Working Paper Series
ISSN 1170-487X

Tree Browsing

**by Mark Apperley and
Michael Chester**

Working Paper 96/13

July 1996

Department of Computer Science
The University of Waikato
Private Bag 3105
Hamilton, New Zealand

Tree Browsing

Mark Apperley & Michael Chester

Department of Computer Science
University of Waikato
Hamilton, New Zealand

m.apperley@cs.waikato.ac.nz

Abstract

Graphic representations of tree structures are notoriously difficult to create, display, and interpret, particularly when the volume of information they contain, and hence the number of nodes, is large. The problem of interactively browsing information held in tree structures is examined, and a design for a tree browser proposed. This design is based on distortion-oriented display techniques and intuitive direct manipulation interaction. The tree layout is automatically generated, but the location and extent of detail shown is controlled by the user. It is suggested that these techniques could be extended to the browsing of more general networks.

Introduction

This report examines the problem of the interactive browsing of information held in tree structures, and proposes a design for a tree browser. Graphic representations of tree structures are notoriously difficult to create, display, and interpret, particularly when the volume of information they contain, and hence the number of nodes, is large (Shneiderman, 1992; Lamping *et al*, 1995). To facilitate browsing and retrieving information from trees it is desirable that a simple intuitive interface be developed which automatically generates the tree layout, and which allows the user to explore the structure interactively, to identify the relationships between nodes, and to examine the data they contain in whatever level of detail is appropriate for the task in hand.

The problem of displaying large data spaces in the relatively limited areas offered by most computer display screens is one which has received considerable attention over the past 15 years (Leung & Apperley, 1994). The majority of the solutions proposed to this problem involve distorting the original space in order that a section can be viewed in detail while an overview is maintained of the entire area and the correct spatial relationships between all elements are retained. One group of techniques tends to involve graphical distortion, and is rather akin to magnifying or distorting a picture. Another group involves semantic intensity distortion, in which important relationships are highlighted, while those which are of less significance move into the background or are suppressed.

The tree browser described here integrates a set of display or representation techniques based on distortion-oriented display mechanisms already mentioned, together with a set of interaction techniques which are largely intuitive or self explanatory for users of modern graphical user interfaces. A very large tree can be displayed graphically and browsed within the confines of a conventional display screen using this browser. Only those nodes in which the user expresses an interest are displayed in any detail; the remainder are iconised, and in some cases whole sub-trees which are of no current interest are "pruned" or suppressed. As the user alters the view while browsing, then so the automatic layout adjusts the tree to fit best within the available space.

In the remainder of this report, the design of the tree browser is described using as an example the class hierarchy from the Symantec™ C++ library (Symantec, 1995). This hierarchy exhibits the classic tree properties commonly encountered; the number of nodes is large (80 for the section of the library used in the example), the relationship between a node and its sub-tree is hierarchical, a user will often be interested in simultaneously examining in detail two or more nodes in different parts of the tree, and the data contained in the nodes is of interest at several different levels of detail. There are many other examples of tree structures to which these techniques could be applied, some, like this one, fixed in the computer context, and others drawn from a diverse range of disciplines such as genealogy, biology and management. It is also suggested that the majority of the techniques proposed and incorporated in the browser design could be applied equally well to more general network structures. The general principles of these applications have already been demonstrated (Furnas & Zacks, 1994; Schaffer *et al*, 1993).

Distortion-Oriented Display Techniques

The term *distortion-oriented display* relates to a range of techniques for providing simultaneous presentation of detail and overview in a single display. These techniques have evolved to assist in browsing, locating, interpreting and retrieving information from large and complex data spaces. Their aim is to provide the ability to examine one or more sections of the space in detail, while simultaneously maintaining an overview of the entire space.

Spence (1993) has proposed three broad categories into which these techniques can be divided. The first of these comprises the strictly graphical distortion techniques typified by the bifocal display (Apperley *et al*, 1982), the perspective wall (Mackinlay *et al*, 1991), and the graphical fisheye view (Sarkar & Brown, 1992). Leung and Apperley (1994) provide a review of these approaches, all of which involve geometric transformations or variable scaling along one or more axes in the data space. A variety of transformation functions have been demonstrated, ranging in complexity from the piece-wise constant bifocal to the polar fisheye and hyperbolic transformations.

