METAPHOR-ENABLED INTERFACE ARCHITECTURES

by

Haseeb Quazi

A thesis submitted in partial fulfilment of the requirements for the degree of

Masters of Information Sciences

Massey University

2005

Approved by

Alexei Tretiakov
Supervisor

Date

Massey University

Abstract

METAPHOR-ENABLED INTERFACE ARCHITECTURES

by Haseeb Quazi

Supervisor: Alexei Tretiakov, Department of Information Systems

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEGMENTS

Praise be to Almighty God, Lord of the Universe.

First and foremost I would like to thank God for the motivation and guidance that He has given me to make this report possible.

My deepest gratitude and thanks is made to the following people in one way or another who have helped in the completion of this report:

To my supervisor Alexei Tretiakov, for his inspiration, guidance and expert advice, whom without none of this would have been possible.

To Dr Roland Kaschek for his valuable assistance and information with the literature of my study.

Finally, I am deeply indebted to my family, who have been the motivational force in my life. I thank them for their prayers, love and support.

Chapter 1

1. INTRODUCTION

In today's contemporary society, the most common method in searching for information is through Web Information Systems (WIS). The Internet is open to people regardless of the different backgrounds one may come from e.g. ethnicities, gender, age, culture etc.

The Internet is also widely used because of its simple layout and to an extent the language used is easy to understand. Although this may not be true in all cases. Some of the key websites such as e.g. Google, Yahoo and Wikipedia are designed to have easy and quick access to information.

Most languages incorporate metaphors and to an extent, so does the Web. Users interact with metaphors and respond to them in different ways and also they can search for information on the Web and use different web-based applications in their daily life such as Internet banking, online flight booking, online libraries etc. The information presented to the users is mainly controlled by the website and it might not be presented in a favourable manner that may suit the user.

Metaphors can be introduced into websites to enhance the presentation of the information and the way users interact with websites. The aim of this thesis is to identify an approach where metaphors can be employed as an extension of websites and adapt to the different user types in order to make their interaction with the website more efficient and effective.

Metaphors have been used in the computing domain since the early days (Jacko & Stephanidis, 2003). One of the common metaphors used in computing is that of the desktop we see when to turn on our computer.

Metaphors can relate concepts from one domain to another. For example a novice user of a particular domain may not be well acquainted with its concepts but using metaphors such as relating the concept at hand to one already known to the user. Metaphors are based on universal or local knowledge of the users and relating to already established concepts could help the users in understanding and grasping new concepts.

Deploying metaphors onto websites is a concept that is still in its early stages, hence metaphors should be carefully chosen. Although we assume that well-chosen metaphors will make the user interaction with a website or any domain much easier and clear. On the contrary, badly chosen metaphors can lead to confusion and misconceptions.

This thesis will give us insight into the assumptions made here.

1.1 Interfaces

The noun *interface* refers to be a discrete and tangible feature that we can map, draw, design, implement and attach to an existing bundle of functionality. We naturally visualise an interface as the place where contact between two entities occurs. An interface is a contact surface. It reflects the physical properties of the interacting entity, the functions to be performed and the balance of power and control.

When designing Interfaces the first and most important question to ask is, *what does the user want to do?* There is still much debate that agrees and disagrees on the designing of Interfaces being a separate thing from applications engineering (Jacko & Stephanidis, 2003).

In the context of this thesis, several Interface Architectures will be investigated which will lead to the proposal of a metaphor enabled interface architecture. Metaphors can be developed for particular domains and user types. In addition, Metaphors designed for particular domains and user type will allow other systems of the same domain to use them. For example, different banks can use metaphors designed for the banking domain. Therefore, the concept is somewhat of a plug and play technology where a metaphor interface application can be plugged into the existing system and subsequently metaphors will be invoked on the existing systems front end.

However in reality, software and hardware specifications have to be met by the domains system in order to access such applications.

In order to make a solid start to the project "Metaphor-Enabled Interface Architectures", selection of a domain for research had to take place. As the domain of electronic/Internet banking gains popularity, some initial investigation was undertaken regarding the way the users interact with such websites. Many banking websites were visited in order to gain an understanding of how the different applications worked. This led to the question of whether using metaphors in website interfaces will make user interaction more effective and efficient.

The next section is the history of the induction of metaphors into user interfaces.

1.2 Metaphors

Metaphors are frequently used in our everyday language. The characteristics of the metaphors in our language are the bases of how metaphors work in an interface. Metaphors occur throughout interfaces that we come across in our daily computing. Taking the example of the email is one of the metaphors, which has been well established in the Internet domain. Emails can be compared to the real life scenario of receiving mail in a mailbox. A user who has a mailbox on their computer can open this to receive their mail. The technology that sends the message to and fro can be taken as the virtual postman.

