# Narcissus: Visualising Information

R.J.Hendley, N.S.Drew, A.M.Wood & R.Beale
School of Computer Science
University of Birmingham, B15 2TT, UK
{R.J.Hendley, N.S.Drew, A.M.Wood, R.Beale}@cs.bham.ac.uk

**Abstract**

*It is becoming increasingly important that support is provided for users who are dealing with complex information spaces. The need is driven by the growing number of domains where there is a requirement for users to understand, navigate and manipulate large sets of computer based data; by the increasing size and complexity of this information and by the pressures to use this information efficiently. The paradigmatic example is the World Wide Web, but other domains include software systems, information systems and concurrent engineering.*

*One approach to providing this support is to provide sophisticated visualisation tools which will lead the users to form an intuitive understanding of the structure and behaviour of their domain and which will provide mechanisms which allow them to manipulate objects within their system.*

*This paper describes such a tool and a number of visualisation techniques that it implements.*

## 1 Introduction

People are increasingly faced with the problem of filtering and interpreting enormous quantities of information. From this mass of data they need to extract knowledge which will allow them to make informed decisions. They need to organise and interpret the mass of detail in order that the information and knowledge that is required to support their tasks is accessible to them.

Although this is not a new problem, the rapid and accelerating increase in the quantity of information available and a growing need for more highly optimised solutions have both added to the pressure to make good and effective use of this information. There has also been an increased requirement for direct access to information by its end-user rather than indirect access through third parties (e.g. librarians or research assistants). Similarly, the increase in electronic access means that information which would previously have been implicitly filtered out through the high cost of identification and access is now available on the desk-top.

This paper reports on work which investigates the use of self organising systems and virtual reality techniques to provide an environment which reveals the structure of the system being explored and provides for the navigation through the system and manipulation of the objects. The system uses a KQML interface to communicate with (possibly several) host systems. Examples are presented from applications of the visualisation tool to the World Wide Web and to a program development environment.

## 2 Large information spaces

### 2.1 Problem domains

The problems of understanding, navigating through and manipulating complex information spaces are now being faced across a wide range of application areas and it is becoming increasingly important to provide tools which provide sophisticated support to users in these tasks.

Software Engineering is, in many respects, a classic example of the domains where these problems exist. The systems being constructed and manipulated can be very large and complex and will often have a large number of engineers working on them through their life. These engineers will need to construct and maintain an overall understanding of the structure, constraints and behaviour of the system with which they are working as well as having a very detailed knowledge of at least parts of that system. They will need to look at the system from several different perspectives. For instance, they may need to understand the control flow, data flow, class structure, profile information and so on. They will also have different information requirements depending upon the task which they are performing - e.g. whether they are constructing,

debugging or maintaining the system. A large number of tools are available which support Software Engineers in some of their tasks (e.g. there are browsers, animators and so on), but these generally work at a relatively low level and are poor in providing support at the higher levels where it is often most important.

Concurrent Engineering, more generally, is an area where there is widespread support for users at the low level and yet where there is relatively little support for engineers in building an overall understanding of their systems. In these domains the overall complexity of the problem has traditionally been addressed through rigid compartmentalisation of the design and manufacturing processes with each sub-task operating within unnecessarily conservative constraints. This can lead to poor overall designs, since there is little chance for opportunistic optimisations across sub-components. Increasingly, economic and other factors are requiring that much more highly optimised designs should be produced and this is leading to many of the same problems that have been encountered within Software Engineering.

The World Wide Web (WWW) [1] is probably the best example of an information space where users need support. The structure of the web has evolved rather than been designed and the quantity of information is both very great and it is changing rapidly. Locating relevant information can be very difficult and although there are tools, such as search engines, which can help users to find information, the user is left without any overall *picture* of the information space. This is a particular problem since, often, a search will fail to locate the actual information required but *will* locate pages which are a small number of links away. A common strategy is to use a search engine to locate the potentially useful areas of the web and then to manually search outwards from these points. This manual search is often near blind since the user has virtually no information to guide them. Furthermore, in order to try to make the search as efficient as possible the user will frequently adopt a fairly complex strategy which places unreasonable memory requirements upon them - for instance, they will explore a branch and then backtrack, intending to resume the search from that point if they are unsuccessful elsewhere. This all leads to searches being much less reliable and efficient that they should be.

