Automatic Visual Mapping of Abstract Information

C. Russo Dos Santos, P. Gros, P. Abel
Institut Eurécom
2229, Route des Crêtes
06904 Sophia Antipolis, France

Abstract *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?*

In this paper we present our strategy for automatically mapping time-varying abstract data onto the visual parameters that are used to display the information. 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.

Keywords: Information Visualization, Automatic Mapping, Visual Metaphors

1 Introduction

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.

In this paper we present the strategy we use for mapping large volumes of dynamic data onto metaphoric virtual worlds. The work presented here is part of a research project, CyberNet[1], that has the broad goal of studying how three-dimensional (3D) information visualization may help in the visualization, understanding and analysis of large volumes of dynamic information. The starting application domain was network monitoring and management to benefit from in-house available 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.

CyberNet uses a distributed object framework composed of three distinct parts:

- The *collecting layer* comprises the data collecting agents used to gather the raw data from the network devices.
- The *structuring layer* structures the raw information in order to be exploited by the presentation layer. The information is structured into what we call *services*, which are represented by a *service tree.*
- The *presentation layer* is responsible for presenting the 3D world. It maps the structured information (service tree) onto the visual parameters of the virtual world. It also provides the user interaction and navigation capabilities.

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 effective data. And, since CyberNet's framework is application-domain independent (with, naturally, the exception of the raw data collecting agents), a way of mapping abstract information on the visual parameters in a manner that is application field independent was also sought for.

Moreover, the fact that the visual mapping of the data is 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 mapping on the fly and to experiment and test different kinds of mappings and different metaphors, in an easy and transparent way. This paper presents the requisites and the mapping process we have devised to automatically map abstract information onto visual parameters.

The paper is organized in sections, as follows: in Section 2 we present the concept of a service, its attributes and elements, and we also address the user's tasks. In Section 3 we discuss our usage of visual metaphors, their attributes, components and constraints. Section 4 describes in detail the mapping types, criteria and steps, and presents an example of a mapping taken from the CyberNet project. Finally, we draw some conclusions in Section 5.

2 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, thus, is 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, relations and service nodes:

- *Entities* are the smallest building blocks that are used to construct the service tree. Entities gather related information concerning an element of the service and may have one or more attributes.

- *Relations* are the "glue" that is used to build the service model tree. Relations gather entities according to a common characteristic (entity attribute). Thus, a relation is a reference to one or more entities and an entity can be involved in one or more relations.

- *Service nodes* are the nodes of the service tree. Service nodes use the service elements (entities and relations) to build the service tree. Each service node is related to at least one relation and is therefore aware of all the entities involved in the concerned relations.

In Figure 1, on the left side, we can observe an example of a service tree for the NFS service. In the service tree we can easily identify the entities (represented by the square boxes) and the service nodes (represented by the elliptical boxes). The relations are figuratively represented by the arrows.

2.1 Service Characterization

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. In our system, data is characterized along three basic dimensions: type, time and semantic.

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 [2], [3] or [4]. The *time* dimension classifies data according to its time-dependent behavior, since our framework has to deal with highly dynamic data. The *semantic* dimension provides the context in which the data is going to be used (useful for task-dependent mapping). Relations are characterized according to their basic types (e.g., dependency, set of).

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 metaphors, without the incurring on the risk of not being able to visualize all the information.

2.2 Tasks

After some initial experiments without taking user tasks into consideration, we came to the conclusion that the tasks that are to be performed on the information should also 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 it to look for a better information visual encoding depending on the user task, even though the service and the service data remain the same. To comply with this, we also have a characterization of the tasks and mapping rules that take them into account. For instance, color is of good use for correlation tasks. For comparison purposes, heights must be positioned all on the same plan – it is very hard to compare height on different plans.

3 Metaphors

Our motivation to use visual metaphors to depict the information is based on the fact that the use of metaphors allows for the presentation of information in a way that the user is already familiarized with. This gains more relevancy if we take into account that most of the 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 to represent a process, for example?

Additionally, the use of real-world based metaphors (e.g., city metaphor, solar system metaphor) or otherwise abstract metaphors (e.g., conetree), provides consistency to the information presentation. A metaphoric world gives a consistent look and feel to the information presentation. This is of great importance since it makes it easier for the user to construct and then maintain the mental model of the visual representation.

