer can enter
- landmarks are reference points, external to the observer, where he cannot enter but rather view from a distance; they must have some kind of uniqueness quality to them

Based on these real world navigation principles, some design principles were established (Darken and Sibert, 1996a) to build easier navigable virtual worlds. These design principles are divided in two categories, map design principles and organizational principles, respectively related to spatial knowledge and environmental design.

### Map Design Principles

Maps are the most common form of acquiring spatial knowledge of an environment, aside from gaining this knowledge by direct experience, which is done usually *a posteriori*. The spatial knowledge acquired from a virtual map should facilitate wayfinding. Thus the virtual map should be design to the following principles:

- show organizational elements and organizational structure
- always show the observer's position
- orient the map with respect to the terrain and the observer

### Organizational Principles

Embedding organizational principles in virtual worlds design has as goal to allow the observer to mentally organize the environment into a spatial hierarchy, by providing a hierarchical structure. These principles are;

- divide the world into distinct small parts, hierarchically organized
- organize the small parts under a simple organizational structure (e.g. grid)
- provide frequent directional cues

These six design principles are intended to provide a foundation for building large navigable virtual worlds. Some experiments were conducted in order to evaluate the efficacy of the design principles presented (Darken and Sibert, 1996a) (Darken and Sibert, 1996b). Amongst general conclusions attained we can refer: in the absence of adequate directional cues, observers will experience disorientation, and navigation and way finding are inhibited; a large virtual world lacking in structure is difficult to search; a map allows for search-strategies optimization.

Navigation improves with knowledge of the surrounding space. In an exercise of navigation in a conference hotel with one member of the exercise pair blindfolded, prior familiarity of the space was the factor cited as being the most helpful (Furnas and Jul, 1997). In the following we describe some methods intended to give the user some feedback regarding position awareness.

Different kinds of solutions have been investigated regarding user's position awareness. The main idea is to provide visual feedback of the user

position. The simplest feedback scheme is to permanently display the 3D coordinate position of the user. This solution is not of great help especially because this position only has a meaning if the user already has an in-depth knowledge of the world geography – survey knowledge.

More elaborated solutions are based on the display of a global, simplified view of the world added in the user's field of view. Stoakley et al. (Stoakley et al., 1995) propose the use of a "World In Miniature". Satalich (Satalich, 1995) studied how two-dimensional maps could help users to navigate in virtual buildings. Another research work presents the concept of a "map view" (Edwards and Hand, 1997), a tool that allows the user to monitor his position (viewed from a "satellite") on a small virtual screen embedded in the 3D world.

Although there are differences among these methods, the underlying idea is to include – in front of the user – a small overall view of the world and a marker showing the position of the user in that world. For orientation awareness, Murta (Murta, 1995) has pointed out the importance of the knowledge of the vertical direction and presented some "upward" cues such as ground planes, backdrops and directional illumination.

The concept of "trailblazing" has also been proposed (Edwards and Hand, 1997). The basic idea is to allow the user to leave graphical markers (that act as user defined landmarks) in the 3D world. The use of landmarks can be compared to the use of hyperlinks in HTML (HyperText Markup Language) documents, trailblazers being some kind of equivalent to the user's bookmarks proposed by HTML browsers.

### 6.2.3 Constrained/Assisted navigation

Allowing a user to navigate freely in the environment is important, but most of us have experienced that being "as free as a bird" is not that easy or, more to the point, useful. Research has been done in order to enhance the navigation activity by taking into account the goals of the user. The solution is generally referenced by the term *constrained navigation*. Although it is true that these methods generally put constraints on the user movements, we prefer to use the term *assisted navigation*, as the final goal is not to constrain the user movements, but to help him navigate more easily and efficiently.

Some early work towards the direction of helped navigation is reported in prior research (Hand, 1997). Kaye (Kaye, 1997) presents the "tracking viewpoint"; the idea is to modify the user's direction of view in order to allow the user to track a specific object – potentially moving – in the scene. In other words, the system provides an automatic cameraman that follows an object

in the scene. This idea was also presented for tracking the user position in order to control the "map view" (Edwards and Hand, 1997). Hanson et al. (Hanson and Wernert, 1997) presents a tool that constrains camera movements so that the position is limited to a surface and the orientation is dependent upon the surrounding objects. This study has been done mainly in the context of terrain navigation. The authors conclude that this kind of help tools should be context-dependent, state-dependent and history sensitive.

### 6.2.4 Semantic navigation

In research work addressing hypertext navigation, Dillon et al. argue that ultimately, the idea of directly navigating semantical space is spurious (Dillon et al., 1993). It is furthermore argued that in effect we cannot navigate semantic space, at least not in the way we navigate physical environments – we can only navigate the physical instantiations that we develop of the semantic space, and (Dillon et al., 1993) recommend that a well designed hypertext system entails a strong correspondence between its semantical and physical structure. It should be noted that this firm opinion on the impossibility of navigating semantically is made in the context of navigation in hypertext systems. Nonetheless, the same recommendation of close binding the semantical structure with the physical structure has been made in related work regarding wayfinding (Passini, 1984), for the cyberspace (Benedikt, 1991), and for effective view navigation (Furnas, 1997). We take this recommendation into consideration in order to provide a combination of physical and semantical navigation, as will be described in the next section.

## 6.3 Our approach

We feel that the navigation schemes provided by traditional 3D browsers are too generic for our specific navigation needs. It is evident that when the user navigates from office to office in a virtual building he does not use the same navigation mechanisms that he uses when he is exploring a landscape of data or studying the topological structure of a cone tree.

In the CyberNet visualization framework, all the metaphoric worlds are constructed using standard well-defined graphical components: layout managers are used to arrange 3D glyphs in space. For example, the same stack layout manager is used to align offices along a corridor and to stack floors on top of each other, in order to construct a building. However, the navigation along a corridor is not the same as the navigation from one floor to another. Our belief is that the navigation mechanisms should be associated to the

graphical components. The objective is that the user navigates in the world with the mechanism most suited to the metaphor. We call this principle *metaphor-aware navigation.*

We use a *distributed navigation system* that takes advantage from the metaphoric components navigation capabilities to implement the metaphor-aware navigation concept. This distributed navigation is supported on the basic principle that each metaphoric components is responsible for the navigation in its own "turf" – usually defined by its bounding box. It further uses routing tables to move the user to the next metaphoric element with navigation capabilities.

In addition, we believe that being completely free to navigate is not always the best solution. It is frequent that the user gets lost quickly and naturally becomes frustrated. Consequently, we have implemented some *assisted-navigation modes* where the system takes care of moving the user or his direction of view to the new point of interest. Of course, this does not in any way forbid the user to move freely if he so wants, and to navigate without any of the assisted-navigation modes provided for him.

Finally, we provide means for the user no navigate based both on the semantical structure and on the metaphorical structure. Semantic navigation in our system is strongly connected to the assisted-navigation movements and takes advantage from the fact that the mapping of the service tree on the metaphor tree is a one-to-one mapping. Metaphoric navigation is related to navigate freely and to direct target selection in the world. The user may thus navigate both the service tree or the metaphor.

All these navigation matters are developed in the next sections.

## 6.4 Metaphor-aware navigation

In a comparative study between different metaphors (Wiss and Carr, 1999) one conclusion was that custom navigation is crucial in 3D user interfaces. We think that when the user navigates from office to office in a virtual building (Figure 6.1) he does not use the same navigation mechanisms as when he is exploring a landscape of data or studying the topological structure of a conetree. Our belief is that navigation mechanisms should be dependent on the metaphor and embedded in the graphical components of the 3D world. The goal is that the user navigates in the world with the mechanism most suited to that particular metaphor. We call this principle *metaphor-aware navigation.*

One of the easiest ways to understand the idea behind metaphor aware navigation is to consider path-based navigation in a building. When a user

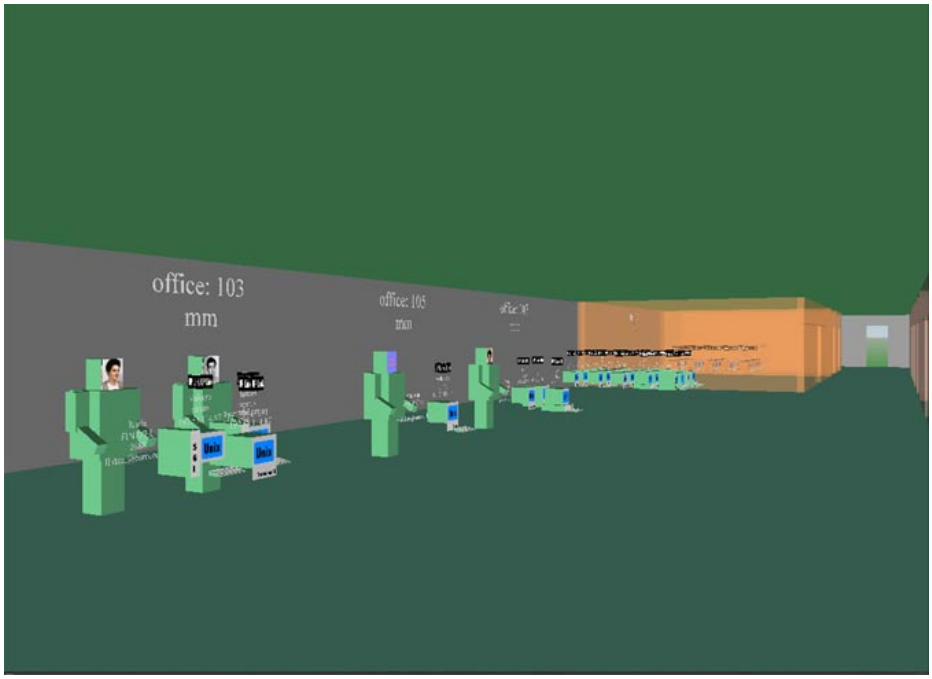



Figure 6.1: Inside the building metaphor – information visualization tool for staff and network devices.

wants to go from his current position to another office, he generally follows some logical path through the stairs and corridors (see Figure 6.2). Following this path (as opposed to instantaneous jumping) is important since it gives to the user the knowledge of the relative location of the objects in the virtual space – route knowledge. Using traditional navigation tools, this kind of navigation is not an easy task. Our metaphor-aware navigation system will help the user accomplish his task: the system will automatically route him along the right path in the building and will take care of the details of the navigation (e.g., turn, go down, and so on). It should be noted that the path taken by the user in our metaphor-aware navigation system is a "feasible" path from the point of view of real-world constraints (e.g., there is no wall traversing, no ceiling or floor crossing, and so on). This is in agreement with research that states that navigation schemes in virtual worlds must be tied to the user's prior knowledge of how to behave in similar real-world environments (Dillon et al., 1993).

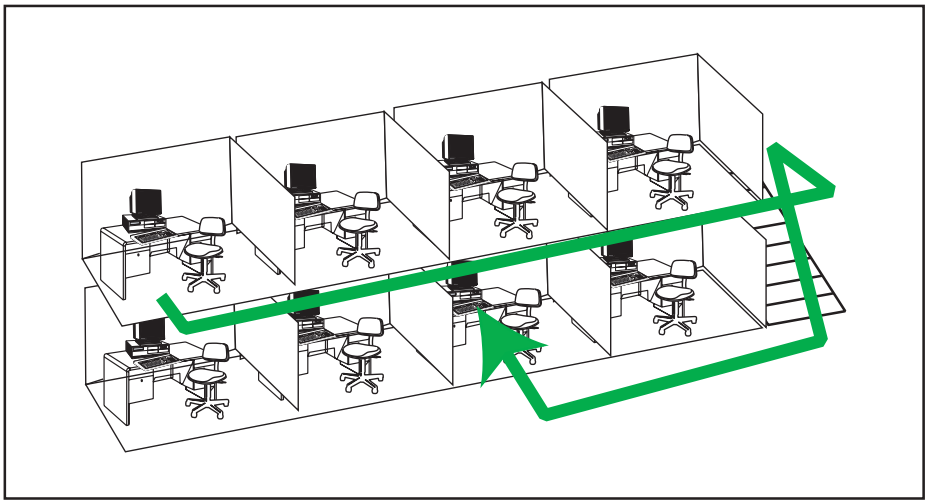


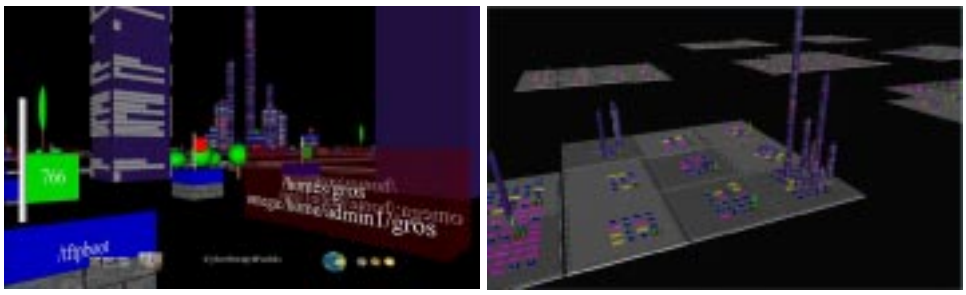
Figure 6.2: The metaphor-aware navigation path from one office to another.

### 6.4.1 User navigation desiderata

The basic principles we took into account when designing the CyberNet navigation tools include the following:

- Navigation tools should be adapted to the metaphoric world and may sometimes mimic real world navigation. This helps preventing user disorientation. We gave an example based on the building metaphor in the previous paragraph, where route knowledge is enforced by the path navigation mode, in which the user navigates in a manner similar to the real world. This goes in the same direction as former research that argued that user knowledge of how to behave in certain environments (e.g., roads must be crossed at a pedestrian crossing) must be tied to the navigational schemata (Dillon et al., 1993).

- In order to ease the navigation, the metaphor-aware mechanism should take into account the actual level of the metaphor hierarchy that the user is currently in. Different navigation schemes may be used for navigating at different levels of the metaphoric world hierarchy. This kind of navigation is helpful to acquire different levels of spatial knowledge. For instance, in a town, the user may navigate at the district level like a bird flying over the town, or he may navigate at the street level walking in the streets in order to go from one building to another (Figure 6.3). The user will have a global view seen from far away – survey knowledge – when he is navigating at the district level, and a



(a) Walking at street level (b) Flying at district level

Figure 6.3: City metaphor – NFS data visualization tool. The two navigation metaphors correspond to the two main types of spatial knowledge: route knowledge and survey knowledge (Thorndyke and Hayes-Roth, 1982).

narrowed, more local view – route knowledge – when navigating at the street level. These two different levels of navigation support the distinction made in previous research (Thorndyke and Hayes-Roth, 1982) between the two main types of spatial knowledge: route knowledge and survey knowledge.