## 2.2 Information visualisation

One approach to these problems is to move more responsibility away from the user towards the machine. Routine tasks can be automated entirely and frameworks have been developed which can support the interaction between these tools and users. More sophisticated agents are also being developed which can locate, retrieve and filter information based upon the particular requirements of their users. These tools can reduce the quantity of detail that a user needs to contend with but, ultimately, the user still needs to process information in order to form an overall understanding of their system, if only to guide agents more effectively.

A complementary approach is to provide sophisticated visualisation tools which will present information to the user in a way in which the global, high-level structure is apparent and yet the low-level detail is still accessible. This will reduce the processing and memory requirements of the user and allow them to operate more effectively and efficiently. Some striking examples of the ways in which a good visual representation can unlock previously hidden structure and yet not obscure the detail are contained in Tufte [7].

Robertson [4] identifies four processes that need to be supported by appropriate visualisations:

- Sense making (building an overall understanding of the information).
- Design.
- Decision making (Building a decision and a rationale for that decision).
- Response tasks (finding information to respond to a query).

These processes usually impose different requirements on the visualisation since the information requirements of the different tasks vary and the mechanisms by which the information is accessed are different. Ideally though, a user should be able to work within one visualisation which is rich enough and sufficiently adaptable to support whatever task the user is performing.

## 2.3 The Narcissus system

Most visualisation systems rely upon a two dimensional representation and use a fixed layout algorithm. The use of a 3-dimensional representation through which users can navigate provides a much richer visualisation. This is partly because of the increased information density, but it is even more true when additional virtual reality techniques are employed to make the visualisation more sophisticated.

Similarly, by giving the objects that make up the information space behaviours that determine their movement through (and hence their position in) the 3-dimensional space (rather than using a global layout algorithm) we can generate much more effective views onto the objects and their inter-relationships.

The Narcissus system uses these techniques of self-organising systems and virtual reality to generate visualisations through which the users can navigate and manipulate objects in the visualisation. The system is

implemented as a process which communicates with applications (e.g. web browsers and programming environments) using KQML [3]. This provides a degree of application independence and also allows the system to work concurrently with several, possibly heterogeneous, applications and also allows collaborative working between several users.

The results that have been produced confirm the intuition that these are powerful techniques. The emergent structure that can be revealed by the system is often remarkable. It is also clear that there are a large number of issues that still need to be explored and we are adapting the system to experiment with some potential techniques for addressing these problems.

## 3 Organising Objects

Many approaches have been used to produce useful spatial layout of objects (for instance, using a global layout algorithm or applying statistical techniques to produce clusters of related objects [6]). The approach that we have adopted is to give each of the objects in the information space a behaviour which determines its movement (and ultimately its position) in the 3-dimensional space. At present the rules which determine the behaviour are common to all of the *active* objects although, in the future, we will modify this so that the rules can vary between objects and between classes of objects. We also intend to experiment with agents that will wander through the information space and modify the rules and other attributes of the objects.

The current model is loosely based upon physical systems with rules defining forces that act between the objects. These forces cause the objects to move in space. There are two classes of force:

- All objects in the system exert a repulsive force on all of the other objects.
- Active relationships between objects lead to attractive forces being exerted between related objects.

By running the model, objects migrate through space so that they are spatial close to those objects with which they are semantically related. Normally, a steady state is reached within a relatively small number of steps through the model. Figure 1 shows the movement from a completely disorganised system towards one where the structure is becoming apparent.

The emergent structure can be striking and it appears that a set of visual clichés will emerge. For instance, Figure 2 shows a structure which is found quite often. In this example (which is from the application of Narcissus to a programming environment), the small objects inherit from the two larger ones. The central objects inherit multiply (e.g. from both of the large objects), while the peripheral ones inherit singly. This overall structure is very hard to interpret from the textual representation of the program and yet becomes immediately apparent from the visualisation.

