ISR-TR-94-45
CAR-TR-716
CS-TR-3288

June 1994

Visual Information Management for Network Configuration

Harsha Kumar*, Catherine Plaisant,
Marko Teittinen*+, Ben Shneiderman*+

Human-Computer Interaction Laboratory
Center for Automation Research
*also Institute for Systems Research
+also Department of Computer Science
University of Maryland

Abstract

Current network management systems rely heavily on forms in their user interfaces. The interfaces reflect the intricacies of the network hardware components but provide little support for guiding users through tasks. There is a scarcity of useful graphical visualizations and decision-support tools.

We applied a task-oriented approach to design and implemented the user interface for a prototype network configuration management system. Our user interface provides multiple overviews of the network (with potentially thousands of nodes) and the relevant configuration tasks (queries and updates). We propose a unified interface for exploration, querying, data entry and verification. Compact color-coded treemaps with dynamic queries allowing user-controlled filtering and animation of the data display proved well-suited for representing the multiple containment hierarchies in networks. Our Tree-browser applied the conventional node-link visualization of trees to show hardware containment hierarchies. Improvements to conventional scrollbar-browsers included tightly coupled overviews and detailed views. This visual interface, implemented with Galaxy and the *University of Maryland Widget Library* ™, has received enthusiastic feedback from the network management community. This application-specific paper has design paradigms that should be useful to designers of varied systems.

1. Introduction

Today's networks are heterogeneous along several dimensions, for example, different transmission media (satellite, fiber optic), kind of data being transmitted (video, sound, images, data), and multivendor networks. This makes networks highly complex in their transmission, performance and communication characteristics. Tens of thousands of elements have to be controlled, each having dozens of parameters to be specified. Almost all of the control and monitoring functions of communication network systems are implemented as software. Therefore, such software systems themselves are of enormous size and complexity.

The ISO / ANSI standards committee has classified the functionality required of network management systems into the six categories of Configuration management, Fault management, Performance management, Security management, Accounting management and Directory management. We concentrated on configuration management, defined by ISO as "Defining, monitoring and controlling network resources and data".

Many network management systems take a database-centric approach, i.e., the network is represented in a database called the Management Information Base (MIB). Designing the user interface for a Network Configuration Management System (NCMS) is a very challenging problem because of the many configuration parameters that need to be entered into the MIB via the user interface, i.e. the task of configuring a network is data-intensive. NCMS users are network operators working under the pressure to keep the network running 24 hours a day under all circumstances. Current user interfaces typically consist of hundreds of forms that need to be filled in order to configure / update the network. Users get lost in piles of forms in the absence of effective organization and information visualization tools. "A fundamental feature of an advanced network management station is the capability to present to the human manager a comprehensible picture of the relevant scenarios" [Cons93].

We worked closely with *Hughes Network Systems* (HNS) who develops commercial network management systems and manages a large number of telecommunication networks. Researchers from the fields of human-computer interaction, databases and networking participated in the project. The work involved the development of a prototype network management system (henceforth referred to as *prototype*) with a graphical user interface, and an object-oriented database with embedded dynamic constraint-checking mechanisms.

Our group from the Human-Computer Interaction Laboratory designed and implemented the user interface. Our focus in the first stage of the project was on satellite networks. Satellite networks, in general, are less complex than heterogeneous networks, because of their *star* topology, as opposed to a *mesh* topology. However, starting with the satellite networks enabled us to design and develop initial prototypes rapidly and obtain immediate feedback from users.

2. Background

2.1. Literature Review

There is some literature that directly addresses the issue of user interfaces for network management. [Beck90] used color and size-coded nodes and links on an underlying geographical map to visualize network statistics. They used direct manipulation widgets like sliders to filter network data interactively. We believe that this approach has merits. [Mart93] applied several visualizations like Cone-trees, circle-diagrams and fisheye views for telecommunications network planning. They designed a language to describe the mapping between input data and visualization features. [Cons93] applied their Hy+ visual database system to manipulate network visualizations through visually expressed queries.

[Moen90] gives algorithms for drawing dynamic trees. [Rada88] describes a system that graphically displays data structures, including trees. [Chig93, Robe91] describe 3D visualizations of hierarchies with their Info-TV and Cone-tree systems respectively. [Mess91] proposes a divide-and-conquer layout algorithm for graphs, [Henr91] presents a methodology for viewing large graphs, while [Ding90] provides a framework for automated drawing of data structures. [Bear90] compares navigational techniques to improve the display of large 2D spaces. [Gedy88] describes the design and implementation of a graphical browser for a large highly connected database. [Holl89, Scha92] present studies of fisheye views of graphs. [Shne92, Turo92] focus on treeemaps, a hierarchical visualization tool that uses a space-filling approach.

Thus, much of the related literature concentrates on the design, algorithmic and browsing aspects of graph and tree structures in general. Our work is different in that we address the specific application of <u>telecommunications network configuration</u>, and present design paradigms and visualization tools that we successfully applied to this application. The two visualization tools that we used were the *treemap* [John92, Shne92] and the *Tree-browser* (introduced in this paper).

2.2. Simplified description of the structure of the network

In the HNS architecture, the satellite networks consists of a centralized hub and many (thousands) of remotes. Each remote communicates with the hub via satellite (Figure 1). This communication is established by setting up a *session.* Any 2 remotes communicate via the hub.

The hub and each remote consist of a complex hierarchy of hardware (Figure 1) and software objects. A remote has many DPCs (Data Port Clusters), each DPC many LIMs (LAN Interface Modules), and each LIM many ports. Similarly, the hub hardware has a similar but larger containment hierarchy: many network groups, many networks, many DPCs, many LIMs and many ports. A hub typically has thousands of ports. A session is established between a port of the hub and a port of a remote.

Inroutes and outroutes are the satellite channels used by hub ports and remote ports to communicate. Each session is assigned an Inroute Group (group of Inroutes).

2.3. Problems with current interfaces for network management

Current network management systems rely heavily on forms in their user interfaces. Forms are easy the develop and customize. They are usually simple to understand and to start using. But complex systems require so many forms that screens quickly become cluttered with dozens of overlapping forms (windows), and window-management becomes a real burden (Figure 2). Operators need to learn the names of all the forms. The problem is often aggravated by the fact that unrelated forms look similar, while there is no visual connection between related forms. The design of forms could be substantially improved in terms of meaningful layout, consistency, and efficient utilization of screen space.

The user interfaces of current systems reflect the intricacies of the network hardware components but provide little support for guiding users through tasks. Each task typically consists of a sequence of forms that need to be filled. No feedback is given as to how much of the task has been completed and how much of the task remains to be done. This is important since operators often have to simultaneously work on several tasks and handle emergency calls as they come. There is a clear need for the ability to group windows together, iconize them and re-open them simultaneously.

Data remains mainly displayed in tables and lists; some of the data is even in incomprehensible hexadecimal format. Better visualization techniques are needed. Many network management systems provide area maps showing the location of the remotes and lines indicating the communication links. In one system that we saw, remotes were represented as dots on a map of the U.S., and the operator could zoom in and out. These maps were seldom used by operators because the drawing of the map was too slow and the information shown was not useful enough. The latter is especially true for satellite networks which have a star topology, i.e., each remote is linked to the hub and it communicates with other remotes through the hub only.

