GENISOM

James Brittle and Cornelia Boldyreff
Department Of Computer Science
University Of Durham
{j.g.brittle,cornelia.boldyreff}@durham.ac.uk

Abstract

GENISOM, is an offspring component of the GENESIS software engineering platform, incorporating the generation, maintenance and viewing of self-organizing maps. The self-organizing map's unsupervised clustering method has been used to visualise a large software collection.

Key Words: Self Organizing Maps, GENISOM, software visualisation

1 Introduction

The Self-Organizing Map (SOM),invented by Prof. Teuvo Kohonen in the early 1980s [4], is an unsupervised, clustering algorithm. It has been demonstrated to aid programmers in the process of reverse engineering by discovering common features within legacy code [2] and to assist in object recovery[1], although visualisation of the associated maps is not explicitly considered in reports of this research.

A prominent problem within the field of Software Engineering concerns reuse. Reusable assets are in abundance, over the web and in libraries but it is extremely difficult to locate reusable software artefacts that are relevant to a particular application. The necessary organisation is often lacking and difficult to achieve given the dynamic nature of such software collections. This problem can also be found where a large evolving software system consists of an ever growing number of components and the managment and hence the comprehension of the associated software artefacts tends to be increasingly difficult.

Having suitable visualisations of such software collections can mitigate the problem identified above.

2 GENISOM

GENISOM is a client/server application designed to manage and enable viewing of SOMs. Figure 1 illustrates a simplified architecture diagram for the GENISOM system.

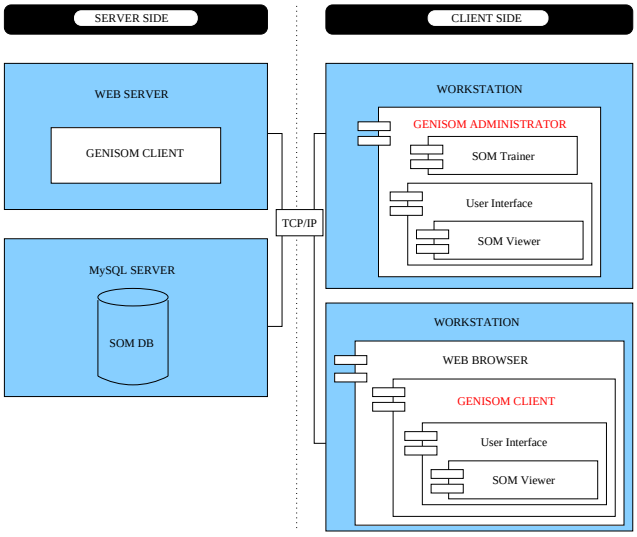

Figure 1. GENISOM architecture

The GENISOM Administrator is used to manage SOMs; it enables their creation and maintenance. The generated SOMs are then stored within a MySQL database, from which the GENISOM Client can then retrieve the data and display the generated maps. The Client is web based using Java Web Start[1] to aid its accessibility to users. The architecture of the system enables distributed software engineering teams to work together.

There are two main use case scenarios for the system; firstly the Administrator could be used by the librarian of a reuse library and the Client by the system developers (see Figure 2), therefore aiding them in the location of reuse candidates. Secondly both tools could be used by members of a software development team to aid program comprehension, or help in decisions regarding restructuring and reengineering of the system.

[1] http://www.java.sun.com/products/javawebstart/

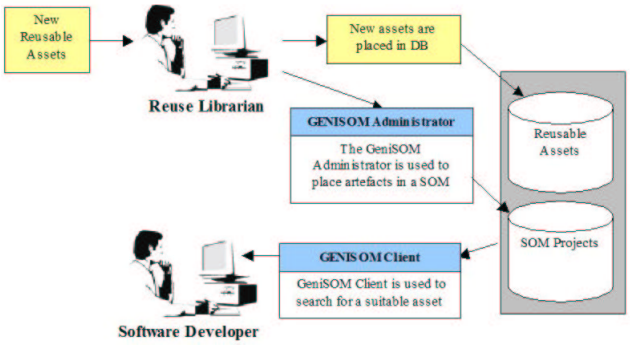

Figure 2. Reuse infrastructure with GENISOM system in place

3 Visualisations of Maps

The GUI for the SOM evolved through a number of stages which resulted in the final 2D map as illustrated in Figure 3.

