Copyright 2001 SPIE

Published in the Visual Data Exploration and Analysis Workshop, Proceedings of SPIE 2001

January 2001, San Jose, USA.

Personal use of this material is permitted. However permission to reprint / republish this material for advertising or promotional purposed or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the SPIE and the authors.

Visual Information; Amplifying and Foraging

Claire Knight[†] and Malcolm Munro

Visualisation Research Group,
Research Institute in Software Evolution.
Department of Computer Science,
University of Durham,
Durham, DH1 3LE, UK.

ABSTRACT

Visualisation is an important weapon in the management and control of the vast flood of data now generated. In order to be effective and useful it is important that such visualisations are designed to accommodate the variabilities of the tasks to which they will be put, and for the data they will be expected to be able to display. Such a view necessarily means that not all visualisations are always applicable. To this end, work has been done on visualising software and systems with the aim of creating intelligence amplifying tools that aid, rather than try to replace the user and his intuition and domain knowledge.

Keywords: Software Visualisation, System Visualisation, Program Comprehension, Intelligence Amplification.

1. INTRODUCTION

Visualisation research has grown and matured quickly in the last decade. Despite these advances, and the well known information overload problem experienced by many, visualisation research is still an area with problems to solve and issues to address. This is not a view held by all. Much of the early visualisation work (for example [2]), which was very successful in opening new arenas and making obvious some of the benefits of visualisation, is now relied upon to be able to solve all visualisation problems. An example of this is the Cone Tree [5] approach for displaying hierarchical data. It is a wonderful approach assuming that the data is hierarchical, and that the tasks required from the visualisation are to manipulate and view that structure. Unfortunately for proponents of such reuse, this is not always the case; both from a data and a user perspective.

There are many factors that are important for visualisation. The main ones in the context of all forms of information visualisation (especially those considering inherently intangible data sources), are:

- Navigation, Interaction, and Orientation
- Scalability
- Evolution
- Automation

Navigation, interaction, and orientation are presented together because they are often highly interconnected. Orientation features are also very often useful for navigation and vice-versa. Interaction mechanisms are also affected by the navigation available to users. The final three bullet points cover issues that should be considered vital for all visualisations, including those that are also used to create virtual environment visualisations. Obviously there may have to

[†] Correspondence should be directed to the first author. E-mail: C.R.Knight@durham.ac.uk. Tel: +44 191 374 2554.

be trade-offs for a particular system with respect to some of these features but striving to achieve all of these goes some way towards reliably creating useful and usable visualisations.

The selection and use of metaphors is not without controversy, not least because the considered suitability of any metaphor is a very subjective issue. As with all representations and interfaces, the level of familiarity often affects how much at ease the user is with them. This obviously means that for any visualisation and hence metaphor and mapping, there is a period of time whilst that mapping is learnt and understood but after that time the nuances and even irregularities of the metaphor become as common to the user as any other interface they use.

This research has acknowledged the difficulties and differences with visualisations and therefore has tried to address certain identified problems with the visualisations developed. In doing this, there has still been the ultimate aim of creating visualisations that can act as information foraging (IF) and intelligence amplification (IA) aids. This can be seen when the example visualisations are presented.

2. IA AND IF

Intelligence amplification (IA) is the use of computers to aid and enhance human intelligence rather than the artificial intelligence (AI) aim of trying to substitute humans with computers. Intelligence amplification builds on the skills that humans already have, and tries to augment the areas that are lacking in some way.

Information foraging (IF) in the context of this work is a broad term that is used to refer to any of the activities (for example browsing or querying) that take place when a user is looking for information within the given datasets. It is usual for the exact information required to be an unknown in this process, and that the *foraging* activity is being carried out in response to a more complex question or task than the one piece of information can answer.

2.1 Intelligence Amplification

Frederick Brooks (documented in by Rheingold [8]) describes his beliefs about intelligence amplification in the following way

> *"I believe the use of computer systems for intelligence amplification is much more powerful today, and will be at any given point in the future, than the use of computers for artificial intelligence (AI). In the AI community, the objective is to replace the human mind by the machine and its program and its data base. In the IA community, the objective is to build systems that amplify the human mind by providing it with computer-based auxiliaries that do the things that the mind has trouble doing."*

Brooks identifies three areas in which humans are more skilled than computers. The first is *pattern recognition* (aural or visual). The second is in performing *evaluations*, and the third is the *overall sense of context* that allows previously unrelated pieces of information to become related and useful in a new situation. Walker [9] also touches on the subject of intelligence amplification in his discussion on the challenges of visualisation.