Operators are given written instructions (suggested ports and inroutes) which are the results of studies of performance data and planning information, always done off-line. The instructions are updated on an irregular basis and operators sometimes have to take decisions based on outdated information. Thus, the process of selection of hardware components is often one of trial and error: configurations are used as long as problems are not encountered. There is a need for the user interface to provide this information on-line so that decisions can be based on current information.

3. Design Methodology

3.1 "Know thy users' task"

The interface should be designed to match the tasks performed by network operators. In order to design a task-oriented user interface, we collaborated closely with operators and engineers at Hughes Network Systems. We took training classes that are taken by new operators, interviewed designers and engineers of the current systems, and observed operators during their shifts as they configured networks and responded to emergency calls from customer sites. Finally manuals and screen prints of the current system were perused to give us mastery of a representative subset of configuration tasks that our prototype would address.

<u>The Task</u>
As mentioned previously, an important task in network configuration is to set up a session between a hub port and a remote port (henceforth referred to as the session task). This translates into the following subtasks:

1. Find the correct remote. The customer usually provides the name of the remote or at least some information like the location and the LAN Group of the remote. Hence, this subtask is usually straightforward.
2. Choose a port on that remote on the basis of a number of parameters, e.g. LAN type, LAN group, and data rate. Configure the port by entering values - or confirming default values - for several attributes such as node addresses, data rates etc.
3. Choose a port on the hub, again on the basis of similar parameters. Configure the port by entering attribute values. Check for compatibility of the remote and hub ports.

4. Choose an inroute group for the session.
5. Establish (or commit) the session.

Essentially, the task is to link *two compatible leaf nodes* (remote port and hub port) from two different trees via a node (inroute group) from a third tree. Thus, the session task involves a combination of:

- *Data exploration* to find trends of use or availability of space,
- *Querying* to find optimum choices of elements,
- *Data entry* to modify the configuration, and
- *Verification* of correctness / monitoring of impact on the physical network.

3.2 Low fidelity paper prototypes

After the task analyses, we made some low fidelity paper prototypes. The use of post-it notes (windows) on paper (computer screen) worked well for us. This enabled us to get rapid feedback from intended users of the system at Hughes and refine our design.

4. A visual information management interface for network configuration

In the past year the Human-Computer Information Laboratory has been designing, developing and describing visual information seeking interfaces [Ahlb94]. We have identified the following features as key elements of successful interfaces:

- *Overview* of the entire search space,
- Ability to *zoom* into regions of interest,
- *Dynamic queries* for filtering the data interactively in real time, and
- *Tight-coupling* between related fields to avoid empty query results.

The term dynamic queries describes the interactive user control of visual query parameters that generates a rapid (100 ms update) animated visual display of database search results [Shne94]. This kind of real-time display maximizes the visual bandwidth of the users, enabling them to catch trends in the data and spot exceptions. Also, irrelevant information is filtered out. The concepts of dynamic querying and tight-coupling are similar to those of *Focusing and Linking* [Buja91]. We have developed prototypes of a HomeFinder [Will92], a FilmFinder [Ahlb94], a health-statistic atlas [Plai93] and other interfaces for searching people, documents and resources.

In our network configuration prototype, we provide several interchangeable overviews not only of the data, but <u>also of the task to be performed</u>, and we merge the environment for exploration and querying with that for data entry and verification.

4.1. Task overviews

After operators select a task they wish to accomplish, a list of subtasks (elements) is displayed at the top of the screen in order to guide operators through the task (Figure 3). The checklist provides guidance as to what subtasks remain to be completed, but does not enforce the order of completion.

For example, in the session task, the subtasks related to the hub are displayed on the left, those related to remotes are displayed on the right, and those related to the inroutes are displayed in the center. A subtask can be completed by typing in directly, e.g., the name of a remote can be typed in directly. Alternatively, each button brings up an overview (visualization) of the data from which a selection can be made. For example, when the DPC button on the hub side is clicked, an overview of the hub is displayed at the DPC level (using a treemap or Tree-browser). Selection of an element makes its name appear in the top check-list, correct configuration of all of its parameters makes the corresponding button turn green indicating that that subtask has been completed. Thus, when the complete line of buttons at the top becomes green, the operators know that they can commit the task (this usually corresponds to commitment of a database transaction).

4.2. Data overviews: General Hierarchy Visualization tools

There are a number of hierarchical structures in this application, e.g. hardware containment hierarchies of the hub and remotes. Therefore, we made use of the treemap and Tree-browser, which are general hierarchy visualization tools. These visualizations provide access to network nodes and links, from where operators can access forms if required. In Figure 3, the treemap was selected to visualize the hub hardware hierarchy while the Tree-browser was selected to visualize the remote hardware hierarchy. Operators can interchangeably use either tool.

The general screen layout is tailored to fit the task (Figure 3). For example, in the session task, all overviews and forms corresponding to the hub open by default on the left, remote overviews and forms open on the right, and inroute overviews and forms open in the center. This simple default positioning of windows has a definite advantage over the typical random window placement because the overhead task of window management is reduced. The operator retains the flexibility to reposition windows.

Treemap:
The treemap maps hierarchical information to a rectangular 2-D display space utilizing 100% of the designated space [John92, Shne92]. Treemaps use a slice-and-dice strategy to partition the display space into a collection of rectangular boxes representing the tree structure. The strengths of treemaps are that they provide access to detail while keeping the global context. Screen space utilization is maximized, and scrolling and panning are not required. The number of nodes that can be displayed by a treemap is an order of magnitude greater than that by a traditional node-link diagram [Turo92]. In this application we applied dynamic queries to the treemap.

Figure 3 shows the following 3 distinct components of the treemap visualization tool:

Treemap Display: The treemap display displays the visualization. For example, in figure 3, the hub hierarchy is displayed down to the port level. The operator can also see the hub at the Network, DPC or LIM levels by clicking on the corresponding buttons at the top. The treemap provides a mechanism to zoom into nodes. Thus, the operator can either choose a port directly (bottom-up approach) simply by clicking on it, or by choosing a network, then zooming into that network, selecting a DPC, and so on (top-down approach).

Display controls: The operator can use the display controls to set the size and color of each node to represent an attribute of that node.

Query controls: The operator can use the query controls to make queries on both numerical and textual attributes of nodes, e.g. baud rate and LAN Group respectively.

