A Project completed as part of the

Requirements for the

BSc (Hons) Computer Studies (Visualisation)

Entitled : Virtual Reality Internet Mapping System

By

Benedict Daniels

In the years 1998 - 1999

Abstract

Using the Internet today requires a certain talent in deciding which links to follow when searching for pages related to a certain subject, some pages cease to exist, and some fail to contain relevant material. Many pages will link themselves to each other, helping the individual in his quest for information, but these are hampered by unfinished pages, or unrelated pages included in link lists.

One way round this is to represent a group of pages as objects, with a visual representation of a page with related content, or that does not exist, with optional depth of mapping to avoid overloading the user with web objects and locations. The use of graphics and maps has been effective in other areas of structured data representation. This paper evaluates some of the other systems in limited use to visualise the actual virtual structure of the Internet, and looks at some common themes with which to design a superior tool.

This Paper researches this field, and discusses a design for a new tool for mapping related sections of the Internet. The original aim of this project was to also code the tool itself, but this has been discarded due to time constraints, and the fact that the development of such a tool requires a higher level of expertise than was available.

The report concludes with some ideas for the advancement of the system, and ideas for a system that were discarded as being unfeasible in the first level of development.

Contents

Chapter	Title	Page
1	INTRODUCTION	4
2	LITERATURE REVIEW	5
3	CONCEPT AND IDEAS FOR A NEW TOOL	10
	3 - 1 Virtual Reality Systems and the Internet	11
	3 - 2 Search Engines and the Internet	12
4	BASIC DESIGN OF THE TOOL, STRATEGIES AND METHODS	14
5	CODING OF THE TOOL, PROGRAMMING AND DEVELOPMENT ISSUES	17
6	EVALUATION	20
7	FURTHER DEVELOPMENT	22
	7 - 1 Individualization of Web Pages	22
	7 - 2 Extending this to Multi-User Environments	23
8	CONCLUSIONS	24
	BIBLIOGRAPHY	25
	APPENDIX A : Project Proposal	26
	APPENDIX B : Project Plan	28
	APPENDIX C : Glossary	30

1 : Introduction

Using search engines on the Internet can often result in URLs (Universal Resource Locations), and pages which may be quite unrelated. A good example is pornographic links with a whole dictionary used in the page's description, to cause the page to appear on results for searches for many different mundane subjects. This always depends on what the user searches for, but can interrupt many simple, but unrelated keyword lists.

Another factor in searching the Internet, is pages which no longer exist in their URL listed location, this can be very annoying, as some link lists now, have lacked maintenance for so long, that almost the whole list refers to non-existent pages. The time taken to follow up all these links can be quite consuming, and some method of eliminating pages from a list of URLs to evaluate would be useful.

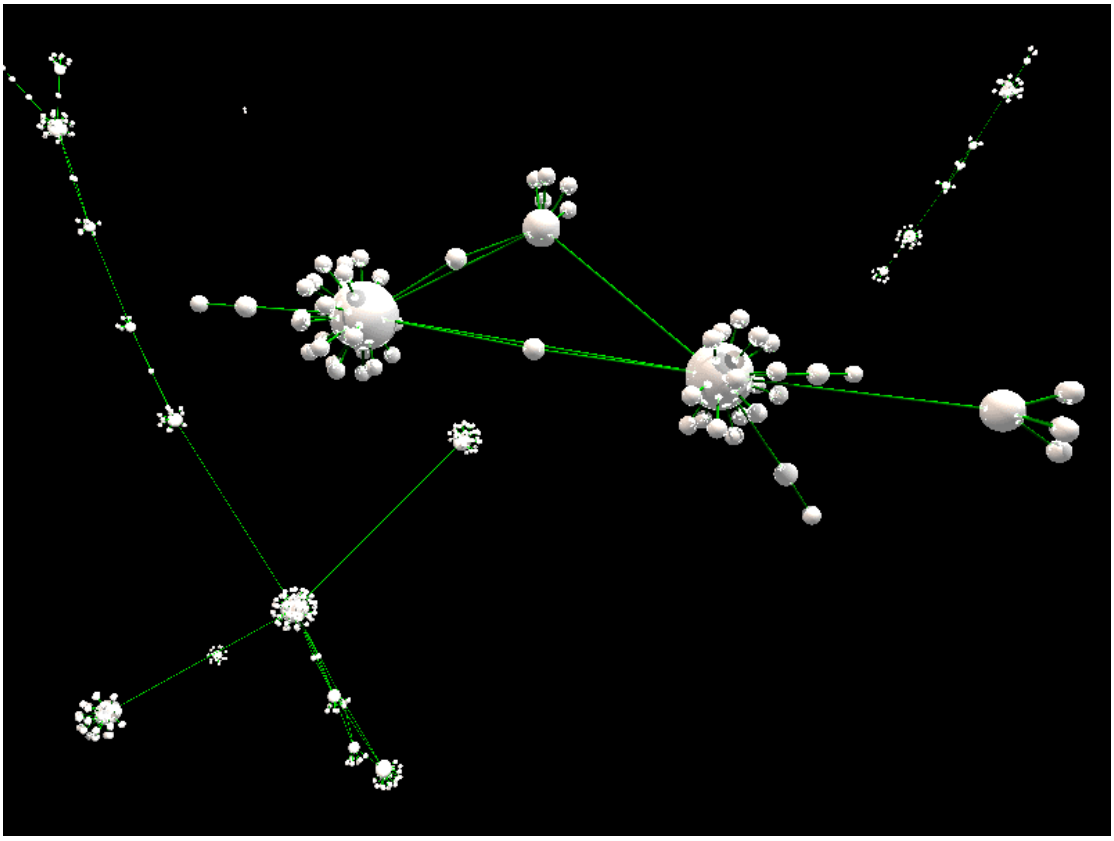
Visual paradigms have been useful in displaying groups of data for the purposes of bringing across the structure, and a data object's context within such a structure. This paper will evaluate previous work using Virtual Reality and 3D systems to represent logical systems of Web based material, for use in navigating such logical systems, and aid the user in selecting pages to view. This paper will also put forward a design for such a system, hoping to build on the points of previous work, and hopefully result in a software tool for achieving a workable system.