Not all interface metáphors are easy to use; some can be misleading. For example, a metaphor debated till now is the Trashcan on the Macintosh desktop. In order to eject a disc inserted into a disk drive one has to drag and drop the disks image on the desktop onto the Trashcan. This cannot be compared to someone throwing a disc into a Trashcan in real life. Hence it is vital that Interface metaphors are well designed in order to make the user interaction with system easier.

It should be taken into consideration that there are thousands of domains that use their products over the web. This thesis will concentrate on targeting a particular domain in order to get the best possible proposal for a Metaphor-enabled Interface architecture.

One of the most important aspects in designing such architectures is to look into the way users interact with the system. The next section discusses the importance and use of User types in the development of such architectures.

1.3 User Types

Over the years, studies on the usability of hundreds of product and web site designs have been carried out. There have been designs that were incredibly effective for users and designs that fell tremendously short. One emerging pattern in our ongoing research is that design teams that know a lot about their users are more likely to produce user experiences where the systems are usable, effective, and pleasing, as users can think in rather familiar ways as compared to a new domain (Kobsa, Koenemann & Pohl, 2001).

Using interface metaphors make the users more comfortable to understand a system. There are diverse users using the Internet to obtain the information they need. These users do not think in similar fashion and they are most likely to carry out tasks in different ways. Therefore, to cater for the different user types, systems could make use of metaphors and mapping them to the compatible users.

Metaphors have been used in teaching novice users as they are based on common knowledge. Different domains have different users and capturing the best way to present the information or functionality to them may be effectively done through the use of metaphors. It must also be made clear that all functionalities of systems may not be modelled with a metaphor.

Capturing information about the users of a particular system can make customisation of Interfaces more suitable and effective. User Types for different domains can be categorized by different information about the user. The term user data introduced by (Kobsa, Koenemann & Pohl, 2001) is to denote information about personal characteristics of the user, while the term usage data is related to user's behaviour.

The user data is directly obtained from the user while the usage data is taken from observing the users movements/behaviours. There have been systems developed using such techniques to obtain valuable information about users (Kobsa & Wahlster, 1989). In this thesis an architecture will be proposed, which will cater for adaptively viewing metaphors according to different customer types.

1.4 Architectures

"The architecture of a software system defines that system, in terms of computational components and interactions among those components" (Shaw & Garlan, 1996). In information sciences, architecture is a term applied to both the process and the outcome of thinking out and specifying the overall structure, logical components and the logical interrelationships of a computer its operating system, a network, or other conception.

Architecture is a model guiding implementation. Usually architecture is dealt with after a conceptual model has been obtained. Once the conceptual model is obtained, the architecture is then validated against it. Architectures consist of Syntax (structure of the architecture), Semantics (meaning of components and interactions) and Pragmatics (reasons behind structure & meaning).

In this thesis our aim is to propose an architecture for the Metaphor-Enabled Interfaces by formulising a conceptual model and validating it against an implementation of the Architecture.

As a result various architectures will be investigated in this thesis in order to develop the best possible architecture for the Metaphor-Enabled Interfaces.

1.5 Technologies

There are a wide range of technologies, which can be used in the development of such systems. The main focus of this thesis is to work towards the proposal of a metaphor-enabled web-based interface architecture. In this case technologies such as XML and XSL currently are in great demand.

Trausan, Novischi, Cerri & Maraschi, have developed a system for processing personalized Metaphors on the Web for learning a Foreign Language. XML was used for the annotation of Metaphors and XSL for visualization (Trausan, Novischi, Cerri & Maraschi, 2000). Other server side technologies have to be taken into consideration so that the dynamic functionality of the system could be handled. In this case, technologies such as Java Server Pages, Active Server Pages, Servlets, PHP, Java Applets and Java Beans will be investigated and subsequently one will be chosen to develop a working prototype of a Metaphor-Enabled Web-based Interface Architecture. In further chapters various technologies will be looked into in detail in order to compare and evaluate in the areas of functionality compatible to the chosen domain.

1.6 The objective of this Research

The aim of this research is to firstly propose a conceptual model that will map User Types to metaphors. The Higher-order Entity - Relationship Modelling (HERM) language was used to formalise the model. The conceptual model also helps in the creation of metaphors because other data regarding metaphors is required in order for it to fit into the model. The model also helps with User Type creation in a similar fashion.

Secondly it was required to design and develop a Metaphor-Enabled Web-based Interface Architecture. Metaphors have been a part of the interfaces with respect to computers from the early days. Now businesses are coming towards the Internet and the confidence is building as more and more commercial sites are setup everyday (Coffman & Odlyzko, 2002) (Odlyzko, 2003).

