GLOBAL JOURNAL OF PURE AND APPLIED SCIENCES VOL. 13, NO. 4, 2007: 585 -590
COPYRIGHT (C) BACHUDO SCIENCES CO. LTD. PRINTED IN NIGERIA. ISSN 1118-0579

ENHANCEMENT OF USER SATISFACTION AND COMPUTER RESOURCE MANAGEMENT THROUGH THE APPLICATION OF FISHEYE INTERFACE TO KNOWLEDGE ENGINEERING SYSTEMS

U. R. OGBUAGU AND J. N. OBIDINNU

(Received10 April 2007; Revision Accepted 15 June 2007)

ABSTRACT

The structure of knowledge engineering systems is usually very complex and is associated with some level of dissatisfaction to end users. It is posited herein that the Fisheye strategy as an information visualization tool can adequately address this problem. We applied this tool in developing a knowledge engineering system, and it produced the desired results by translating and scaling the entire information into one screen. This method produced better satisfaction and enhancement to end users by reducing the mental burden of browsing and memorizing the steps towards a set goal.

KEYWORDS: Fisheye Strategy, Resource Management, Knowledge Engineering, Visual Enhancement

1. INTRODUCTION

The Fisheye strategy is the embedding of a specific piece of information within its larger context. Furnas (1986) observed that a fundamental feature of the Fisheye strategy is to provide a balance of local details and global context of data in one screen. This is achieved by displaying the local data with much emphasis, even as the global context of the data is also displayed on the same screen with reduced emphasis. The implication of this is that Fisheye representations will, more successfully communicate information to its viewers than a series of separate dialog boxes popping up as several screens, one after the other. It is called Fisheye views because it has a similar effect on the view of data that an optical Fisheye lens has on a photograph (Furnas, 1986).

By inspection, it is noticed that workers in large corporations apply the principles of the Fisheye information pattern. They at least know their local Heads of Departments, but in most cases may not know the Directors in the remote divisions. Similarly, an examination of knowledge pattern in the academic world reveals a Fisheye strategy, such that, in similarity ratings, the disciplines near one's area of specialization are better understood by him than those distant away from his specialty. In the same vein, order of stories in newspapers has shown that news editors evolve a Fisheye editorial strategy. The papers contain more of their locally based news stories, and this coverage thins down as the distance increases, such that only important news items that are compensating are published. While there are many interesting processes behind these results, we assert that many naturally occurring phenomena of knowledge do exhibit similar tendencies. Hence, we are motivated to investigate the appropriateness of the Fisheye strategy to provide a good viewing interface for knowledge engineering systems.

The interface between a user and a machine can determine, to a great extent, the success or failure of harnessing the resources within a computer. The structure of knowledge engineering systems is usually very complex. Consequently, designers employ different tools to enhance users' satisfaction. Some of the typical concerns for users in knowledge engineering systems include: Complex interfaces that may require specialized training, getting lost in the process of browsing through several steps to reach a target, and the time wasted in going through the steps. The Fisheye strategy is an information visualization tool that can adequately address these problems. It is an interface mechanism that can be employed to facilitate faster access to large data sets. This is achieved by providing a balance of local detail and global context of data in one screen. The significance of this work includes, among others; a reduction in the length of time of accessing data stored in a computer, as well as the mental burden on a user trying to memorize the integration of the various components of data appearing in separate screens.

2. LITERATURE REVIEW

Furnas (1986), introduced the idea of Fisheye strategy and applied it to text documents. The display was based on a combination of Degree of Interest (DOI) and proximity to the focus. The DOI was decided prior to inspecting the document. The function assigns to each point in the structure, a number telling how interested the user is in seeing that point, given the current task. A display of any desired size n, can be made by simply showing the n "most interesting" points, as indicated by the DOI function.