2 : Literature Review

This is a fairly unexplored area, with some of the most relevant material only being published in the last year. The most similar work done, is the Narcissus Visualisation system, (Hendley et al) is based along similar lines to the system this paper has in mind. A system that maps out the collection of web pages, and their relation to each other. The resulting system was written using an independent graphics model, that uses KQML (a Knowledge Query and Manipulation Language) to obtain the mapping information from which to build the 3D model. This results in fairly large, and often hard to view models, which, while displaying hierarchies of pages, and the relationships between pages, offers fairly little in the way of information. The option to obtain and display the information is there, but when used, can make the model even more difficult to interpret. The paper does not specify what system was used to create the model, but a fairly simple collection of spheres and lines has been used to create the model itself. The book "Information Visualisation" by Card, Mackinlay and Shneiderman has several papers on visualising the Internet. A common theme is to use objects of differing size, colour and orientation to represent the status, size, and position in a hierarchy of the object in question. Obviously, simple shapes used in different combinations are useful as learned points of references. Colour as used in many situations can have an to effect on positioning and orienting the user within a mapped space.

In "Visualising Cyberspace" (Andrews, K.), The 3D maps have been reduced to 2D maps, with a 3D element, giving height, and the facility for colour to objects. This makes for very impressive maps, with layouts made quite clear using a tree diagram. However, tree diagrams cannot describe *inter*-connections very well, as on a 2D map, this will result in crossed lines, and a map more difficult to interpret. This mapping technique is therefore less suited to our problem.

The Narcissus system touches on searching for information pages, and discusses the backtracking method commonly used, in which a link is tested for its usefulness, and if it fails to provide adequate levels of help, the user will backtrack to the last URL visited.