This means the user base will also expand, bringing diverse users to the already diverse Internet. In order to cater for such a diverse audience, an architecture can be designed where common knowledge is displayed as Metaphors. This will make the interaction easier and customisable, which will be a great step towards making websites friendlier. This can eventually bring in more customers and more returns to businesses. Therefore with the proposed conceptual model a

Metaphor-Enabled Web-based Interface Architecture was proposed that enables metaphors on a website to be invoked according to the User types.

Analysis on the feasibility of the architecture and the conceptual model are given throughout this thesis and a formal user evaluation exercise took place in order to validate the concept.

Chapter 2

2. LITERATURE REVIEW

2.1 Metaphor Theory

The word *metaphor* was defined as a novel or poetic linguistic expression where one or more words for a concept are used outside of its normal conventional meaning to express a *similar* concept (Lakoff & Johnson, 1980). Metaphors have always been a part of everyday speech. In the 1980's George Lakoff and Mark Johnson developed a logical and methodical way to express all the traditional theories about metaphors.

Lakoff and Johnson claimed that; metaphors is a property of concepts and not of words; the function of a metaphor is to better understand certain concepts and not just some artistic or esthetic purpose; metaphor is always not based on similarity as the same metaphors can be used in different contexts, metaphor is not bound to be used in certain areas where there are physical or linguistic similarities; metaphor is used effortlessly in everyday life by ordinary people, not just by special talented people; and metaphor, far from being a superfluous though a pleasing linguistic ornament, is an inevitable process of human thought and reasoning (Lakoff & Johnson, 1980).

Michael Reddy also came to the party in 1993 and clarified that Metaphor is a thought, not a language. Metaphor is a major and indispensable part of our ordinary, conventional way of conceptualising the world and that our everyday behaviour reflects our metaphorical understanding of experience. This opened up the idea of conceptual metaphors.

A conceptual metaphor is defined as understanding one conceptual domain in terms of another conceptual domain; e.g. using one person's life experience to understand a different person's experience. A conceptual domain can be any logical organisation of experience (Reddy, 1993).

2.2 Incorporating Metaphors in Systems design

The role of metaphors in the design of Information systems has been approached in many different ways. The main area where metaphors have been of success is in the design and development of user interfaces. When people interact with a system the place where the interaction takes place is the interface.

Interfaces are mainly what the users see and the processing of a user request are not usually seen. From an interaction point of view the interface is a crucial part of the entire system. Hence, the incorporation of metaphors into the interfaces of systems can enhance the interaction between the users and systems. The creation of metaphors for interfaces has been discussed below in the light of research that has taken place (Erickson, 1990) (Johnson, 1994) (Madsen, 1994) (Raskin, 1997).

There have been proposed frameworks for creating effective interface metaphors. Erickson approached this in three steps: the functional definitions, user problem identification and metaphor generation. The issue that came to surface was the generation of metaphors which was highly dependent on the designers' knowledge of the system. The designer has to completely understand the functionality of the system in order to identify effective metaphors. This point can be argued as there can be several effective metaphors and then choosing the best one in order to keep the user interested would not be so easy.

Identifying the user problem is one of the key aspects as this may not be possible to identify before the system is created. Even though user testing can be undertaken, the users movement across the system can be monitored to which metaphors can be associated accordingly. Erickson's third step "Metaphor generation" is interesting, as it combines the first two steps but makes it clear that the metaphors should not overshadow the aim of the system itself.

Similar work has taken place by Madsen (Madsen, 1994) who came up with a formal methodology for metaphorical design. It had similar constructs to those of Erickson. Madsen based his research and findings on pre-existing cases such as, the design of a small command language, the design of a task in which users can define links between parts of different computer documents, the design of a bank automated teller machine, the initial design of a production planning system and different services at libraries.

These cases were looked at with respect to fitting them into different metaphors such a travel agency, a bakery, house cleaning, cooking etc and then choosing the one that best fits the case. These cases led to the proposal of the guidelines on how to derive metaphors.

The guidelines were structured with three main activities, which are generating metaphors, evaluating metaphors and developing metaphors. Madsen stated "generating metaphors concerns getting ideas for potential metaphors to be used in the design process. Evaluating metaphors concerns choosing from among the potential metaphors the one (or the ones) that may fit the particular design task in a productive way. Developing metaphors addresses using the metaphors chosen in the actual design work" (Madsen, 1994).

These researchers made it clear that a more systematic qualitative study on the effectiveness of using metaphors in design needed to be undertaken. In this thesis one of the main aims is to try and physically demonstrate how metaphors and metaphorical structures can be used in web based interfaces and how they work.