> *"A natural and intuitive visual interface can retain the critical contribution from human perceptual skills, ensuring that opportunities for lateral thinking or perhaps an unexpected leap of imagination are not lost. Programming a computer to "look for something interesting" in a database is a major undertaking, but given appropriate tools, it is a task for which humans are well equipped."*

The first sentence can be seen to be similar to the third skill identified by Brooks, that of a sense of context. The second sentence by Walker is essentially talking about the pattern recognition skill specified by Brooks (in [8]).

Intelligence amplification is of importance to software visualisation (and any other form of visualisation) because in representing large and complex data sets graphically the aim is to help the user to get a better understanding of content of the data sets. By aiding the user in this way visualisation tools are acting also as intelligence amplification tools. Reading through many thousands of pieces of information and then summarising them in a finite graphical space would be an

immense, complex and possibly tedious task. For a computer with the right "instructions", it is a simple data processing exercise.

The role of a visualisation system as an intelligence amplification tool rather than as a system that tries to second-guess the information the user requires is emphasised by Crossley et al. [10]:

> *"…the role of the system is not to select documents similar to a user-supplied query but to organise and display information about many documents in such a way as to assist users to select useful documents on their own."*

This shows that the important challenges and research issues for visualisations are to be able to handle such tasks well and provide the necessary support as transparently as possible. Changing the query mechanism in order to improve performance (for example in the situation above) is not going to help in another situation or be widely applicable to other visualisations.

2.2 Information Foraging

Several data mining techniques have been suggested for integration with virtual environments in order to alleviate some of the problems associated with information location and comprehension (information foraging, IF) [6]. There is already a body of visualisation literature that deals with the visual representation and display of the results of mining large data sources, such as visual query environments. The use of the term IF is made because of the distinction between systems producing a subset of their data for user review based on various mining algorithms and the view preferred by the authors of using the visualisation systems to uncover the relevant data in context, thus creating information.

Chen [7] refers to information foraging theory being based on ecological perspectives from studies on information-seeking behaviour. Such strategies were originally developed in biological and anthropological work, and information foraging is adapted from optimal foraging theories. Essentially this focuses on a trade-off between information gain and retrieval cost. Despite this detail Chen also acknowledges that information foraging is a broad term and can be associated with a wide variety of assessing and seeking activities.

As defined above, in this research, information foraging is used to refer to any of the activities that take place when a user is trying to locate some information within the dataset for the purposes of achieving some goal of fulfilling a task. The activity breakdown (query, browse, …) is not considered as it is felt that such distinctions do not add to the use of the term, or the support that visualisations should provide to facilitate such activities. By supporting many activities the information foraging activities can all take place as the user sees fit to utilise the visualisation; this is turn leads to visualisations that can be perceived as intelligence amplifying.

2.3 Foraging to Amplify

IF and IA can be considered to be linked terms, at least in the context of visualisation. This is because in order to facilitate IA, visualisations are produced that allow for the various activities that can be associated with IF. In this way, IA tools are created through enabling IF activities to take place in the context of complex and large dataset representations.

It could also be suggested that these terms are linked at a fundamental level because if a visualisation is hard to use (from a perspective of locating information, interface usability aside) then it is not something that augments a user's working practice. In fact in such cases it is highly likely that the tool becomes a hindrance and then is not utilised. If visualisations (or any tool) allow for a range of foraging activities in a supportive manner this is when the aim of IA can start to be achieved.

3. SYSTEM AND SOFTWARE VISUALISATION

In addressing identified problems within the program comprehension and software maintenance arenas it meant that certain assumptions could be made as to the type of tasks and the range of information required. Whilst this approach

would not enable the development of a "silver bullet" (for the interested reader see [1]) of visualisation it does allow for the development and refinement of techniques for intelligence amplification and information foraging.

Software visualisation is a powerful way of viewing the many and varied artefacts that composed together form a computer system. It also has much power and use in the representations of the complex interrelationships between these artefacts. Software visualisation is very much concerned with the visual representation of not only the data items, but also the relationships between them that can be large in number, variety and complexity. To add to this problem, each is different in their importance depending on the task that the visualisation is being used for. This is very much a metaphor and representational issue, especially when considering three-dimensions and as yet (if there ever will be) there is no "recipe book" set of steps that can be followed to lead to the creation of appropriate visualisations.