An example of the Narcissus/Hyperspace model

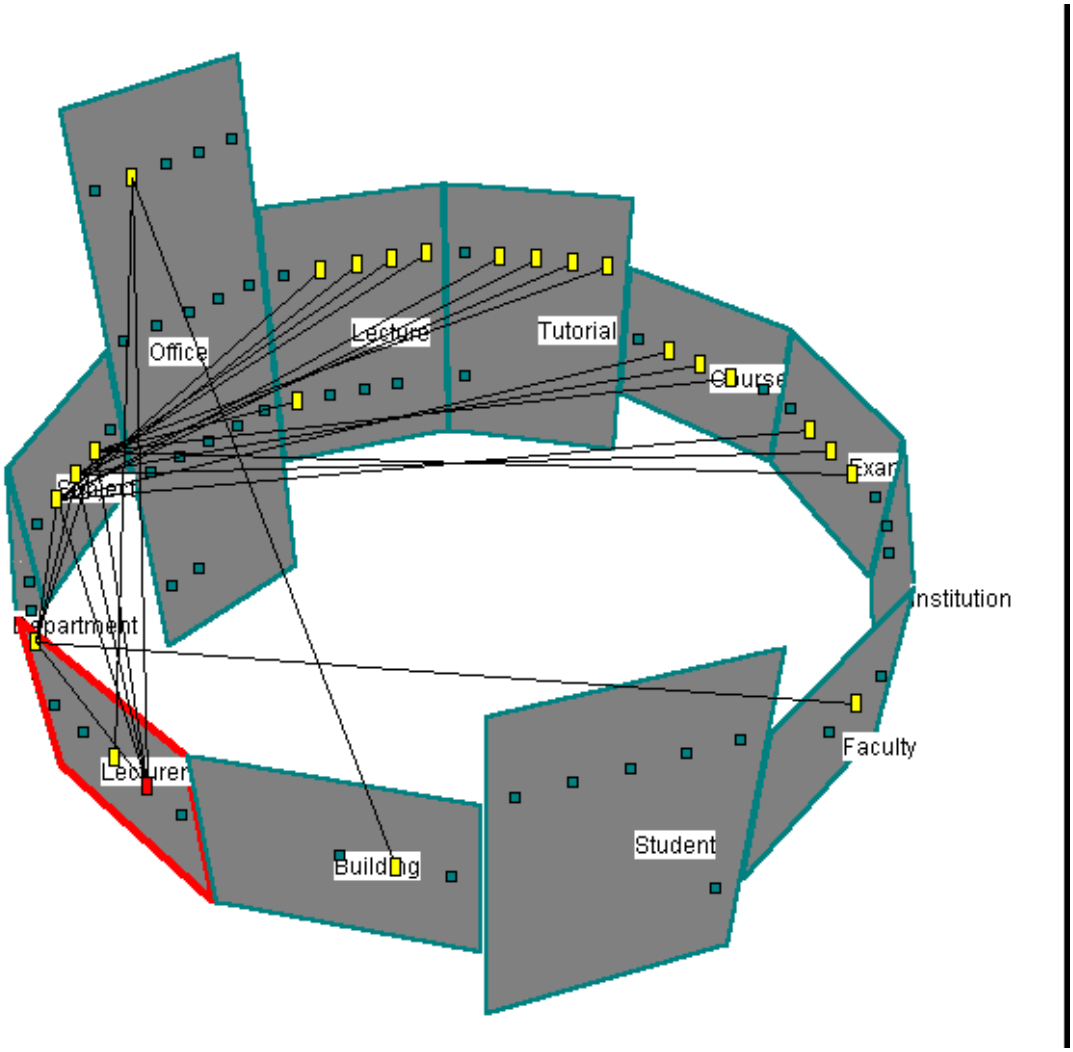
A system called “Hyperspace” (Drew et al) an offshoot of the Narcissus system, dedicated towards web mapping, and subject matching offers some ideas on extracting the required information. This system reads out information from pages **as** they are accessed, storing the information in a local cache, so every page needs to be read in at least once before the structure starts to become apparent. This system produces large complex systems again, but produces models based on a larger view than small clusters of related subject matter, these clusters show up and large spheres surrounded by small groups of related pages, branching out to meet other such groups. Several un-connected groups may begin to show up through the use of browse sessions to explore different subjects and web-networks. Hyperspace is

also geared up to display the titles of the web pages being displayed, however, these may make the map unpresentable as in earlier cases.

"We believe that allowing users to visualise the web in this way will help them to orient themselves within its information landscape - allowing them to make more effective use of the many resources that the web provides."

Hyperspace : Drew et al (199?)

"Winona" is an experimental system also from Birmingham University is a database viewer with two different viewing systems. One is the Circular wall system, which displays various tables, with points on each wall segment representing records. Instead of links to the relevant related table, it draws in connections to related records on the relevant table. Although this results in a lot of crossed lines, it actually makes for an easy to view model. The only limitation is the number of tables that can be represented before the wall becomes difficult to interpret. The advantage of the Winona System is that it can display another view which is the table hierarchy view. The wall view might be a good view in our system, but would probably require more complicated coding to make it possible.