Getting back to the session task, a good strategy for selecting a port is to first find the LIMs which are not over-utilized and then within those LIMs to look for an available port with high baud rate. The following is a possible sequence of steps that the operator might take in order to find a good port:

1. Ask to see the hub at the LIM level by clicking on the LIM button at the top. In Figure 4a, the outer box corresponds to the network. That outer box is divided vertically into 3 DPCs. Then the space for each DPC is split horizontally, and each resulting rectangle represents one LIM that the DPC contains.
2. Query the LIMs on the basis of utilization using the double-box slider. Those LIMs not satisfying the query are grayed out (Figure 4b).
3. Ask to see the hub at the port level by clicking on the Port button at the top (Figure 4c). Note that the LIMs that did not satisfy the query in step 2 remain grayed out.
4. Set both size and color of the ports to represent port baud rate. The bigger the port, the higher the baud rate. Similarly, brighter reds represent higher baud rates and deeper blues lower baud rates (Figure 4d). It is also possible to set the size and color of nodes (i.e. LIMs, ports etc.) to represent different attributes.
5. Choose a big red port (Figure 4e).

When a port is clicked, a form pops up to show the attributes of that port. These can be modified. A port can be selected for the session with a different action (currently by shift-clicking on it, but the drag and drop metaphor seems more appropriate to drop a port into in the checklist). When a port is selected, the names of the corresponding Network Group, Network, DPC, LIM and Port appear in the checklist. If all attributes have been entered or confirmed, all the hub buttons in the checklist also turn to green, signifying completion of this subtask.

<u>Tree-browser:</u>
A node-link representation of trees is composed of nodes and links, where nodes represent individual nodes in the tree, and links represent interrelationships between nodes (is-child-of and is-parent-of).

Node-link diagrams make inefficient use of screen space, which means that even trees of medium size require large areas to be completely displayed. Scrolling and panning is usually needed, and global context is lost in the absence of an overview. Treemaps overcome these deficiencies of node-link diagrams by using a space-filling approach. On the other hand, node-link diagrams are well known, intuitive and clear. They allow better depiction of ordering amongst siblings and links can be used to display additional information when appropriate.

Figures 5a and 5b show node link diagrams of a remote hierarchy at the LIM and Port levels. The nodes are color-coded to represent the configuration status, i.e. whether a particular node (e.g. port) is used, unused, or undefined.

The Tree-browser is a visualization tool for tree structures which makes use of the node-link representation. It overcomes the problem of loss of global context by providing tightly-coupled detailed views and overviews, and is envisioned to provide dynamic querying and *semantics-based browsing* features (Section 5.1).

In figure 6, the Tree-browser shows a remote at the port level in two views. A detailed view shows the node-link diagram in full zoom with the node names displayed, and an overview shows a miniature version of the node-link diagram without the node names. A field-of-view (the black rectangle on the overview) indicates what part of the tree is seen in the detailed view and can be moved to pan the detailed view. Similarly, when the detailed view is scrolled by using the scrollbars, the field-of-view moves in the overview to provide global context feedback. This direct linkage between views is called *tight-coupling*.

In section 5, we discuss design issues for the Tree-browser.

4.3. Unified interface for exploration, querying, data entry and verification

Another design principle that we applied was to give a unified interface to the users for exploration, querying, data entry and verification. In other words, the Tree-browser and treemap act not only as means of output, but as means of input as well. The operators can click on any node, e.g. remote port, and get a form that gives the identity and attributes of that port. (Figure 7). Updates can then be made to the database via the forms. When an update is attempted, the embedded constraints in the database are checked and if they hold, the update is accepted and the buttons Remote, RCPC, LIM and Port all turn green, else an error message is displayed in a popup window. Our interface provides an easy mechanism for users to activate or deactivate constraints on objects at run-time.

Thus, when all three subtasks (selection and configuration of hub port, remote port and inroute) are completed, all the buttons at the top are green, and the operator is in a position to commit the session. When the "Commit" button at the lower right corner is clicked, a session is set up from the remote port to the hub port via the inroute.

4.4. Implementation of the prototype

Our prototype runs on Sun SPARCStations. C++ and ObjectStore, an Object-Oriented database, were used for the MIB. The user interface was developed using C and Galaxy, a platform-independent user interface builder.

5. Current directions

5.1. Enhancements to the Tree-Browser

Dynamic querying

As demonstrated in the case of treemaps, a majority of tasks in browsing information spaces can be facilitated by dynamic querying. We are currently implementing two types of dynamic queries on the Tree-browser:

Attributes-based: Queries on node attributes, for example, give me all high baud rate ports.

Topology-based: Queries based on tree topology, for example, give me the LIM that has maximum number of available ports.

Semantics-based browsing

Generic 2D browsers [Plai94] treat the information space being browsed as images only. We believe that browsing of trees can be facilitated by taking advantage of the underlying structure of the tree. We are exploring ways to enable fast navigation between siblings, up to parents and grand-parents etc., without having to manually scroll and pan. Traversal of the tree in preorder, postorder and inorder, and tours of nodes marked either manually or by a query are being investigated.

Coping with varying size and structure

As the size and complexity of a tree increases, the problem of visualizing it effectively becomes more and more challenging. This applies to all visualization methods. There is a clear need for guidelines on the design of overviews, as the size of the tree increases. As the aspect ratio (fan-in / fan-out) of a tree varies, new layout strategies are needed. An intermediate view, which provides more detail than the overview, but less detail then the detailed view might be of help. With respect to dynamic queries, too many nodes might make the visualization cluttered and color-coding the results of queries might not be effective enough. Hierarchical clustering is a possible solution. The current implementation of the Tree-browser has uniform size for all nodes and uniform width for all links. Using the size, shape, color and texture of nodes and the width, color and texture of links would allow many attributes to be displayed simultaneously but might become difficult to interpret.

5.2. Hybrid (Mesh) networks

Our focus has now shifted to hybrid networks that include terrestrial links and ATM switches. From the user interface perspective, this means that the network topology is mesh, not star. Whereas in satellite networks, a communication link directly connects one remote port and one hub port, in the case of hybrid networks, a link between two nodes might be implemented as a dozen links through as many intermediate nodes.

Even in that case, the general hierarchy visualization tools described above can be used for many subtasks. But graph browsers are needed in order to see the overall structure of the network. We will extend dynamic queries and visual information management interfaces to browsing graphs.

6. Conclusions

We have received very encouraging feedback from Hughes Network Systems and other people from the network management community. This leads us to believe that a task-oriented approach to designing user interfaces, coupled with appropriate visualization tools is an effective strategy for successful user interfaces. Our proposed information management user interfaces should dramatically improve the time to learn, speed of use and error rate of next generation NCMS.

The combination of the treemap and dynamic queries was well received. We feel that expert operators will experience increased productivity in their tasks of configuring and managing networks. Formal usability evaluations are needed to quantify such productivity gains. The Tree-browser, when coupled with dynamic queries and semantics-based browsing features, promises to be a powerful, yet intuitive tool for visualizing hierarchies. Usability studies and experiments need to be conducted to access its strengths and weaknesses, and come up with more concrete design guidelines.

Acknowledgments:

We presented our work on the design of the user interface but the prototype was produced by a much larger team. We wish to thank Mulugu Srinivasarao, Shravan Goli and Konstantinos Stathatos, who worked closely with Harsha Kumar and Marko Teittinen to implement the prototype, Dave Whitefield of Hughes Network Systems who provided invaluable feedback, and George Atallah, Mike Ball, John Baras, Anindya Datta, Ramesh Karne, and Steve Kelley for their help managing the team and involvement in database issues. Thanks also to HCIL members who reviewed the paper and provided invaluable feedback.