Metaphorical structures are the newfound branch in metaphor design. Thalheim and Dusterhoft (Thalheim & Dusterhoft, 2000) have done considerable work in this direction. They defined metaphorical structures and gave practical examples.

Carroll and Thomas (Carroll & Thomas, 1982) brought up the Structure principle that "people learn structures, not isolated pairings (fallacy of composition) – a system interface should be a coherent structure". When defining metaphorical structures, the notions of metaphors, allegories, metonymies or synecdoche's are considered. Metaphorical structures have the ability to communicate and aid learning. When introducing metaphorical structures into websites the domain should be well researched in order to integrate the structure, functionality and interfaces.

Thalheim and Dusterhoft came up with a co-design approach for consistent development of metaphorical structures in an integrated manner. Co-design approach itself uses metaphors such as the dialogue approach, where users are looked at as actors in a certain role (Thalheim & Dusterhoft, 2000).

Erickson previously discussed the methods of evaluating interface metaphors. He came up with four main aspects of metaphors. The first one was that metaphors must have structure. Followed by the second one which was to apply the structure, the third one which states, how representative is the metaphor? The forth and final one was extensibility, as the need for metaphor to be expandable to related domains (Erickson, 1990). This brought the usability effect of metaphors in interfaces to attention.

2.3 Interface Metaphor Usability

In this thesis one of the aims is to research how metaphors affect the usability of systems when introduced into interfaces. Previous studies (Thalheim & Dusterhoft, 2000),(Erickson, 1990) and (Wells & Fuerst, 2000) claim that there are significant positive effects when introducing metaphors into interfaces.

These findings are mostly based on theory, as no significant empirical data is available to satisfy such claims. In the "Metaphor-enabled Interface Architecture" project, one of the aims is to evaluate the effectiveness of such Interface metaphors towards the usability of the system. In later chapters the process of evaluating the prototype and extracting valuable data to validate such claims will be discussed in more detail.

There has been some criticism as to the limits of interface metaphors. Kay said that the initial metaphors are effective for example, a paper in a typewriter as to a page in a word processor, but the computers can do a lot more than just simulate a type writer. Now word processors are used more frequently in comparison to a typewriter which was widely used in the past. Hence, there is definitely some computer processing and it is not all just metaphors, but instead the metaphors can help in extending the systems as they are used through time.

Metaphors can be very powerful tools in extending currently used systems as a user get familiar with structures that can be extended with the arrival of new and more powerful technologies (Kay, 1990).

2.4 Personalisation of Metaphors in websites

One of the most important ways to add value to a system or business is to serve the customers like you would know them. Users need to have a sense of ownership or friendship with the system or business. Kobsa, Koenemann and

Pohl very clearly and comprehensively discussed the value of personalization and the World Wide Web (Kobsa , Koenemann & Pohl, 2001). They looked into the World Wide Web from a business perspective and claimed that it supports the entire sales cycle, such as:-

- *Pre sale phase:* it establishes and strengthens corporate and brand identities, draws customers attention to new products and services.
- *Sales phase:* enables customers to select the desired products and buy them.
- *Post sale phase:* Reassures customers of their purchase decision and to deliver additional values through services like product-support for user groups and loyalty programs.

Customer relationship software have supported sales and marketing divisions in their tasks to provide individualised customer care. They integrate and utilise information from various sources to create targeted information, services and product offers. The question that arises now is that, how do metaphors fit into this?

With such customer relationship software available on the web, it enables the collection of information about customers (interests, online purchase behaviour, support needs etc) and this also gives us the opportunity to dynamically create the content and presentation formats for narrow-targeted and/or personalised information delivery.

Now the notion of introducing metaphors into interfaces can be looked into as personalisation of metaphors. People from different backgrounds can understand metaphorical structures, but there are cases where metaphors can be only understood by a particular group of people, who can be categorised by their interests, locality, age, sex, religion etc.

Hence gathering such information through customer-relationship software can result in the creation of personalised metaphors as well as more robust metaphorical structures. In the prototype developed for the "Metaphor-enabled Interface Architectures", the initial design supported categorization of users by different parameters. However, in order to test the prototype and due to time limitation the users were categorized by country, user types of countries such as Pakistan and New Zealand were created.

2.5 Domain-Oriented Interface Metaphors

The use of the Internet is becoming tremendously common meaning that a large variety of people are using it. These users will have different requirements and characteristics and will interact with different domains via websites.

Metaphors are used in computer interface extensively but, at the moment there is an increasing need to identify and develop more effective metaphors that will maximise the users' interaction with the systems and make the experience pleasant, efficient and effective.