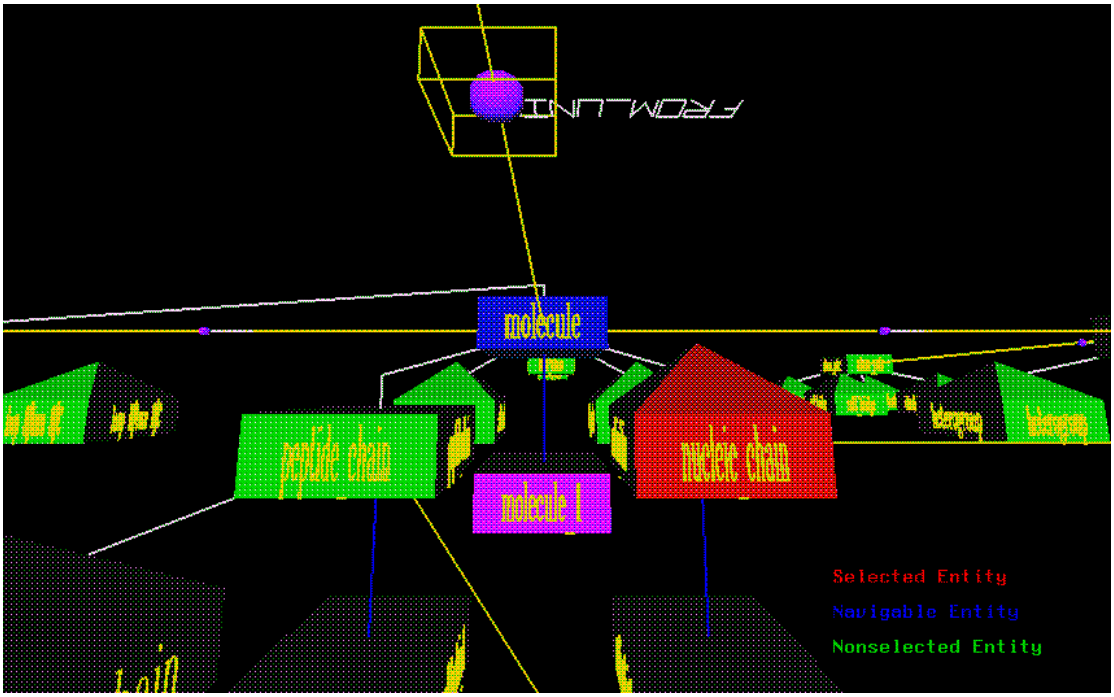


An example of the Winona "Circular Wall Approach

"Amaze" is the last example system, also from Birmingham University, this system used a grid system to lay out 3D data objects, in a similar principle to ours, without the 3rd dimensional element which we require for our model. The basic results are also fairly crude, and best suited to small scale data structures. One notable feature of Amaze, is the use of mapped text, being used to label objects. The use of texture or even text mapping would be a slowing factor in the system, making smooth navigation difficult.

All of these systems seem to have overheads which are suited to programs which either run off-line on cached files, or over longer periods of time than most users would be

happy with. Obviously, most of these systems have been run and tested as research tools, rather than as end user aids.



An example of the “Amaze” Schema.

All the information on “Winona”, And “Amaze” taken from HTTP://www.biochem.abdn.ac.uk/~john/ifdbvis.html. The Qpit section is unavailable at this time.

3 : Concept and Ideas for a new tool

So far, there has been no commercial system built along these lines, at least, none that have been marketed with the success of other internet programs. The inspiration came when I found a number of websites, purporting to be on one subject, with a large list of links to other sites on the same subject, and many links towards pay-per-usage sites. I found that by following some of these links, they either pointed to each other, to sites that did not exist, and to the pay-per-usage sites. Visualising this, what would be seen would appear to be a cluster of sites, pointing to themselves and nothing else, creating, essentially, a closed system of files and links. As Lea et al(1996) discuss in their introduction to VRML, Virtual reality has it's roots firmly embedded in popular science fiction, in several cases as a means for navigating data storage spaces, in "Disclosure" by Michael Crichton, and in "Neuromancer" by William Gibson, we have fine examples of both a local database, *and* what could be described as the internet, being accessed using virtual reality representations of logical data systems. In "Disclosure" the film, we see a Database organised as a set of virtual filing cabinets, accessed by use of an "Avatar". The word "Avatar" is used to describe a users representation of themselves within a multi-user 3D environment. Lea et al(1996) discuss the development of multi-user 3D communities using Java to create dynamic VRML scenes, this isn't necessary for our system, but may be discussed in **Further Development**. In "Neuromancer" the representation is closer to the representation intended for our system, the describing of objects in a 3 dimensional space using geometric objects, connected by lines of communication. In this case however, the objects represent closed systems, and the lines represent physical communication systems, but the analogies are very similar to ours. In fact, Gibson's representations of the "Matrix", are probably responsible for some of the initial concepts leading to VRML

3, 1; Virtual Reality Systems and the Internet

Virtual reality and the internet is still a relatively small area at the moment, even though VRML has been around for the last 4 years, use is still rather limited, this is mainly due to the fact that the common internet browsers, such as Microsoft's Internet Explorer, and Netscape, require specialist "Plug-Ins" to be capable of viewing these files. With the recent development of the VRML 2.0, this requires new plug-ins to be produced, slowing down the spread of VRML sites. VRML stands for Virtual Reality Mark-up Language, and is closely related to HTML (Hypertext Mark-up Language). Using simple un-compiled scripts to describe a 3-dimensional scene of objects, with the applications of URLs to objects. This allows the user to browse the web by clicking on 3D objects, rather than highlighted text, as in HTML's case. VRML is a good language for creating simple scenes, as the graphics engine, used to navigate the scene is already set up and active, and the tools for assigning behaviour and URLs is also built in. Texture mapping is also possible in VRML, allowing for complex imagery to be created with less complicated 3D objects.

VRML is the only Virtual Reality system to be applied successfully to the Internet, although there are possible replacements. When deciding on the languages I could use in the development of this system, I found that there are some other systems that could provide the functionality required, and could allow for the browsing system as well. Java now has the capability to be embedded in VRML, to provide dynamic VRML scenes, and this is the system I finally decided on, but Java also has a port of OpenGL that could be applied to it, and while this would result in a fairly efficient system, this would require high level development of functions to track objects and the mouse position in a Java window. Without this, there is no way to decide when and how a user clicks on an object within a 3D scene. Now however, there is a set of class libraries for Java for the implementation of 3D graphics, which would probably make the development and design of a 3D browser using Java exclusively more viable.