The second category encompasses those techniques where the distortion is not geometric, but is achieved by using different encoding or representations for different objects in the information space. Thus a text document of interest would be shown in full detail, character-by-character. A similar document which was of little current interest might be encoded minimally as an icon, whose colour, shape, and perhaps label, might reflect characteristic properties of the document it represents. Such techniques are commonly used in the representation of directory hierarchies in graphical user interfaces, and indeed permeate many other aspects of window-based interfaces. Spence (1993) has described these as transformations in the Z-dimension, as opposed to the X and Y transformations associated with geometric distortion.

The third group of techniques involve thresholding the significance of items within the data space, and the suppression of those which fall below this threshold. Thus in Furnas's (1986) fisheye views each information element is assigned a value based on its perceived relevance with respect to the current focus of interest. A degree of interest function of this relevance and the distance from the focus is then computed, and if it falls below a specified threshold, the element is suppressed in the display.

The Representation of Trees

A range of distortion based techniques, and other novel representations, have been applied to the display of hierarchically related information or trees. Lamping *et al* (1995) have commented that those techniques based solely on transformations in two-dimensional space are generally unsatisfactory when it comes to representing trees because of the characteristic exponential growth in the number of nodes (see for example Figure 1). TreeMaps (Johnson & Shneiderman, 1991), another two-dimensional representation, progressively sub-divides the space available with each node containing its entire sub-tree. Although TreeMaps cope with rapid node expansion, they too suffer from the drawback that the further down the hierarchy, the less space that is available for the node. TreeMaps are, however, particularly useful for applications such as directory hierarchies, where the "size" of a node is indeed the size of its sub-tree and is a parameter of significant interest.

Because trees tend to be pyramidal in shape, they do not map well onto two-dimensional surfaces. Cone Trees (Robertson *et al*, 1991) address this problem by mapping trees into a three-dimensional space, so that the tree spreads as a cone. Rotation of the cone can be used to bring a node of interest to the front, and so into the detail focus region. Other techniques attempt to accommodate the awkward shape of trees by mapping them onto hyperbolic surfaces (Lamping *et al,* 1995), by using fractal techniques (Koike & Yoshihara, 1993), and by using a space subdivision approach, similar to TreeMaps, but with each node comprising a three-dimensional bubble (Boardman, 1995).

Although they are not directly involved with the representation of trees, other studies have examined the use of Furnas's (1986) fisheye concept in browsing hierarchically structured networks (Schaffer *et al,* 1993; Brown *et al,* 1993). Results suggest that the increased context information provided by the fisheye view over traditional views of such networks significantly improves user's navigation.

All of these techniques share a common disadvantage in that the distortions make the trees less than recognisable, and produce gross shape changes as the focus of interest is moved.