- Navigation tools should take advantage of the metaphor hierarchical world and provide schemes to navigate through the various levels of the hierarchy. The user will then benefit from his knowledge of the underlying structure of the world, which matches the service hierarchy, to navigate semantically in the virtual world. He may just use a navigation interface proposing standard movements in the hierarchy (i.e., up, down, next, and previous).

- Navigation tools should help the user when he has a specific navigation task and should handle the navigation details for him – i.e., they provide the locomotion for the user.

- The navigation tools should take advantage from the fact that, although sometimes based upon the real world, the virtual worlds pose fewer constraints to the user (e.g., gravity, size, and speed). This is the case with the "jump to" navigation mode, which basically takes the user to the target destination instantaneously; this is in contrast to the path mode, that takes the user along a metaphor-dependent feasible path. The navigation tools must also take into account the fact that the user may change the surrounding virtual world and adapt accordingly.

### 6.4.2 Metaphoric components

The aforementioned principles led to the design of what we call *metaphoric components*. Metaphoric components are the graphical elements of a metaphor, i. e., 3D glyphs and layout managers, with embedded interaction and navigation capabilities. Thus, according to our terminology, metaphoric components and graphical elements of a same metaphor do not exactly mean the same thing, from a theoretical point of view. It should be noted though that not every metaphoric component is constrained to intervene in the navigation. Section 6.10.4 explains the concept of *transparent nodes* from the navigation point of view.

Graphical elements are just the basic elements that are used to construct a metaphoric virtual world and serve to visually map the information. Metaphoric components are all the graphical components of the metaphor, but we consider that this type of components must have embedded interaction and navigation features, if the case applies. As an example, a rectangular box may be a graphical element (3D glyph) for constructing a given metaphoric world (e.g., a virtual building). Nevertheless, that rectangular box when instantiated in the same metaphor to play the role of a floor with navigation features (e.g., moving the user along the floor's corridor) is then perceived by our system as a metaphoric component.

This latent ambiguity of the metaphoric components goes to the encounter of the argument that contrary to Graphical User Interfaces (GUIs), electronic worlds contain objects function both as navigational and destinational (Benedikt, 1991). This kind of multipurpose property of electronic worlds elements has also been advocated in other research (Alexander, 1982) (Furnas, 1997).

## 6.5 Task-driven navigation

For the user's movements to be efficient, it is important for the user to have a spatial knowledge of the environment and a clear understanding of his location. Much effort as been put in the "wayfinding" tasks (Darken and Sibert, 1996a). The interest on this topic is mainly related to immersed interfaces for virtual reality.

Wayfinding is obviously not the only task a user may want to do when navigating in virtual spaces (Gabbard and Hix, 1997). In addition to "be as free as a bird", which should be always possible, following are several user tasks related to our field of interest (we do not pretend to be exhaustive):

- inspect an object either by jumping to a predefined viewpoint attached

to that object, or by looking in the direction of that object from the user's current position. In the latter, a tracking mechanism should handle moving objects.

- travel to an object following a logical path according to the metaphor in order to be aware of the objects relative positions or to monitor other objects along that path.
- scan, traverse or visit several objects according to some criteria:
  - scan all the objects that are children of a given object
  - traverse all the objects that have a common ancestor in the underlying hierarchy (depth first)
  - visit all the objects from a user defined list or that the user has already inspected. This scanning requires jumping, traveling or looking at the successive objects
- have a global view of a set of objects. What is viewed from a parent is dependent upon the metaphor itself; the system may give an overview of all the children. This is useful for examining relations and correlated objects.
- navigate in the 3D world relatively to the underlying hierarchy. The user can take advantage of his knowledge of the underlying hierarchical structure of the metaphoric virtual world. This is what we call semantic navigation and will be further detailed in Section 6.6.5.

## 6.6 Assisted navigation

In order to assist the user in his navigation task, the system must know the *user's current state*. We define the user's current state as constituted by three parameters: the *user's current location* in the 3D world, his *current node of interest* (the node that currently has his attention) and a *new node of interest* (the next target of his attention).

When the user is simply moving around in the world, the user's current node of interest and his current location are identical. Even so some navigation tasks require the user to be located in a place and to have his node of interest on another object; an example is the "look at" navigation mode, which is a case of exocentric navigation by viewpoint manipulation. The basic functionality of the navigation system is to allow the user to modify his current state (current location and node of interest), by choosing a new node of interest.

### 6.6.1 Target selection

Our assisted navigation mechanism is hence target oriented: the user should specify a *node of interest*. This node of interest is a node in the metaphor hierarchy that can either be the destination of a movement or an object to look at. In CyberNet, the node of interest can be defined using two basic mechanisms.

*Absolute selection* allows the user to identify an object by its name (using a hierarchical VRML-like viewpoint menu) or by direct selection in the 3D space. *Relative selection* allows the user to identify an object by its position in the hierarchy relatively to the previous node of interest. Once a target is defined, the system automatically handles the details of the navigation for the user. For instance, the user is transported to the target location by following a logical path according to the metaphor.

### 6.6.2 Automating the selection

It is important to be able to automate the selection of the new node of interest. The first use of automation is to be able to go back and forth in the historical list of already visited nodes. Another important automation application is for defining round trips in the world in order to monitor a set of nodes. Simple navigation tasks such as glancing at all the nodes that are children of a given node also require automation. Automation involves getting to the next target (when several objects are involved) or back to the original location (when the end of the movement is reached). The automation requires the system to manage the time spent at each intermediate node of interest as well as the possibility to interrupt the navigation tasks (e.g., suspend, resume, stop).

### 6.6.3 Assisted movements

So far, we have only discussed the schemes provided to select the new node of interest. Further detailed explanation on these schemes in given in the next section (Section 6.7). The system will now have to determine a 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 on combined use of a *user mode* and a *movement mode*. The description of both the user modes and movement modes is done in Section 6.8 and Section 6.9, respectively. The movements obtained by logically combining these two modes are represented in Table 6.1.

We have seen how our helped navigation system is based on the choice of a target destination – node of interest – with the possibility of specifying intermediate passing points, in the case of the "interpolated" movement mode (Section 6.9.2). If we consider navigation as being the sum of wayfinding (knowing where to go) plus locomotion (getting there), as Darken (Furnas and Jul, 1997) defined it, in our system, wayfinding is provided by target selection (Section 6.6.1) and locomotion by the movement modes (Table 6.1).

We define physical or, perhaps more accurately, metaphorical navigation as the case when the user is navigating the virtual world, and his wayfinding decisions (i.e., choosing his target destination) are based on the visual information available, that is, on the metaphoric representation of the data. We define semantic navigation as the case when the navigation decisions made by the user are based on the underlying information structure, as opposed to the visually rendered world.

### 6.6.4 Physical (metaphorical) navigation

The user can effectively navigate in the metaphoric world – *physical navigation* – either by his own means or by taking advantage of the navigation mechanisms that the metaphor provides for him. Namely, the already referred-to path-based navigation where the metaphoric components take care of moving the user to his target destination according to a logic physical path in the metaphor (e.g., no walls traversing in a city metaphor). The target may be chosen directly in the virtual world by clicking on the desired destination object. This is what we mean by physical navigation, as opposed to semantic navigation – the user is navigating based on the visual information displayed and not based on the underlying information structure and semantics.

Since virtually every metaphoric component has navigation capabilities the whole metaphor structure may be traversed – this accounts for the physical navigation (navigation in the visual presentation). For instance, in the city metaphor, represented in Figure 6.3, the user may directly select a given building as target destination because of its height (height is mapping a data value which accounts for choosing the highest value). This is navigating based on visual information. These are only examples of the metaphor-dependent navigation mechanisms that we provide for the user physical navigation. We also provide mechanisms for navigating based on the actual information structure – semantic navigation.

### 6.6.5 Semantic navigation

As the service tree is mapped on the metaphor tree, navigation in the service tree is also possible. This accounts for *semantic navigation*. The user may navigate thus based essentially on the service information rather than on the rendered visualization of that data. For instance, the user has the possibility to choose to go to every element (service node or entity) of the service tree. The navigation mechanism will take him there in the virtual world according to his choice of navigation mode (e.g., by flying or walking the user to the target). This target option is given in the form of a hierarchical menu that displays the whole service hierarchy completely expanded – as it "unfolds" as the user advances/goes down in the service hierarchy.

Alternatively, the user might also choose to go to the ancestors or to the descendants of a service node or entity (note that an entity, being a leaf node, does not possess any descendants). Similarly, the built-in navigation mechanisms embedded in the metaphor will take him there. This allows for a powerful way for exploring and examining relations and dependencies. We also provide semantic information/cues for the user to navigate (and not get lost), mostly in the form of labels, either permanent or pop-up.

### 6.6.6 Historical navigation

Finally, to conclude our presentation of the different navigation types that the user may take advantage of, we should refer that the user may also decide to navigate according to navigation history. We call this type of navigation, *historical navigation*. Historical navigation allows the user to go back and forth in a metaphoric path. In our current implementation, this type of navigation only involves relative selection: backward/forward. Variations of historical navigation might include remembering user-navigated viewpoints or priority ordering based on usage history.

## 6.7 Navigation modes

There are two types of navigation modes: absolute and relative navigation. Both modes are explained in the next two sections.

### 6.7.1 Absolute navigation

*Absolute navigation* requires the user to select a new node of interest using absolute selection – i.e., giving a specific location in the world as target destination. For this purpose, when a metaphoric component is created it should

notify its existence to the system. The system is then responsible for offering a mechanism to select that new object as the new node of interest. This selection can be done, either by using an embedded 3D interface (by clicking on an object for example), or by using an external menu that is hierarchical and context sensitive. This general mechanism looks like the mechanism supported by most VRML browsers for handling viewpoint selection.

Once the user has selected a new node of interest, the system has to determine what steps to take according to the current navigation task. In absolute navigation the target destination is chosen independently of current location.

### 6.7.2 Relative navigation

*Relative navigation* requires the user to select a new node of interest using relative selection: the user chooses the target destination relative to his current location. The new node of interest is chosen relatively to the underlying world hierarchical structure 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 take the user to the next office in the corridor, while "up" should take the user to the corridor or to the next floor (according to what the designer has specified as being the higher level). In other words, up/down as navigational aids take the user to the ancestors/descendents of the node in question; in a similar manner, next/previous, will take the user to the adjacent/preceding sibling (element at the same level in the hierarchy).

This relative navigation scheme is important since CyberNet users are aware of the hierarchical service model represented by the metaphoric world. Since the virtual world structure follows the hierarchical service model they may rapidly access points of interest. When using this mechanism the user is actually navigating semantically – he is navigating the service hierarchy, which matches the metaphor hierarchy. The translation of the user action (i.e., "up") into a visual node of interest is metaphor dependent. In fact, what the system does is take the user to the upper hierarchical level in relation to the service structure (and mapped onto the visual metaphor structure). For instance, as is shown in Figure 6.4 for the virtual building metaphor, the floor (corridor) upper level is the building itself, and not the upper floor in the visual representation. This is implemented using a neighboring table (see Section 6.10.2) provided by each metaphoric component. The navigation system uses this table to determine the new node of interest and then the absolute navigation algorithm is used.

## 6.8 User modes

Once the system knows the new node of interest, it supports two user navigation modes: *go to* and *look at*. The selection of the user navigation mode is done via a radio button interface.

### 6.8.1 Go to

The first user mode is the *go to* mode. In this mode the user's location is modified in order to be transported to the location of the new node of interest. Basically, the user is moved to the target destination. The actual way in which the user moves depends on the movement mode (see description in Section 6.9).

Independently of the movement mode, the *go to* user mode is an egocentric navigation mechanism. The user viewpoint is moved through the world, creating in the user the sense of being at the center of the space.

### 6.8.2 Look at

The second user mode is called *look at*. In this mode the user stays at his current location and his direction of view is modified in order to look at the new node of interest. Like the *go to* mode, the *look at* mode is an egocentric movement – the user's direction of view is moved through the world. The precise movements are dependent on the movement mode described in the next paragraphs.

## 6.9 Movement modes

The system supports three possibilities for movement modes: *jump*, *interpolated*, and *path*. Similarly to the user mode selection, the movement mode is selected via an option menu – radio button interface.

### 6.9.1 Jump

Using the *jump* mode (or *point to point* mode), the user can directly *go to* (or *look at*) the node of interest. This is very simple to implement since it only requires binding to the new viewpoint. Although we all dream of "teletransportation" in our car every morning, this navigation scheme has some drawbacks: the user tends to lose spatial knowledge of the world since, as he cannot observe other objects along the path, he does not have information

on the relative position of the objects anymore. Jump tends to disconnect the user from previous context.

### 6.9.2 Interpolated

In the *fly to* mode, some VRML browsers support interpolation from the starting viewpoint to the destination viewpoint. We implemented a mode based on this approach that we called *interpolated* mode. The result is somewhat unpredictable as soon as the viewpoints have different directions of view: the user gets the impression of doing some strange looping when navigating from one to another. This is the reason why we have implemented the path mode.

### 6.9.3 Path

The main idea behind the *path* mode is that the user should follow a logical path in order to move from one node of the hierarchy to another; this path relies on the metaphor itself and cannot be independently determined. For example, when a user wants to go from one office to another, he will automatically be routed through the corridor. If the office is located on a different floor, the corridor will have to route the user to the elevator (or stairs), the elevator will route the user to the desired floor and then to the desired corridor, and the latter will take the user to the destination office (Figure 6.2).

### 6.9.4 Combining user and movement modes

By combining the user modes with the movement modes, we obtain the navigation techniques described in Table 6.1.

## 6.10 Implementation

In this section we are going to present the CyberNet's navigation wizard and describe the implementation of our distributed navigation system.

### 6.10.1 The navigation wizard

A *navigation wizard* manages the CyberNet navigation system.

Upon creation, each metaphoric component notifies the wizard that it is a potential node of interest and provides, at least, a predefined viewpoint location and an examine point. The navigation mechanism requires the user

| User modes | Movement modes: Jump | Movement modes: Interpolated | Movement modes: Path |
|---|---|---|---|
| Go to | The user jumps to the new node of interest and gets attached to it. | The user flies in a straight line from his current location to the new node of interest and his orientation is modified. | 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 looks in 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 6.1: Navigation obtained by combining the different user modes and movement modes.

to define a node of interest and a navigation mode. The role of the navigation wizard is:

- to manage the navigation user interface. This interface must allow the user to have access to both relative and absolute navigation. Relative navigation only requires the use of standard navigation buttons (that is, up, down, next, and previous). In order to support absolute navigation, the wizard has to make each metaphoric component viewpoint available to the user by inserting an item in a hierarchical menu and/or provide direct selection of the same object.