The reason why Java is more suited to the task of analysing web structures is it's inherent binding to the Internet, and its multi-platform usage. Java is already equipped with the tools required to read web pages from the net, and due to its object oriented basis, would be useful for keeping track of each web "object" and the various data to be associated with it. C or C++ might also be useful for this system, but produce standalone systems, which require re-compiling on different systems to be able to work. Java also has an edge over C in that with its connections to other Internet technologies, such as browsers, it can be used to administrate their use, such as executing a browser window with a specific URL, or an associated program such as ftp, with the appropriate address.

In the extreme, a bonus for VRML is that tools are in place to use physical VR immersive equipment, such as VR goggles, glove etc. In fact, if it were possible to display a HTML web page on the face of a VRML object, and to navigate around that page, while displaying it on such an object, it would be possible to represent our system as a complete VR system, without the requirement for normal keyboard access (except for our original URL input, or search engine keywords). However, reasonably speaking this is not feasible, without major additions to the VRML language. The use of full immersion just for the purposes of browsing is a little Over The Top, however, some applications are discussed in **Further Development**.

3, 2; Search Engines and the Internet

Search engines are very good at matching keywords that the user provides, with descriptions, and/or keyword lists held that refer to a particular website. And when searching for a common subject, this can provide a user with a list of several pages of list results, sometimes number in the hundreds, or even thousands. Depending on the search engine used, these may come with confidence ratings on how well the page's description matches up to the users request. In this case, results will be listed in order of confidence, making it more likely

the user will reach the right page in the first links tried out. Even with a high confidence result, backtracking through results is quite common, even when several keywords have been used in the search. If the information about a page to be referenced could be seen *before* the user views the page itself, this would save time in evaluating a page. While the system being suggested here would perform virtually the same task as the search engine, there are some factors which could help limit the pages that are returned as having similar content. For instance, many web pages now carry a meta-tag at the start of the page. This is invisible when browsing the page, but holds a set of keywords for web-crawlers to associate with the page itself. Using these meta-tags, it should be possible to transcend the users search requests, and match these meta-tag lists with a page that the user has a high confidence in anyway. This means that instead of relying on the users supply of keywords, the system can base the checking of pages to be viewed against a whole page worth of material, or just the meta-tag's list. This sort of functionality is much more complicated than it sounds, requiring multiple word matching and decision making to decide the relevance of a page. This sort of extra requirement involves another increase in the turnaround between reading a page, and deciding on its content.

4 : Basic Design of the tool, Strategies and methods

The are three main stages in the process of forming a 3D VRML scene from an Internet structure. The final result desired is 3D objects, with characteristics bound by the status of the object on the internet, lines (or thin cylinders) connecting the structure together, laid out to create the scene. So, for starters, we need the information on the web pages themselves. The system needs to be given a URL origin, the URL to start the whole mapping process from in the first place. This needs to be scanned into either a local file, or in situ on the net. It needs to be scanned in whichever location, word by word. (Word by word referes to each solid unbroken string of characters, which would include the long strings, which make up the HTML hyperlinks which we need to describe the web structure being investigated.
If we treat each page as a separate object, we can associate each page (or node, this may be an easier way to refer to each object in the scene, as it forms a logical network of connections) with the connections it provides. If we are to link pages back to this, (in the occasions there is a reference both ways between two nodes) we need to store the URL of each page. This is also necessary if we wish to use the final model as a browsing aid. We also need to understand how we are going to represent an individual page. If the page doesn't exist (the file no longer exists at the URL) we need to show this to the user, eliminating the page from the possibilities, and reducing the evaluation time that the user might consume going through a link list.

So, the basic storage method for the web pages requires several elements, these are

1. URL of the web page to be displayed
2. Basic Status of the web page (existent, non-existent, relevant, non-relevant, type of reference)
3. A basic co-ordinate in 3D space for the page
4. A list of external URLs that the page may point to

5. A list of internal nodes (other objects in the scene) that the page points to.

So if we are going to deal with multiple nodes using this same data, using a structure or class command in Java would make administration of our node system far simpler.

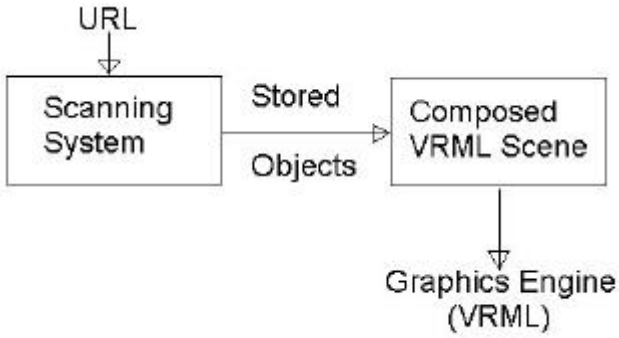
A cut down version of this map could be used to store information on each of the connector sections that fit in between each node, even though the link information isn't required, the position and orientation will be. Depending on whether the page is updated, or verified, after it has been set up to start with, these links may change, they may be discarded, and in this case, it might be useful to keep the link, but using some sort of visual cue, indicate the change on the model.