3.1 Software World

The *Software World* visualisation [3, 4] is a virtual environment based visualisation that represents a static analysis of Java source code. These data items are then visualised for the user to explore and query as necessary to be able to better comprehend the code and then perform some comprehension or maintenance task. Since *Software World* is targeted at the visualisation of Java code, the levels of metaphor mapping are tailored to suit the language. The language definition of Java allows certain assumptions to be made and these have influenced the decisions made.

3.1.1 Basic Mapping Decisions

Software World is based on a real world metaphor that encompasses city objects and urban development ideas. The main reason for this choice of metaphor was the power it afforded in scaling so that software systems of varying sizes could still be accommodated coherently by the one metaphor and related group of representations.

For ease of implementation (and user navigation) the world is considered to be a flat plain rather than a sphere. This also allows the use of Euclidean Geometry, an accepted standard for architectural (and VR) purposes. The world has different levels, which can be briefly described as

- World; flattened, overview picture, atlas style, not necessarily countries as would be known in standard geography but shows different elements of the visualisation at a very high level and the relationships between those elements.
- Country; each element shown in the world view is a country. It provides a way of splitting the items in the world down one level without the detail that is provided by the next level down.
- City; shown within countries, as the next level of granularity. These cities are composed of sub-areas but try to ease the navigation burden through the use of standard urban navigational aids (covered in more detail below).
- Districts; there can be several of these in a city, the number depending on the information to be represented in the visualisation. They group together related aspects of the software and provide groupings to be used when moving from a higher level of abstraction to a more detailed level.
- Streets/Buildings/Gardens/Monuments; these show the detail of the visualisation and provide the next level of abstraction down from the districts. They also act as legibility features and landmarks of the city.
- Inside Buildings/Gardens; this is the finest level of detail, where detailed direct mappings from the code to the visualisation can be made.

Detail of the mapping is available in [11], but for demonstrating use of the visualisation they are not specifically considered here.

3.1.2 Foraging within Software World

This is presented from the point of view of looking to solve a common software engineering question; what is the impact of making this change. The process of using the visualisation is presented, as well as having images from a *Software World* visualisation generated during the process of visual representing the results of a static analysis of over 17,000 lines of source.

The actual process of comprehension and the use of auxiliary comprehension tools are not covered as it is assumed that the reader can make judgements about the extra information (if any) they need to complete the task. It is also because people differ in their approach to this sort of task and to dictate a way of working would drastically reduce the overall effectiveness of the tool in a wider view. The visualisation is there to support the process of comprehension in any way the user sees as beneficial to them, their way of working and the task in hand.

It is important to consider impact when making any change to the code. It is highly unlikely that the change will be completely isolated, especially if it requires change to data items (variables and data structures). Impact analysis is something that is a major consideration when doing any form of program comprehension and having tool support in this area, even if a human needs to make the final judgements, is supportive to that process.

Problem

An example task that falls into the category of impact analysis is when a change involves altering the type of a class variable. Such a task may impact any code contained within that class, and should the variable be static (regardless of whether or not the class is static) it could have much wider impact throughout a system. All known usage of this variable needs to be examined to see how the change may affect the rest of the code. This is turn may require restructuring of other areas of code but for the moment this will not be considered for the scenario.

Solution

Assessing the possible impact that altering the type of a class variable would cause;

- There is an assumption that the class is known in this case and that the start of the process is to search for that class using the class and package name. From this the class is highlighted in both the world and country views. The class can then be selected and the outside of the district within the city boundary is shown.
- The user can then go into the district and find the monument that represents the variable being considered for change. This monument would then provide information about the current type and use of that variable. An example monument is shown in Figure 2, whilst a wider district view can be seen in Figure 1.

Figure 1 - Scene from Software World showing central garden, with varying sizes of buildings in the foreground

Figure 2 - Monument in *Software World*

- From here the usage of the variables locally can be examined by visiting each of the (possibly) affected buildings representing the methods that contain usage of this variable. If the variable is static and is used in the wider scope of the system then the searching facility can be used to explore these other areas.
- Detail of impact where the variable is used can then be judged (for methods) through the use of the source code and method information presented in the buildings. For other areas of code the package, class and method in which the usage is located can all be used to build up a picture of the impact. Figure 3 shows the various pieces of information available for a variable. The actual image is of a desk in building, showing local variable information but the same issues are important whatever the type of the variable. This image shows the name and type of the variable, whether it is an array, the line number it was declared on, it's usage, any initial value, and the scoping information (which is not as pertinent to class variables).