According to Furnas (1986), generalized Fisheye views are achieved by decomposing the DOI function into two components, the a priori and the a posteriori component. The a priori component is that contribution to interest, which largely transcends the particulars of a given interaction with a structure. Major, critical, or particularly informative pieces are presumed to be of greater a priori interest, than parts that might be considered only finer details. The a posteriori component is the contribution to determining what parts are of most interest that, very specifically depends on the current interaction. In its simplest additive form, the generalized Fisheye DOI model is:

$$DOI_{fisheye}(X|.=Y) = API(X) - D(X,Y)$$

where $DOI_{fisheye}$ is, according to the fisheye model, the user's DOI in a point, X, given the current point of focus, Y. API (X) is the global a priori importance of X and D (X,Y) is the distance between X and the current point Y. That is, the interest increases with distance. This simple formulation allows Fisheye structures to be defined in any sort of structure where the necessary components can be defined.

Sarka and Brown (1992) extended the Fisheye strategy to viewing and browsing graphical representations. They introduced layout considerations into the Fisheye formalism, so that the position, size, and level of detail of objects displayed are computed based on client-specified functions of an object's distance from the focus and the object's pre-assigned importance in the global structure. They used a large graph representing major cities in a country to demonstrate their idea. In their demonstration of the graphical Fisheye views, the vertex with the thick border is the current point of interest to the viewer, and is called the *focus*. In this prototype system, a viewer selects the focus by clicking with a mouse. The focus changes the display, which is updated in real time whenever the mouse is dragged. The size and detail

U. R. Ogbuagu, Department of Computer Science, Cross River University of Technology, Calabar- Nigeria
J. N. Obidinnu, Department of Computer Science, Cross River University of Technology, Calabar- Nigeria

of a vertex in the Fisheye view depend on the distance of the vertex from the focus.

Lamping, Rao and Pirolli (1995) described an implementation for presenting a 2-D graph through a Fisheye zoom. The hyperbolic browser provides a smoothly varying "focus plus context" view where the display space allocated to a node decreases continuously with the distance from the focus, yet does not disappear abruptly. This application of Fisheye zooming according to Hane (2000) is one of the best known and successfully commercialized hyperbolic tree forms. The integration of details within context by means of a spherical distortion proves to be a good way of looking at those details without losing awareness of the overall picture. Collaud (1995) discussed an interesting application of the Fisheye view and introduced the CZWeb tool that makes use of two techniques – Fisheye views and continuous zoom – to help users navigate the web. The prototype graphically displays a network in a rectangular 2-D display space. A hierarchy is used to display some web pages in great detail and the others in less detail or no detail at all. This study is important in that it was one of the first attempts to apply the Fisheye strategy to the web domain.

Bederson (1998) reported the development of a Web-browsing prototype called Pad++, which is a multiscale graphical environment. This prototype displays multiple Web pages and the links between them instead of showing one page at a time. A Fisheye view approach is used in this display method where the page in focus is clearly readable whereas the others are shown in smaller scales to provide context. This approach was compared to the traditional display method of Netscape in several different scenarios, and the authors found that after some changes to the prototype, subjects using Pad++ answered questions 23% faster than those using Netscape. Ozgur and Schuff (2004) suggested the use of Fisheye views to alleviate the problem of information overload and disorientation in understanding business processes in organizations. According to them, "*information overload*" is defined as, excessive demand for information comprehension beyond the cognitive capacity of an information user. Any effort that will simultaneously explore all relevant business processes in detail would usually create an excessive amount of information that cannot be processed easily. To avoid information overload, therefore, Data Flow Diagrams (DFDs) are simplified by splitting the representations into different levels, depending on the degree of detail. A context diagram shows the entire system in the context of the organization. A Level 0 (zero) diagram contains elements describing the major sub processes of the system. A series of Level 1 (one) diagrams contain further details regarding each sub process of the Level 0 diagram. For DFDs representing large-scale systems, there can be a large number of levels. It is left to the viewers of these diagrams to integrate context and details in their minds while switching back and forth between different levels. Such integration is not always easy, and often causes disorientation where viewers feel lost within the complexity of the hierarchy. System analysts are therefore faced with a dilemma regarding the level of details to show in representing business processes. They believe that the application of Fisheye views to business processes would make it possible to see different levels simultaneously with an overall view of the system. To create a Fisheye view of a series of DFDs, the detail of the sub process (the focus) is embedded into a higher-level diagram (the context). Furthermore, expanding one or more clusters, while leaving the remaining clusters collapsed, can create the Fisheye view of an Entity Relationship Diagram (ERD). Creating Fisheye views of ERDs aids the understanding of data models.