Each time a page is read in, we need to keep a record of how far (in order of connections between pages) the page is from the origin URL. In order to scale down the magnitude of the model (and to ensure that mapping does not continue into infinity), we need to restrict the number of jumps that the system takes from the origin. Looking at the maps produced by "Hyperspace", the average size of the maps produced looks to be using an average number of jumps under 10. The maps being produced look to be of reasonable size to navigate and map in a short amount of time. The option is always there to extend the jump limit to 20 or so, but the time to analyse all the pages that would be shown up in this case would increase exponentially, and the model would be hard to view with all the model in a view window. This is not a barrier to navigation, in VRML viewers, the user has the option to fly around to orient themselves to view only a small portion of the model, but the time constraints on building a model that big would probably rule it out. Depending on the software being used for viewing the model, large structures could also slow the processing and animation frame rates down exponentially, making smooth viewing of the model very difficult at the same time.

Once we have a logical storage of our map, we need to convert this into VRML, and this involves using a build up of static strings (basic building blocks of standard shapes) and combining this with a position in 3D space, and the URL that the object represents. Once these pieces are built up, we add this to a basic string representing the entire scene, and in the end, after all the objects and connections have been built up, this large string can be returned to the VRML viewer to be displayed.

This requirement, appears to reflect many characteristics common to object oriented languages, which is why Java was selected as the most effective programming language to be used.

The basic process stages, simple, but complicated to implement.



Alongside the object storage, once we have the position of each of the objects, we calculate a mid point between each object, and position a cylinder of appropriate length at this point, and rotate it in the 3 dimensions so that the ends of the cylinder end within the confines of each box object. This means the final string will be broken into 2 stages, the first half describing the page objects, and the second half describing the connection lines. After all the nodes have been described, and the connections described, we need a loop to add each section of the scene into one string, which at the end of the program, we return **as** a string, to the VRML scene.

5 : Coding of the tool, programming and development issues

Coding of the tool is simply a question of transferring the last chapter's designs into Java, with the VRML string compilation being returned at the end of the program.

In Java, the first step is to set up some form of Structure or Class information for the objects themselves. In this instance, because the objects may have an indeterminate number of pages connected to them, a dynamic list of items needs to be part of the object, so using the Class statement would be useful in this instance. The VRML 2.0 scene I have used is modified from an example taken from Lea (1996). A simple sphere to start with, using a Sensor statement to respond when the object is clicked on by the user. Using the URL statement in VRML 2.0, we assign a Java class to the object to be instantiated when the user clicks on the sphere, setting the program in motion.

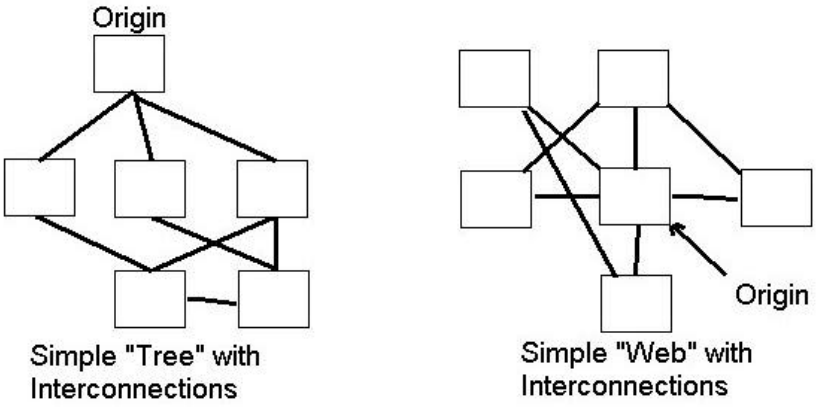
We need to store a basic shape for use representing the Web pages, to start with, a simple cube should suffice, with colours being used to differentiate between active and dead pages. So an active page could be

```
DEF PAGEn Transform {
        Children [
                Shape {geometry Box {size 1 1 1}}
                Appearance Appearance {
                        url PAGEnURL
                        material Material {diffusecolor {1 0 0}}
                }
        ]
}
```

This is perhaps no syntactically correct, but it defines a VRML object defined as PAGEn where n is the page in sequence of being built into the model. It has the shape of a geometric Cube, with the URL PagenURL assigned to it, with an RGB colour of 1 0 0. This base code cannot be stored like this, but broken up so that the PAGEn and PAGEnURL values can be entered at a later stage. An inactive URL would use the same code, but the material values would be replaced to display the object using a different colour. The only major problem with this code, is that if we want to associate this object with the rest of the scene, we need to build it into the VRML hierarchy, which is what I have had most difficulty with.

The first piece of code to be executed is to ask the user for a URL to start the system with. With this, we can use the Java.net library to copy a URL to a local file, renamed to match a code stored with the Class instance, so we can identify the file with the object being used in the program. We loop round this scanning of pages until we reach the jump limit (also entered by the user) away from the original URL.