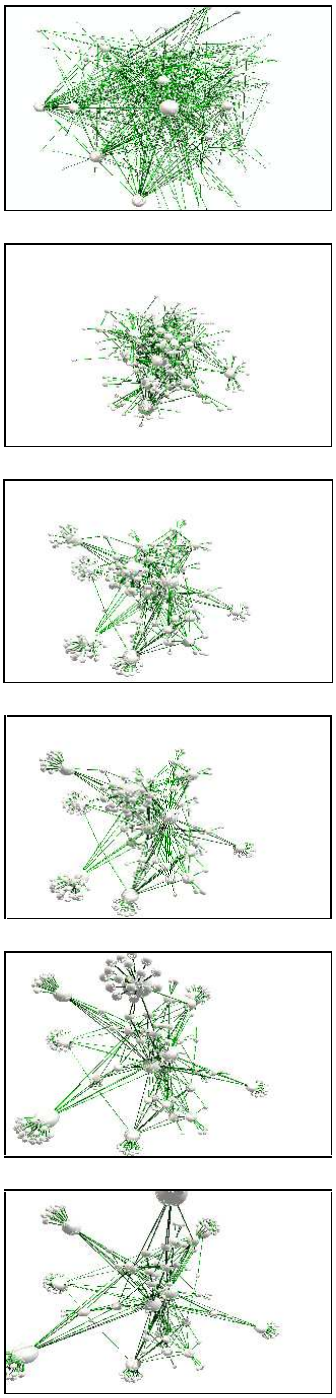

**Figure 1: Moving from disorganised to organised structure**

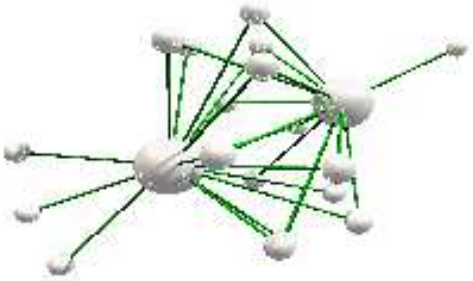

**Figure 2: The barrel structure**

The user can determine which relationships are active within the model and hence which relationships will be responsible for generating attractive forces between objects. The rules used to determine the forces are still subject to experimentation and it seems likely that there will be no single set of rules that are appropriate to all circumstances.

If the force is used to determine an object's acceleration, then a stable state can still include some motion with some clusters of objects pulsating whilst others may include orbiting satellites and so on. This motion can be a very valuable part of the visualisation since this dynamic behaviour can be useful for the recognition of individual clusters of objects and of similar structures. However, for the large systems at which this work is aimed, the computational cost can be very high and it can take a long time to reach a stable state when the force determines the acceleration.

An alternative is to have the force determine the velocity of an object. This leads to stable states in which there is no residual motion and so the model can be run until this quiescent state is reached and then turned off whilst the user navigates through and manipulates the system. Whilst this is a more practical solution, it does lose some of the richness of the first alternative and we will seek ways to optimise the system in order to leave some residual motion without incurring an unreasonable performance penalty.

# 4 The visualisation

## 4.1 The basic model

Providing a 3 dimensional virtual reality through which the user can move to explore and manipulate information is a potentially very powerful technique. The Narcissus system provides the user with a window onto their information space and allows the user to navigate through this space and to select and manipulate objects. The user is able to control some aspects of the behaviour of the objects and can select individual objects and classes of objects which should be visible. By default, the system draws arcs to represent the active relationships in the system although in some cases it is clearer when the user turns this off. Similarly, the system can label objects with their attributes (e.g. their URL) but this is often confusing and can obscure the structure. The user can either just display these attributes for selected objects or can set a distance threshold beyond which they are not displayed.

Figure 3 shows the representation of a collection of web pages. In this example, representing several hundred nodes, important structural information becomes obvious. For instance, the structure at the right of the visualisation is a collection of manual pages and examples. The large nodes are indexes into the pages, the ball-like structure represents the set of cross referenced manual pages and the structure below represents the examples.