3. APPLICATION OF FISHEYE STRATEGY TO KNOWLEDGE ENGINEERING

As is evident from the presentation in section 2.0, Fisheye systems have found applicability in information visualization systems, especially in the visualizations of complex spaces. The idea to smoothly integrate the context and details is promising, but needs to be modified for specific implementations. So far, no available literature has revealed the application of Fisheye views to knowledge engineering systems.

3.1 Knowledge Engineering Concepts

In this paper, we shall adopt the definition by Locke (1999). Knowledge engineering is the process of codifying the knowledge of a human expert in a form that is accessible to a non-expert through an expert system. It follows a specialized procedure for analyzing information systems and design. The information system being analyzed is an abstraction of a real-world issue of interest. Blythe, Ramachandran and Gil (2001) identified some typical concerns that users may have in using knowledge engineering systems. These include, but not limited to, getting lost as it takes several steps to get to desired information, and spending longer time browsing through several steps to reach a goal. Coulson (2002) categorized knowledge engineering activities to typically consist of the following stages:
Acquisition of knowledge from human experts; (ii) Analysis and synthesis of acquired data;
(iii) Integration and interpretation of knowledge; and (iv)

Knowledge representation
Contemporary computer scientists have agreed with Coulson (2002) that these stages are relevant to successful implementation of knowledge engineering systems. We shall therefore adopt them in our design.

3.2 Data Collection

The information system of interest is the Braking system of a motorcar. This is a collection of mechanisms put together in a motorcar to ensure that it stops whenever the brake system is applied. The data used for our system were obtained from motor vehicle mechanic experts. We adopted several information gathering methods to ensure adequacy. These data were treated through the stages listed in section 3.1. Eventually we arrived at a comprehensive volume of data, with a sample of it presented in Table 1.

