Visualising Software – A Key Research Area

Claire Knight and Malcolm Munro
Technical Report 5/99

Visualisation Research Group,
Research Institute in Software Evolution.
Department of Computer Science,
University of Durham,
Durham, DH1 3LE, UK.
{C.R.Knight, Malcolm.Munro}@durham.ac.uk

Keywords: Software Visualisation, Metaphors, Program Comprehension, Software Maintenance

Abstract

The visualisation of software systems has long been investigated, primarily through the use of graph structures representing at most a couple of aspects of the system under consideration. It has been claimed that such visualisation cannot be successful because of the size and complexity of software; this is true of purely graph displays. This position paper makes the case for three-dimensional visualisations that try to overcome these problems and then poses the question that if these visualisations are successful is it not then true to say that software visualisation has the ability to be both successful and a useful tool.

1. Introduction

Visualisation has long been used to illustrate facts about programs and software systems graphically, but it is only in recent years that visualisation research has moved to consider the use of three dimensions. The consideration of structures other than graphs within these three dimensional spaces is an even newer concept.

The use of the more traditional graphs (for example call graphs) is not discredited by the 3D visualisation community, but seen more as one facet amongst many useful techniques. The sheer size and complexity of many modern systems (and even legacy systems after many years of evolution and maintenance) very often illustrate the shortcomings of the graph-structured approach because of the cluttered and confusing pictures generated.

Whilst using three-dimensions and/or other display techniques is not going to magically remove these problems, they do provide reason enough to investigate other visualisation styles and methods.

Proponents of the graph-based displays have been known to discredit such research purely on the basis of lack of empirical evidence as to the benefit of three dimensions. Whilst such evidence is in short supply in the software maintenance and program comprehension fields there is much psychological information to be found and reused from studies relating to graphics, colours, navigation and general information visualisation.

This paper addresses some of the many issues relating to three-dimensional software visualisation and then discusses some of the main aspects relating to this sort of visualisation. The question of whether software visualisation is even feasible is then addressed, based on the information presented earlier in the paper.

2. Software can't be Visualised!

In his paper *No Silver Bullet* [1], F. Brooks wrote

"Software is invisible and unvisualizable."

and

"…software is very difficult to visalize. Whether one diagrams control flow, variable-scope nesting, variable cross-references, dataflow, hierarchical data structures or whatever, one feels only one dimension of the intricately interlocked software elephant. If one superimposes all the diagrams generated by the many relevant views, it is difficult to extract any global overview."

Some of these points are very valid, but if they can be overcome, or minimised, does that then mean that software visualisation is possible and effective?

Theoretically the answer is yes, but there are many considerations and concessions that must be made to make the visualisation effective at communicating information to the viewer.

In the above excerpt the visualisations referred to by Brooks are all graph based data sets, displayed visually as such. At the time Brooks was writing, visualising software meant displaying some information about (or some aspect of) the software in a graph structure. This now need not be the case with the advances in computer hardware and graphics technology. As can be seen in Figure 1, Brooks has a valid point about the problem of visualising complex pieces of software with graphs.

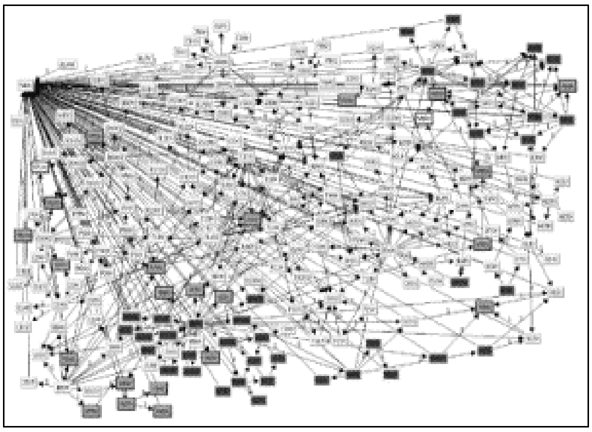

Figure 1 - Call graph of a commercial system

Figure 1 shows a call graph of a small well-maintained commercial system. If there is this much visual overcrowding of information from such a system then it is clear that new techniques for visualisation are needed.

Visualisation has the possibility of becoming a very powerful and much used tool by software engineers and maintainers. The following sections of the paper present information on three main areas of visualisation and then considers whether visualisation of software is feasible or whether Brooks' comments still hold true.

3. Visualisation Hypothesis

The premise of this paper is that visualisation is possible and effective. To argue this case, three important aspects are considered; intelligence amplification, metaphors and abstract and real world visualisations. Conclusions are then drawn based on the evidence presented in the preceding sections.

The definition of visualisation that will be used throughout the remainder of this paper is:

"Software visualisation is a discipline that makes use of various forms of imagery to provide insight and understanding and to reduce complexity of the existing software system under consideration."