- to always track the user's current position and current node of interest in relation to the metaphor hierarchy.
- to determine the set of movements that must be done within the context of the current metaphor and according to the current movement mode.

### 6.10.2 Embedded distributed navigation

In order to avoid complex centralized algorithms, the management of the path-navigation algorithm is distributed between all the metaphoric components. Each metaphor component implements a routing table, very much like the one used by IP (Internet Protocol) routers in order to route packets on the Internet. The navigation wizard basically follows the routing and activates the traversed nodes. The effective movements are handled by each activated node within the space it has under its responsibility (usually related to its bounding box). For this purpose, each metaphoric component has three tasks:

- to provide a *neighboring table* used to support relative navigation. The goal of this table is to be able to translate user action regarding relative selection (e.g., up, down, next, previous) into a destination node according to the metaphor.
- to implement a built-in *routing table*. The goal of this table is to define which is the next node of the hierarchy that should be traversed to go from the current position to the desired destination. In the previous corridor example, going to an office on the next floor requires to go to some intermediate node like the elevator.
- to control the navigation in the part of the space it has under its responsibility. In the previous corridor example, the corridor should effectively translate the user onto a path from one point to another. Of course care should be taken during the design to insure that the path is continuously connected when moving from one node to the next.

### 6.10.3 Path navigation via localized control

The general path-navigation mechanism uses these built-in mechanisms in combination as follows: the routing tables are used as a mechanism to find the metaphoric route that links two nodes (see Figure 6.4). The navigation wizard basically follows that route and activates the traversed nodes along

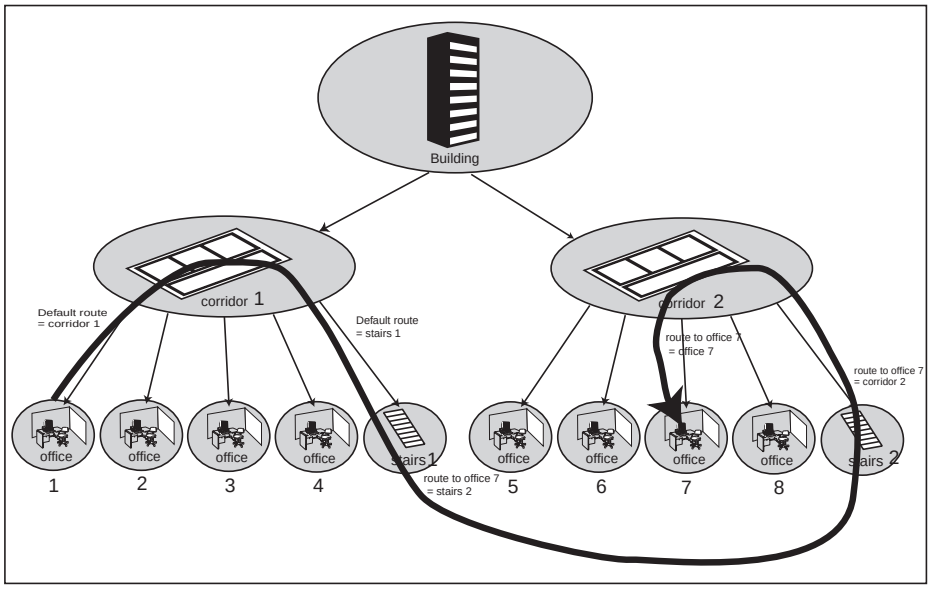



Figure 6.4: Path navigation mode: the metaphor hierarchy tree is traversed using routing tables.

the route. Each activated node handles the effective movements. The current node first has to use its routing table in order to determine the next traversed node. When this node is identified, then the current node animates the user's viewpoint from its current position to the position provided by the next node.

The path followed is dependent on the metaphor but is generally simple to implement because our basic graphical components have usually simple geometric characteristics. The complexity of the resulting world is an effect of the combination of a high number of simple components. Thus this approach is very scalable and provides an intuitive response even for very large visualizations.

### 6.10.4 Transparent navigation nodes

It is important to note that it is not mandatory that every node of the graphical hierarchy participates in the navigation mechanism. One reason is that it may not make sense, in the context of a given metaphor, to provide some glyphs with navigation capabilities (e.g., usually a window does not handle navigation).

There is also another type of metaphoric components that do not possess navigation features: some intermediate graphical nodes are inserted in the hierarchy only for visual purposes and they are transparent from the user

perspective since they have no real-world meaning. For example, the "fit" layout manager is inserted to force the child hierarchy to fit into a specific space allocated by the parent layout manager. It is generally implemented by scaling down its child elements. In that case, the graphical node becomes "transparent" in terms of navigation or even visual representation, and the user does not even notice its existence – only its effect.

In order to be transparent, a node does not notify its existence to the navigation wizard and, thus, is not referenced in any of the neighboring and/or routing tables of other nodes. It may also happen that a node acts as transparent in a determined movement node, and must handle navigation when in a different movement mode.

### 6.10.5 Automated navigation

The ultimate task of the navigation wizard is to manage navigation automation. The navigation wizard also maintains a history buffer in order to go back and forth to previously accessed locations. The navigation wizard is able to manage looped sets of nodes of interest and to loop through them in order to implement round trips. It also supports automatic scanning (one level in the hierarchy) and traversing (depth first traversal of the hierarchy) of the children of a given node. For all these tasks, the navigation wizard exploits the information stored in the metaphor hierarchy which is a faithful mapping of the service hierarchy, and then translates the metaphor information to actual movements that occur in the visual world.

It should be pointed out that the design of the metaphor-aware navigation is truly a part of the metaphoric world design. It has the same level of importance as the purely graphical modeling work. An important design goal is to define a coherent navigation scheme for the metaphoric world as a whole. When designing a metaphoric navigation, the mechanisms to navigate in the world should be developed with the same care that is put on the pure 3D modeling of the metaphoric world.

## 6.11 Examples

In this section we present two examples of user navigation using quite different visual metaphors.

Section 6.11.1 describes user navigation in a real-world based metaphor: a virtual building. In Section 6.11.2 we describe how the user is able to take advantage of our assisted-navigation mechanisms to navigate in a quite abstract metaphor: an information-pyramids metaphor. The examples will

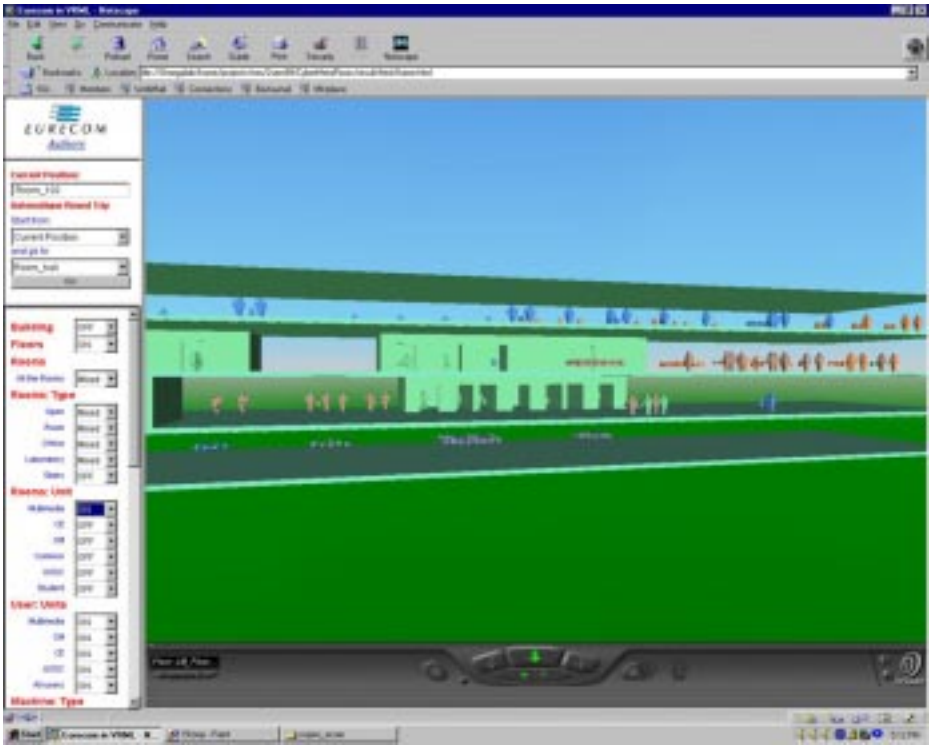


Figure 6.5: 3D visualization tool of the Eurécom Institute building with visual information on the staff and network infrastructure.

be followed by a brief discussion regarding user navigation in both real-world and abstract based metaphors.

### 6.11.1 Metaphor-aware navigation in a virtual building

As part of the CyberNet project we have developed a demo tool to visualize and inspect the physical location of network devices, according to their actual location in the Eurécom Institute building (Figure 6.5 and Figure 6.6). This demo also contains information regarding the personnel and the physical structure of the building, with the correct relative location of offices, labs, and so on.

Although, at first glance, the virtual building may seem like an faithful three-dimensional reproduction of the actual building, the interaction possibilities with the virtual representation go further than those of the real life building. The user may render transparent any wall, or he may decide to hide/show information (for instance, rooms or entire departments) and he may even change the whole configuration of the building by displaying a de-

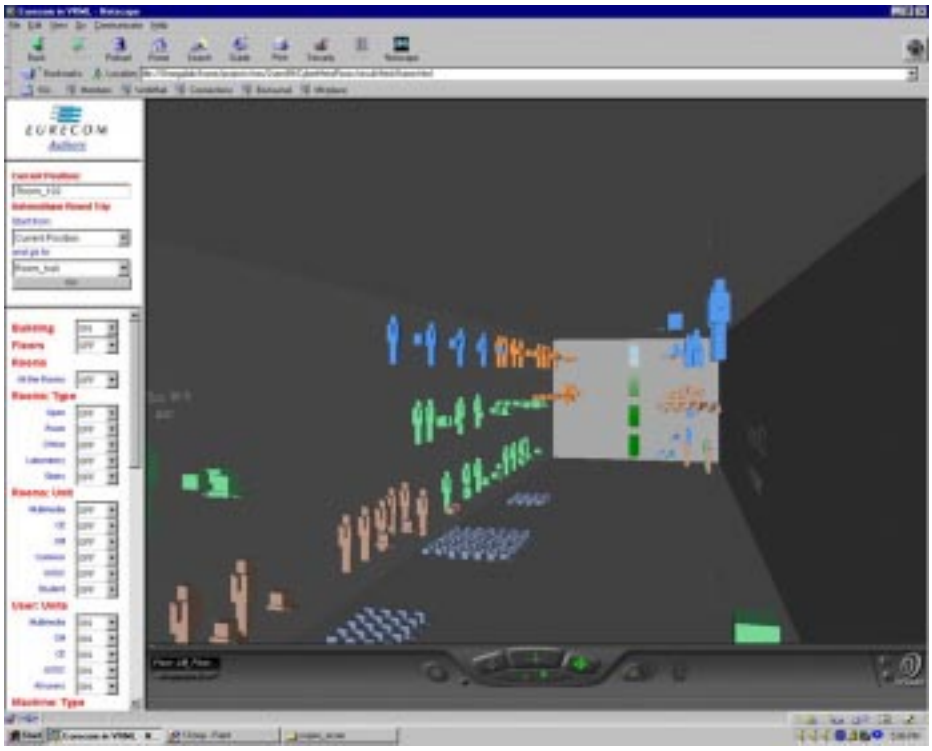


Figure 6.6: The Eurécom building with transparent walls and floors – a suitable environment for testing the *look at* user mode.

partment per floor (in real life and in the default visualization, departments are spread across all floors).

These interaction capabilities have already been discussed in Section 5.6 so we will not further develop this subject here. Nonetheless, we must bear in mind that logical information changes also have an impact on the navigation (e.g., determining which navigation path is followed). For instance, if the user chooses to visualize one department per floor, all the neighboring and routing tables used to determine the intermediate and destination nodes have to be updated, since almost all the elements' locations are bound to be affected.

The main reason for developing the virtual building demo was to test a first version of the navigation wizard and some of our ideas regarding the navigation in 3D virtual worlds. Mainly, we wanted to test the embedded aspect of the navigation, where the graphical components are responsible for handling the navigation in their "territory", and the selection of points of interest and the subsequent determination of the metaphoric path. So, in order

to be able to assist the user navigating, we have implemented the metaphor aware navigation mechanism in the context of the building metaphor.

As previously explained, the underlying idea of our navigation system is to track the user current position so that the system is always able to associate the user's current position with a node in the hierarchy. In addition, the system provides the user with a tool that allows him to select a destination node. This selection can be done, either by using a 3D embedded interface (e.g., by clicking on an object), or using an external menu. This menu is both hierarchical and context sensitive.

In the two-dimensional interaction interface that is visible on the left of Figure 6.5 and Figure 6.6 the user may choose a destination point. The navigation mechanism then takes him there following the shortest "real-world feasible" path (i.e., without traversing walls, taking the stairs whenever it is necessary to change floors, and so on). The selection menu is context sensitive and hierarchical giving the current position as default for the departing point.

The system determines the set of movements that must be done within the context of the current metaphor in order to go from the user's current node of interest to the new node of interest. For instance, if the user wants to go from his current location to a new location on the upper floor, he just has to choose the new target in the hierarchical menu on the 2D interface (the default departing position is always the user's current location).

The navigation wizard will then handle all the cumbersome details for the user, such as moving along the corridor, finding the stairs, climbing up the stairs, and moving along the upper floor's corridor until the destination point. The user will be automatically taken along this path, thus getting visual information on the relative location of the elements he passes by, which will enable him to build an accurate spatial model of the elements' locations – route knowledge.

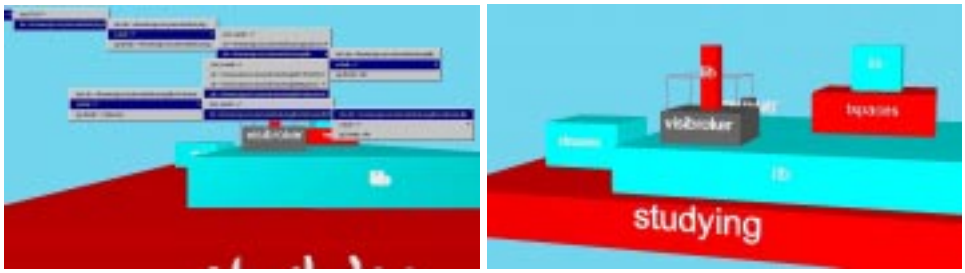
### 6.11.2 Navigating in the pyramids metaphor

As we have seen in Section 6.11.1, path navigation plays a very important role in the virtual-building metaphor. The user is tied to his real-world navigational schemata and can easily relate to the same kind of navigation in a virtual building. However, for an abstract metaphor, where there may be no real-world *a priori* knowledge of how to navigate, the user can still take advantage of our navigation system.