Table 1: Sample Data for the braking system

S/N	Problem	Evidence	Resolution(s)
1.	Brake pedal goes to the floor when it is depressed	-Brake fluid is very low -The master cylinder is bad -Air in the hydraulic system - Leakage in hydraulic system	-Fill the master cylinder -Replace the master cylinder -Bleed the hydraulic system -Locate and repair leakage
2.	The brakes hardly stop the car or won't hold it at a stop	-Brake fluid is very low -The master cylinder is bad -Air in the hydraulic system -Brake pads/shoes are worn out	-Fill the master cylinder -Replace the master cylinder -Bleed the hydraulic system -Replace brake pads/shoes
3.	Brake pedal is very hard to push down	-Bad power brake booster -No vacuum to power brake booster -Brake line is pinched -Some objects stuck under brake pedal	-Replace power brake booster -Replace vacuum line as required -Replace brake line -Remove interfering object
4.	Brake pedal fades	-Brake pads/shoes don't line up -Brakes overheating due to dragging -Brakes overheating due to airflow -Brake fluid has the wrong colour -Leakage in hydraulic system	-Replace brake pads/shoes -Service the brake system -Remove interfering objects -Replace brake fluid -Locate and repair leak
5.	Brake pedal goes down much further to apply the brakes	-Brake fluid is very low -Brake fluid has the wrong colour -Air in the hydraulic system -Brake pads/shoes are worn out -Bad power brake booster check valve	-Fill the master cylinder -Replace brake fluid -Bleed the hydraulic system -Replace brake pads/shoes -Replace power brake booster or valve
6.	Parking brake won't release	-Parking brake cables are frozen -Parking brake linkage needs lubricants -Broken parking brake mechanism	-Replace parking brake cables -Lubricate parking brake linkage -Repair or replace broken brake parts
7.	Brake pedal feels spongy	-Leakage in hydraulic system -Air in the hydraulic system -Brake fluid has the wrong colour -Brake pads/shoes are worn out	-Replace leaking brake lines/hoses -Bleed the hydraulic system -Replace brake fluid -Replace brake pads/shoes
8.	Brakes Grab	-Parking brake not releasing -Brake pads/shoes are worn out -Bad front disc caliper or rear drum brake wheel cylinder	-Lubricate parking brake -Replace the brake pads/shoes -Replace or rebuild front calipers -Replace or rebuild both rear wheel cylinders
9.	One or more brakes lock up	-Brake pads/shoes are worn out -Loose or bad front wheel bearings -Brake pads/shoes are contaminated -Bad front disc caliper or rear drum brake wheel cylinder	-Replace brake pads/shoes -Replace bad wheel bearings -Repack/retighten loose wheel bearings -Replace or rebuild both front calipers -Replace or rebuild rear wheel cyl.
10.	The brakes squeal when they are applied	Dirt/dust accumulating on the pads/shoes -Brake pads/shoes are worn out -Rotors/drums glazed -Audible wear indicators contact rotors	-Clean/sand brake pads -Replace the brake pads/shoes -Clean/sand the rotor/drums

The program design will utilize the data in Table 1 to form a base for the system, which will be connected to a Fisheye interface through the inference engines. The Table is made up of series of problems that can be encountered by a car driver, the evidence that will lead to the solution, and finally the resolution of the problem. The eventual computer program will be a car braking diagnostic system that can be used for troubleshooting brake problems.

4. PROGRAM DESIGN

This section presents the structure of the Fisheye program. The program combines the features of the Fisheye strategy (for the interface) in addition to the rules that determine the option of evaluating the braking problems (inference engine). The data stored in a database, serve as the back end. We begin with a Fisheye algorithm formulated for this purpose. This is followed with a diagnostic tree representing each of the items in Table 1.

4.1 Fisheye Algorithm

Different authors like Storey et al (1997) formulate suitable algorithms for the implementation of the Fisheye strategy. Each one is modified according to the specification requirements of the application of interest. One of such algorithms that we found suitable for this work is from Storey et al (1997) and has its steps listed below.

Steps

1 Represent the data in the form of a graph, with each parent enclosing its child nodes,

2 At any level, all the children node should be at the same level from their parent.

3 Select the node of interest among the child nodes, which grows by a scaling factor, and pushes its neighbours' nodes outside, depending on the translation vector,
4 Scale the node and its neighbours' to fit inside the parent (root). The screen refers to the root (parent) as used above. This is the only step visible to the user.
{See section 4.1.2 for illustrations of the use of child/children node and parent (root) nest}.

4.1.1 Mathematical Representation

Each neighbour is pushed outward by adding a translation vector [T = Tx, Ty] to its coordinates; it is then scaled by a factor, *s*, around the centre of the screen (Xp, Yp). The scale factor, *s*, is equal to the size of the bounding box (visual display unit) before translation (T) divided by the size of the bounding box after applying T. Equations (1) and (2) show the functions applied to the coordinates (X, Y) of the neighbour nodes to determine the new position $X^1 Y^1$:

$X^1 = X_P + s\,(X + T_x - X_P)$ .... (1)

$Y^1 = Y_p + s\,(Y + T_y - Y_p)$ .... (2)

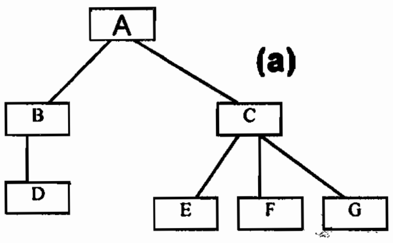
To shrink a node that has previously been enlarged, the following inverse equations are used:

$X = (X^1 - X_P)\,/s + x_p - T_x$ .... (3)

$Y = (Y^1 - Y_p)\,/s + Y_p - T_y$ ....(4)

The idea in this model emanates from the work of Storey et al (1997). We developed it further to implement a new system. The functionality of the system is discussed in section 4.3.