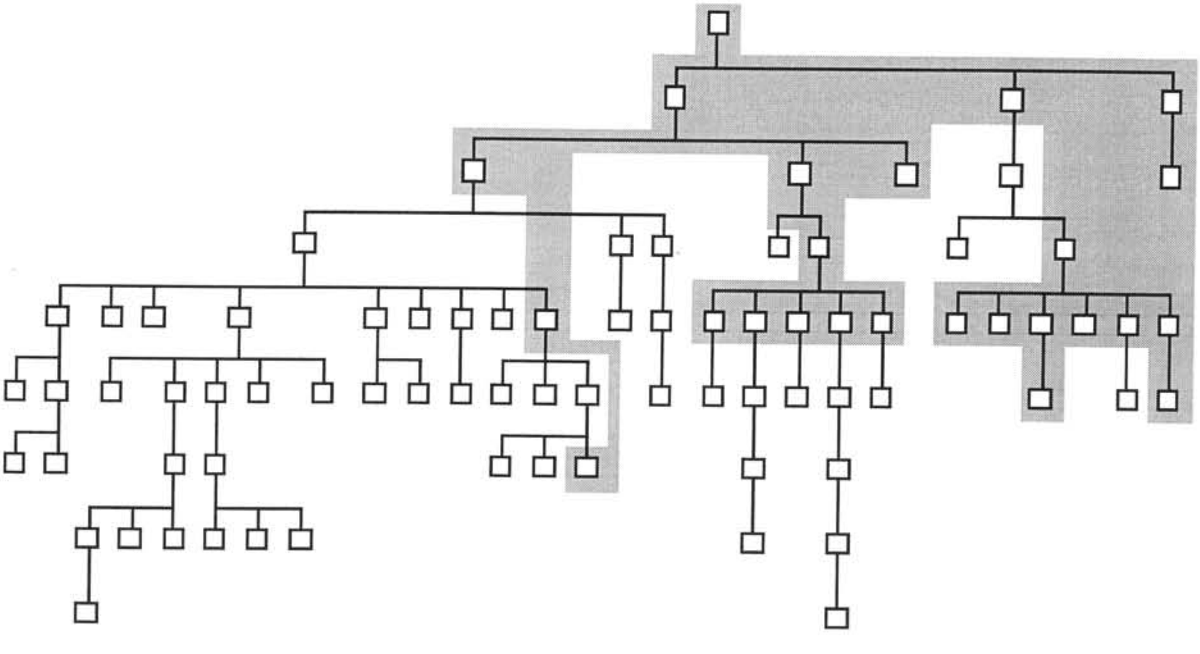

Figure 1: A section of the Symantec™ C++ Class hierarchy, showing the rapid growth in the number of nodes between levels in the hierarchy. The shaded area indicates the nodes which are explicitly represented in the browser window of Figure 2.

The Tree Browser Display

The tree browser design described here attempts to utilise the best features of the three different approaches to distortion-oriented displays described by Spence (1993); geometric distortion, semantic intensity distortion, and thresholding. While it is recognised that these display