Partial support for this project was provided by Hughes Network Systems, Maryland Industrial Partnerships, the Center for Satellite and Hybrid Communication Networks, and the NSF Engineering Research Center Program (NSFD CD 8803012).

References:

[Ahlb92] Ahlberg, C., Williamson, C., Shneiderman, B., Dynamic Queries for Information Exploration: An Implementation and Evaluation. *ACM CHI '92 Conference Proc. (Monterey, CA, May 3-7, 1992)* . ACM, New York. (1992) 619-626

[Ahlb94] Ahlberg, C., Shneiderman, B., Visual Information Seeking: Tight Coupling of Dynamic Queries with Starfield Displays. *Proc. of CHI '94*, ACM, New York (1994)

[Bear90] Beard, D.V., Walker, J. Q. II., Navigational Techniques to Improve the Display of Large Two-Dimensional Spaces. *Behavior & Information Technology 9*, 6, (1990) 451-466

[Beck90] Becker, R.A., Eick, S.G., Miller, E.O., Wilks, A.R., Dynamic Graphics for Network Visualization. *Proceedings of the first IEEE conference on Visualization*, San Francisco, California, October 1990, pp 93-96

[Buja91] Buja, A., McDonald, J.A., Michalak, J., Stuetzle, W., Interactive Data Visualization using Focusing and Linking. *Proceedings of IEEE Visualization '91*, San Diego, California, October 1991, pp 156-163

[Chig93] Chignell, M. H., Poblete, F., Zuberec, S., An Exploration in the Design Space of Three Dimensional Hierarchies. *Proceedings of the Human Factors Society*, 1993

[Cons93] Consens, M., Hasan, M., Supporting network management through declaratively specified data visualizations. *Proceedings of the third {1FIP / IEEE} International Symposium on Integrated Network Management*, 1993, pp 725-738

[Ding90] Ding, C., Mateti, P., A Framework for the Automated Drawing of Data Structure Diagrams. *IEEE Transactions on Software Engineering*. Vol. 16, No. 5, May 1990

[Gedy88] Gedye, D., Browsing the tangled web, Master's thesis report, Division of Computer Science, University of California at Berkeley, May 1988

[Henr91] Henry. T. R., Hudson, S. E., Interactive Graph Layout. *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software and Technology*, November 1991

[Holl89] Hollands, J.G., Carey, T.T., Matthews, M.L., McCann C.A., Presenting a Graphical Network: A Comparison of Performance Using Fisheye and Scrolling Views. *Designing and Using Human-Computer Interfaces and Knowledge Based Systems*, Elsevier Science publishers B.V., Amsterdam, 1989

[John92] Johnson, B., Shneiderman, B., Tree maps: A space-filling approach to the visualization of hierarchical information structures. *Proc. of IEEE Visualization' 91*, San Diego, CA (Oct. 1992) 284-291.

[Mart93] Martin, J.C., Visualization for Telecommunications Network Planning, *IFIP Transactions on Graphics, Design and Visualization*, Bombay, India, February 1993, pp 327-334

[Mess91] Messinger. E. B., Rowe, L. A., Henry, R. R., A Divide-and-Conquer Algorithm for the Automatic Layout of Large Directed Graphs. *IEEE Transactions on Systems, Man, and Cybernetics 21*, 1, (1991)

[Moen90] Moen, S., Drawing Dynamic Trees, *IEEE Software*, vol 7, num 4, July 1990, pp. 21-28.

[Plai93] Plaisant, C., Facilitating Data Exploration: Dynamic Queries on a Health Statistics Map. *Proc. of the 1993 American Statistical Association conference - Section on Government Statistics, San Francisco, Aug. 1993*. A.S.A Alexandria, VA. (1993) 18-23

[Plai94] Plaisant, C., Carr, D., and Shneiderman, B., Image browsers: Taxonomy, guidelines, and informal specifications, Department of Computer Science Technical Report CS-TR-3282, University of Maryland, College Park, MD, (March 1994)

[Rada88] Radack, G., Desai, T., Akrti: A System for Drawing Data Structures, *IEEE Languages for Automation Workshop* (Aug 29, 1988, College Park, MD), IEEE Press, pp 116-120

[Robe91] Robertson, G. G., Mackinlay, J. D., Card S. K., Cone Trees: Animated 3D Visualizations of Hierarchical Information. *CHI '91 Human Factors in Computing Systems*, ACM, 1991

[Scha92] Schaffer, D., Zuo, Z., Bartrum, L., Dill, J., Dubs, S., Greenberg, S., Roseman, M., Comparing Fisheye and Full-Zoom Techniques for Navigation of Hierarchically Clustered Networks. *Research report No. 92/491/29*, Department of Computer Science, The University of Calgary

[Shne92] Shneiderman, B., Tree Visualization with Treemaps: 2-d Space-Filling Approach. *ACM Transactions on Graphics*, Vol. 11, No. 1, January 1992, Pages 92-99

[Shne94] Shneiderman, B., Dynamic queries for visual information seeking. *IEEE Software*, in press, 1994

[Turo92] Turo, D., Johnson, B., Improving the visualization of hierarchies with treemaps: design issues and experimentation. *Proc. of IEEE Visualization '92*, Boston, MA, (Oct. 1992) 124-131

[Will92] Williamson, C. and Shneiderman, B., The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System. *Proceedings of the ACM SIGIR T92*, Copenhagen, Denmark (June 21–24, 1992) 338–346.