4.1.2 Graphical Explanation



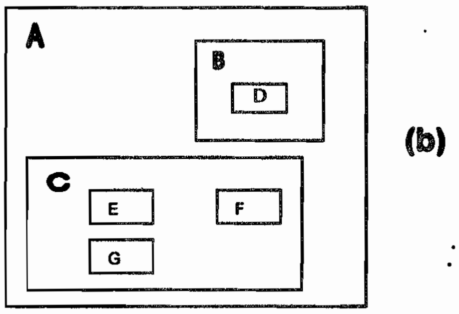


Figure 4.1: (a) A Graph representation of the Data level (b) A nested graph view of the same Data levels with composite nodes A, B, and C

Figure 4.1 (a) shows a graph representation of a single component (problem) of the data in Table 1.This implies that each of the problems can also be sketched accordingly. The root node, A, represents the PROBLEM column of the data. The next level, B, and C represent the EVIDENCE column of the data, and suggest that there are two evidences for the selected problem. Different problem components have varying number of evidences. Finally, the last level, D, E, F, and G represent the resolution column. Figure 4.1 (b) shows the nesting of the children node within their parent node. At each level the nodes are adjusted to fit inside the available space.

4.2 Diagnostic Tree

Diagnostic trees consist of flows and troubleshoot steps. Flows in a diagnostic tree can also contain sub flows representing common troubleshoot sequences. Figure 4.2 is a diagnostic tree representing one of the braking problems listed in Table 1. It begins with the problem type, and then continues with the next levels in the solution sequence. From the tree, it is easy to visualize the steps involved in diagnosing a particular brake problem. We present Figure 4.2 as representative of the other brake problems listed in Table 1.

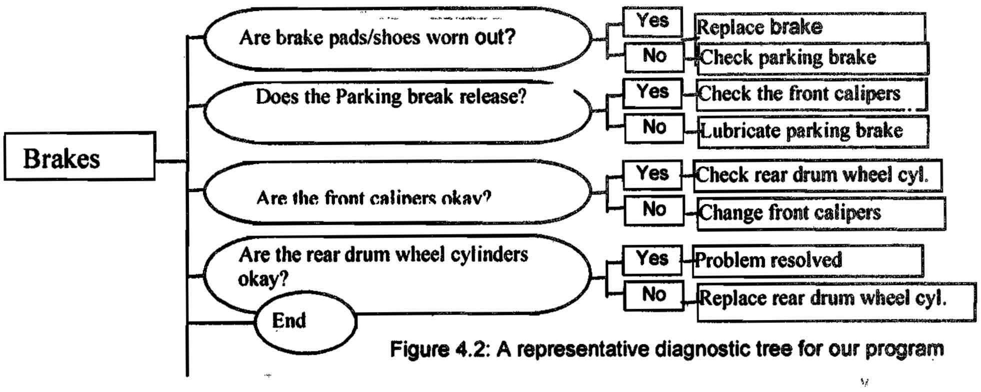

Brakes
Are brake pads/shoes worn out?
Yes
Replace brake
No
Check parking brake
Does the Parking break release?
Yes
Check the front calipers
No
Lubricate parking brake
Are the front calipers okay?
Yes
Check rear drum wheel cyl.
No
Change front calipers
Are the rear drum wheel cylinders okay?
Yes
Problem resolved
No
Replace rear drum wheel cyl.
End