The system allows for an easy way of navigating through metaphors that do not possess an intrinsic real-world based navigational schemata, by

providing means for traversing and exploring the whole service model or metaphor tree. This can be done via absolute or relative navigation, as the navigation wizard always provides feedback for the user's current object of interest. The navigation feedback mechanism, fostering user's awareness of the current target, is given by a red bounding box drawn around the current object of interest.

Let us use the example of the pyramids metaphor for exploring a file system, already presented in Chapter 5. While exploring the virtual world, the user may either navigate by his own means (i.e. means that the 3D viewer provides), or he may choose to take advantage of CyberNet's navigation assistance.

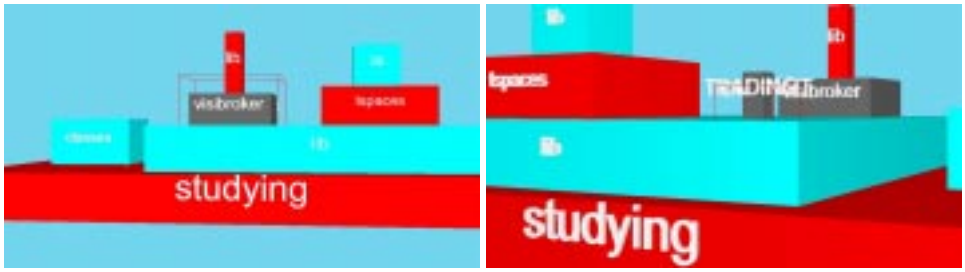



(a) Choosing an absolute target (b) Resulting destination

Figure 6.7: Absolute navigation with location feedback (bounding box) in the pyramid metaphor – file system visualization tool.

As illustrated in Figure 6.7, the user can choose a target in the absolute navigation mode – i.e., the target destination is chosen independently of the current user's location. The destination is specified either by selecting an object directly in the world, or by using a menu that unfolds the hierarchy of the service – Figure 6.7 (a). The user is then automatically moved by the navigation system to the final destination – Figure 6.7 (b). The visual feedback of the current object of interest is given by a red bounding box that encircles the object.

On the other hand, the user may also choose to navigate relatively to his current position (relative navigation). For this, the user has a 2D interface that allows him to traverse the service tree. Figure 6.8 shows the result of relative movements `up` and `next`, respectively on the left and right, starting from the user's location depicted in Figure 6.7 on the right.




(a) Moving `up` in the service (b) And moving `next`

Figure 6.8: Relative navigation in the pyramid metaphor – file system visualization tool.

### 6.11.3 Discussion

We have presented two distinct examples that illustrate some of the capabilities of our navigation system. Although path navigation is very useful, its major utility is for navigating in real-world based metaphoric worlds – such as the city metaphor (Figure 6.3) or the virtual building metaphor (Figure 6.1. For more abstract metaphors, for which the user does not have an *a priori* navigational schemata knowledge, the path navigation is eventually not the best solution, this is notwithstanding the fact that they may also be real-world based (e.g. the pyramid metaphor). There is no predefined way of navigating a pyramid, for instance, that the user can relate to.

Hence, for the metaphoric worlds that do not have an intrinsic real-world based way of navigating, we feel that the user can rather take advantage of our navigation aids by using a combination of absolute and relative navigation, to explore the world and traverse the whole service model. The path navigation is substantially less intuitive for these cases.

Two additional notes should be made, though. The first is that, for either case, CyberNet's navigation system complies with our metaphor-aware navigation concept – i. e., there is a clear dependence of the way the user navigates in relation to the metaphor. The second observation, that has already been made for the user's navigation desiderata (Section 6.4.1), is that a metaphoric world which relies heavily on its real-world counterpart should not limit or constrain the navigation, at least not entirely. In other words, user navigation must take advantage of the potentialities due to the fact that the world is a virtual one, and need not always be constrained by the feasibility of the same kind of navigation in the real-world.

## 6.12 Conclusion

In this chapter we have presented the fundamentals of the CyberNet project's navigation system. We have stated our views regarding enhancing and making it easier to navigate in a 3D virtual world. Our major contribution is the metaphor-aware navigation concept and the fact that we are able to implement this in an automatic manner.

The metaphor-aware navigation concept is basically stating the principle that the way we navigate in a given virtual world depends on the characteristics of said world, namely on the metaphor employed to depict the information. In other words, the solution to an easier 3D navigation does not lie in one generic way of navigating that is applicable and appropriate for every single case. Furthermore, in the same world, navigation may also take different modes, depending on the hierarchical level of the metaphor in which the user is moving; e.g., flying at district level and walking at street level in a metaphoric city (Figure 6.3).

The task-dependent navigation was also approached. If the user is doing some monitoring work, he probably wants to navigate in the world in a serendipitous mode, just making sure that he passes by all the devices that must be monitored, in a kind of a journey. On the other hand, if he wants to do some specific management task, he probably wants to get to the final destination by taking the fastest/shortest path to get there. We thus have different navigation modes available that take into account these differences.

In order to implement automatic navigation mechanisms that follow those requirements, we have described how we embed the navigation capabilities in the individual graphical elements. We have also described how we delegate simple navigation tasks to the graphical elements and make them responsible for all the navigation that must occur within their "turf" (usually defined by a graphical object's bounding box). This avoids implementing a large centralized algorithm, enables scalable growth at run time and speeds up the navigation.

We have shown, based on the concept of neighboring tables and routing tables, how we are able to determine the metaphoric path that is the support of the path-based navigation. We have also stated the role played by the navigation wizard, specially regarding keeping updated information on the user's state.

We have further described how, by tightly binding the service model to the metaphor hierarchy, we are able to coherently combine physical and semantic navigation. In this way, we provide the user with a powerful way of exploring the 3D data visualizations by exploiting either the metaphor

tree (physical navigation), or the service tree (semantic navigation). The fact that both ways of navigation are combined and available in a single visualization makes CyberNet's navigation mechanism a very useful resource for data exploration and information visualization.

# Chapter 7

# Dynamics

> Visualization links the two most powerful information processing systems known – the human mind and the modern computer. (Gershon and Eick, 1997)

## 7.1 Introduction

The demand for the visualization of large volumes of information has been growing at a fast pace in recent years. This is due to various factors: the growing availability of information everywhere within a mouse click and to a such degree that a new term was coined – *information overload* – and the fact that the computing and display power are not hindering anymore, are some of them.

Furthermore, nowadays there is also demand for information visualization systems that are able to supervise large amounts of *dynamic* information such as stock exchange data (Delaney, 1999), for example. As we have seen in Chapter 3 the broad goal of the CyberNet project (CyberNet, 2001) is to deliver an interactive 3D dynamic information visualization tool designed to evaluate the added value that this technology brings to the visualization of large volumes of dynamic information. We now provide explicit descriptions of the mechanisms that we have developed to deal with the visualization of dynamic data.

Specific problems arise when dealing with dynamic data visualization in real time. The fact that the data is dynamic has a traversal impact on every step of the visualization. It influences data collection, structuring, mapping, and visualization. Dynamic data also imposes additional requirements on user navigation and interaction. This chapter presents the mechanisms that have been developed to deal with data dynamics in CyberNet.

### 7.1.1 Motivation and problem statement

The motivation to study mechanisms to deal with the visualization of dynamic data came naturally from the first application domain for the CyberNet project: network monitoring and management. Network data is highly dynamic and changes must be seen in real time, in order to detect eventual problems. Since we developed our own collecting mechanisms for collecting data in real time, it only made sense to research ways for also visualizing it in real time.

Visualizing dynamic data in real time, however, poses several problems. Some of them are: how to make sure that relevant data changes do not pass unnoticed, while irrelevant ones do not waste processing time? How to maintain a stable and coherent virtual world regardless of the visual changes necessary to keep up with the data changes? How to reduce the visual impact of changes in order not to confuse the user? These and other problems caused by visualizing dynamic data in real time are attacked by different dynamic handling mechanisms in our system.

### 7.1.2 Chapter organization

This chapter is organized as follows: Section 7.2 briefly presents some related work in the area of visualization of dynamic data. Section 7.3 our approach to the problem of real-time dynamic data visualization. The next section, Section 7.4 presents the mechanisms we have developed to deal with dynamic data collection, and Section 7.5 presents the mechanisms related to handling the dynamic data model. In Section 7.6 we introduce the impact of data dynamics in the visualization model and in Sections 7.7, 7.8, and 7.9 we describe and discuss the techniques and strategies we used to deal with that impact. Section 7.10 relates dynamics to navigation and Section 7.11 relates dynamics to interaction. Finally, in Section 7.12 we draw some conclusions.

## 7.2 Prior work

Lots of research effort has already been put in the field of scientific visualization for the representation of dynamic data. The applications are very diverse and comprise different application fields such as geographic information systems (Kanellopoulos et al., 2001), the offshore industry (Chapman et al., 2001) or medicine (Chapuy et al., 2000). However, even in scientific visualization, few applications deal with real-time visualization, such as (Chapuy et al., 2000).

In information visualization, the state of the art is quite similar, in the sense that there are very few applications that deal with the visualization of large volumes of dynamic data in real-time, such as (Delaney, 1999). Most of the related work on the field of dynamic data visualization deals with recorded data and *a posteriori* analysis as, for instance (Wegenkittl et al., 1997b) (Wegenkittl et al., 1997a). Additionally, frequently the term dynamic visualization means that the visualizations are interactive (Rheingans, 2002) responding, for instance, to user input and not that the data itself is dynamic.

## 7.3 Our approach

The impact of data dynamics affects the entire visualization system: it affects data collection, the information structuring, the visual mapping, the visualization, and also the navigation and interaction.

Our approach was to develop separate mechanisms to deal with the different effects of data dynamics. For data collection we have implemented a data pushing mechanism that reduces interlayer data traffic, while making sure that important data changes do not unnoticed. The information structuring relies heavily upon the services of an entity repository, that keeps track of the entities' different life cycles. For the mapping, we have derived some mapping rules to deal with data dynamics and we also count on adaptors (Section 4.8.6) reutilization for mapping new entities.

As a step further ahead for the mapping, the actual data representation we have designed has different propagation strategies for dealing with the impact of updating the virtual world according to the data changes. We also designed a novel approach by mixing propagation strategies according to direction. The problem of navigating in a dynamic world has been dealt with by constantly updating user's position and direction of view in order to take into account the updated world layout. Finally, we have designed a quite simple algorithm to deal with world updating and a process for adding persistency to selection.

## 7.4 Collecting dynamic data

We focus this chapter on the visualization of dynamic data. Dynamic data is data that varies in time. Data may vary in amount (new data may appear and old data may disappear) as well as in value (the value of existing data may vary in time). In addition to the time dependency, a fundamental point is that the system must process data on-line and, thus, is not aware of the

modifications before they happen. As we have seen from Chapter 3, for the data collection we have developed collecting agents that we have called *informators*.

Regarding data collection, the most important mechanism we use to deal with data dynamics is the choice of data transfer method, from the collecting layer to the structuring layer. As already referred to in Section 3.4.1, instead of data pulling which is more common in traditional network monitoring systems, we use a data pushing method. This means that instead of having to poll the devices, we rely on the informators to push the data whenever there is a change that is deemed important. An important design question remains: who defines what is important and what is not?

We have defined data pushing policies that judge of the importance of the data change and then the informators act accordingly: either by pushing the new data or by being oblivious to the change. These data pushing policies use criteria based on data changes and/or on some imposed time scheduling – and may be more or less refined according to the desired system's behavior.

Some examples of data pushing policy criteria are:

- data value changed
- data value changed above a given percentage
- every two seconds
- every two seconds and data has changed

These criteria combined with the data-pushing technique ensure that no data change thought as important passes unnoticed, while keeping the interlayer traffic load to a minimum.

## 7.5 Dynamic data model

In the next section we will explain how we take advantage of an entities repository with a persistent query capability to manage the data structuring.

### 7.5.1 How to handle data dynamics?

As we have seen from Chapter 3, online processing of dynamic data requires translating the dynamic data into visual elements. This translation requires several steps.

The first step toward this process is to logically structure the data in order to obtain a data model – this step is called *data transformations* according to the terminology of (Card et al., 1999). Basically, structuring the data requires to group data according to common properties and to identify relationships. In our structuring process the final data model is a tree (Section 3.6).

The data model is also used to handle the dynamic nature of the data: each time new data is created or existing data is modified, the system will automatically update the data model to reflect this modification. Because the data model is automatically translated into a visual representation, the modification will be visualized.

The data model we have developed is based on the concept of *entities* (Section 3.6.2). Entities are used to group all the values that are necessary to describe some logical element of the data model. An entity can represent a physical device (e.g., a router, a hub) or a conceptual item (e.g., a process, a file-handle). The entities are created by an entity collecting process described in (Abel et al., 2000).

Since we are dealing with a dynamic environment, entities have a life cycle: new entities may be created, existing entities may disappear, and the values (attributes) of the entity may vary. In order to cope with this dynamics all the entities are stored in an *entity repository*. The role of the repository is to keep track of all the existing entities and to be able to answer to queries concerning entities (i.e., all the hubs that have an IP address in a given range).

The difference between the repository and traditional database management systems is that the queries are persistent. When an object makes a query to the repository, all the entities that match the query are returned to the caller; if, later on, a new entity that matches the query is created, the caller will be notified. This *persistent query* capability is the basis of the dynamic data model, and the entity repository plays a fundamental role.

The data model tree is composed of *service nodes* (Section 3.6.2). A service node is an object that knows how to interpret part of the data model. The service node acts as follows: the service node issues a persistent query to the entity repository, the repository returns what we call a *relation* (Section 3.6.2) which groups references to all the entities that satisfy the query. The service node adds that relation as one of its children, hence constructing a tree-like structure.

Figure 7.1 depicts the mechanism of creating a new service node when a new entity is created.