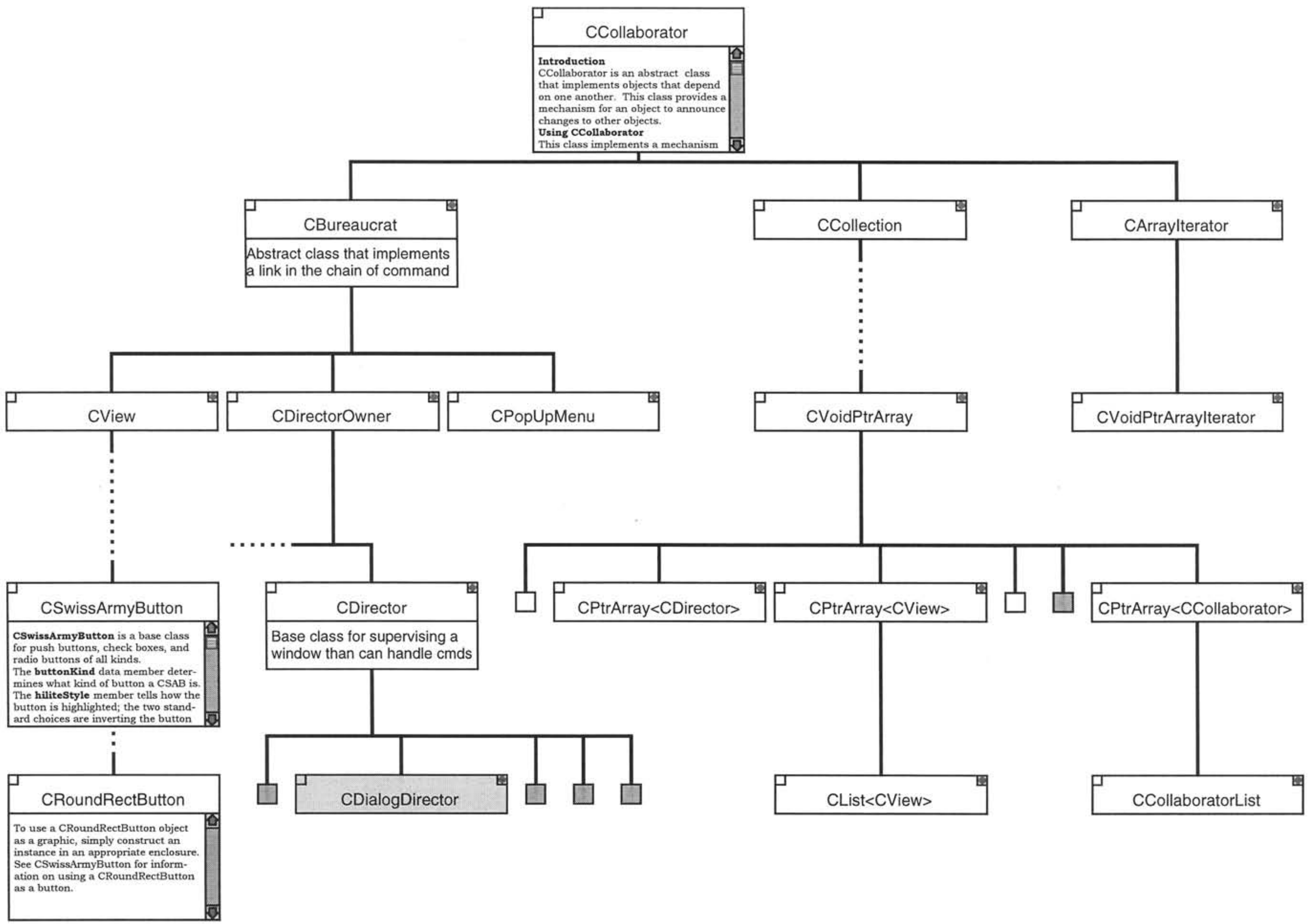

Figure 2: A tree browser display of the section of the Symantec™ C++ class library illustrated in Figure 1.

features have almost all been seen in tree browsers before, this browser leaves the user firmly in control of the distortion parameters, and automatically maintains the tree in a recognisable fashion. Simple and intuitive user interaction with the tree is seen as being vital, so the display and interaction techniques are tightly integrated.

Figure 2 shows an example tree browser display, a representation of the section of the Symantec™ C++ class library illustrated in Figure 1. Remember that Figure 1 contains 80 nodes. Figure 2 shows in detail only those sections of Figure 1 contained within the shaded area; nevertheless, the user retains an overview of, and access to, the entire tree. The user could have started with the view of Figure 1, and by a series of interactions, have arrived at Figure 2. In this example the user is attempting to find an appropriate class from which to begin building a new interaction button.

Automatic layout generation for trees has been addressed by others (Battista et al, 1994; Eades & Sugiyama, 1990). However, of particular concern in this application is the need to maintain consistency of node position and relationships and overall tree shape as browsing proceeds. As will be seen from the description that follows, entire sub-trees may disappear or reappear with a single button-click. A layout algorithm has been developed which provides good consistency and minimal positional change (Chester, 1995).

There are a number of different node representations included in Figure 2, as well as some arc constructions which require explanation. Figure 3 illustrates the Z-transformations, or semantic intensity distortions which can be carried out on nodes, giving rise to the differing node appearances in Figure 2. Figure 3(a) shows an "iconised" node; there is no label or descriptive information provided, although different icons could be used to represent different types of node in a non-homogenous tree. In Figure 1, only iconised nodes were used, and because of the space saving, the entire tree could be represented on the display.

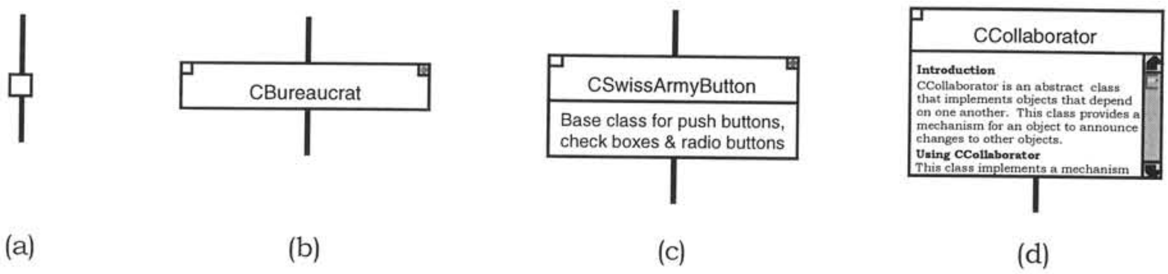

Figure 3: The hierarchy of node detail provided by the tree browser.