Figure 4.2: A representative diagnostic tree for our program

4 DISCUSSION

Translation and Scaling are types of Geometric Transformations in Computer Graphics. Translation involves displacing an object a given distance and direction from its original position, while Scaling involves the process of expanding or compressing the dimensions of an object. These principles are applied in our model as seen in section 4.1. Figure 4.3 (a) shows an example of a grid layout of nine nodes. A look at equations 1-4 reveals that both translation and scaling are combined in each case. In Figure 4.3 (b) the node of interest assumes the centre and scaled, while the adjoining nodes are displaced according to a translation vector. As the central node grows, it pushes its neighbour nodes outward as if there were infinite screen space. The last part of equations 1 and 2 will then scale the central node and its neighbours so that they fit inside the available space, as shown in Figure 4.3 (c). In nested graphs, as shown in Figure 4.1 (b), the node of interest pushes the boundaries of its parent node outward and so on, until the root is reached. As a final step, everything is scaled to fit inside the root node since it cannot grow any larger due to limited screen space. Equations 3 and 4 are reverse equations for Equations 1 and 2 respectively.

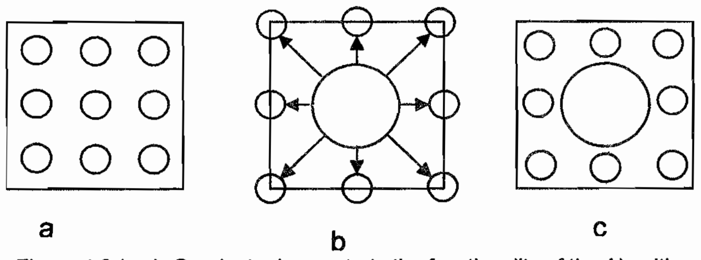
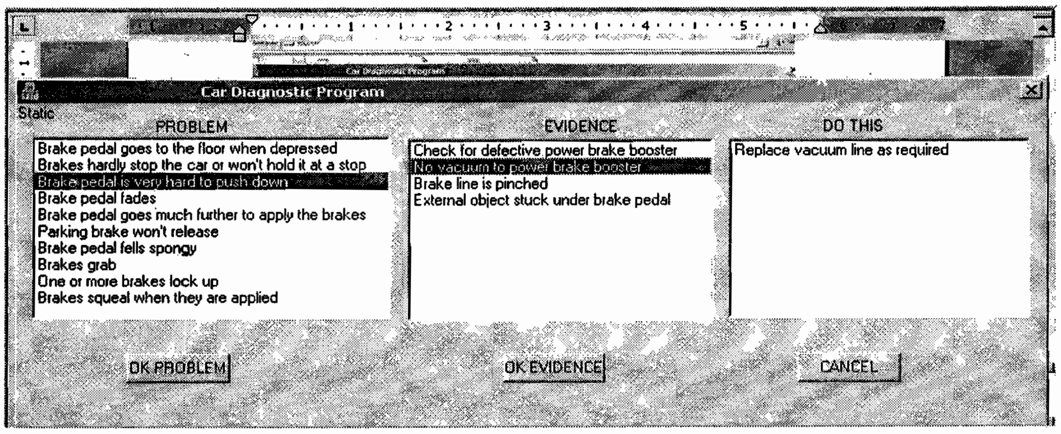


Figure 4.3 (a-c): Graphs to demonstrate the functionality of the Algorithm

The program provides an interface that ensures that all data are presented in one screen, and observes the principles of translation and scaling, as analyzed above. The window is partitioned into three columns, as in nested graphs, with the following column headings; PROBLEMS, EVIDENCE, and DO THIS. A sample of the output is shown in Figure 4.4. The windows grow or shrink depending on the volume of data available. When the space in a particular window is filled, the height of the window begins to grow. This trend continues until the maximum allowed screen space is reached. At this point, further additions to a window will cause distant options from the focus to be scaled down gradually, while ensuring that all the options fit into the screen. With this feature, the same screen is seen by the user, while the options keep changing according to the selections made.