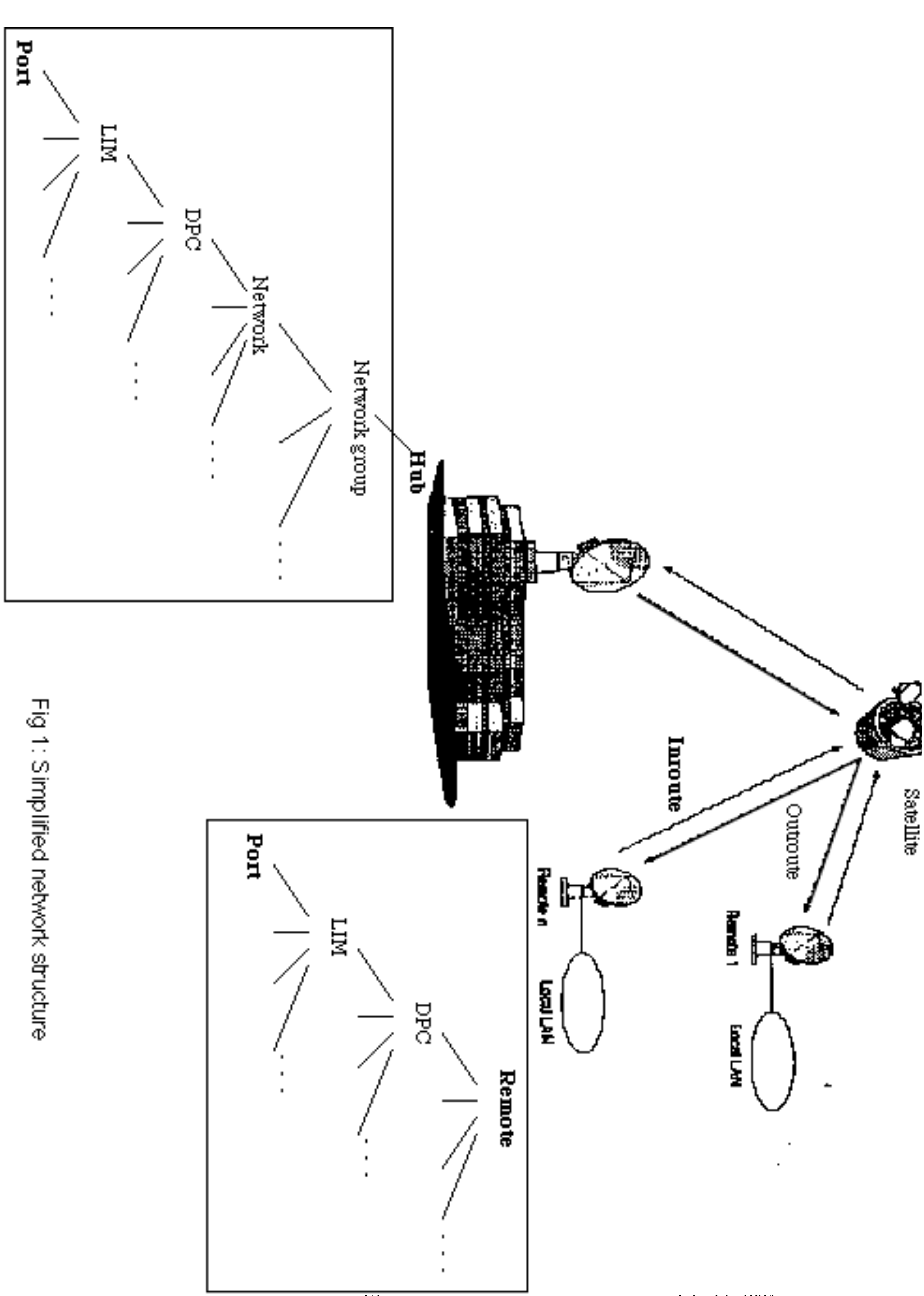

Fig 1: Simplified network structure

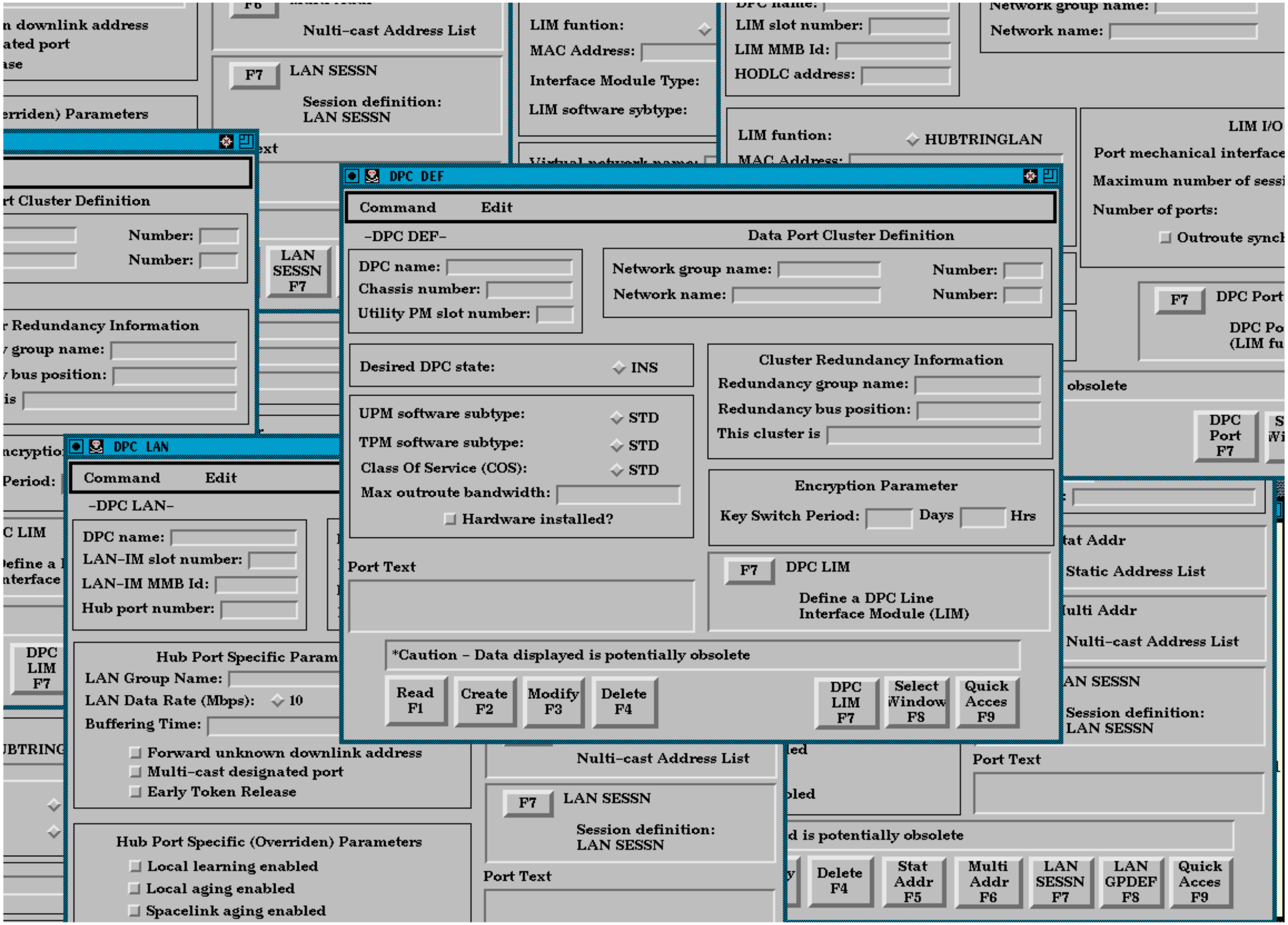

Fig 2: Example screen of a forms-based network configuration system

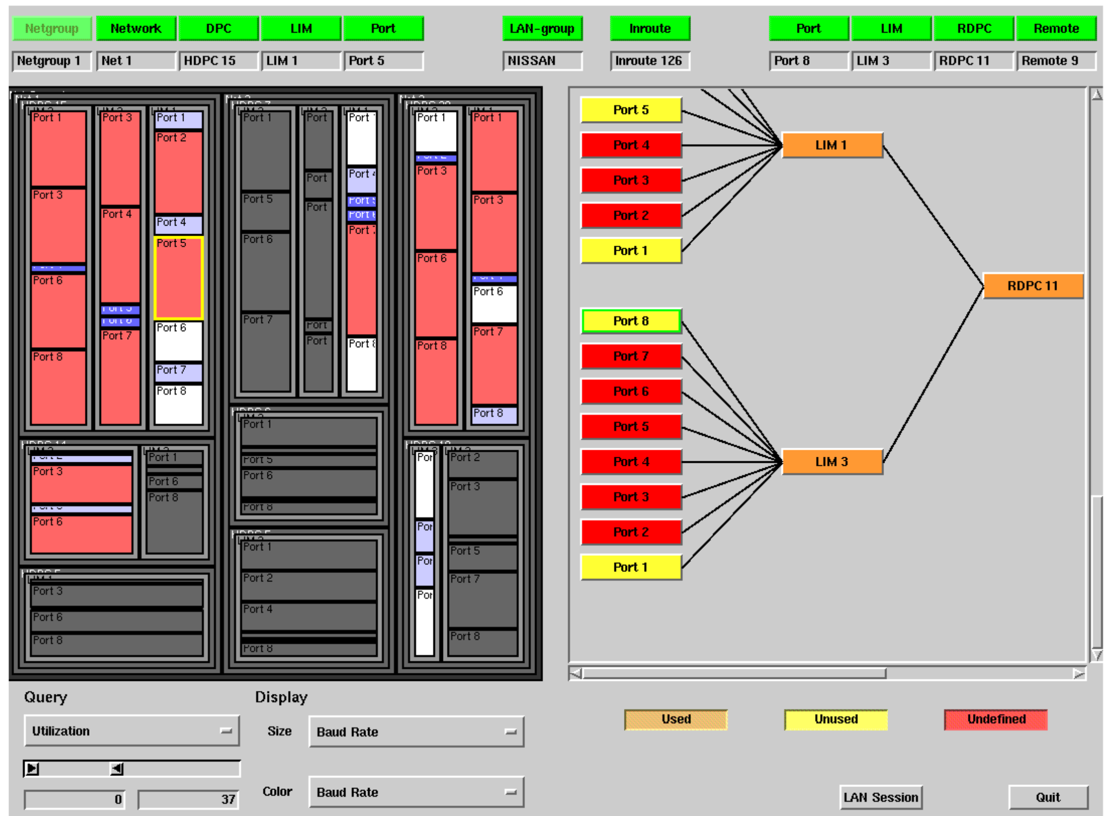