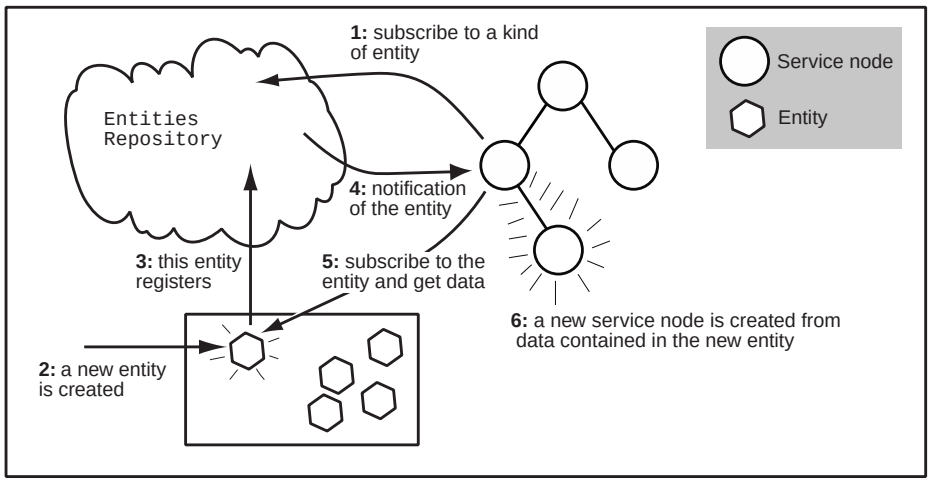



Figure 7.1: Dynamic creation of a new service node in the model tree when a new entity is created.

### 7.5.2 Slaving the visualization to the dynamic data model

Using the previous mechanisms, the data model is always reflecting a hierarchical structure based on the existing entities. This model must be translated into a 3D world. This translation is called the *mapping process* and was discussed in Chapter 4.

The visualization is slaved to the dynamic data model and the system will continuously update the world in order to present an up-to-date visualization of the data model. We take advantage of the hierarchical structure of the data model that matches quite naturally the hierarchical representations of 3D scenes generally supported by most 3D-scene description libraries.

## 7.6 Dynamic visualization model

Our visualization model is based on the concept of visual metaphors and may described by an acyclic directed graph of layout managers and 3D glyphs (Section 3.7). Since the layout managers are responsible for space allocation, they play an important role in managing dynamics.

### 7.6.1 Mapping dynamics

In order to dynamically map new arriving data, our system takes advantage of the adaptors (Section 4.8.6). The adaptors are typed – i.e., for each entity there is an adaptor. Every time a new entity is created, it is mapped by

invoking the corresponding adaptor. This also applies for already mapped entities, in the case of a value's change: the entity adaptor is called and the new value for the corresponding visual parameter is computed. This new value is then propagated to the metaphoric world, where its impact will be handled.

### 7.6.2 Impact of data dynamics

While the system automatically modifies the 3D world, it must take care of global coherency in the world. For example, when an object is growing in size, the system must verify that its geometry does not intersect other objects in the world. If it detects a potential intersection, the system should be able to modify the neighboring objects (for instance it can slightly push them apart in order to make some room for the growing object). Thus, the modification of one object may have impact on its neighborhood, and the modified neighborhood may require the modification of other objects. This phenomenon is called *modification propagation*. Several types of causes lead to this propagation, namely:

- the size of a child is modified
- a new child is created
- an existing child is removed

Managing the modification propagation is an important topic directly related to dynamic layout management. The traditional way to handle static layout management is to use global layout constrained optimization algorithms. However, because of performance problems, dynamic management requires avoiding centralized solutions since global optimization algorithms cannot be re-evaluated each time a small modification arises in the world. We chose to develop a distributed propagation mechanism: in our system each layout manager (Section 3.7.3) has the responsibility of modifying itself and its direct children layout. The next section will discuss the constraint propagation mechanism in detail.

## 7.7 Constraints propagation strategy

Potentially, each data value change may provoke a modification that must be propagated across the world. This process should be as least disruptive as possible and should not bring instability to the virtual world. There are two

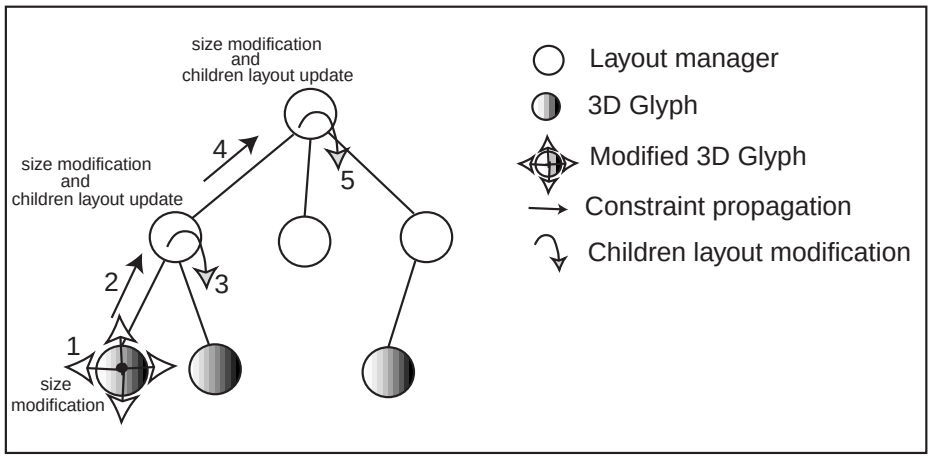



Figure 7.2: Bottom-up constraint propagation.

basic strategies for propagating the modifications: *bottom-up* and *top-down*. However, neither strategy in isolation is sufficient to handle all cases.

### 7.7.1 Bottom-up propagation

The first solution to handle constraint propagation is to use a bottom-up approach. The main idea is to let the children influence the upper part of the hierarchy. For instance, if the size of a 3D Glyph (Section 3.7.3) increases, the space allocated by its parent layout manager (LM) to the 3D Glyph (3DG) should be modified accordingly. Thus, the parent LM should also modify its size and its own parent must take into account this modification. The modifications are propagated to the upper part of the hierarchy until the root (Figure 7.2).

An important consequence of the use of this propagation mode is that the overall world structure may be modified as soon as a leaf node is modified. This is due to the fact that when a child modifies its size, the information is transmitted to its father that in most cases has to modify the position of all its children according to its layout policy. In many cases, this global propagation mechanism can produce globally unstable 3D worlds.

### 7.7.2 Top-down propagation

The second solution to handle constraint propagation is to use a top-down approach. The advantage of this approach is to limit the influence of modifications. In this case, it is the role of the parent to allocate space for its

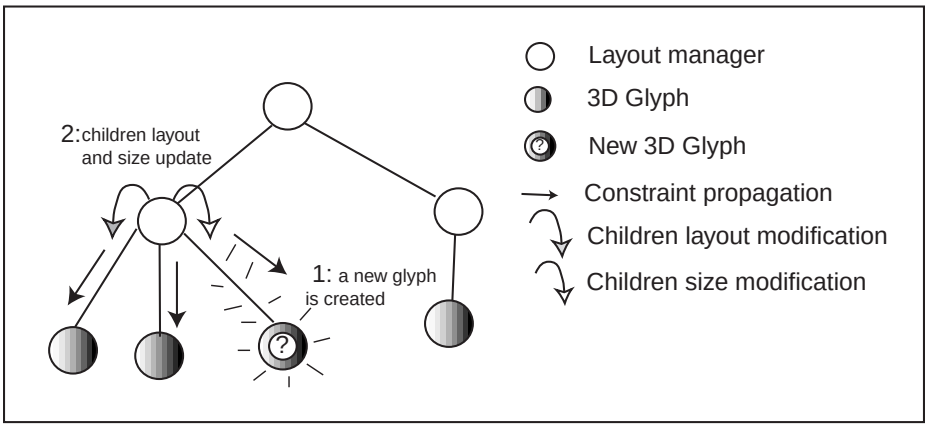



Figure 7.3: Top-Down constraint propagation.

children and to control that they actually fit in the region defined. Since it is the parent that determines the size of its children, the propagation of the space constraints is going from the root to the leaves (Figure 7.3). Each child should be able to fit itself into the region of space that its parent has reserved for it. When a child is not able to do so, it is the role of its parent to modify its size in order to fit the child to the region.

The main problem of this propagation mechanism is that the world loses size coherency: since it is the parent that defines the space allocated to its children, two conceptually identical leaf nodes may differ in size and thus cannot be compared by the user. Another related problem is that when a part of the world is growing in number of elements, each element is generally decreasing in size in order to keep the aggregate size constant. These local modifications (as opposed to the global modifications generated by the previous strategy) can produce local instability in the 3D worlds.

## 7.8 Directional constraints

In the presentation of the two previous constraint strategies, we did not make any references to the constraints themselves or to the scheme used to define the regions of space. Our basic statement is that the volume managed by a parent should include all the volumes managed by (or allocated to) its children. How these volumes are defined is a matter of implementation. However, why should the propagation be identical according to every direction in space?

A good example of non-uniform propagation can be found in the city

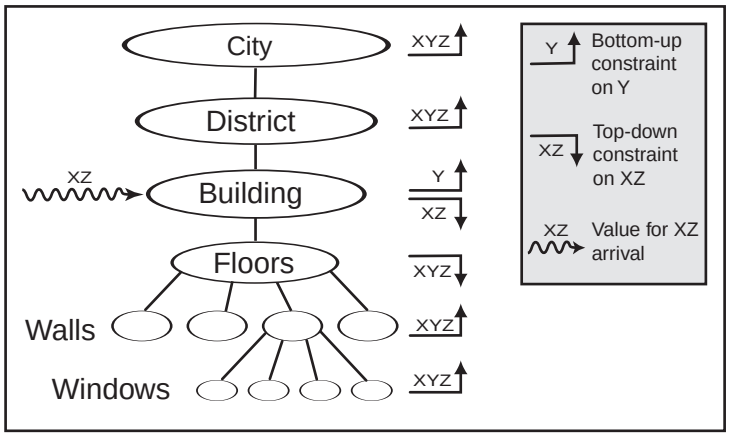



Figure 7.4: Constraints in a city metaphor.

metaphor implemented in the CyberNet project. In this metaphor, a district is defined as a LM that places buildings in space. This can be implemented either by using a top-down or a bottom-up propagation strategy.

The top-down strategy will fix the size of districts (once created a district will not be able to grow in size, and adding new buildings will be done by resizing down existing ones). The bottom-up strategy will conserve building size and allow district to grow (when a district grows, the neighboring districts will be pushed apart in order to make room).

In the first case, even if the districts are limited in the ground plane, there is no reason to limit the buildings in height. Hence, the district LM is designed to apply a top-down constraint to the buildings in the ground plane and leave freedom to the height direction by using a bottom-up strategy for that direction (Figure 7.4).

In other words, buildings have a defined ground size (`X` and `Z`), but they are free to grow in height (`Y`) according to the number and size of the floors. Thus, each floor is constrained to have the same (`X` and `Z`) size as the building.

The walls are constrained in each direction because they should exactly match the floor size. They are composed of windows and the number of windows is used to represent some information. Because the walls size is constrained and since the number of windows may be variable, the size of the windows is not relevant (i.e., cannot be used to map information) although their relative size at a given floor may be.

Our implementation of the city metaphor is thus an example of a successful combination of different constraint-propagation strategies according to the direction in space.

## 7.9 Mixing strategies

The design of complex 3D virtual worlds requires being able to use both constraint propagation strategies within the same metaphor hierarchy, as we have seen above. By mixing both constraint strategies, several propagation configurations may arise – Table 7.1 depicts the different potential configurations.

| **Parent** / **Children** | **Top-Down** | **Bottom-Up** |
|---|---|---|
| **Top-Down** | The parent allocates space for its children – the children respect the allocated size and constrain their own children. | The children decide their own size (they constrain their own children)– the parent allocates space accordingly. |
| **Bottom-Up** | The parent allocates space for its children – the children decide their own size. The potential conflict is solved by the parent using a scale operation to constrain the size of its children. | The children decide their own size (they compute their size from their own children) – the parent allocates space accordingly. |

Table 7.1: Mixing propagation

### 7.9.1 Discussion

We have shown how we are able to combine both top-down and bottom-up constraints propagation strategies according to the direction in space. This combination is specially useful for the case of metaphoric worlds since these do not evolve in the same way for all directions of space. The definition

of what strategy applies at a certain point of the metaphor hierarchy is, naturally, a choice of the metaphor's designer.

We believe, however, that there is no ideal solution to manage real-time dynamics propagation in a virtual world. The fact that the information is changing over time may always cause instability in the virtual worlds, since there is no *a priori* knowledge regarding the value of these changes or when they may occur. Our strategies, nevertheless, minor the possibility that such an instability actually happens.

A further note is worth referring. The fact that in our system we visualize dynamic data in real time represents a step further in comparison to prior related work that also addressed the visualization of dynamic data but within a *a posteriori* analysis context (Zhou and Feiner, 1996) (Wegenkittl et al., 1997a) – i.e., by replaying animations of recorded data. In these cases the data changes are known in advance and allowances can be made for the virtual world to reflect the information changes and propagate the modifications in a perfectly stable manner.

## 7.10 Navigating in a dynamic world

In the CyberNet visualization framework, all metaphoric worlds are constructed using *metaphoric components* (Section 3.7). Metaphoric components (MCs) have mechanisms that help the user to navigate logically inside the metaphor. The goal is that the user navigates in the world with the mechanism most suited to the metaphor itself. We call this principle *metaphor-aware navigation* (Russo Dos Santos et al., 2000).

In order to assist the user in his navigation task, the system maintains three parameters regarding the user: the *user's current location* in the 3D world (the closest object to the user – the user is always associated with a MC), his *current node of interest* (the node that currently has his attention) and a *new node of interest* (where the navigation mechanism is requested to take the user to). When the user is simply moving around in the world, the user's current object (current location) and his current object of interest are identical. Even so, some navigation tasks require the user to be located in a place and to have his center of interest on another object (an example is the *look at* navigation mode).

The difficult part is to know what is the movement that the system must apply to the user when the world evolves. If the current object and the object of interest are identical, the simplest solution is to move the user move in the same way that his current object is moving. This strategy produces good results but may not be sufficient when the current object and the object of

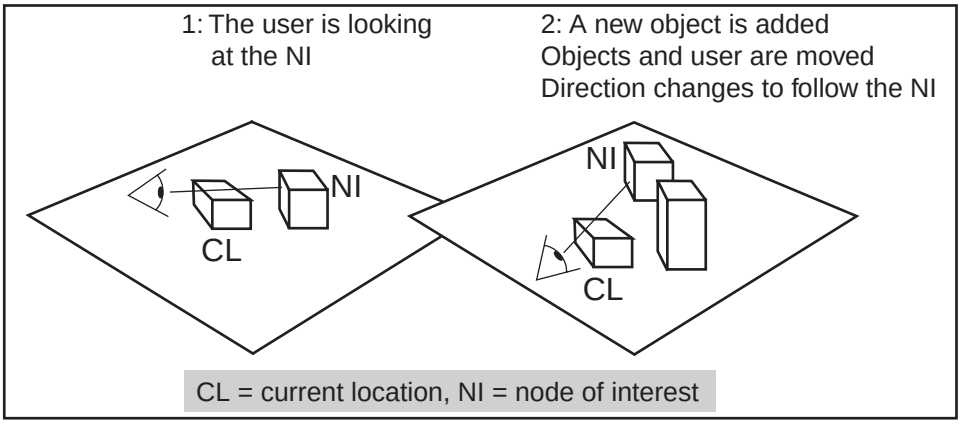



Figure 7.5: Automatic movement of the user when objects are moving.

interest differ, since the movement of the object of interest and the movement of the current object may also differ. In this case, for instance for the *look at* navigation mode, the direction the user is looking at must also be changed towards the new position of the object of interest (Figure 7.5).