Figure 3(b) shows a labelled node; at the expense of screen space, a unique identifying label for a node can be displayed. At the next level of

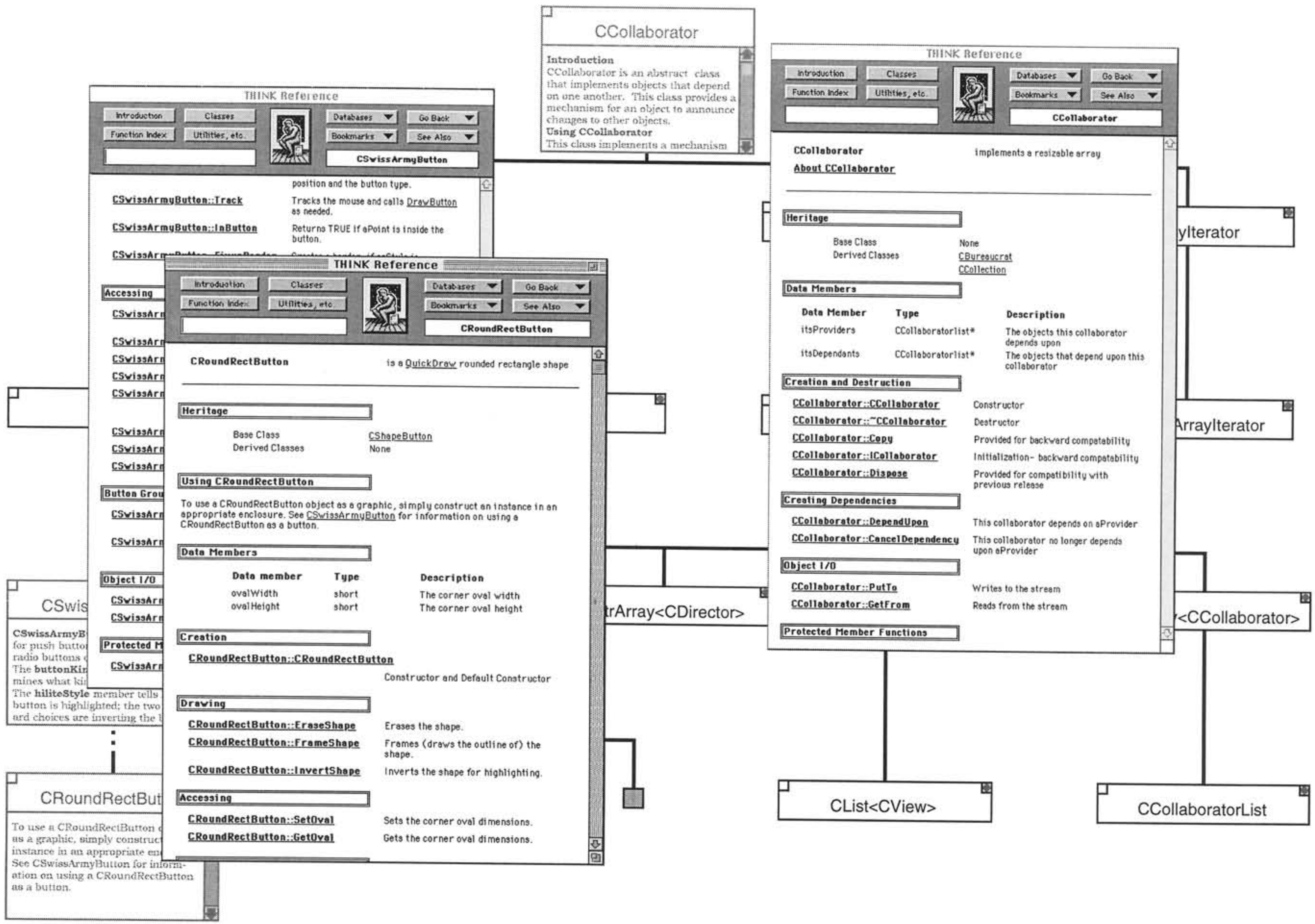

Figure 4: Three nodes from the tree of Figure 2 have been opened as windows, enabling their descriptions to be studied in detail.

detail, Figure 3(c), the label is augmented by a brief abstract, in this case describing the purpose of the class represented by the node, and finally at the highest level of detail. Figure 3(d), a comprehensive description of the node is provided in a small scrolling window.

It should be noted that these four representations are just views of a node at different levels of detail. Any node can be moved between these levels of detail under user control; the appearance of the node changes *in situ*, with all tree relationships remaining intact.