Visual metaphors also provide new ways of not only depicting, but also interacting with and navigating in the virtual world. For instance, in a solar system based metaphor, we can use a virtual spaceship to navigate in the world, and we can interact with the data by creating new galaxies with different data attributes. We have already used several different metaphors, such as a virtual city, a solar system, a library, or a pyramid.

3.1 Metaphor Characterization

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.

A metaphor is described by an acyclic graph, composed of the metaphors elements: *layout managers* and *glyphs* – see Figure 1 for an example of the city metaphor description graph. To characterize a metaphor in order to know what visual parameters are available for mapping, we characterize its components, as described below. Nonetheless, we found that this components characterization was not sufficient. There are some generic attributes, that are characteristic of the metaphor itself, that influence the mapping (e.g., is the metaphor recursive or not).

Aside from the attributes, we also take into account constraints that are specific of the metaphor as a whole and not of its elements. Such constraints may be related to the comprehension of the metaphor; e.g., if, in a cityscape metaphor, we map a wide range data value onto the width of a building and a low range one onto the height, we risk losing the resemblance of a building, since we may end up with a representation that resembles more a wide flat platform that a skyscraper.

Other constraints may also be related to the consistency of the metaphor and to avoid disturbing the user's mental model of the visual world; for instance, again for the example 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 actual position, we would see buildings moving all over the city! This kind of constraints is thus taken into account by our system and is a part of the characterization of the visual metaphors.

3.2 Graphical Components

A 3D metaphoric virtual world is built using two types of graphical components: the 3D glyphs (3DGs) and the layout managers (LMs). A *3D glyph* is a graphical element that offers a number of visual parameters to map information on. A *layout manager* is responsible for placing its children – other layout managers or glyphs – in the 3D space. LMs also have additional interaction and navigation capabilities. In the acyclic graph that describes the hierarchy of the metaphor, the 3D glyphs correspond to the leaf nodes of the tree, while the layout managers correspond to the intermediate ones (Figure 1).

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), text (e.g., label, textColor) and so on.

Layout managers are characterized according to their skills in 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 attributes (e.g., oriented, not ordered), its mapping parameters (e.g., position along one dimension, two dimensions fixed), and the task it may be used for (e.g., sorting). On the other hand, a chess LM, that places children in a grid-like manner on a plane, is not oriented, positions children along two dimensions and may be used comparison purposes (specially in the case where it places its children on a horizontal plane).

4 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.

4.1 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.

4.2 Criteria For Mapping

As mapping criteria we use the expressiveness criterion and the effectiveness criterion, in a manner similar as they are used in [2].

- The *expressiveness* criterion makes sure that the chosen metaphor is capable of representing all the service information. 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.

- 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 (both for glyphs and layout managers) to represent different types of information (entities and relations). It is also based on the relative importance the data has, in the context of a given service.

4.3 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.

We apply both criteria to both of the 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, also referred above.

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.

The *second* step is then to apply the criterion of effectiveness to the result of the structural mapping, obtained from the first 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 step 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.

The *third* 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.

The *last* stage of the mapping process is evaluating the efficacy of the metaphor with respect to the data encoding. For this we 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. Thus, 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.

4.4 Mapping Rules

The mapping rules are implemented by the *adaptors*. Adaptors are non-graphical elements of the presentation layer. 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 [2], or knowledge-based visualizations, such as [5], and extrapolated for 3D visualization where the case applies.