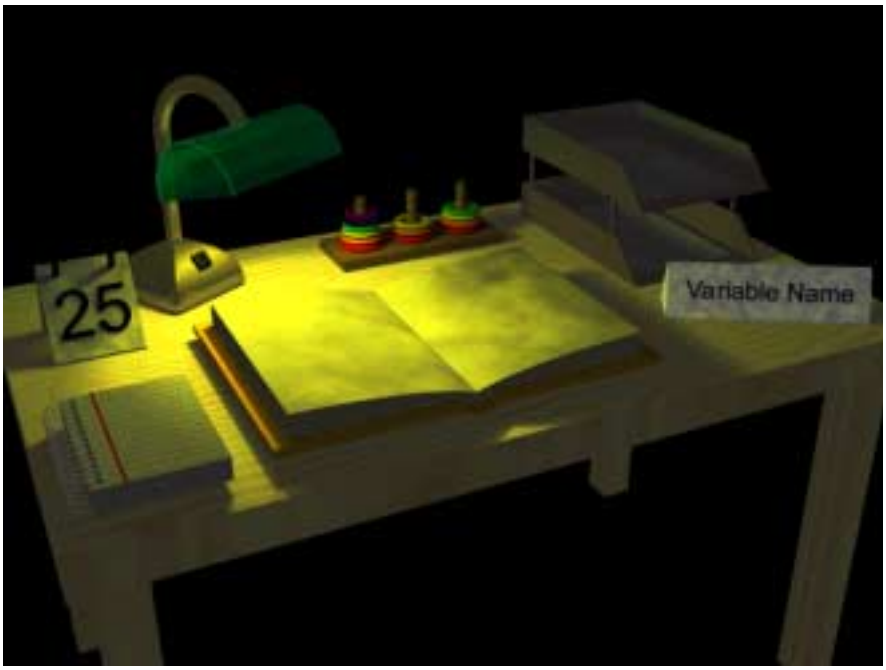

Figure 3 - Variable information available

This scenario provides an illustration of the goal directed investigation that the visualisation supports. It also shows how it is difficult in the graphical context of this mapping to explicitly show the syntactic and semantic relations between data items; this would perhaps make the tracing of class variable usage easier. The scenario solution description also provides an indication of the various interaction mechanisms that the visualisation supports.

3.2 System Ribbons

An alternative view of systems, and taking a wider stance than of just visualising code, has been developed in conjunction with some industrial partners. This focus has been on the management of systems as a whole; code, functionality (at a business level), future plans, and organisational directions. This visualisation uses the three-dimensions to not only create code based visualisations within a certain sub-part of the visualisation (where all three of the x, y, and z co-ordinates are important), but also to show the progress of time and the system at various points through that time. Again, this visualisation focuses on providing views of the data that the user can then interrogate and manipulate in order to retrieve the necessary information. It also aims to make easily accessible the directly related information at any stage, thus providing an environment that is carrying out the hidden cross-referencing to be able to aid the user transparently.

A good reason for visualising this information is that it allows for the integration of various filtering and overview mechanisms to be used to be able to grasp limited portions of the data. It also allows for the detail to be seen when necessary. An added benefit is that visualisation allows these filtering/detail decisions and the data focus to be user driven. Other techniques, for example data mining, seek to find what the algorithms think of as interesting patterns in the data and then present these to the user. For management of complex systems such as software this may not be adequate, as the underlying data factors may not accurately reflect the management needs.

3.2.1 Views of, and Management of, the Information

The images seen in Figures 4 and 5 show an example of a system visualisation at the highest level. These images are part of a sequence of events/action and are described in the following text. The system objects contained within the images could be for example point of sale systems in a retail business.

Figure 4 shows one system object across two snapshots of the system (represented for illustration purely as a sphere within the constraints of the system visualisation). The bounding boxes in show where the system visualisation exists when looking in detail at this part of the time stack. Between the two system object instances (one in each snapshot) is a time ribbon that connects the object through the snapshots, the forthcoming release (shown as the transparent plane) and higher through the visualisation (not shown). There are three system objects traced through the system snapshots, each with an individual time ribbon. There is also the plane at the top, which for the purposes of this visualisation is the highest part of the time stack that is displayed. This plane represents a fixed date in the history/future (depending on the viewing date) of the system; this could be the introduction of a major new customer loyalty scheme. The sphere has two predicted future time paths shown to be starting from it. These predicted paths do not have any more instances of the system object attached since this is not the route that was either taken or forced upon the object. These paths existed from the time of the first visible system snapshot, and there were three ways in which the system could have progressed. As it was, one of these was correct and the other two exist only as historical information to guide future decisions. An example application of these paths is the possible routes to be taken with the introduction of a customer loyalty scheme. The one chosen may have been the most viable for business reasons at the time, but it is important to retain the rejected routes for information should the business or market change and require that the loyalty scheme also change.