There are still occlusion problems that may occur, for instance, when the new object that is created occludes the current object of interest. The solution is to move the user to a new location from where the occlusion does not occur.

## 7.11 Dynamics and interaction

The changes in the world due to the dynamic data may not be previewed – both in time and in nature – since we are dealing with real-time visualization. In order to minimize their disruptive effects, smooth transitions must be presented to the user. Furthermore, the unpredictable nature of the changes renders interaction more complex. For instance, the selection mechanism must possess some kind of "memory" so that a newly created sibling of a selected object will automatically appear as selected also.

### 7.11.1 Movement animation

Movements of objects in the world must be presented in such a way that they do not confuse or disorient the user. A well-known technique to try overcoming this problem is to animate the movement of objects in the visual representation in order to make the visual representation evolve over time in

a way that is not disruptive to the user (Robertson et al., 1991). Moreover, moving the viewpoint while animating the data is difficult to interpret (Marshall and Carswell, 1989).

The evolution in time is usually slowed down to make the changes in the world appear as a smooth transition. In other words, objects do not appear/disappear instantly, nor are they moved around simultaneously. We have designed a four-step algorithm for a transition from a current state to a following state. First, all deleted objects are removed. Second, remaining objects are moved to their new position. Third, all resize operations are done. Finally, new objects are added to the world. This animation is also used to move the user as the world evolves.

### 7.11.2 Dynamic selection

In addition to the built-in navigation capabilities, the metaphoric components also provide some interaction features. We found it important to be able to interact with the world in order to modify its appearance. However, the dynamic nature of the world implies that the current appearance parameters must be saved in order to be later applied to the new objects that may be created. If newly created objects do not inherit those parameters, the world can lose its coherency.

One of most common ways of interacting with a visualization is the selection mechanism (Section 5.6). The user may select objects to cluster, to render transparent or to isolate from other objects in the world, for instance. As the world evolves the selection also evolves. For instance, if the user selects all the elements of a district in a city metaphor, this selection must be "memorized" so that, for instance, each new building added to this district is automatically displayed as selected by the visualization model. In the case of selection across multiple views, this selection "memory" must be retro-propagated all the way back to the entities repository (Section 5.6.2).

## 7.12 Conclusion

In this chapter, we have seen that visualizing dynamic information in real-time leads to some requirements that do not exist in the case of static information or off-line visualization. These new requirements traverse the three layers of our visualization system's framework: the collecting, the structuring, and the visualization layer.

Regarding data collection, we have presented a data-collecting strategy that seeks to minimize the interlayer traffic. For managing the dynamic data

model we have described our use of an entities' repository to keep track of the data changes. We adopted a visualization model slaved to the data model, so that data changes have an immediate repercussion in the metaphoric world.

For the impact of data dynamics on the visualization layer we have discussed two different constraints' propagation strategies, we have introduced the concept of directional constraints, and we have shown how we can combine constraint strategies in order to build coherent complex worlds. We have also attacked the problem of navigating in and interacting with such a dynamic world.

To our knowledge, there is no ideal solution to overcome the problem of visualizing a world that evolves. However, a well-designed metaphor with special emphasis on the layout management constraints helps the visualization to be more stable, and thus more comprehensible and insightful.

# Chapter 8

# Conclusions and Future Work

> Our technological culture is drowning in its own success – masses of data and information are accumulating everywhere. As time goes on, it is apparent that these problems are only likely to become much worse, not better – unless, of course, the entire system somehow collapses of its own weight. (West, 1998)

## 8.1 Conclusions

This dissertation examines 3D metaphoric representations for visualizing large volumes of abstract data. Our society has become increasingly rich in information and new analysis tools are demanded that enable users to make sense of it. Visualization goes beyond using computer graphics to communicate information; visualization enables us to think. This dissertation has studied how to improve visualizations in order to enable users to think clearly, with insight, and with conviction to act.

The work done in this thesis focused four main challenges in 3D metaphoric information visualization: mapping the information, visualizing it, navigating the representations, and managing dynamics. For each of these challenges we have examined the problem, analyzed related work, proposed and designed our own solutions, and discussed the rationale and experimental basis for our approach.

Before attacking the first challenge we presented a flexible framework that takes advantage of virtual worlds interaction capabilities for displaying large volumes of multidimensional information. We have designed a system that is able to automatically build virtual worlds for information display. The system is able to update data modifications in real-time, without requiring user intervention. This visualization system was used as both development

framework and test bed for our proposed solutions.

A key research problem in visualization of abstract information, since it has no natural and obvious physical representation, is to discover visual mappings for representing the information and to understand what analysis tasks they support. In Chapter 4 we attacked this problem. We presented a strategy and framework for automatically mapping abstract information onto 3D virtual worlds.

We analyzed and characterized abstract data in order to extract the relevant features for the mapping process. We also analyzed the influence of and characterized the user's task in the visual mapping. We presented our motivation to use visual metaphors for representing the information and discussed the pros and cons of this approach.

The visual metaphors play a very important role in the visualizations as they provide the media for conveying the information. We did a detailed characterization of the metaphors and their components regarding information mapping. We determined to what constraints they must honor and presented a mapping desiderata concerning the metaphoric components.

In possession of characterizations for the three major components in the visual mapping – data, tasks, and metaphors – we proceeded to present our mapping engine.

Our mapping approach is founded on the separation of two kinds of mapping: the mapping of the information structure and the mapping of the data values themselves. To implement this approach we defined two types of mapping – structural and visual parameters mapping, respectively – and established criteria applicable to both types.

We fully described the mapping process in several steps and we presented our classification of visual parameters according to the data types described earlier in the chapter. We developed special non-graphical elements, called adaptors, that implement the mapping strategy and rules. We concluded Chapter 4 by giving several examples of implemented visual mappings that illustrate and validate our strategy.

The fact that the information mapping is done automatically by our system allows us to create different visualizations on the fly. Furthermore, since our mapping strategy is based on metadata that characterizes the actual data, our mapping engine is essentially application-domain independent. We take advantage of both features to develop multiple views for different applications and research their usefulness in information visualization. This work is presented in Chapter 5.

Having different perspectives on a subject is considered as fundamentally important for making judicious decisions. In information visualization, mul-

tiple views allow the user to have different perspectives of the information. However, multiple views are not achieved without cost. In Chapter 5 we discussed the motivation for using multiple views.

We briefly described all the metaphors that we implemented so far and that are used by our different visualization tools. When appropriate we have suggested ways of improving our current implementation of the metaphors, as some of them are still quite simplistic.

The use of multiple views was demonstrated in a number of different visualization tools. We implemented these tools for different application domains, which agrees with the application domain independence that characterizes most of our work. We validated the multiple views in a number of distinct application scenarios, and provided a short critical evaluation of their benefits.

User interaction is one of the most valuable and powerful assets of information visualization. In Chapter 5 we discussed interaction mechanisms, especially user selection. We established two concepts of selection: metaphor based and service model based selection. User selection across different worlds was tackled by a multi-world selection mechanism, that propagates the selection to the data repository level. This fosters mixed criteria selection. We concluded Chapter 5 by giving other interaction mechanisms examples that we have implemented, such as the horizontal slide.

In order to gain insight into the data and fully interact with the visualizations, the user must be able to navigate the metaphoric virtual worlds. We believe that the default navigation mechanisms available in current 3D viewers are too generic for many applications. Also, navigation is sometimes referred to as one of the weakest links in 3D virtual worlds. We attacked the problem of user navigation in 3D metaphoric worlds in Chapter 6.

Our visual representation is based on the use of metaphors. In accordance with this we created a novel concept of metaphor-aware navigation. The metaphor-aware basic statement is that user navigation in a virtual world should not be oblivious of the metaphor used for the visualization. We believe that we can promote easier 3D navigation by developing navigation mechanisms that take into account the metaphor. We also discussed the need and presented the motivation for task-dependent navigation.

In order to implement the mechanisms described above, we developed a helped navigation system that uses three main inputs: the user's current location, the user's current node of interest, and a target node. We designed and implemented mechanisms for easing and automating the target selection. We also implemented different assisted-movement modes to move the user or his direction of view – depending on the user's navigation mode – from

his current location to the selected target.

We encourage a combination of physical (metaphoric) and semantic navigation by allowing the user to explore both the metaphor hierarchy and the information structure. This combination constitutes a powerful way for exploring data and examining relations. This combination of metaphoric and semantic navigation is supported by two navigation modes that we further designed and implemented: relative and absolute navigation. In the first case the user navigates relatively to his current location; in the latter, the target destination is chosen independently of current location.

Our navigation system relies on a navigation wizard that keeps track of the user's current state defined by the user's current location, current node of interest, and next node of interest. We used a embedded distributed navigation strategy that delegates simple navigation tasks on the metaphoric elements. Each metaphoric element with navigation capabilities is responsible for the user's movements within its bounding box and must provide a neighboring table of other metaphoric elements. The navigation wizard uses metaphor-dependent routing tables and the elements' neighboring tables, to guide the user along a metaphor-aware path.

The embedded distributed navigation strategy in conjunction with the metaphor-dependent routing tables allows for an automatic implementation of the metaphor-aware navigation concept, regardless of a particular metaphor. This makes our navigation scheme portable to other metaphors with minor overhead – only the routing tables, that are naturally dependent on the metaphor, have to be designed for each metaphor. We provided examples to illustrate the concepts of relative and absolute navigation and metaphor-aware navigation.

As technology evolves, new horizons for information visualization are coming within reach. The real-time visualization of large volumes of dynamic information is an example. However, dynamic information brings additional constraints to the visualization system. Moreover, if the visualization is done in real time, the data changes are not known in advance. In Chapter 7 we attacked the subject of visualizing dynamic data in real-time.

We examined the impact of dynamic data on the real-time visualization system: it is traversal to the entire visualization process. It affects data collection, data structuring, data mapping, data visualization, and user navigation and interaction. For each one of these subject matters we analyzed the requirements, proposing and designing solutions for managing dynamics.

We devised a data-collecting strategy that is based on data-pushing policies and uses different pushing criteria to minimize the data transfer while ensuring that no data change considered relevant is overlooked. For man-

aging dynamics in the data model we used a repository that keeps track of all the data entities. We showed how the system is able to always keep an updated image of the current status by taking advantage of the persistent queries capability of the repository.

The impact of data dynamics in the visualization was attacked with more detail. We devised two constraint-propagation strategies to minimize the disruptive effects that changes may have on the metaphoric's world stability, and then we discussed their shortcomings. Since our representations are metaphoric, the virtual-world layouts obey not only space constraints, but also metaphor constraints, in order that its coherence and recognizability is not endangered. This led us to develop different constraints-propagation strategies according to scene changes, in order to cope with the fact that metaphoric worlds do not evolve in the same way for each direction in space. The combination of both strategies constitutes a powerful tool to build and maintain complex worlds. The navigation and interaction in dynamics worlds are also attacked in Chapter 7 and we proposed some solutions for both cases.

## 8.2 Future work

### User testing

Information visualization is still an emerging discipline. Its evolution is tracking the networking and computing revolution. Some applications are still designed as a craft, using mostly heuristics. The power of visualization resides in exploiting human perception. At the same time this also constitutes a challenge, since existing knowledge about human perception and visualization design is still insufficient. Understanding the cognitive impact of visualization is fundamental.

One direction for further work that should be explored and that addresses the relationship between user's cognition and perceptual stimuli is user testing. Preliminary feedback from users indicates that 3D metaphoric visualization is promising. It eases problem detection and understanding. However, thorough user testing, with perceptual evaluation, is desirable in order to further validate and improve our visualizations.

### Information workspace

Another direction for further work is to build an integrated information workspace. This workspace would use our previous work on multiple tools and views to build an integrated information visualization system. The tools

would all be linked in a single system so that the user might navigate seamlessly between the various tools and views. The metaphor-aware navigation mechanism can also be adapted to work in an integrated information workspace. User navigation might be automatically switched according to the metaphor, as the user switches between different views or tools.

Interaction is fundamental for gaining insight into information. Up until now the interaction mechanisms in our system are fairly simple and mostly restricted to selection. Further development of the interaction mechanisms is thus desirable – we might also envisage using such mechanisms for reacting and to interact with the real world (e.g., kill a process).

### New metaphors and refined mapping

New and more creative ways for visualizing information can be envisaged as another further research direction. This would probably require multidisciplinary collaboration, to find original and inventive visual metaphors for representing abstract information. This might entail an evolution in the mapping process, as a more refined metaphor characterization will eventually be required.

Additionally, the mapping should be tested for different application domains. So far, all the applications targeted a computer science domain, with an emphasis on networking. The whole system, and the mapping process in particular, needs to be applied to completely different application domains (such as financial data for instance). It is reasonable to expect that new requirements (e.g., regarding data characterization) will appear.

The use of XML (eXtensible Markup Language) for the mapping process is also envisaged as object of further study. There is a need for consistent interfaces and web accessibility. This implies the need to establish implementable design patterns for information visualization, and also the broad need for interoperable standards for web-based 3D graphics such as X3D (eXtensible 3D).

### Scalability

Finally, further work should be done regarding how the system scales up to the visualization of very large volumes of highly dynamic information in real-time. An added value would be to add persistency to our system. This can allow to record, playback and analyze off-line a crucial sequence of events. Test-case scenarios might also be envisaged for experimental testing and improvement, in order to accomplish a robust prototype of our current framework.

# Bibliography


3DTF (2002). Images of the 3dtf vr environment.
<http://www.asymptote.net/3DTFV.htm>.

Abel, P. (2001). *Supervision d'informations dynamiques et distribuées à l'aide de mondes 3D interactifs : Application à la gestion de réseaux.* PhD thesis, EPFL – Swiss Federal Institute of Technology of Lausanne.