Metaphors are defined by how users perceive them, in their own environment. However the domains still play an important role in extracting potential interface metaphors. Some research has taken place (Wells & Fuerst, 2000) in order to answer some of the questions that are as follows:

- How effective is a metaphorical interface that utilizes objects derived from the user domain? If it proves to be effective, how so?

- Does it improve a users' ability to retain information?

- How does it affect a users' attitude or impression of an organisation?

John D. Wells and William L. Fuerst conducted an experimental study. Two front-end interfaces were designed to present the exact same information. The experiment used two independent variables to measure the information retention of the users, which were Mode of Interface and Mental Model Type.

Modes of Interface:

- **Domain-Oriented Interface Metaphors:**

The design principles developed by Erickson & Madsen (Erickson, 1990) (Madsen, 1994) were used to create this interface. It used objects and processes that were specific to the user domain. In this experiment the user domain was a vacation resort.

- **Frame-Oriented Interface Metaphors:**

This interface was designed using the HTML design principle and the now common concept of frames in web interface design. Both Interfaces displayed same graphics and text. The main difference between the two interfaces was that the information was organised in a different way.

Mental Model Type:

Users were separated into two mental model groups: weak and strong. The users were separated using a questionnaire that was developed using the mental model concept of knowing a person's ability to interpret a metaphor effectively (Gentner, 1983) (Gillan, 1995).

After these users were categorised by their mental model, users within each group were randomly assigned to either a domain-oriented interface or a frame-oriented interface.

Three types of information retention were measured:

- Textually explicit – Easy to read as text.
- Graphically explicit – Easy to see through a visual.
- Graphically implicit – Not easy to see.

The results from the study indicated that there was a significant difference between the two interface types while the mental model type did not stimulate a significant difference. It was also concluded that domain-oriented interfaces provide a more effective presentation language for the interface while keeping the action language consistent. The creation of domain-oriented interfaces adds structure to the existing methodology for creating interface metaphors.

The Mental model theory was justified, as significant differences between the two types of interfaces were found (Wells & Fuerst, 2000). The navigation of the domain-oriented Interfaces was found to have no advantage over the frame-oriented interface. However it was suggested that textually explicit information would be more effectively served by frame-oriented interfaces and graphically explicit information would be better served by domain-oriented interface.

Even users developed a more positive attitude towards the organisation when they used the domain-based interface. This research contributes to the understanding of the importance of users and domains in developing Interface metaphors. In the development of the metaphor-enabled interface architecture it is vital to understand who the users are and what the domain is.

The conceptual model in the future chapter will deal with the mapping of users to metaphors, although these users will belong to a particular domain, therefore investigating the domain models and user models will play a vital role in defining the requirements for the architecture.

2.6 Interface Architectures

Developing an architecture to cater for enabling metaphors on an interface and managing them, led to the investigation of different User Interface architectures. In the ideal case, metaphors are introduced to a web-based application as another layer, it was crucial to somehow separate the metaphor side of the system as much as possible from the actual system. Most recently developed user interface architectures separate the interface from the application, which helps in portability of applications, reuse of components, use of multiple interfaces and customisation of interfaces. Firstly the Seeheim model for user interface architectures was investigated.

2.6.1 Seeheim Model of UI Architecture

It's made up of three components

- presentation system
- dialogue control system
- application interface

Presentation System

The main purpose of this component is to convert the external physical representation to an internal logical one. It can generate images on the display and also read data from input devices and convert the raw data into forms for dialogue control component to read.

Dialogue control system

This defines the structure of the dialogue between the user and the application. It accepts input from the presentation system and from the application (data to be displayed or data requests) and routes it to the appropriate destination.

Application Interface Model

This is a representation of the application from the user interface's perspective. This includes the specifications of application-significant objects, application operations and the mapping from objects and operations in the interface to the actual application data and routines (Brewster, 2001).

The Seeheim model was developed in 1983 and works adequately for simple command-language based user interfaces. Since then extensions of the Seeheim model have emerged and formalised such as, the knowledge-based front ends (NBFE) model and Persona. They used several concepts including user modelling, which is of interest as in this thesis the aim is to model different User types to corresponding metaphors.

The proposed architecture will be able to recognize the user, know which metaphor is of use to the user, display them and determine what one may want to see next. The other interesting and useful concept used in Persona was that of the tailoring agent (Reynolds, 1997).

"A tailoring agent would acquire information from the user and the user's persona and then use an expert system to determine which combination of user-interface settings would be most appropriate for the user. Before permanently altering a user's persona, the tailoring system would verify the accuracy of its choices by polling the user to see if the modified user interface is to the user's liking. Using this technique, the tailoring agent adapts the user-interface to the user instead of forcing the user to adapt to the interface" (Reynolds, 1997).