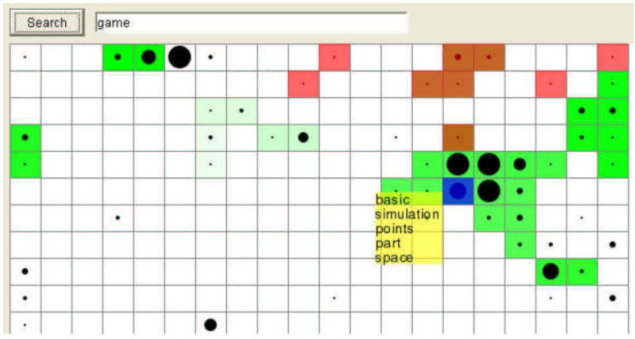

Figure 3. Screenshot of 2D Map

The neural net is arranged as a grid with the inputs (e.g. reusable artefacts) being attached to the neurons. The black dots on grid cells (i.e. neurons) indicate that inputs have been matched to them, with the size of the dot representing the number of them. The green shading of the grid cells indicates the boundaries of similar clusters of neurons.

The map is interactive allowing the user to select a particular neuron (the blue cursor indicating the selection), this displays the neuron's labels, five or less words that best describe the inputs matched to it. As well as displaying the labels for a selected neuron, details of the inputs matched to it are also displayed in a side bar though this is not depicted in the figures.

Browsing is aided by a search system coupled to the map which highlights the results in red for a certain search string.

Using 2D limits the amount of information that can be visualised. Improvements to the GENISOM GUI therefore naturally led on to the development of a 3D map using the Cityscapes technique which has already found application in software visualisation [3]. Using this technique in GENISOM each value is plotted as a column (or 'building'). The 'buildings' are plotted on the same horizontal plane, allowing differences in height and position to be analysed. In relation to a SOM, each building represents a neuron and the height of the building relates to the number of inputs that are matched to it. The implementation of this used Java3D[2] and in the final system the option was made available to switch between using 2D and 3D interfaces.

Figure 4 illustrates the 3D map; the user's view can be rotated and zoomed in and out. Furthermore, the user can also interact with the 3D map in the same manner as the 2D map following the same colour scheme.

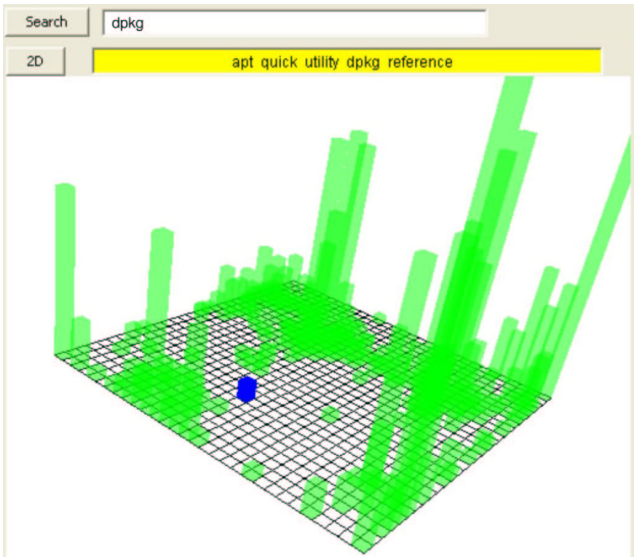

Figure 4. Screenshot of 3D Map

References

[1] A. Chan and T. Spracken. Discovering common features in software code using self-organizing maps. In *Proceedings of the International Symposium on Computational Intelligence*, Kosice Slovakia, August 2000.

[2] A. Chan and T. Spracklen. Object recovery using hierarchical self-organizing maps. In *Proceedings of the International Conference on Engineering Applications of Neural Networks*, Kingston Upon Thames UK, July 2000.

[3] C. Knight. *Virtual Software In Reality*. PhD thesis, Durham University, 2000.

[4] T. Kohonen. *Self-Organizing Maps*. Information Sciences. Springer, second edition, 1997.

[2] http://java.sun.com/products/java-media/3D/