Abel, P., Gros, P., Russo Dos Santos, C., Loisel, D., and Paris, J.-P. (2000). Automatic construction of dynamic 3D metaphoric worlds: An application to network management. In Erbacher, R., Chen, P., Roberts, J., and Wittenbink, C., editors, *Visual Data Exploration and Analysis VII*, volume 3960, pages 312–323, San Jose, USA. SPIE - The International Society for Optical Engineering.

AdventNet (2002). AdventNet, inc.
<http://www.adventnet.com/>.

Alexander, C. (1982). *Humanscape – Environments for People*, chapter A City is Not a Tree, pages 377–402. Ulrich's Books Inc.

Andrews, K. (1998). Visualizing rich, structured hypermedia. *IEEE Computer Graphics and Application*, 18(4):40–42.

Andrews, K. (2000). Information visualization – lecture notes (draft version). IICM / Graz University of Technology.
<http://www.iicm.edu/ivis/ivis.pdf>.

Andrews, K., Pichler, M., and Wolf, P. (1996). Towards rich information landscapes for visualizing structured web spaces. In *Proceedings of IEEE Symposium on Information Visualization '96*, pages 62–63.

Andrews, K., Wolte, J., and Pichler, M. (1997). Information pyramids (TM): A new approach to visualizing large hierarchies. In Varshney, A. and

Ebert, D. S., editors, *Proceedings LBHT IEEE Visualization '97*, pages 49–52. IEEE.

Asymptote (2002). Asymptote architecture. <http://www.asymptote.net/>.

Atlas (2002). An atlas of cyberspaces webpage. <http://www.cybergeography.org/atlas/atlas.html>.

Baldonado, M. Q. W., Woodruff, A., and Kuchinsky, A. (2000). Guidelines for using multiple views in information visualization. In *Advanced Visual Interfaces*, pages 110–119.

Becker, B. G. (1997). Using mineset for knowledge discovery. *IEEE Computer Graphics and Applications*, 17(4):75–78.

Becker, R. A., Eick, S. G., and Wilks, A. R. (1991). Basics of network visualization. *IEEE Computer Graphics and Applications*, 11(3):12–14.

Becker, R. A., Eick, S. G., and Wilks, A. R. (1995). Visualizing network data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):16–28.

Benedikt, M. (1991). *Cyberspace: First Steps*, chapter Cyberspace: Some Proposals, pages 119–224. MIT Press.

Benyon, D. and Höök, K. (1997). Navigation in information space: supporting the individual. In Howard, S., Hammond, J., and Lindgaard, G., editors, *Human-Computer Interaction: INTERACT'97*, pages 39–46. Chapman & Hall.

Bertin, J. (1967). *Sémiologie Graphique*. Gauthier-Villars.

Bertin, J. (1983). *Semiology of graphics*. University of Wisconsin Press, Madison, Wisconsin.

Card, S. K. (1996). Visualizing retrieved information: a survey. *IEEE Computer Graphics and Applications*, 16(2):63–67.

Card, S. K., Mackinlay, J. D., and Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think*. The Morgan Kaufmann Series in Interactive Technologies. Morgan Kaufmann Publishers, San Francisco, California.

Casner, S. M. (1991). A task-analytic approach to the automated design of graphic presentations. *ACM Transactions on Graphics*, 10(2):111–151.

Chalmers, M. (1993). Using a landscape metaphor to represent a corpus of documents. In Frank, A. and Caspari, I., editors, *Proceedings of the European Conference on Spatial Information Theory*, volume 716, pages 377–390. Springer Verlag, Lecture Notes in Computer Science: Spatial Information Theory.

Chapman, P., Stevens, P., Wills, D., and Brooks, G. (2001). Real-time visualization in the offshore industry. *IEEE Computer Graphics and Applications*, pages 6–10.

Chapuy, S., Dimcovski, Z., Do, H., Mirimanoff, R., and Raimondi, S. (2000). Progress of quality control in radiotherapy daily control thanks to online dynamic visualization. In *Proceedings of the Nuclear Science Symposium Conference*, volume 3, pages 57–58. IEEE.

Chen, C. (1999). *Information Visualisation and Virtual Environments*. Springer-Verlag, London.

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

Cleveland, W. S. and McGill, R. (1984). Graphical perception: Theory, experimentation and application to the development of graphical methods. *J. American Statistical Association*, 79(387):531–554.

Cockburn, A. and McKenzie, B. J. (2001). 3D or not 3D? Evaluating the effect of the third dimension in a document management system. In *Proceedings of SIGCHI'01*, pages 434–441.

Conklin, J. (1987). Hypertext: a survey and introduction. *IEEE Computer*, 20(9):17–41.

CORBA (2000). Common object request broker architecture. <http://www.corba.org/>.

Cox, K. C., Eick, S. G., and He, T. (1996). 3D geographic network displays. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 25(4):50–54.

Crossley, M., Davies, N. J., Taylor-Hendry, R. J., and McGrath, A. J. (1997). Three-dimensional internet developments. *BT Technology Journal*, 15(2):179–193.

Crutcher, L. A., Lazar, A. A., Feiner, S. K., and Zhou, M. X. (1993). Management of broadband networks using a 3D virtual world. In *Proceedings of the 2nd International Symposium on High Performance Distributed Computing*, pages 306–315.

Crutcher, L. A., Lazar, A. A., Feiner, S. K., and Zhou, M. X. (1995). Managing networks through a virtual world. *IEEE parallel and distributed technology: systems and applications*, 3(2):4–13.

Crutcher, L. A. and Waters, A. G. (1992). Connection management for an ATM network. *IEEE Network*, 6(6):42–55.

Cubeta, J., Kern, K., Egts, D., and Obeysekare, U. (1998). Venom – virtual environment for network monitoring.
<http://www.nrl.navy.mil/CCS/people/cubeta/venom/paper.html>.

CyberNet (2001). Cybernet project webpage.
<http://www.eurecom.fr/~abel/cybernet>.

Darken, R. P. and Sibert, J. L. (1996a). Navigating large virtual spaces. *The International Journal of Human-Computer Interaction*, 8(1):49–72.

Darken, R. P. and Sibert, J. L. (1996b). Wayfinding strategies and behaviors in large virtual worlds. In *Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems*, volume 1 of *PAPERS: Virtual and Computer-Augmented Environments*, pages 142–149.

Delaney, B. (1999). The NYSE's 3D trading floor. *IEEE Computer Graphics and Applications*, 19(6):12–15.

Dieberger, A. and Frank, A. U. (1998). A city metaphor to support navigation in complex information spaces. *Journal of Visual Languages and Computing*, 9(6):597–622.

Dillon, A., McKnight, C., and Richardson, J. (1993). *Hypertext: A Psychological Perspective*, chapter Space - the Final Chapter or Why Physical Representations are not Semantic Intentions, pages 169–191. Ellis Horwood series in Interactive Information Systems. Ellis Horwood.

Domik, G. (1996). Tutorial on visualization. University of Paderborn / Paderborn, Germany.
<http://www.css.tayloru.edu/instrmat/graphics/hypervis/domik/folien.html>.

EAI (1999). External authoring interface.
<http://www.web3d.org/WorkingGroups/vrml-eai/>.

Edwards, J. D. M. and Hand, C. (1997). MaPS: Movement and planning support for navigation in an immersive VRML browser. In Carey, R. and Strauss, P., editors, *VRML 97: Second Symposium on the Virtual Reality Modeling Language*, New York City, NY. ACM SIGGRAPH / ACM SIGCOMM, ACM Press. ISBN 0-89791-886-x.

Eick, S. G. and Wills, G. J. (1993). Navigating large networks with hierarchies. In *Proc. IEEE Conf. Visualization*, pages 204–210.

Erickson, T. (1993). *Virtual Reality : Applications and Explorations*, chapter Artifical Realities as Data Visualization Environments: Problems and Prospects, pages 3–22. Academic Press Professional.

Erickson, T. D. (1990). *The Art of Human-Computer Interface Design*, chapter Creativity and Design – Introduction, pages 1–4. Addison-Wesley.

Fairchild, K., Serra, L., Hern, N., Hai, L., and Leong, A. (1993). *Virtual Reality Systems*, chapter Dynamic FishEye Information Visualizations, pages 161–177. Academic Press, London.

Fairchild, K. M. (1993). *Virtual Reality: Applications and Explorations*, chapter Information Management Using Virtual Reality-Based Visualizations, pages 45–74. Academic Press Professional.

Fairchild, K. M. (1998). Explorations of 3D virtual spaces: applications of VR. In *Computer Vision for Virtual Reality Based Human Communications*, pages 94–98, Bombay, India. IEEE and ATR workshop on.

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

Foley, J. and Ribarsky, B. (1994). *Scientific Visualization: Advances and Challenges*, chapter Next-Generation Data Visualization Tools, pages 103–127. Academic Press.

Furnas, G. and Jul, S. (1997). Navigation in electronic worlds. In *Proceedings of ACM CHI 97 Conference on Human Factors in Computing Systems*, volume 2 of *Workshop 9*, page 230.

Furnas, G. W. (1986). Generalized fisheye views. In *Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems*, Visualizing Complex Information Spaces, pages 16–23.

Furnas, G. W. (1997). Effective view navigation. In *Proceedings of ACM CHI 97 Conference on Human Factors in Computing Systems*, volume 1 of *PAPERS: Information Structures*, pages 367–374.

Gabbard, J. and Hix, D. (1997). Taxonomy of usability characteristics in virtual environments. Technical report, Virginia Polytechnic Institute and State University. Final Report to the Office of Naval Research.

Gershon, N. (1998). Visualization of an imperfect world. *IEEE Computer Graphics & Applications*, 18(4). ISSN 0272-1716.

Gershon, N. and Brown, J. R. (1996). Computer graphics and visualization in the global information structure – special report. *IEEE Computer Graphics and Applications*, 16(2):60–75.

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

Gershon, N. and Eick, S. G. (1997). Information visualization. *IEEE Computer Graphics and Applications*, 17(4):29–31.

Gibson, J. J. (1979). *The Ecological Approach to Visual Perception.* Houghton Mifflin, Boston.

Goldstein, J., Roth, S. F., Kolojejchick, J., and Mattis, J. (1994). A framework for knowledge-based interactive data exploration. *Journal of Visual Languages and Computing*, 5(4):339–363.

Halasz, F. and Moran, T. P. (1982). Analogy considered harmful. In *Proceedings Human Factors in Computer Systems Conference*, pages 383–386.

Hand, C. (1997). Survey of 3D interaction techniques. *Computer Graphics Forum*, 16(5):269–281.

Hanson, A. J. and Wernert, E. A. (1997). Constrained 3D navigation with 2D controllers. In *IEEE Visualization '97.*

Henderson, Jr., D. A. and Card, S. K. (1986). Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Transactions on Graphics*, 5(3):211–243.

HTTP (2000). Hypertext transfer protocol. <http://www.w3.org/Protocols/>.

Inxight (2002). Inxight software, inc. <http://www.inxight.com/>.

Java (2002). The source for java technology.
<http://java.sun.com/>.

JavaScript (1997). Javascript reference.
<http://developer.netscape.com/docs/manuals/communicator/jsref/contents.htm>.

Johnson, B. and Shneiderman, B. (1991). Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Visualization '91*, pages 284–291.

Kahani, M. and Beadle, H. (1996). WWW-based 3D distributed, collaborative virtual environment for telecommunication network management. In *Proceedings Australian Telecommunication Networks and Applications Conference (ATNAC'96)*, pages 483–488.

Kanellopoulos, I., Stein, A., and Turatti, M. (2001). Visualization of geographic information in a dynamic 3-dimensional environment. In *Proceedings of the Geoscience and Remote Sensing Symposium, 2001. IGARSS '01*, volume 1, pages 201 – 203. IEEE.

Kaye, T. (1997). The tracking viewpoint.
<http://reality.sgi.com/tomk/demos/vrml2/TrackingViewpoint>.

Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8.

Keller, P. R. and Keller, M. M. (1993). *Visual Cues: Practical Data Visualization*. IEEE Computer Society Press and IEEE Press.

Koutsofios, E. E., North, S. C., and Keim, D. A. (1999a). Visualizing large telecommunication data sets. *IEEE Computer Graphics and Application*, 19(3):16–19.

Koutsofios, E. E., North, S. C., Truscott, R., and Keim, D. A. (1999b). Visualizing large-scale telecommunication networks and services. In Ebert, D., Gross, M., and Hamann, B., editors, *IEEE Visualization '99*, pages 457–462, San Francisco.

Kuipers, B. J. (1982). The "Map in the Head" metaphor. *Environment and Behavior*, 14(2):202–220.

Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*. Chicago : University of Chicago Press, Chicago.

Lamping, J., Rao, R., and Pirolli, P. (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. ACM.

Mackinlay, J. (1986). Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110–141.

Mackinlay, J. D. (2000). Opportunities for information visualization. *IEEE Computer Graphics and Applications*, 20(1):22–23.

Mackinlay, J. D., Card, S. K., and Robertson, G. G. (1990). Rapid controlled movement through a virtual 3D workspace. *Computer Graphics*, 24(4):171–176.

Mackinlay, J. D., Robertson, G. G., and Card, S. K. (1991). The perspective wall: Detail and context smoothly integrated. In *Proceedings of ACM CHI'91 Conference on Human Factors in Computing Systems*, Information Visualization, pages 173–179.

Marketmap (2002). Smartmoney.com. <http://www.smartmoney.com/marketmap/>.

Marshall, R. E. and Carswell, P. G. (1989). Alternative views of a hurricane. In *Proceedings of the Conference on Three-Dimensional Visualization and Display Technologies – Visualization Applications*, volume 1083. SPIE.

Masui, T., Minakuchi, M., IV, G. R. B., and Kashiwagi, K. (1995). Multiple-view approach for smooth information retrieval. In *ACM Symposium on User Interface Software and Technology*, pages 199–206.

McCormick, B. H., DeFanti, T. A., and Brown, M. D. (1987). Visualization in scientific computing - a synopsis. *Comput. Appl.Graph.*, 7(4):61–70.

Merriam-Webster OnLine (2002). Merriam-webster online. Merriam-Webster, Incorporated. <http://www.m-w.com/>.

Miller, G. A. (1956). The magic number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97.

Modjeska, D. (1997). Navigation in electronic worlds: Research review for depth oral exam. Technical report, University of Toronto – Department of Computer Science.

Mukherjea, S. and Foley, J. D. (1995). Visualizing the World-Wide Web with the Navigational View Builder. *Computer Networks and ISDN Systems*, 27(6):1075–1087.

Mukherjea, S., Foley, J. D., and Hudson, S. (1995). Visualizing complex hypermedia networks through multiple hierarchical views. In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1 of *Papers: Creating Visualizations*, pages 331–337.

Mullet, K., Schiano, D. L., Robertson, G., Tesler, J., and Tversky, B. (1995). 3D or not 3D: More is better or less is more? In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 2 of *Panels*, pages 174–175.