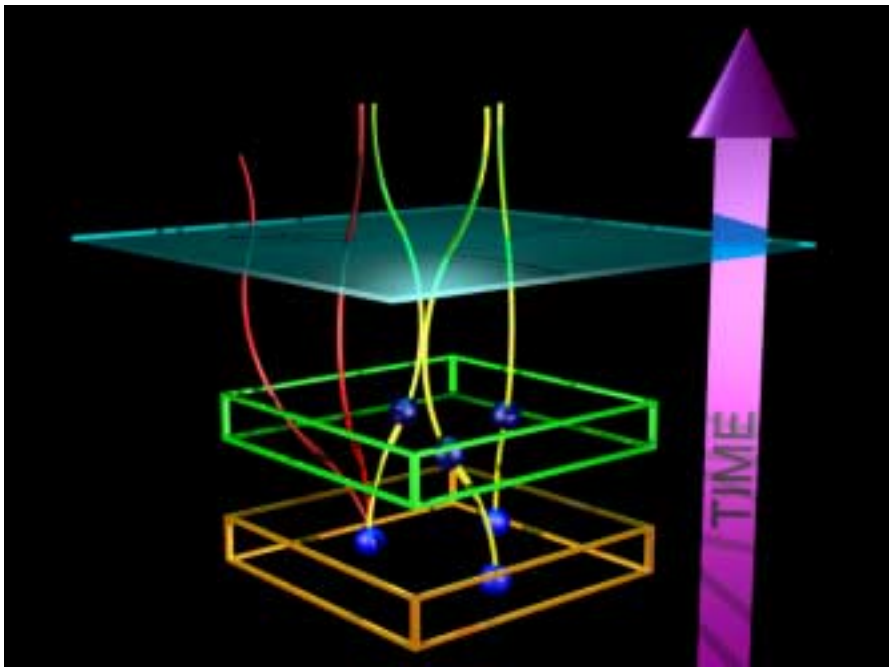

Figure 4 - System visualisation showing several system snapshots (in overview) through time

Figure 5 shows the same visualisation, but with the insertion of an extra plane between the last system snapshot before the next release and the plane that represents the release. The arrow shows where the plane is to be inserted. It can be seen that this will cut through the time ribbons, and then for analysis the system objects at the point of insertion can be traced from further down the ribbon. The insertion of the plane at this point would allow for business analysis on different forms of customer loyalty schemes at this point in time. It is also the case at the system snapshot (below the plane) that there are many system objects that are not currently visible. It is also the case that these system objects have their own histories, dependencies, and impacts. These are seen when the system is viewed in detail. They can also be seen at the inserted plane when it is viewed in detail as they are then projected up.