This led to the investigation of service agents, which is discussed later in section 2.8 of the thesis and also included in the proposed architecture.

2.6.2 Model View Controller (MVC) architecture

Next the Model View Controller (MVC) architectural pattern was investigated. This is an Object-Oriented architectural model, which encompasses interacting objects.

The *Model* is any object (application) – It defines the state of the system, in other words the underlying logical representation, business logic etc.

The *View* is an object, which provides a visual representation of a model (output) – It defines how users see the model/front-end Interface.

The *Controller* is an object, which handles input actions, sending messages to the view or model, as appropriate (input) – It defines how user interacts with the model.

In the MVC architectural pattern, the view is dependent on a model. The model sends a message to the view when it changes. The view re-displays whenever the model is changed. Changes to the model that affect the controller may be registered as a dependent of the model (Cavaness, 2004).

One of the main attractions for looking into this architectural pattern was that a view may contain sub-views. In the case of having a metaphor layer on top of an existing interface, this kind of flexibility is of interest as metaphors can be invoked as sub-views or completely new views without disturbing the original/standard view.

A Single model can be associated with different views and controllers, hence having the same applications with different interfaces (each View-Controller pair only associated with one Model). A separate view could be created for metaphors, which may have a model of its own within it or may use the main model.

The Model View Controller architectural pattern has been discussed on a higher level in this section; hence this led to the investigation of how to use this architectural pattern in developing web applications (Cavaness, 2004).

2.6.3 JSP Model 1 and Model 2 Architectures

As the scope of this thesis is mainly directed at web applications, using technologies such as JSP (Java Server Pages) was favourable. Currently two JSP architectures namely Model 1 and Model 2 are widely used by the web-development community.

In the Model 1 architecture the JSP page handles the requests and the output of the user. The JSP page may talk to JavaBeans or other services but, the JSP eventually determines what to display to the user next.

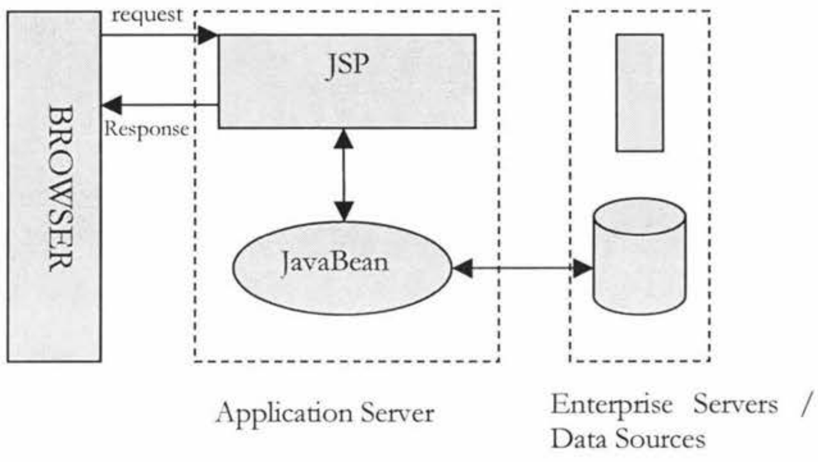


Figure 1: Model 1 architecture (Cavaness, 2004)

In the Model 2 architecture the users request is intercepted by a controller servlet, which handles the request and determines what to display next. The user cannot send a request directly to a JSP page. This allows the servlet to carry out processes such as user authentication, internationalization etc.

This architecture was of interest, as it seemed to be relatively simple to set up and the controller servlet might be able to process metaphors at the front-end which may be faster and could be kept separate from the under-lying business logic (Cavaness, 2004).

Maverick is an implementation of the Model 2 architecture, which is easy to use and can easily be acquired from the Internet. It combines features from Struts, WebWork, and Cocoon2, which are well-established frameworks in the web development community (Schnitzer, Hernandez & Moore, 2001).