Fig. 3: Overall Screen Layout

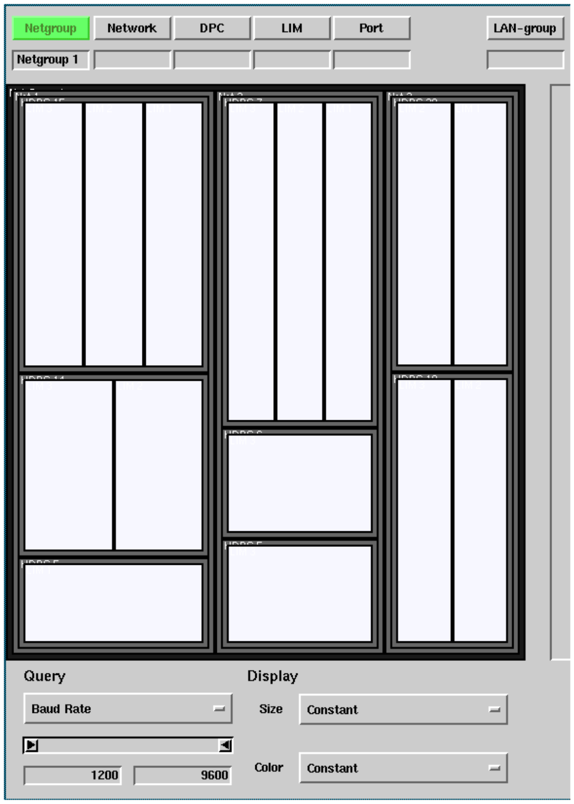

Fig 4a: Hub hierarchy at LIM level

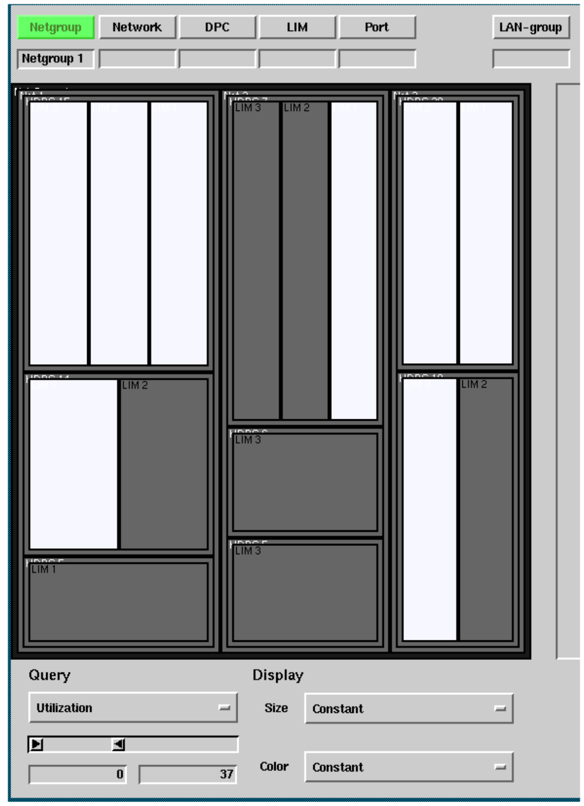

Fig 4b: Query LIMs on the basis of utilization

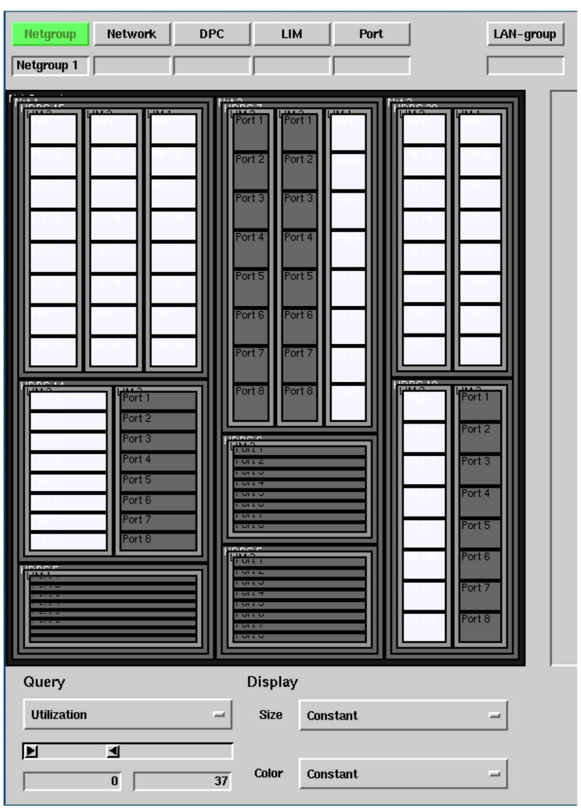

Fig 4c: Go to port level