Once this is completed, we need to decide how we are going to position the elements of the model in 3D space. Lea et al (1996) describe a 3D directory browser which uses a combination of variable sized circles to position elements of each directory, and then a tree formation to lay them out. This looks quite clear, but again, like other layouts we have looked at, does not allow for *inter*connections between pages. The end result will probably look like a spiders web, multiple pages expanding for the centre, with ample room for interconnections between satellite nodes. The problem comes when there is a connection between nodes on opposing side of the origin. This is where we need to introduce a third dimension to the display, raising or lowering one or both of these nodes, so the connection between them can be displayed without intersecting with another. This was another problem which hampered the development of the system, trying to develop a system to build a dynamic model which could be built up bit by bit, and still retain a degree of formality, without either making a huge program to do a simple task, or using some form of behaviour built into the objects, to make sure they do not overlap each other. A form of behaviour was used in “Narcissus” and “Hyperspace” to ensure spatial separation between objects.



Simple diagram showing the problems of each type of organisational approach.

This coding area is where my project has fallen down, in the case of describing an entire VRML string, I have found it almost impossible to develop the right hierarchy of objects and behaviour. Embedding the Java has also been difficult, with some software compilation errors that cannot be resolved at this time. At least in this system, there has been less of a complication than trying to integrate systems with major differences, rather than using a set of components that are *designed* to be integrated.

6 : Evaluation

This project has been effective in its research areas, combining many ideas from various sources to provide an effective design for a mapping tool. What has prevented me from producing a working piece of software has been a combination of several factors.

In looking back over the research I have done for this project, it appears that a lot of the papers and previous work has been done at a higher level than my capabilities. The Narcissus system from Birmingham university, in the form that the paper has taken in my research has no indication of what Academic level it was taken at, but my evaluation of it struck me as being of a higher degree than mine. The design of the system as far as this work has been concerned is fairly simple, looking at it from an engineering point of view, the building blocks for the system are fairly basic, however, final implementation of the system became very difficult, due to increasing time constraints, and a degree of software problems.

The software intended to be used to display the final product is a VRML 2.0 viewer supplied with Lea (1996) on CD-ROM. Over the development of the software, several problems became apparent, difficulty of compiling programs, and on compilation, some of the supplied examples of software failed to execute properly, and these problems I was unable to overcome. Even when using another VRML 2.0 compatible viewer, these problems failed to be resolved.

A major factor that I underestimated was how difficult I found programming in VRML 2.0. rather than just initiating an object into the scene, to create objects that connect to each other, and keep this integration when moving objects in space, VRML 2.0 uses a complex object hierarchy of objects, nodes and behaviour which I found very difficult to understand. This makes it very difficult indeed to write a program that essentially needs to be able to write VRML 2.0 in a dynamic situation.

The project plan that I devised at the beginning of the project envisaged coding to start towards the New Year, with debugging beginning in March in time for the final report in April. Unfortunately, my research over-ran, and the design I was developing required more research in order to make an effective system. This meant that the actual coding stages of the system did not start until a month and a half after they should have, meaning that I had no time to get around the software and coding problems that I have already described in time to finish the system. Even without a period of time to debug the system, integrating the VRML 2.0 and a Java program capable of doing the job has proved almost impossible for me to achieve. A lot of this has to do with my underestimation of the difficulty of the task I was undertaking. Looking back at my original design, and some of the ideas I had for the system (that are now discussed in the next chapter) I did not fully grasp the finer points and details of the functions required in the design to be able to write them in Java.

The plan as I presented it in my original proposal, and in my Interim report was also very vague, but at the time, I saw no need for greater detail, the project was capable of being broken down into the 3 basic stages of research, the design and coding

7 : Further development

In the course of this project, I have found other concepts from different systems that could be used to enhance the system in many useful ways, I would like to use this section to convey some of these, and discuss some of the ways the system could be made more commercially attractive.

7, 1; Individualisation of Web Pages

When first developing the concept of the system, it struck me that what is being created is a “map” of an area of web space, a virtual representation of a system, far removed from the actual physical world. In the “Real” world, navigation is carried out using landmarks. Now, in a perfect world, the net would be stable, with very few changes to it’s structure, but in reality, pages are created, deleted, modified on a fairly regular basis. However, the rate of deletion of pages is lower than the rate of creation, meaning on average, that the Internet is growing, and that most pages will continue to exist for a reasonable amount of time.