PROBLEM	EVIDENCE	DO THIS
Brake pedal goes to the floor when depressed	Check for defective power brake booster	Replace vacuum line as required
Brakes hardly stop the car or won't hold it at a stop	No vacuum to power brake booster	
Brake pedal is very hard to push down	Brake line is pinched	
Brake pedal fades	External object stuck under brake pedal	
Brake pedal goes much further to apply the brakes		
Parking brake won't release		
Brake pedal fells spongy		
Brakes grab		
One or more brakes lock up		
Brakes squeal when they are applied		

Figure 4.4: Option 2 is selected from among the symptoms listed for problem 3, leading to a resolution highlighted under DO THIS column.

The PROBLEMS column represents the distinct braking problems that can be identified by a driver. The options under EVIDENCE can only be displayed when an option has been selected in the PROBLEMS section. The numbers of options that can be displayed here vary, as it is a factor of the selected problem. This is deliberate in order to allow the user initiate the action of making a choice, based on the closeness of the SYMPTOMS observed as a result of the identified problem. The DO THIS column can only accommodate one field at a time corresponding to a chosen option in the EVIDENCE column. The options here provide the RESOLUTION approach to the symptom that has been distinctly isolated under the evidence column.

CONCLUSION

Using the information from expert auto engineers and mechanics on the breaking system of a car, we designed a computer diagnostic system that can be used to resolve car braking problems through the Fisheye strategy. The presentation of the structure in one window allows even non-expert computer user to operate it. In this regard, available visual resources are better managed and end user satisfaction is enhanced.

REFERENCES

Bederson, B., 1998. A Zooming Web Browser, In Forsythe, C., Ratner, J., and Grose, E. (eds.). Human Factors and Web Development; pp.255-266. Lawrence Erlbaum, New Jersey..

Blythe, J., S., Ramachandran, K., and Gil, Y., 2001. An integrated environment for knowledge acquisition. Int. Conf. on Intelligent User Interfaces; pp. 13–20. San Francisco.

Collaud, G., 1995. "A distorted-view approach to assisting web navigation". New Paradigms in Information

Visualization and Manipulation; pp.255-266. ACM Press, New York.

Furnas, G. W., 1986. Generalized Fisheye Views. Morristown, Bell Communications Research, New Jersey.

Hane, P. J., 2000. LEXIS-NEXIS Tests Data Visualization Technology. Information Today. Retrieved on 17th January, 2006 from http://www.infotoday.com/newsbreaks/nb1122-2.htm.

Locke, J., 1999. Basics of Knowledge Engineering. Kindred Communications Troubleshooter team, Microsoft Support Technology. Nottingham, NH. Retrieved on 17th January, 2006 from http://www.lcc.uma.es/~eva/doc/materiales/microsoft.pdf

Lamping, J., Rao, R. and Pirolli, P., 1995. A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. In Proceedings of ACM CHI, pp. 401-408.

Coulson, R. N., 2002. Knowledge Engineering For Ecosystem Management, Knowledge Engineering Laboratory, College of Agriculture and Life science, Texas, A & M University. Retrieved on 18th January, 2006 from http://www.kelab.tamu.edu/standard/approach.html

Ozgur, Turetken and David, Schuff, 2004. Supporting Systems Analysis and Design through Fisheye Views. Communications of the ACM, 47 (1): pp72 – 75.

Sarkar, M. and Brown, M., 1992. Graphical Fisheye Views of Graphs. In Proceedings of ACM CHI. pp. 83-91.

Storey, M.A. D., Wong, K., Fracchia, F., and M¨uller, H., 1997. On Integrating Visualization Techniques for Effective Software Exploration. InfoVis '97, Phoenix, AZ, pp 38–45.