There is one further representation of a node which the tree browser provides. At any of the above four levels of detail, a node can be "opened" into a window (see Figure 4). In this case a new window is opened for the node, and the most detailed descriptive information is displayed. The node window so created may be superimposed on the browser window, but it is an independent window that can be moved, scaled or closed under user control; the original node symbol in the tree remains.

Thresholding or suppression of parts of the tree is also evident in Figure 2; not all of the original 80 nodes are represented. Those parts of the diagram which represent or suggest suppressed items are reproduced in Figure 5.

Figures 5(a) and 5(b) show "stumped" versions of iconised and labelled nodes respectively. These result from sub-tree pruning; the entire sub-tree emanating from the node in question has been flagged as being of little interest to the user at the present time, so is not displayed. It is better to consider the sub-tree as having been collapsed into the root rather than cut off; it still exists, but is not displayed. Pruned sub-trees can be easily restored by the user at any time; considerable savings of screen real estate can be achieved through this means, as is evident from Figure 2.

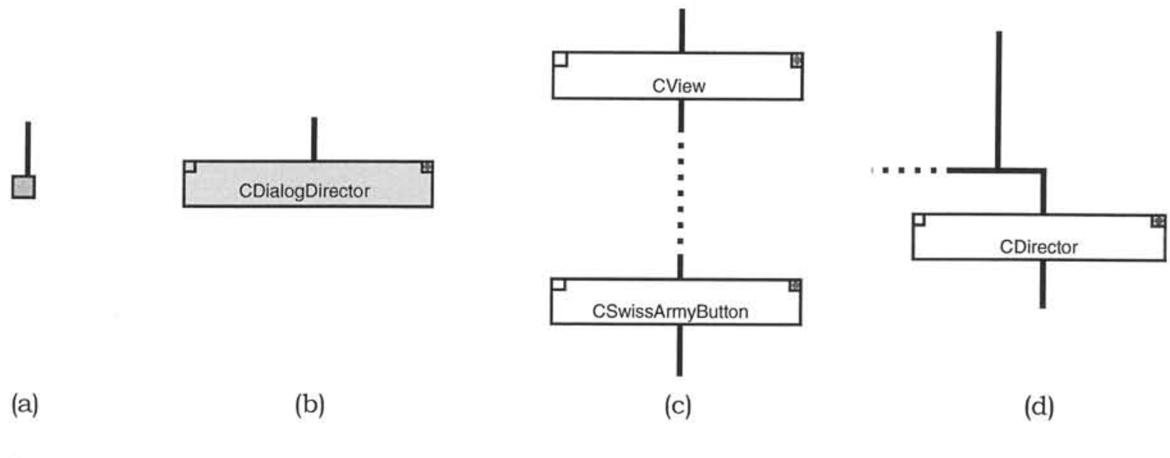

Figure 5: The representation of thresholding or suppression in the tree.

Stumped nodes allow for the suppression of entire sub-trees. Often, however, the user does not wish to suppress a sub-tree, but the detail that comes between a node and one of its descendants. This is the situation described as "vertical telescoping" and represented in Figure 5(c). In this case, all nodes between the two in question are suppressed, including the siblings of the suppressed nodes (and their sub-trees) and the siblings of the target descendant node. Any children of this latter node will still be shown, unless suppressed by pruning.

A further form of suppression is required when some but not all of the children of a given node are of interest. This "horizontal telescoping" is represented in Figure 5(d). Here there is just one child node which is to be followed; its siblings (and their sub-trees) are suppressed.

Interacting with the Tree

As much as possible, interaction with the tree is provided with simple and intuitive mouse actions, directly on the nodes or braches of interest. For example, movement between the four levels of node representation is achieved via expand and contract boxes in each of the icons. The smallest "iconised" node (Figure 3a) is expanded to a "labelled" node (Figure 3b) by simply clicking on it; it can only get bigger. A labelled node has both an expand box (top right) and a contract box (top left) as shown in Figure 6. Display of the full window for a node (see Figure 4) is achieved by a double-click action on any of the four node forms.