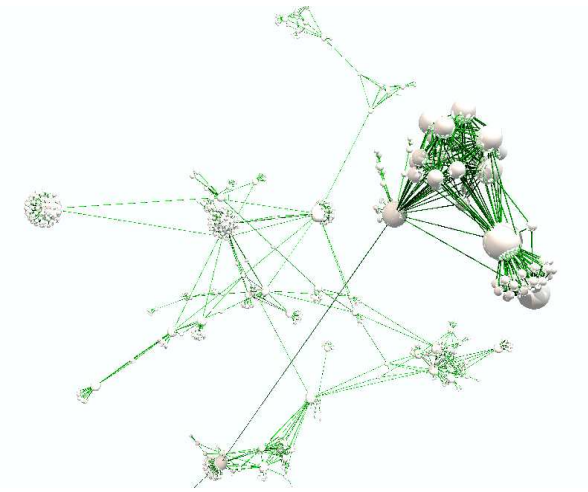

**Figure 3: Representation of complex web structure**

## 4.2 Extensions to the basic model

A number of other techniques have been developed to improve the visualisation of the system and the recognition of objects or clusters or classes of objects. Figure 4 shows an example where an icon has been used to render the surface of an object (In this case a world map has been used to represent a global object). In some cases this could be a useful and practical approach, for instance, in web browsing a user's image could be used to render the surface of their home page or a rendered image of a page could be projected onto the object which represents it. There are only pragmatic considerations which prevent this from being extended to movies, animations and so on. Empirical evaluations will be required to determine just

how useful these techniques might be. Other alternatives include the use of 3 dimensional icons. Vion-Dury et al [8] report work where they have used a function of an object's name to generate a distinct polyhedral shape for the object. This is potentially useful, particularly when the name carries some semantic information which might lead to related objects having common physical features (In their domain of programming environments this is often the case).

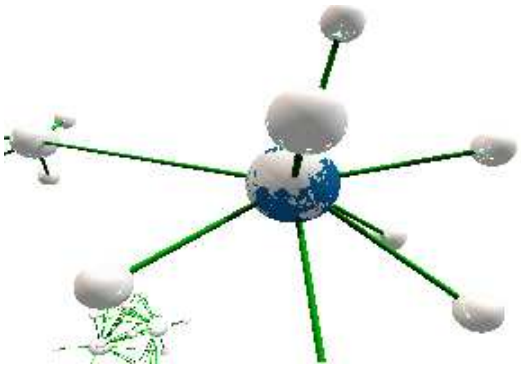

**Figure 4: Icon rendered onto surface of an object**

Figure 5 shows an example where the system has been used with a program development environment and colour has been used to overload additional information onto the visualisation. In this case, profile information has been used to colour objects according to the frequency with which they receive messages. This is a potentially useful technique in other domains, such as web browsing, where statistical information and other information is required as well as basic structural information.

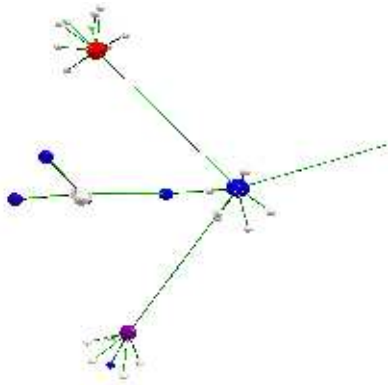

**Figure 5: Visualising profile information**

Another technique with which we are experimenting is to merge a cluster of individual objects into one compound object. This agglomeration removes some of the detail from the visual representation so that the overall structure is more clearly visible. These compound objects are formed by placing a translucent surface around the cluster so that from a distance it appears as one distinctive object but, as it is approached, the internal structure becomes more apparent and the user can smoothly move from a high-level view to one in which all the detail is available. Figure 6 shows such an agglomeration. These agglomerations might be given more than just a visual reality - it may be that it is more effective to jump to another space when the agglomeration is entered or to give the agglomeration some semantic significance which might, for instance, affect their behaviours.