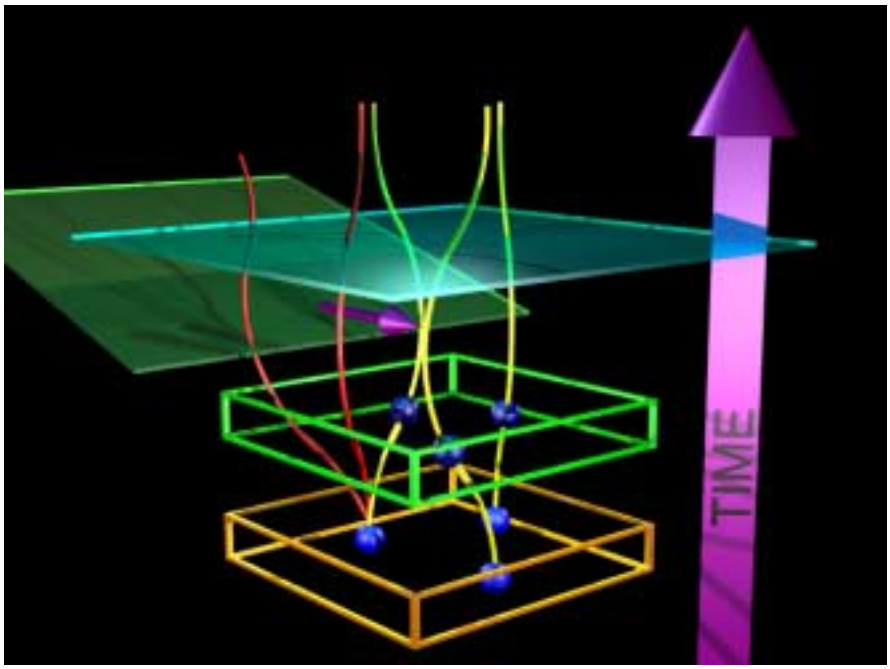

Figure 5 - Inserting a query plane at the specific time point

This section has shown from another side of system and software visualisation how visual representations can be used to provide information location and analysis (foraging) and management and comprehension of the information presented (amplification). Processing tasks such as data location, aggregation, and visual display are provided by the computer. This then frees the user to interpret and managed the data presented to them and to therefore utilise the information obtained from the visualisation.

CONCLUSIONS

Using restricted domains for the investigation and refinement of visualisation techniques allows assumptions to be made which can simplify certain aspects of the problem. This approach is vital to be able to further the techniques, but it is also important to realise the impact task and data have on visualisations. Therefore visualisation researchers should be aware of the limitations of their approaches and where they are more widely applicable. They must also acknowledge such.

It is important to remember that the underlying data set influences the applicability of a metaphor. *Software World* works well for Java because of the strictly enforced hierarchy. Without this, either a different metaphor or the way in which it is mapped would need to be used. It seems that, at least at the moment, that the metaphor construction and mapping are an art form. This is not necessarily ideal, but by using guidelines to guide the process towards using ones that deal with contextual visualisations, navigation and orientation, different levels of abstraction, aiming towards successful IA and IF and a structure suited to the data, then the poorer representations and metaphors will be weeded out at an earlier stage.

This paper has shown two visualisations from the program comprehension and software maintenance domain. These visualisations have been used to show that despite the variability of task and data which undoubtedly affects visualisations (and certainly should be seen as having an impact), it is possible to create visualisations with common goals; in this case IA and IF.

ACKNOWLEDGEMENTS

This work is financed by an EPSRC ROPA grant: VVSRE; **V**isualising **S**oftware in **V**irtual **R**eality **E**nvironments.

REFERENCES

1 **F. P. Brooks**, *No Silver Bullet*, IEEE Computer, pp10-19, April 1987.

2 **J.D. Mackinlay**, **S. Card** and **G.G. Robertson**, *Perspective Wall: Detail and Context Smoothly Integrated*, Proceedings of the ACM SIGCHI '91 Conference on Human Factors in Computing Systems, pp. 173-179, April 1991.

3 **C. Knight** and **M. Munro**, *Comprehension with[in] Virtual Environment Visualisations*, Proceedings of the IEEE 7th International Workshop on Program Comprehension, pp4-11, May 5-7, 1999.

4 **C. Knight** and **M. Munro**, *Should Users Inhabit Visualisations?*, to appear in Proceedings of IEEE WETICE, June 14-16, 2000.

5 **G.G. Robertson**, **J.D. Mackinlay** and **S. Card**, *Cone Trees: Animated 3D Visualizations of Hierarchical Information*, Proceedings of the ACM SIGCHI '91 Conference on Human Factors in Computing Systems, pp. 189 - 194, April 1991.

6 **J. Leigh** and **S. Bailey**, *A Tele-Immersive Environment for Collaborative Exploratory Analysis of Massive Data Sets*, ASCI '99, June 1999.

7 **C. Chen**, *Information Visualisation and Virtual Environments*, Springer Verlag London Ltd., 1999.

8 **H. Rheingold**, *Virtual Reality*, Mandarin Science, 1992.

9 **G. Walker**, *Challenges in Information Visualisation*, British Telecommunications Engineering Journal, Vol. 14, pp17-25, April 1995.

10 **M. Crossley, N. J. Davies, R. J. Taylor-Hendry,** and **A. J. McGrath**, *Three-dimensional Internet Developments*, BT Technology Journal, Vol. 15, No. 2, pp179-193, April 1997.

11 **C. Knight**, *Virtual Software in Reality*, Ph.D. Thesis, Department of Computer Science, University of Durham, June 2000.