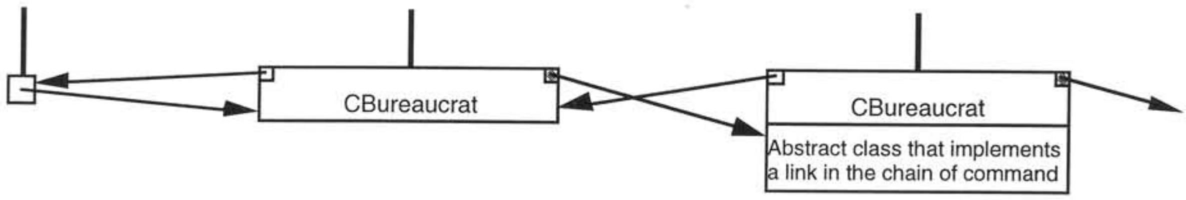

Figure 6: A single mouse-click is required to move a node from one level of detail to another.

Pruning is achieved through a combined keyboard-mouse sequence. When the option key is held down, the cursor changes in appearance from the browser icon to a pair of snips. With the cursor in this state, clicking on the descending arc from a node causes that node's sub-tree to be pruned. With the same cursor, clicking on a stumped node causes its sub-tree to be redisplayed.

Vertical telescoping, as shown in Figure 5(c), is achieved by a simple dragging action between the two nodes. The cursor is placed over the lower of the nodes, the mouse key pressed, and the cursor dragged to the ancestor node. As the dragging takes place, then so those parts of the tree that will be suppressed are greyed as the cursor crosses each node. When the target ancestor node is reached, and the mouse key released, then the unwanted part of the tree disappears. The vertical distance between the two selected nodes is reduced and a dotted arc appears between them. The suppressed section of the tree can be restored to view by a double-click action on the dotted arc.

Horizontal telescoping, where a number of sibling nodes and their descendants are suppressed, is handled in a similar fashion. However, in this case it is not a simple dragging action between two nodes that is required, but the tracing of a path which passes through each of the nodes to be suppressed (Spence, 1995). Figure 7 shows an example where a path is traced through three non-adjacent nodes. When the mouse button is released, the sub-tree reduces to the one shown in Figure 7(b), with the three selected nodes suppressed. Restoration of the suppressed nodes is achieved by a double-click action on the dotted stump extending to the left in Figure 7(b).

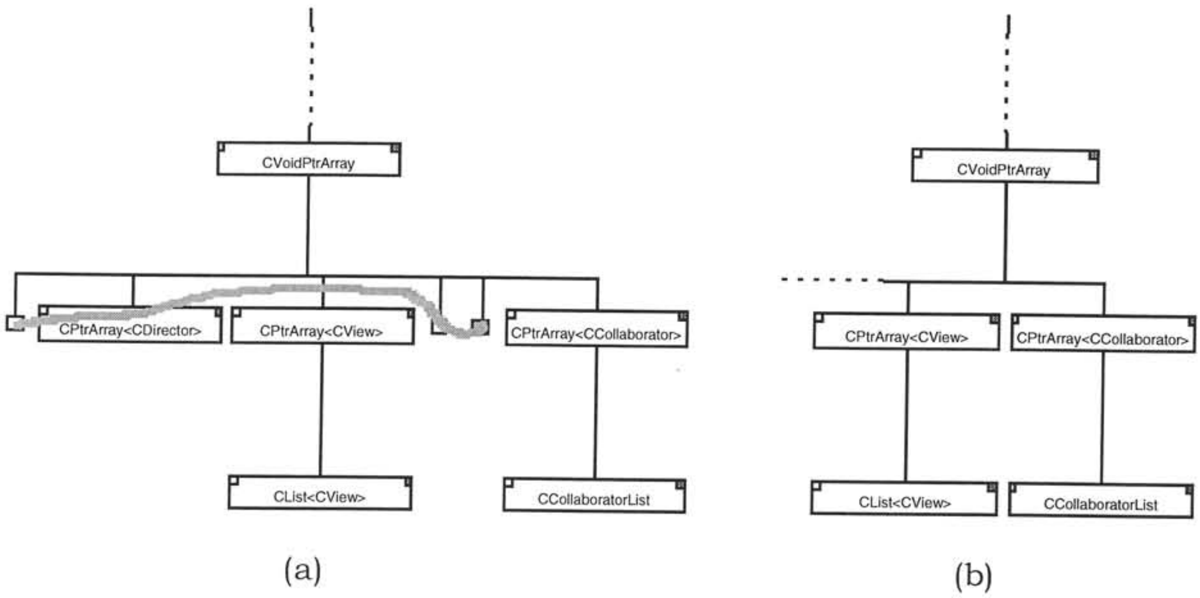

Figure 7: (a) A path is traced though a group of sibling nodes, (b) resulting in their suppression.