This definition, with justifications for the various parts of it, as defined by Knight can be found in full in [12]. The basis for this definition is that the term visualisation should not be restricted to only computer displayable images, and that visualisation should have a purpose; i.e. to help users investigate complexity.

3.1 Visualisation for Amplification

The production of "nice" pictures is no reason to advocate research into visualisation, although visually appealing imagery can be put to good use in visualisations. An identified benefit, and therefore use, of visualisation is for intelligence amplification. Intelligence amplification is the use of computers to enhance the user's understanding and comprehension rather than the artificial intelligence view of trying to replace the human.

Intelligence amplification is an importation issue to consider when discussing software visualisation. Visualisation is the process of displaying graphically a large and complex data set to try and aid the users understanding and comprehension, i.e. acting as an intelligence amplification tool.

Brooks, the author of the above comments in Section 2, believes (documented by Rheingold [9]) that intelligence amplification is, and always will be, much more powerful then using computers for artificial intelligence techniques. As a comparison, consider creating a graphical display of a 10,000 line program. Reading through the 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.

Hubbold et al. [3] make a connection between intelligence amplification and virtual reality (VR). This therefore holds true for visualisations that make use of virtual reality as an enabling technology. The authors identify the pattern recognition and contextual abilities of humans that can be harnessed through the use of suitable intelligence amplification tools.

> *"In our everyday existence we cope with, and filter out, tremendous amounts of information almost effortlessly and with very little conscious thought. Indeed, if the same information, in all its detail, were to be presented in a form that we had to think about consciously, then we would be overwhelmed quite easily. Spatial awareness, pattern recognition, information filtering, coordination of multiple information streams are things we take for granted. Rather than look for a solution in AI, part of the VR thesis is that information presented in a suitable way can be processed far more effectively and directly by people."*

Chalmers [6] makes the link that the visualisation display, and the navigation of that display, is important as to whether the visualisation can be utilised as an intelligence amplification tool.

> *"In designing an information display, we should support movement and exploration through the space so as to let people build up their own models of the information. By moving and searching through a complex environment, looking in detail as some parts, and in overview at others, we make sense of it and make our our decisions about how to use it and work with it.*
>
> *Note that it is not enough to have an information space through which people can move. One has to give thought to what people will see from different positions and angles."*

This, whilst containing basic information, is very often ignored or not implemented properly and both the visualisation displays and the possible uses to which they can be put are well below the potential level of visualisations in general. This forces the user into patterns of working they are not familiar with and can decrease the effectiveness of that work. In effect the tool cannot then act as an aid to the user and enhance their work.

3.2 Visualisation Metaphors

The use of a metaphor creates a logical framework within which the mappings between the data items and the graphical representations can be made. This mapping tends to suggest some form of analogy between the two concepts, although this may be at a higher level of abstraction than individual data items and graphic objects. The need for a metaphor was identified by Levialdi et al. [5] in the construction of their database visualisation system.

> *"Using VR visualization techniques to represent the results of queries implies the definition of a mapping, or* metaphor*, among the objects of the database and the objects of some virtual world."*

It is important to note that most (if not all) three-dimensional visualisations are implemented in some form of virtual reality system, however basic in terms of equipment and the realities presented. This means that discussions and issues relating to virtual reality are applicable when considering visualisation.

Pettifer and West [7] are of the view that the potential power of virtual reality is related to the metaphor used to create the graphical reality. The strength of the metaphor, and the closeness to natural interaction set virtual realities and three-dimensional visualisation apart from other graphics based tools and techniques. The authors also identify the benefits of natural (real world) metaphors, and of making use of perceptual and spatial skills learnt and used in the real world in the virtual reality.

> *"A three-dimensional world metaphor has much more scope for direct human/computer interaction than the two-dimensional desktop because it engages in us those perceptual and spatial faculties that allow us to comprehend our surroundings and to process effortlessly the vast amounts of information that are presented to our senses second by second. It is the potential to directly engage these faculties that is the defining characteristic of virtual reality. As the immersive environment is far richer than the desktop, the metaphors for interaction assume a far greater significance. … The role and management of metaphors for the virtual environment therefore assumes key significance."*

A benefit is that in using three-dimensional visualisations some of the cognitive processing needed for navigation and visual interpretation can be shifted to the sub-conscious as these are activities that are carried out daily with no real thought. Metaphors not only provide a basis for the mapping from the data to the visualisation, but also for the virtual reality in which the visualisation is situated.