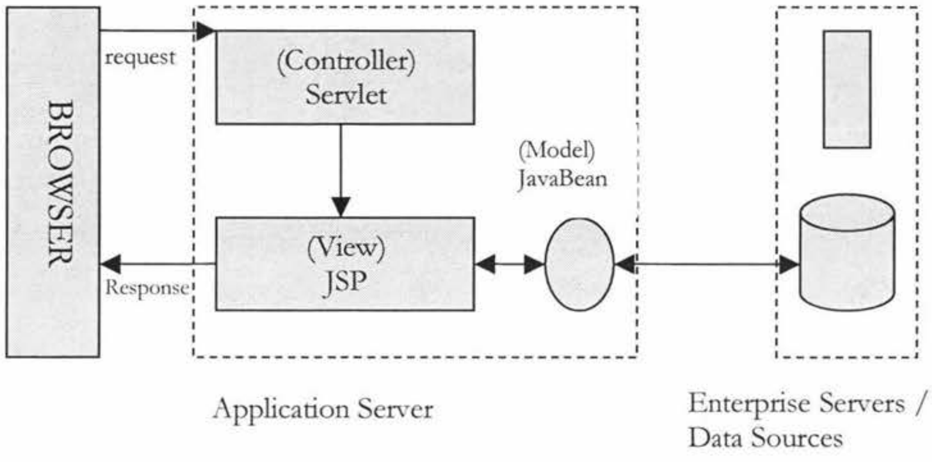


Figure 2: Model 2 architecture (Cavaness, 2004)

2.7 Clickstream Analysis

The clickstream analysis was looked into for the purpose of analysing users' behaviour at a website and then generating metaphors in relation to a behavioural map created from the clickstream analysis.

On a Web site, clickstream analysis (sometimes called clickstream analytics) is the process of collecting, analysing, and reporting aggregate data regarding the various visitors of the website and lists them in their respective order.

These are the result of the succession of mouse clicks each visitor makes (that is, the clickstream). There are two levels of clickstream analysis, traffic analysis and e-commerce analysis (Banerjee & Ghosh, 2001).

Traffic analysis operates at the server level by collecting clickstream data related to the path the user takes when navigating through the site. Traffic analysis tracks the number of pages served to the user, how long it takes pages to load, how often the user hits the browser's back or stop button and how much data is transmitted before a user moves on.

E-commerce-based analysis uses clickstream data to determine the effectiveness of the site as a channel-to-market by quantifying the user's behaviour while on the website. It is used to keep track of what pages the user lingers on, what the user puts in or takes out of their shopping cart and what items the user purchases.

As large volumes of data can be gathered through clickstream analysis, many e-businesses rely on pre-programmed applications to help interpret the data and generate reports on specific areas of interest. Clickstream analysis is considered to be most effective when used in conjunction with other, more traditional, market evaluation resources (Banerjee & Ghosh, 2001).

Carrying out a clickstream analysis to determine the use of metaphors can be a very useful tool however it is outside the scope of this thesis but can be taken up as an extension to this research in the future.

2.8 Service Agents

Business components often access internal and external services or applications. A service agent is a component that encapsulates the interface, protocol, and code required to use such services. For example, a business solution often requires information from the accounting system to complete a business process.

The solution would delegate all interaction with the accounting system to a service agent. The service agent makes it much easier to change the external service provider. Service agent can channel requests to the appropriate receivers at the backend. In the system being proposed in the thesis, the service agent communicates with a proxy that has a filter on it. This is an intercepting filter that can read through web pages and strip them of GIF's and compress documents on the fly.

The filter will be modified in order to enable metaphors on the interface. The metaphor application will be called upon once the service agent has passed the message through. In order to customise the filter to suite the objective of this thesis, a filter was investigated, which is discussed in more detail in the next section (Silva and Delgado, 1998) (Guttman, Perkins, Veizades & Day, 1999).

2.9 Intercepting filter

In the Intercepting filter pattern existing system resources can be wrapped within a filter that intercepts requests and responses. An intercepting filter can pre-process or redirect application requests and can post-process or replace the content of application responses. Intercepting filters can be deployed on webs application services without any changes to the existing source code (Sun Microsystems, 2002).

The intercepting filter that was researched was an open source filter named WebCleaner (Webcleaner, 2003). It is a filtering HTTP proxy that can disable animated GIFs, compress documents on-the-fly (with gzip), add/remove HTTP headers, and remove unwanted HTML (adverts, Javascript, etc.).

One of the aims of this thesis was to modify the filter so it could enable and disable metaphors on a webpage on the fly. WebCleaner was written in a programming language called Python (Webcleaner, 2003). For the purpose of this thesis Python was studied in detail but due to technical problems it was not used in the development of the metaphor enabled interface prototype. Although a proficient Python developer may not have any problems to further extend this intercepting filter with the metaphor model proposed here. Therefore instead of extending WebCleaner, a JavaBean was written and used instead.

Chapter 3

3. ANALYSIS AND DESIGN OF THE PROTOYPE

3.1 Research Area

After reviewing a considerable amount of literature in the area of Metaphors and computer interfaces, it was made clear that this area still requires a great deal of research. The use of metaphors in interfaces, in order to assist different user types is the major idea that will be developed throughout the passage of this thesis. So far not much research has taken place in this area and not many empirical studies have been under taken (Wells & Fuerst, 2000).