Implementation

A demonstration prototype of the tree browser is currently being implemented in Symantec™ C++ on a Power Macintosh™ computer.

Conclusions

This report has described a design for a tree browser based on distortion-based representations, automatic tree layout, and well-known and intuitive interaction techniques. Although the demonstration prototype system is not yet complete and no usability testing has been carried out, experience with similar techniques in other applications suggests that users should be able to locate items and information of interest more quickly than with conventional undistorted tree representations.

It is also suggested that the majority of the techniques proposed here for browsing trees could readily be extended to deal with more general network structures.

Acknowledgments

The research described in this report was carried under a sub-contract to the Massey University FRST-funded project "Advanced Information Technologies Through CASE". The contributions of Robert Spence and Paul Lyons in formulating the browser concepts are gratefully acknowledged.

References

Apperley, M.D., Tzavaris, I. & Spence, R. (1982): A bifocal display technique for data presentation. *Proceedings of Eurographics '82*, 27-43.

Battista, G., Eades, P., Tamassia, R. & Tollis, I.G. (1994): Algorithms for drawing graphs: an annotated bibliography. *Computational Geometry Theory and Applications*, **4**, 235-282.

Boardman, R. (1995): *Visualisation of Information Trees*. ISE3 Individual Project Report, Information Engineering, Imperial College.

Brown, M.H., Meehan, J.R. and Sarkar, M. (1993): Browsing Graphs using a Fisheye View. *INTERCHI '93 Proceedings*, 516.

Chester, Michael (1996): *The Display and Manipulation of Abstract Trees*. 420 Progress Report, University of Waikato, Department of Computer Science.

Eades, P. & Sugiyama, K. (1990): How to draw a directed graph. *Journal of Information Processing*, **13**(4), 424-437.

Furnas, G.W. (1986): Generalised fisheye views. *CHI '86 Proceedings*, 16-23.

Furnas, G.W. & Zacks, J. (1994): Multitrees: Enriching and Reusing Hierarchical Structure, *CHI '94 Proceedings*, 330-336.

Johnson, B. & Shneiderman, B. (1991): TreeMaps: A Space-Filling Approach to the Visualisation of Hierarchical Information. *CHI '91 Proceedings*, 401-408.

Koike, H. & Yoshihara, H. (1993): Fractal approaches for visualising huge hierarchies. *Proceedings of 1993 IEEE Symposium on Visual Languages.*

Lamping, J., Rao, R. & Pirolli, P. (1995): A Focus+Context Technique Based on Hyperbolic Geometry for Visualising Large Hierarchies. *CHI '95 Proceedings*, 401-408.

Leung, Y.K. & Apperley, M.D. (1994): A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Transactions on Computer-Human Interaction*, **1**(2), 126-160.

Mackinlay, J.D., Robertson, G.G. & Card, S.K. (1991): The Perspective Wall: Detail and context smoothly integrated. *CHI '91Proceedings*, 173-179.

Robertson, G.G., Mackinlay, J.D. & Card, S.K. (1991): Cone Trees: Animated 3d visualisations of hierarchical information. *CHI '91 Proceedings*, 189-194.

Sarkar, M. & Brown, M.H. (1992): Graphical fisheye views of graphs. *CHI '92 Proceedings*, 83-91.

Schaffer, D., Zuo, Z., Bartram, L., Dill, J., Dubs, S., Greenberg, S. & Roseman, M. (1993): Comparing Fisheye and Full Zoom Techniques for Navigation of Hierarchically Clustered Networks. *Proceedings of Graphics Interface 93*, 87-97.

Shneiderman, B. (1992): Tree Visualisation with Tree-Maps: 2-d Space-Filling Approach. *ACM Transactions on Graphics*, **11**(1), 92-99.

Spence, R. (1993): *A Taxonomy of Graphical Presentation.* Information Engineering Section Report #93/1, Imperial College.

Spence, R (1995): *The Care and Maintenance of Trees.* Private Communication.

Symantec (1995): *SYMANTEC C++ Development System for Power Macintosh: User's guide and reference.* Symantec Corporation, Cupertino, CA.