The use of three dimensions for visualisation can add an element of familiarity and realism into systems, especially when using a metaphor approaching a real world experience. The world is a three-dimensional experience and by making the visualisation more like the "real" world means there is less cognitive strain on the user. This in turn makes the system easier and more comfortable to use, regardless of data analysis and understanding, because of all the experience and knowledge the user has built up elsewhere. This means that the aim of the visualisations is to aid the comprehension of complex phenomena can be achieved without adding unnecessary (virtual) environmental complications.

Chalmers [6] addresses the use of three dimensions (and the use of real world style metaphors) as a familiar way of dealing with information and writes:

> *"For such uses of an information space, a naturalistic 3D view seems a powerful but familiar way of controlling information detail. Perspective lets us gain an overview of distant regions and detail of what is close."*

In a paper about the Information Visualizer System from Xerox Parc [8] the following is written about the use of three-dimensional displays:

> *"…the three-dimensional displays help shift the viewing process from being a cognitive task to being a perception task. This transfer helps to enable humans' pattern matching skills."*

This is only true of well designed graphics, but supports the views that images that are known to the user can aid them in understanding the images presented. This also ties visualisation back to the intelligence amplification ideas presented in the previous section.

3.3 Abstract and Real World Visualisations

Much of the visualisation of programs done to date has been abstract in nature; geometrical shapes in either two or three dimensions, possibly of differing colours, have been used to represent elements of the program code. It has, however, been shown that simply transferring the traditional two-dimensional abstract graphs into three-dimensional space does not generate effective and usable visualisations in most cases [4].

Some more recent work on three-dimensional abstract visualisations moves away from these graph oriented displays and an example of this can be seen in Figure 2. This work is documented in more detail by Young [4], but briefly this shows the modules of the software system laid out around the calling structure of the program. On some of the modules more detailed information can be seen on top of the module blocks.

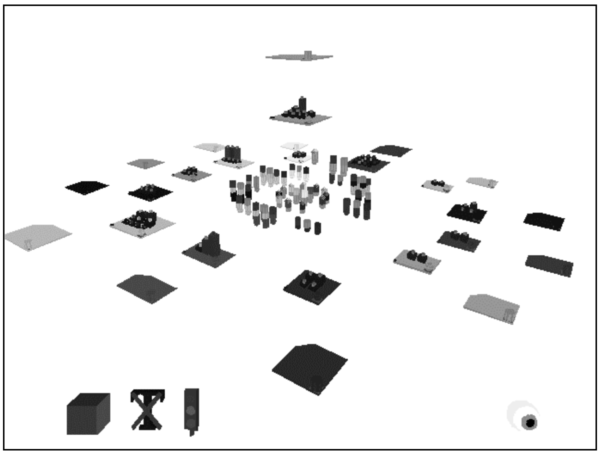

Figure 2 - 3D Abstract Visualisation

Real world visualisations rely on metaphors and mappings that translate between the data items and elements from the everyday human world. These data items are then visualised through the mappings to create a scene that does not, at first glance, appear alien to the human eye. The commonality of the scene then allows the thought and analysis to be directed to the underlying data and, for example, any incompatibilities between the scene and the real world can indicate certain properties about the underlying data.

An example of real world software visualisation can be found documented by Knight and Munro [2] and an example of this can be seen in Figure 3. The *Software World* visualisation uses a city metaphor to represent the software system and here the buildings represent methods, with the doors and windows on the buildings representing the number of local variables and the number of parameters to the method respectively.

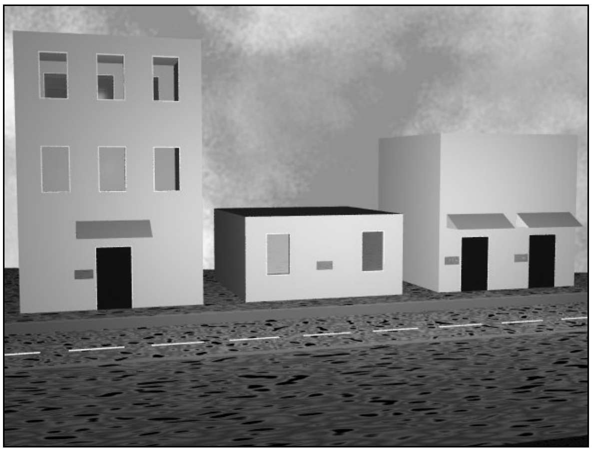

Figure 3 - *Software World* Real World Visualisation

Whilst these abstract and real world approaches differ in the representations used, they can be complimentary systems of visualisations. It is very likely that solutions to any visualisation problem will encompass some elements of both. The underlying data set can also have an effect on the visualisation style utilised. With program code, either style is acceptable because the software is intangible, but for more tangible data, the actual data values may indicate the style to be used. As Feijs and de Jong [11] note in their recent visualisation paper:

> *"But there is no such thing as the one right view;…"*

4. Visualisation is Possible!

Along with the comments presented in Section 2, Brooks [1] also writes

> *"A geometric reality is captured in a geometric abstraction. ... The reality of software is not inherently embedded in space. Hence it has no ready geometric representation in the way that land has maps, silicon chips have diagrams, computers have connectivity schematics."*

This tends to suggest that the use of metaphors, and hence real world mappings, are of no use because of the intangibility of software. But it is precisely this intangibility of software that makes the use of metaphors and abstractions such a powerful technique.