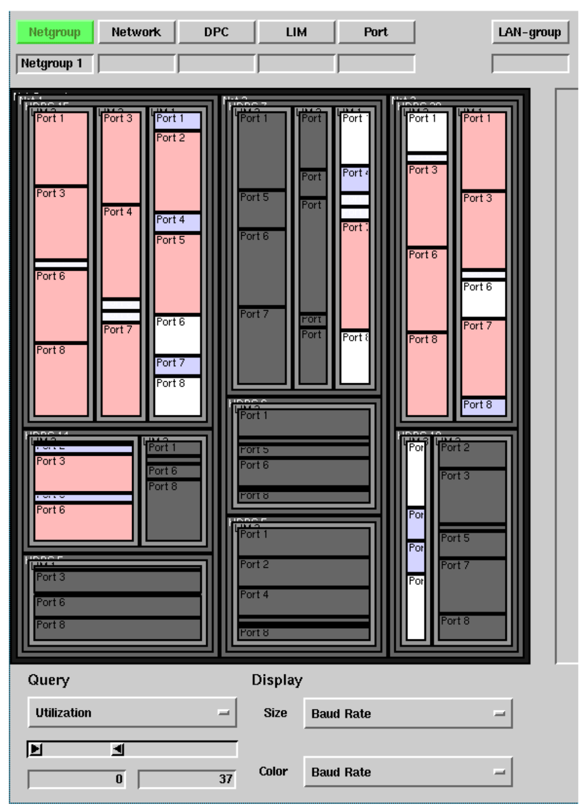

Fig 4d: Set size and color of ports to port baud rate

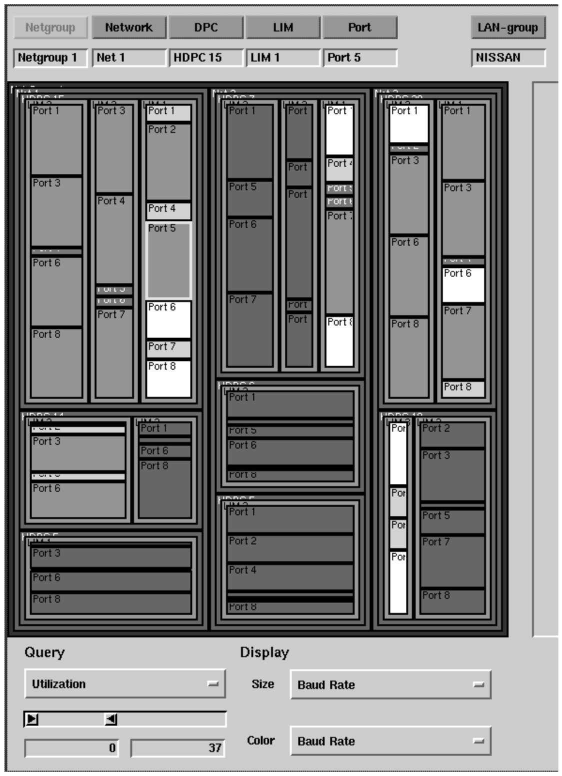

Fig 4e: Choose the "best port", i.e. one whose LIM has low utilization, and which has high baud rate