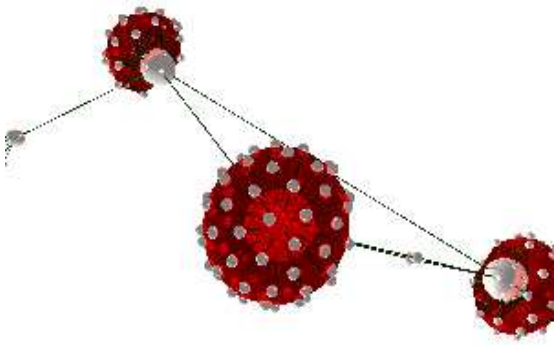

**Figure 6: Clusters enclosed by translucent surface**

## 5 The Narcissus system

### 5.1 Implementation

The first implementation of the Narcissus system was as a hard-wired visualisation for the SELF [5] programming environment. Since then it has been re-implemented as an independent system which communicates with other applications using KQML [3]. This implementation has been applied to web browsing and to program development. Narcissus itself is independent of the application(s) with which it is being used, although work is required to enable new applications to interact with the system. Figure 7 shows the system working with a modified version of xmosaic to provide visualisation support for web browsing.

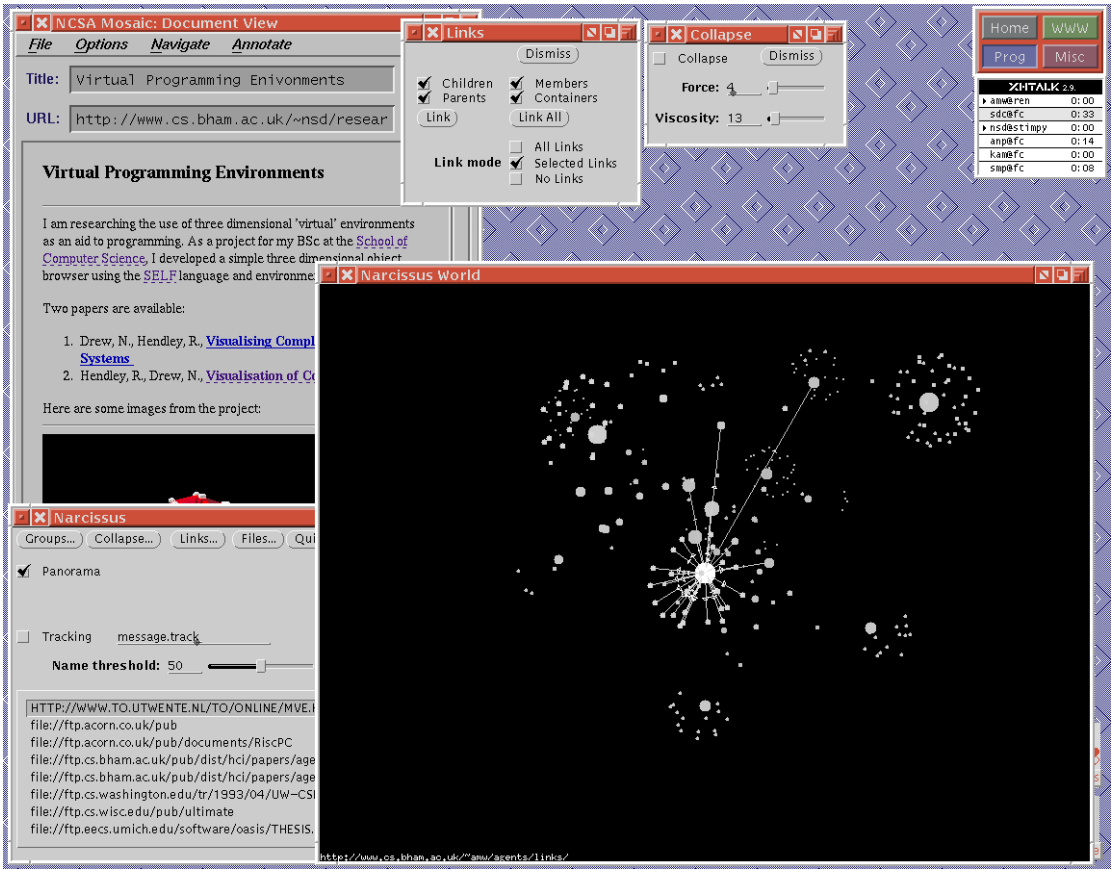


**Figure 7: Narcissus applied to Web browsing**

The use of KQML is useful not just because it provides application independence. Many of the domains in which visualisation is most important involve collaboration between multiple users and between multiple applications. Visualisation support in these domains should ideally integrate the work of these several users across all of their applications. At present this has only been validated across multiple instantiations of the modified xmosaic to provide a visualisation of concurrent web browsing.

The system allows the user to manipulate various parameters of the visualisation and of the forces acting on the objects. The user is able to move through the 3 dimensional representation of the world in order to change their view and to select objects (e.g. web pages) which are then manipulated using an appropriate tool (e.g. web pages are displayed and can be manipulated by a browser). An implementation is also presently being undertaken for an immersive VR system.

The performance of the system is dependent upon the size of the information space being manipulated, but with several hundred objects it provides acceptable interactive performance.

### 5.2 Evaluations

A formal empirical evaluation is presently being undertaken to assess the value of the visualisations provided by the system. Informal tests suggest that, for web browsing, the visual representations provided and the mechanisms for manipulating the objects and visualisation are useful.

## 6 Conclusions

The use of virtual reality and self organising systems is a powerful technique for information visualisation. The applications that we have built do show that important, high-level structure can become obvious using these techniques and that the low-level detail is still seamlessly accessible. It is also clear that navigating through the information space can be a relatively straightforward and natural task.