One method that is used a lot to navigate the web is by the use of bookmarks. In some cases, every single page that is of use is entered as a bookmark, but this can lead to multiple pages of bookmarks which make finding the right page even more difficult than using a search engine. The alternative is to bookmark “Landmark” pages, from where a known route to the page required is known. Now while the analogy is a little thin, using a similar tactic in 3D VRML space would be possible, but only if objects have unique properties, and areas of similar subject are to be visited fairly often. One way to do this, would be to store what would essentially be a VRML scene in the header of a page, so that when that page is accessed by our modelling system, instead of a standard representation, which would look identical to every other page’s representation in the model, we have a

"unique" object in our 3D space. The word unique is used loosely, as there is no way to enforce this, just by encouragement. This means that in a commonly used web-space environment of pages, it would be possible for a user to navigate our 3D space by visual landmarks, rather than logical ones. In an immersive system, this would be even more useful to minimise the requirement for input of text strings.

7, 2; Extending this to Multi-user environments

The use of Avatars was discussed earlier, and while these are used in Multi-user environments, our system is essentially a browsing and search aid for an individual. However, in the last section on making the Scenes we view more individual and "realistic", we have a larger scope for extending this into the domain of browsing the Internet in real time, and representing ourselves in this "virtual" web space.

The Ultimate extension of this type of system would be to use local servers to store information on the "landscape" of the Internet, much as DNS servers describe the local physical domains.

8 : Conclusions

Overall, there is a lot of scope and requirement for a mapping system such as this, in both the browsing, *and* the subject searching areas of the Internet. At the moment, there is still a rapid growth in the number of people getting connected to the Internet, with major advertising campaigns across the country aimed at novice users and families. With this type of growth comes two things. First of all, users with little knowledge of the net, who would benefit from a visual tool for browsing the internet, and who probably require the most help in selecting pages best suited to their needs. And then a rapid growth in homepages, and web development, increasing the number of pages we have to trawl through from link pages to get what we need.

The use of 3D systems is becoming more widespread, VRML will become easier to use, and easier to design for with the introduction of authorship tools, and programming aids. Which means an increased market for visualisation systems, and for navigation aids. It is too much to hope for that the Internet comes to the point where this system is the basic method of navigation, but the speed of the Internet will probably (hopefully) increase to the point at which the development of large, easily navigable models based around this system design take no more than a few minutes. At the moment, this system is still very time and Internet demanding, with evidence that the "Hyperspace" method, of waiting for every page to be viewed at least once before integrating it into the model, is the most efficient. This unfortunately means that the user has to evaluate the page themselves, instead of some form of pre-evaluation that would save time and internet resources. I think that on the whole, for the purposes of *my* project, VRML 2.0 and Java were reasonable selections for developing the system, but if a system were to be commercially viable, use of more efficient compiled languages such as C++ would result in a much faster and "finished" system.

Bibliography

Lea, R. Matsuda, K. Miyashita, K. (1996). *Java for 3D and VRML Worlds.* New Riders Publications: Indianapolis, USA.

Flanagan, D. (1997). *Java in a nutshell.* O'Reilly and Associates Inc: Sebastapol, CA, USA

Flanagan, D. (1997). *Java examples in a nutshell.* O'Reilly and Associates Inc: CA USA

Card, S.K. Mackinlay, J.D. Shneiderman, B. (1999). *Readings In Information Visualisation; Using Vision to Think.* Editors: Card, S.K., Mackinlay, J.D. Shneiderman, B. Morgan Kaufmann Publishers Inc.: California, USA.

Brown, J.R. Earnshaw, R. Jern, M. Vince, J. (1995). *Visualization; Using Computer Graphics to Explore Data and Present Information.* John Wiley and Sons Inc. USA.

Adams, E. Doherty, D. (1996). *Moving Worlds: Complete Coverage of VRML 2.0 Specification.* Prima Publishing: CA, USA

Hendley, R.J. Drew, N.S. Wood, A.M. Beale, R. *Narcissus: Visualising Information.* School of Computer Science: University of Birmingham.

Andrews, K. (1995). *Visualising Cyberspace: Information Visualisation in the Harmony Internet Browser.* In: IEEE Symposium on Information Visualisation, New York .

Drew, N.S. Hendley, R.J. Wood, A.M. Beale, R. (1995) *Hyperspace; Web Browsing with Visualisation.* URL : http://www.cs.bham.ac.uk/~amu/hyperspace/www95/

Appendix A : Original Project Proposal

Appendix B : Original Project Plan

Appendix C : Glossary

URL ; Universal Resource Locator/Location. This is a unique reference which points to a page on the Internet, usually made up of the server on which the page is stored, and the subdirectory(s) to find the file.

Meta-Tag ; A section of the HTML header, used to store a description of the Page, often in Keyword form.

VRML ; Virtual Reality Mark-up Language, the basic language for describing 3D scenes for use on the Internet.

HTML ; HyperText Mark-up Language, the basic language of the Internet, a hypertext system stored as ascii text.

OpenGL ; Open Graphics Library. A set of libraries re-written for use with different languages, used almost exclusively for 3D graphics development.

Java ; An Object Oriented Language developed by Sun Microsystems for use on the Internet. Well suited to this as the Interpreter is usually embedded inside Web browsers, meaning it can be used across several hardware and software platforms.

Avatar ; A term used for a model or object used to represent the user within a multi-user 3D environment.