Munzner, T. and Burchard, P. (1995). Visualizing the structure of the World Wide Web in 3D hyperbolic space. In *Proceedings of the VRML 1995 Symposium*, pages 33–38. ACM Press.

Murta, A. (1995). Vertical axis awareness in 3D environments. In *Framework for Immersive Virtual Environments '95*, pages 169–176, London.

Newman, W. M. and Lamming, M. G. (1995). *Interactive System Design*. Addison-Wesley Publishing, Wokingham, England.

Norman, D. (1993). *Things That Make Us Smart*. Addison-Wesley.

North, C. (2001). Information visualization – class online. Dept of Computer Science / Virginia Tech.
<http://people.cs.vt.edu/ north/infoviz/>.

North, C. and Schneiderman, B. (1997). A taxonomy of multiple window coordinations. Technical Report CS-TR-3854.

North, C. and Shneiderman, B. (2000). Snap-together visualization: A user interface for coodinating visualizations via relational schemata. In *Advanced Visual Interfaces*, pages 128–135.

Passini, R. (1984). *Wayfinding in Architecture*. Van Nostrand Reinhold.

Performance Co-Pilot (2002). Sgi - performance co-pilot: Home page.
<http://www.sgi.com/software/co-pilot>.

Perlin, K. and Fox, D. (1993). Pad: An alternative approach to the computer interface. In Kajiya, J. T., editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 57–64.

Rekimoto, J. and Green, M. (1993). The information cube: Using transparency in 3D information visualization. In *Proceedings of the Third Annual Workshop on Information Technologies & Systems (WITS'93)*, pages 125–132.

Rheingans, P. (2002). Are we there yet? Exploring with dynamic visualization. *IEEE Computer Graphics and Applications*, pages 6–10.

Roberts, J. (2001). Issues of Dataflow and View Presentation in Multiple View Visualization. In *2001 International Conference on Imaging Science, Systems and Technology (CISST)*, CISST Annual Conference, Workshop: Fundamental Issues of Visualization, pages 177–183, Las Vegas, NV.

Roberts, J. C. (2000). Multiple-View and Multiform Visualization. In Erbacher, R., Pang, A., Wittenbrink, C., and Roberts, J., editors, *Visual Data Exploration and Analysis VII, Proceedings of SPIE*, pages 176–185. IS&T and SPIE.

Robertson, G. G., Card, S. K., and Mackinlay, J. D. (1993a). Information visualization using 3D interactive animation. *Communications of the ACM*, 36(4):56–71.

Robertson, G. G., Card, S. K., and Mackinlay, J. D. (1993b). Nonimmersive virtual reality. *IEEE Computer*, 26(2):81–83.

Robertson, G. G., Mackinlay, J. D., and Card, S. K. (1991). Cone trees: Animated 3D visualizations of hierarchical information. In Robertson, S. P., Olson, G. M., and Olson, J. S., editors, *Proc. ACM Conf. Human Factors in Computing Systems, CHI*, pages 189–194. ACM Press.

Rogowitz, B. and Treinish, L. (1996). How not to lie with visualization. *Computers in Physics*, 10(3):268–274.

Rogowitz, B. E. and Treinish, L. A. (1998). Data visualization: the end of the rainbow. *IEEE Spectrum*, 35(12):52–59.

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

Roth, S. F., Chuah, M. C., Kerpedjiev, S., Kolojejchick, J. A., and Lucas, P. (1997). Toward an information visualization workspace: Combining multiple means of expression. *Human-Computer Interaction*, 12(1/2):131–185.

Roth, S. F., Kolojejchick, J., Mattis, J., and Goldstein, J. (1994). Interactive graphic design using automatic presentation knowledge. In *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*, volume 1 of *Active Support for Interaction*, pages 112–117. Color plates on page 476.

Roth, S. F. and Mattis, J. (1990). Data characterization for intelligent graphics presentation. In *Proceedings of ACM CHI'90 Conference on Human Factors in Computing Systems*, End User Modifiable Environment, pages 193–200.

Russo Dos Santos, C., Gros, P., Abel, P., Loisel, D., Trichaud, N., and Paris, J.-P. (2000). Metaphor-aware 3D navigation. In Roth, S. F. and Keim, D. A., editors, *Proceedings of the IEEE Symposium on Information Visualization,2000 – InfoVis2000*, pages 155 – 165, Salt Lake City, USA. IEEE.

Sarkar, M. and Brown, M. H. (1992). Graphical fisheye views of graphs. In *Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems*, Visualizing Objects, Graphs, and Video, pages 83–91.

Sarkar, M., Snibbe, S., Tversky, O. J., and Reiss, S. P. (1993). Stretching the rubber sheet: A metaphor for viewing large layouts on small screens. Technical Report CS-93-39, Department of Computer Science, Brown University. Sun, 13 Jul 1997 18:30:14 GMT.

Satalich, G. (1995). Navigation and wayfinding in virtual reality: Finding proper tools and cues to enhance navigation awareness. Master's thesis, University of Washington.

Senay, H. and Ignatius, E. (1994). A knowledge-based system for visualization design. *IEEE Computer Graphics and Applications*, 14(6):36–47.

Senay, H. and Ignatius, E. (1996). Rules and principles of scientific data visualization. In ACM SIGGRAPH HyperVis Project – Teaching Scientific Visualization Using Hypermedia.

SGI (2002). Silicon graphics inc. <http://www.sgi.com/>.

Shaffer, E., Reed, D., Whitmore, S., and Schaeffer, B. (1999). Virtue: Performance visualization of parallel and distributed applications. *IEEE Computer*, 32(12):44–51.

Shneiderman, B. (1994). Dynamic queries for visual information seeking. Technical Report UMCP-CSD CS-TR-3022, College Park, Maryland 20742, U.S.A.

Shneiderman, B. (1996). The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings of IEEE Symposium on Visual Languages*, pages 336–343. IEEE.

SNMP (2002). Simple network management protocol.
<http://snmp.cs.utwente.nl/ietf/rfcs/rfcbytopic.html>.

Spence, R. (2001). *Information Visualization*. ACM Press.

Startree (2002). Inxight star tree.
<http://startree.inxight.com/>.

Stoakley, R., Conway, M. J., and Pausch, R. (1995). Virtual reality on a WIM: Interactive Worlds In Miniature. In Katz, I. R., Mack, R., Marks, L., Rosson, M. B., and Nielsen, J., editors, *Proceedings of the Conference on Human Factors in Computing Systems (CHI'95)*, pages 265–272, New York, NY, USA. ACM Press.

Tesler, J. and Strasnick, S. (1992). FSN: The 3D file navigator. Silicon Graphics Inc. Available by anomymous ftp from,
<sgi.sgi.com in directory sgi/fsn>.

The monk on the mountain (2000). The monk on the mountain problem. The Math Forum.
<http://mathforum.org/workshops/lucent/monk.html>.

Thorndyke, P. W. and Goldin, S. E. (1983). Spatial learning and reasoning skill. In Pick, H. L. J. and Acredolo, L. P., editors, *Spatial Orientation: Theory, Research, and Application*. Plenum Press, New York.

Thorndyke, P. W. and Hayes-Roth, B. (1982). Differences in spatial knowledge acquired from maps and navigation. *Cognitive Psychology*, 14:560–589.

Treemaps (2002). Treemaps for space-constrained visualization of hierarchies. Human-Computer Interaction Lab / University of Maryland.
<http://www.cs.umd.edu/hcil/treemaps/>.

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut.

Tufte, E. R. (1990). *Envisioning Information*. Graphics Press, Chechire, Connecticut.

Tufte, E. R. (1997). *Visual Explanations – Images and Quantities, Evidence and Narrative*. Graphics Press, Chechire, Connecticut.

VRML (1997). VRML 97, International Specification ISO/IEC IS 14772-1. <http://www.web3d.org/>.

Walker, G. (1995). Challenges of information visualization. *British Telecommunications Engineering Journal*, 14:17–25.

Ware, C. (2000). *Information Visualization: Perception for Design*. The Morgan Kaufmann Series in Interactive Technology. Morgan Kaufmann.

Ware, C. and Franck, G. (1994). Viewing a graph in a virtual reality display is three times as good as a 2D diagram. In *IEEE Conference on Visual Languages*, pages 182–183.

Wattenberg, M. (1999). Visualizing the stock market. In *ACM SIGCHI 1999 Extended Abstracts*, pages 188–189. ACM SIGCHI.

Wegenkittl, R., Groller, E., and Purgathofer, W. (1997a). Visualizing the dynamical behavior of wonderland. *IEEE Computer Graphics and Applications*, 17(6):71–79.

Wegenkittl, R., Löffelmann, H., and Gröller, E. (1997b). Visualizing the behavior of higher dimensional dynamical systems. In *IEEE Visualization'97 Proceedings*, pages 119–125. IEEE Computer Society.

Wehrend, S. and Lewis, C. (1990). A problem-oriented classification of visualization techniques. In Kaufman, A., editor, *Visualization'90*, pages 139–143. IEEE Press.

Wei, B., Silva, C., Koutsofios, E., Krishnan, S., and North, S. (2000). Visualization research with large displays. *IEEE Computer Graphics and Applications*, 20(4):50–54.

West, T. G. (1998). Knowing what you don't need to know. *Images and Reversals – Computer Graphics Newsletter SIGGRAPH ACM*, 32(1).

Wiss, U. and Carr, D. A. (1999). An empirical study of task support in 3D information visualisations. In *Proceedings of the IEEE International Conference on Information Visualisation*, pages 392–399, London. IEEE.

Wood, J., Brodlie, K., and Wright, H. (1996). Visualization over the World Wide Web and its appplication to environmental data. In Yagel, R. and Nielson, G. M., editors, *Proceedings of IEEE Visualization 96*, pages 81–86.

Wright, W. (1997). Business visualization applications. *IEEE Computer Graphics and Applications*, 17(4):66–70.

Wurman, R. S. (1989). *Information Anxiety*. Doubleday, New York.

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

Young, P. (1996). Three dimensional information visualization. Technical report, Department of Computer Science - University of Durham.

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

Zhou, M. X. and Feiner, S. K. (1996). Data characterization for automatically visualizing heterogeneous information. In *Proceedings IEEE Symposium on Information Visualization*, pages 13–20. IEEE.

Zhou, M. X. and Feiner, S. K. (1998). Visual task characterization for automated visual discourse synthesis. In *Proceedings of ACM CHI 98 Conference on Human Factors in Computing Systems*, volume 1 of *Supporting the Design Process*, pages 392–399. color plate on p. 643.

# Publications

1. C. Russo Dos Santos and P. Gros. Multiple views in 3D metaphoric information visualization. In *Proceedings of IEEE IV2002 (to appear)*, London, UK, July 2002.

2. C. Russo Dos Santos, P. Gros, and P. Abel. Combining physical and semantical navigation in three-dimensional information visualization. In *Proceedings of SPIE Visualization and Data Analysis Conference*, vol. 4665, pp. 296-307, San Jose, USA, January 2002.

3. C. Russo Dos Santos, P. Gros, and P. Abel. Dynamic information visualization using 3D metaphoric worlds. In *Proceedings of the International Conference in Visualization, Imaging, and Image Processing*, pp. 60-65, Marbella, Spain, September 2001.

4. C. Russo Dos Santos, P. Gros, and P. Abel. Automatic visual mapping of abstract information. In *Proceedings of CISST'2001 – Workshop on Fundamental Issues in Visualisation*, vol. II, pp. 619 - 625, Las Vegas, USA, June 2001.

5. C. Russo Dos Santos, P. Gros, P. Abel, D. Loisel, N. Trichaud, and J.-P. Paris. Metaphor-aware 3D navigation. In *Proceedings of IEEE InfoVis2000*, pp. 155 - 165, Salt Lake City, USA, October 2000.

6. C. Russo Dos Santos, P. Gros, P. Abel, D. Loisel, N. Trichaud, and J.-P. Paris. Mapping information onto 3D virtual worlds. In *Proceedings of IEEE IV2000*, pp. 379 - 386, London, UK, July 2000.

7. C. Russo Dos Santos, P. Gros, P. Abel, D. Loisel, N. Trichaud, and J.-P. Paris. The navigation wizard: Helped metaphor-aware navigation in virtual worlds. In *Proceedings of the International Workshop on Virtual Reality (Virtual Environments)*, pp. 69 - 78, Brest, France, July 2000.

8. C. Russo Dos Santos, P. Gros, P. Abel, D. Loisel, N. Trichaud, and J.-P. Paris. Experiments in information visualization using 3D metaphoric worlds. In *Proceedings of IEEE WET ICE 2000*, pp. 51 - 58, Washington, USA, June 2000.

9. P. Gros, P. Abel, C. Russo Dos Santos, D. Loisel, N. Trichaud, and J.P. Paris. Experimenting service-oriented 3D metaphors for managing networks using virtual reality. In *Proceedings of the Virtual Reality International Conference*, Laval, France, May 2000.

10. P. Abel, D. Loisel, P. Gros, C. Russo Dos Santos, and J.-P. Paris. Enhancement of network and system management using virtual reality. In *Proceedings of IEEE Network Operations and Management Symposium (NOMS 2000)*, pp. 975 - 976, Honolulu, USA, April 2000.

11. P. Abel, P. Gros, D. Loisel, C. Russo Dos Santos, and J.P. Paris. Cybernet: A framework for managing networks using 3D metaphoric worlds. *Annales des Telecommunications*, April 2000.

12. P. Abel, P. Gros, C. Russo Dos Santos, D. Loisel, and J.-P. Paris. Automatic construction of dynamic 3D metaphoric worlds: An application to network management. In *Proceedings of SPIE Visual Data Exploration and Analysis VII*, vol. 3960, pp. 312-323, San Jose, USA, January 2000.

13. C. Russo Dos Santos, P. Gros, P. Abel, D. Loisel, and J.-P. Paris. Using virtual reality for network management: Automated construction of dynamic 3D metaphoric worlds. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology (VRST'99)*, pp. 184-185, London, UK, December 1999.

14. C. Russo Dos Santos, P. Gros, and P. Abel. Three-dimensional visualization of large volumes of information. In *Proceedings of CLME'99*, vol. 2, pp. F73 - F87, Maputo, Mozambique, September 1999.

15. P. Abel, P. Gros, D. Loisel, J.P. Paris, and C. Russo Dos Santos. Construction automatique de mondes virtuels représentant le comportement dynamique d'un réseau. In *Proceedings of Cinquèmes journées d'études et d'échanges sur la Compression et la Représentation des Signaux Audiovisuels (CORESA'99)*, France, June 1999.