This is not to say that a successful visualisation mapping, and implementation of that mapping, is an easy task. There are many problems associated with visualisations of all forms. A basic one when visualising code is deciding on the best mappings between language elements and the metaphor. Knight and Munro [2] point out that one of the most difficult problems is in dealing with the possible range of data items to be visualised.

> *"The hardest problem in visualising anything is that, theoretically, the visualisation has to be able to deal with the range of items from one to infinity. This massive range means that automation and layout algorithms (both for two and three-dimensional visualisations) are hard problems. It also means that the visualisations need to be defined with this in mind; or to provide a way of dealing with very large numbers and indicating this fact to the user."*

This sizing issue can be a hindrance to any metaphor, but it is no reason to discredit the use, or investigation, of visualisations.

Later in his paper, Brooks [1], writes

> *"In spite of progress in restricting and simplifying the structures of software, they remain inherently unvisualizable, thus depriving the mind of some of it's most powerful conceptual tools. This lack not only impedes the process of design within one mind, it severely hinders communication among minds."*

This makes a link to both intelligence amplification, and also the power that metaphor is able to bring to visualisations.

Section 3 of this paper uses three aspects with which to argue for and justify software visualisation. It appeals to the notion of intelligence amplification as a means to enhance understanding and comprehension of complex systems, and to the essence of defining suitable metaphors. It is these metaphors that will form the heart of a successful visualisation, and these metaphors are therefore an important (and hard) research area.

From the information presented in this paper it is possible to conclude that software visualisation, especially with the newer techniques of three-dimensional non-graph based imagery, is indeed a feasible research area.

There have been a few forays into the use of visualisations of software that (a) make use of three dimensions and (b) do not explicitly draw any form of graph (or other node and arc) structure but it is this sort of research that should be encouraged. It is in doing this that the valid criticisms made by Brooks can be avoided and the complexity of the software system portrayed without hindering comprehension.

Acknowledgements

This work is partly financed by an EPSRC studentship.

References

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

[2] C. Knight, M.Munro.
Comprehension with[in] Virtual Environment Visualisations.
To appear in Proceedings of the IEEE 7th International Workshop on Program Comprehension, Pittsburgh, PA, May 5-7, 1999.

[3] R. Hubbold, A. Murta, A. West, T. Howard.
Design Issues for Virtual Reality Systems.
Presented at the First Eurographics Workshop on Virtual Environments, Barcelona, 7th September 1993.

[4] P. Young, M. Munro.
Visualising Software in Virtual Reality.
Proceedings of the IEEE 6th International Workshop on Program Comprehension, Ischia, Italy, June 24-26, 1998, pp19-26.

[6] M. Chalmers.
Design Perspectives in Visualising Complex Information.
Proceedings IFIP 3rd Visual Databases Conference, Lausanne, seizerland, March 1995.
http://www.ubs.com/info/ubilab/print_versions/ps/cha95.ps.gz.

[5] S. Levialdi, A. Massari, L. Saladini.
Visual Metaphors for Database Exploration.
A position paper relating the Virgilio system, available on-line from
http://www.darmstadt.gmd.de/~hemmje/Activities/Virgilio/Publications/virgilio.ps.

[7] S. Pettifer, A. West.
Deva: A coherent operating environment for large scale VR applications.
Presented at the first Virtual Reality Universe conference in Santa Clara, California, April 1997.

[8] J. T. Stasko.
Three-Dimensional Computation Visualization.
Georgia Institute of Technology, Technical Report GIT-GVU-92-20, 1992.

[9] H. Rheingold.
Virtual Reality
Mandarin Science, ISBN 0-7493-0889-3, 1992.

[10] K. Lynch.
The Image of the City.
The M.I.T. Press & Harvard University Press, Cambridge Massachusetts 1960.

[11] L. Feijs, R. de Jong.
3D Visualization of Software Architectures.
Communications of the ACM, December 1998, Vol. 41, No. 12, pp72-78.

[12] C. Knight.
Visualisation for Program Comprehension: Information and Issues.
University of Durham, Computer Science Technical Report 12/98.
http://www.dur.ac.uk/~dcs3crk/workfiles/documents/Lit_Survey_Tech_Reports/Tech_Report_12-98.ps.gz