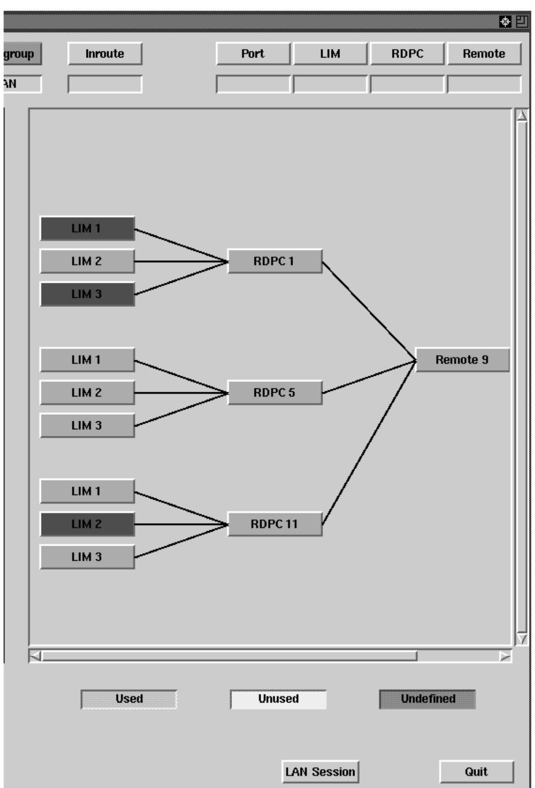

Fig 5a: A remote at LIM level

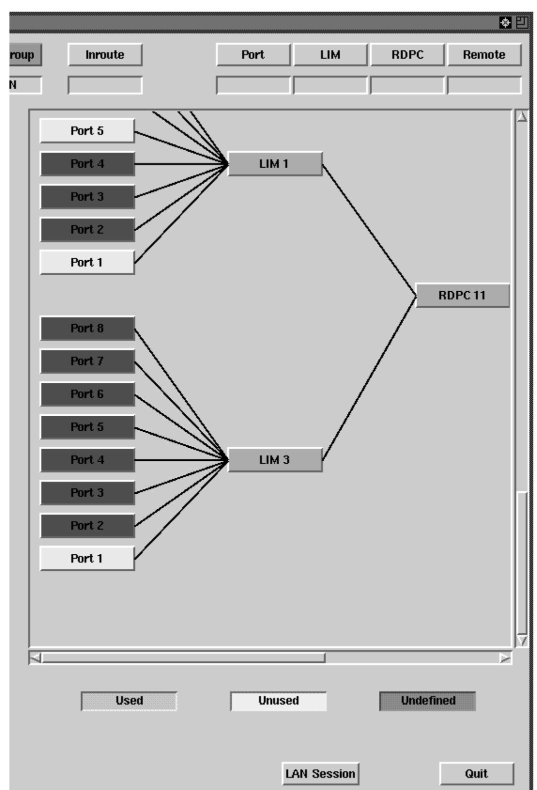

Fig 5b: A remote at port level

Red: Undefined　　Yellow: Unused　　Orange: Used

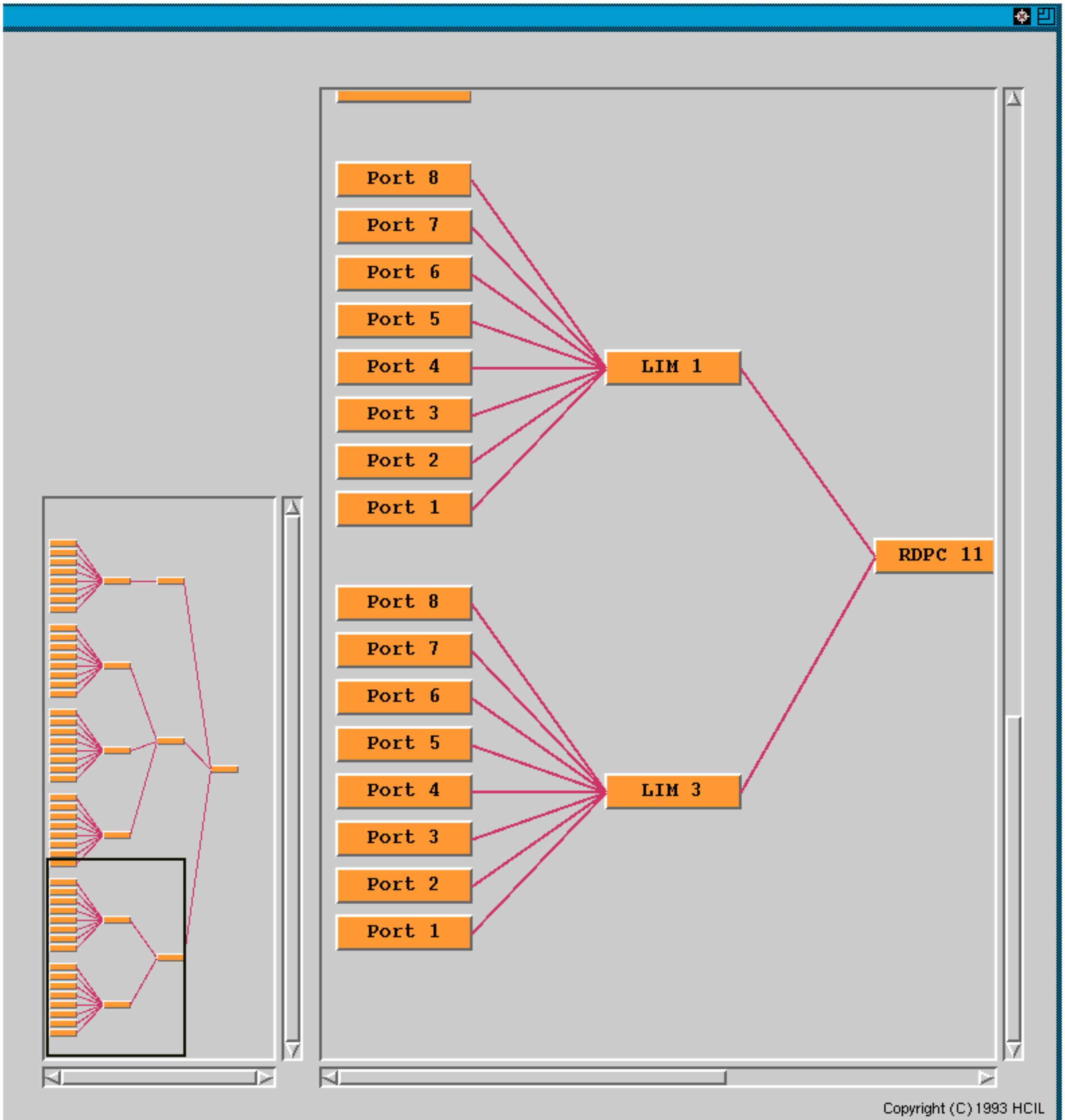

Fig 6: Tree-browser with overview and detailed view tightly-coupled

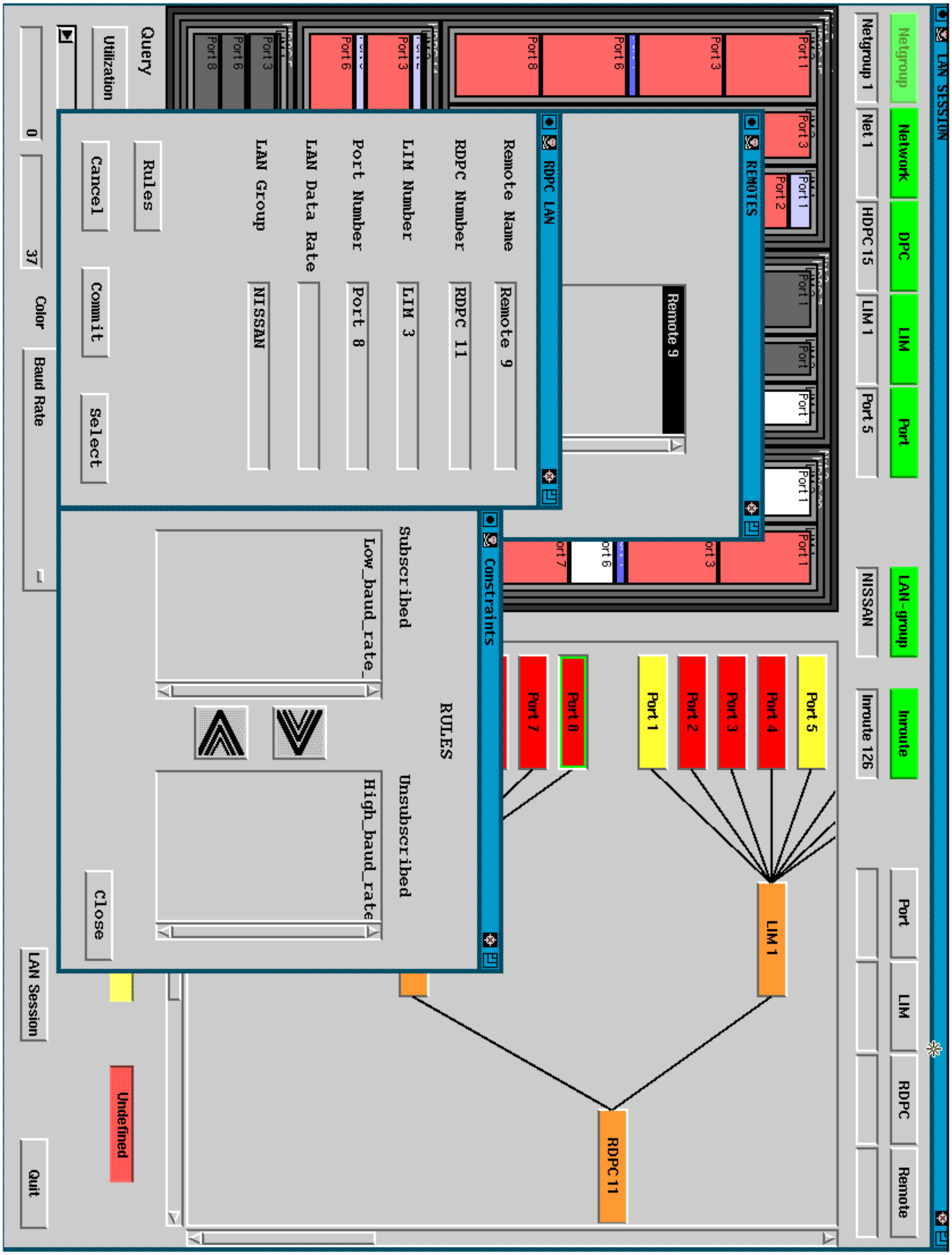

Fig 7: Unified Interface for exploration, querying, data entry and verification