The concept of best mapping is a not a static

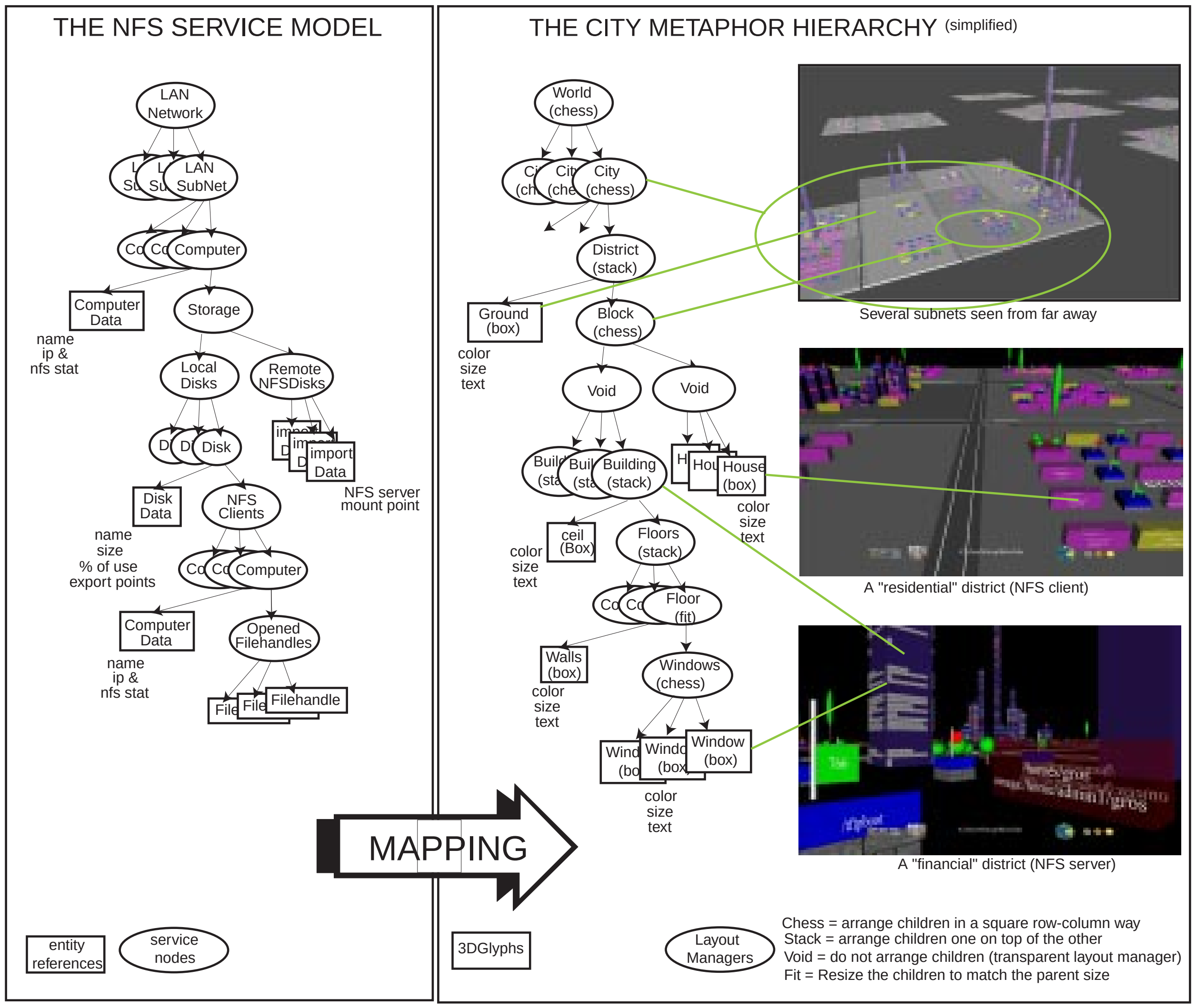

Figure 1: Example of a visual mapping between the NFS service and the city metaphor (CyberNet project).

one in the context of our system. We can have different types of "best" mapping. The efficacy of the 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) or purely aesthetic concerns. For the moment, though, only visual perception concerns and the users tasks are taken into account when finding the best mapping.

4.5 Mapping Example

Figure 1 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 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.

5 Conclusions

We presented a framework and a strategy for automatically mapping abstract information onto metaphorical virtual worlds. The fact that we 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.

Our mapping engine is based on metadata that characterizes the actual data – thus making the mapping engine application domain independent, since it makes use of a higher abstraction level – and on the visual parameters available from the virtual world. We also take into account the tasks that must performed on the data. The mapping criteria allows us to define several kinds of "best" mapping.

References

[1] Pierre Abel, Pascal Gros, Cristina Russo Dos Santos, Didier Loisel, and Jean-Pierre Paris. Automatic construction of dynamic 3d metaphoric worlds: An application to network management. In Jonathan Roberts, Robert Erbacher, Philip Chen and Craig Wittenbink, editors, *Visual Data Exploration and Analysis VII*, volume 3960, pages 312–323. SPIE - The International Society for Optical Engineering, January 2000.

[2] Jock Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110–141, April 1986.

[3] Steven F. Roth and Joe Mattis. 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, 1990.

[4] Michelle X. Zhou and Steven K. Feiner. Data characterization for automatically visualizing heterogeneous information. In *Proceedings IEEE Symposium on Information Visualization*, pages 13–20. IEEE, 1996.

[5] Hikmet Senay and Eve Ignatius. A knowledge-based system for visualization design. *IEEE Computer Graphics and Applications*, 14(6):36–47, November 1994.