The approach appears to scale reasonably well to systems with many hundreds of nodes but with larger information spaces the techniques that we are evaluating

for hiding low-level detail will become increasingly important.

One of the benefits of making abstract information structures more concrete is that it provides the user with the ability to recognise objects by their relative positions and by the distinctive shapes that clusters of objects form. Again, the formation of agglomerations of objects (either by forming surfaces around them or through attaching some iconic representation to them) will support this recognition. One problem that occurs with dynamic systems (that is systems where objects are being added, removed or have their relationships modified) is that a small change can cause the system to re-organise in a way which may significantly affect the visualisation. This may not be a major problem in practice and one virtue of the approach is that the user can at least watch the re-organisation taking place (e.g. when a new URL is added to the web visualiser, the user sees it moving into position and any other consequential changes taking place). One approach may be to freeze the system after an initial organisation and then to just allow local changes to the structure when modifications are made. The best solution will need to be determined empirically, but it is likely that incorporating other cues to recognition will, in any case, be valuable.

The behaviours of the objects are presently very simple and the rules which determine the behaviours are global to all objects. Extending this to provide a richer set of behaviours and allowing these to vary across the objects is part of our current work.

## References


[1] Berners-Lee, T., Cailliau, R. Luotonen, A., Nielsen, H.F. & Secret, A. *The World Wide Web*. Communications of the ACM 37 (8), pp 87-96. 1994.

[2] Drew N.S.., Hendley R.J. *Visualising Complex Interacting Systems*. Proceedings of CHI95 (to appear) 1995.

[3] Finin T., Fritzon R., McKay D. et. al. *An Overview of KQML: A Knowledge query and manipulation language*. Technical Report, Department of Computer Science, University of Maryland. 1992.

[4] Robertson G., Card S., Mackinlay J. *Information visualisation using 3D interactive animation*. CACM Vol. 36, No. 4. 1993.

[5] Smith, R.B. & Ungar, D. *SELF, The power of simplicity*. in Proceedings of OOPSLA'87, published as SIGPLAN Notices 22 (12). 1987.

[6] Snowdon D., Benford S., Brown C., Ingram R., Knox I., Studley L. *Information visualisation, browsing and sharing in populated information terrains*. 1995.

[7] Tufte E. *Envisioning Information*. Graphics Press. 1990.

[8] Vion-Dury J., Suntan M. *Virtual images: Interactive visualisation of distributed object-oriented systems*. Proceedings of OOPSLA-94, ACM Press. 1994.