Systems where metaphors can be managed and maintained have not been developed yet. Therefore, systems need to be developed that solely look into the nature of metaphorical structures in information systems and especially user interfaces. As the Web is now easily accessible to users from different walks of life, it is of great value to capture information about these users in order to provide them with better means to view and access information.

3.2 Project content

The project "Metaphor-Enabled User Interfaces" has received funding from the Business Research Fund under the College of business at Massey University. Associate Professor Roland Kaschek of the information systems department is the Project Leader. The project was approved on the bases of opening new opportunities for software reuse and user interface testing. The project will also develop the track record of research on metaphors in user interfaces initiated by Dr. Roland Kaschek in his recent work (Tretiakov & Kaschek, 2005).

3.3 Project Scope and Feasibility

Metaphors are in use for various purposes in computer science. Their value as a tool to construct systems is contended. However, Thalheim and Dusterhoft suggest that usability of Web Information Systems may drastically increase by incorporating metaphors in their user interface. With existing architectures and interfaces, one cannot easily test or modify the metaphors used. The scope of the architecture being proposed in this thesis will allow easy deletion, addition or updating of metaphors used in an information systems user interface. The architecture will be applicable to both new and existing information systems. The prototype developed should have the following benefits:

1. The ability to test metaphors with the aim of determining the best metaphors for a certain group.

2. The ability to configure an information system with different metaphors for different user groups.

3. The ability to dynamically change the metaphors offered by an information system to accommodate changing requirements, without reinstalling or even restarting the core information system.

There will be great emphasis of the reuse aspect of such interfaces. Therefore efforts on separating the interface from the core information system by realizing the business functions on a façade, which will be designed to incorporate metaphors while offering full access to the capabilities of the core information system.

3.4 Prototype Analysis

The first stage of developing the prototype was to come up with a conceptual model that would be able set the basis for the further development of the prototype. In the following sections the conceptual model will be discussed in detail.

3.4.1 Conceptual model for a system managing metaphorical structures in WIS

The initial scope of the prototype was identified through a simple requirement analysis exercise. After reviewing the literature, analysing other systems and a few brain storming sessions the initial idea was put across with three major requirements. This is at abstract level and the idea is further clarified in later parts of the thesis.

R = Requirements:

R1 = Transform and Ship Information.
R2 = Transform and Display Information.
R3 = Change Metaphors in Runtime.

I = Interface

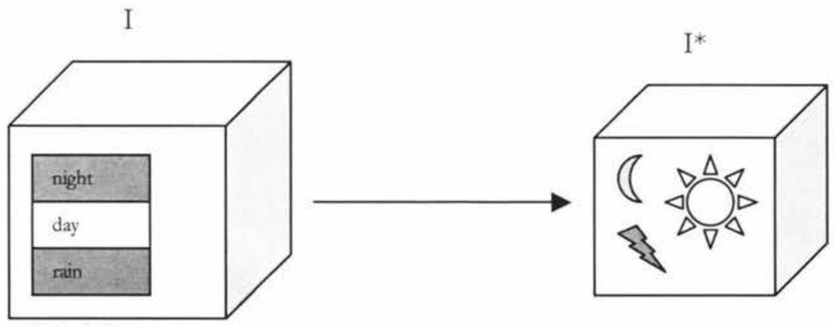


Figure 3. Initial concept.

R1: The prototype should be able to transform the interface, such that metaphors are added along with the requested information. The interface may also contain the information requested by a user in the form of an appropriate metaphor.

R2: The prototype should be able to transform interfaces in order to display the metaphors. Hence transforming will be an intermediate step before displaying the metaphors on the interface.

R3: The prototype should be able to change metaphors on the interfaces at runtime. Hence allowing updates to be made on the fly as the system grows.

As the research progressed, our understanding of the requirements above have changed. In the following section we will look into the design analysis in more detail with the help of uses cases.

3.4.2 Design Analysis

The design analysis stage started once the requirements of the project and prototype were made clearer. It is vital in any software development project to get clear requirements in order to succeed. In this case several brainstorming sessions with my project leader and supervisor took place.

In order to design the prototype a specific domain needed to be chosen. As Internet Banking has gained popularity in recent times, it seemed appropriate to choose it as the domain to be researched. This initiated a use case analysis of the domain. Many online banking systems were analyzed but due to confidentially reasons, their respective names will not be disclosed.

The following figure shows the use case for an online banking system with an adjacent metaphor enabling system. Figure 2 graphically shows the interaction between the two systems. Examples of the written use cases are as follows and complete documentation of the use-cases can be found in the Appendix.

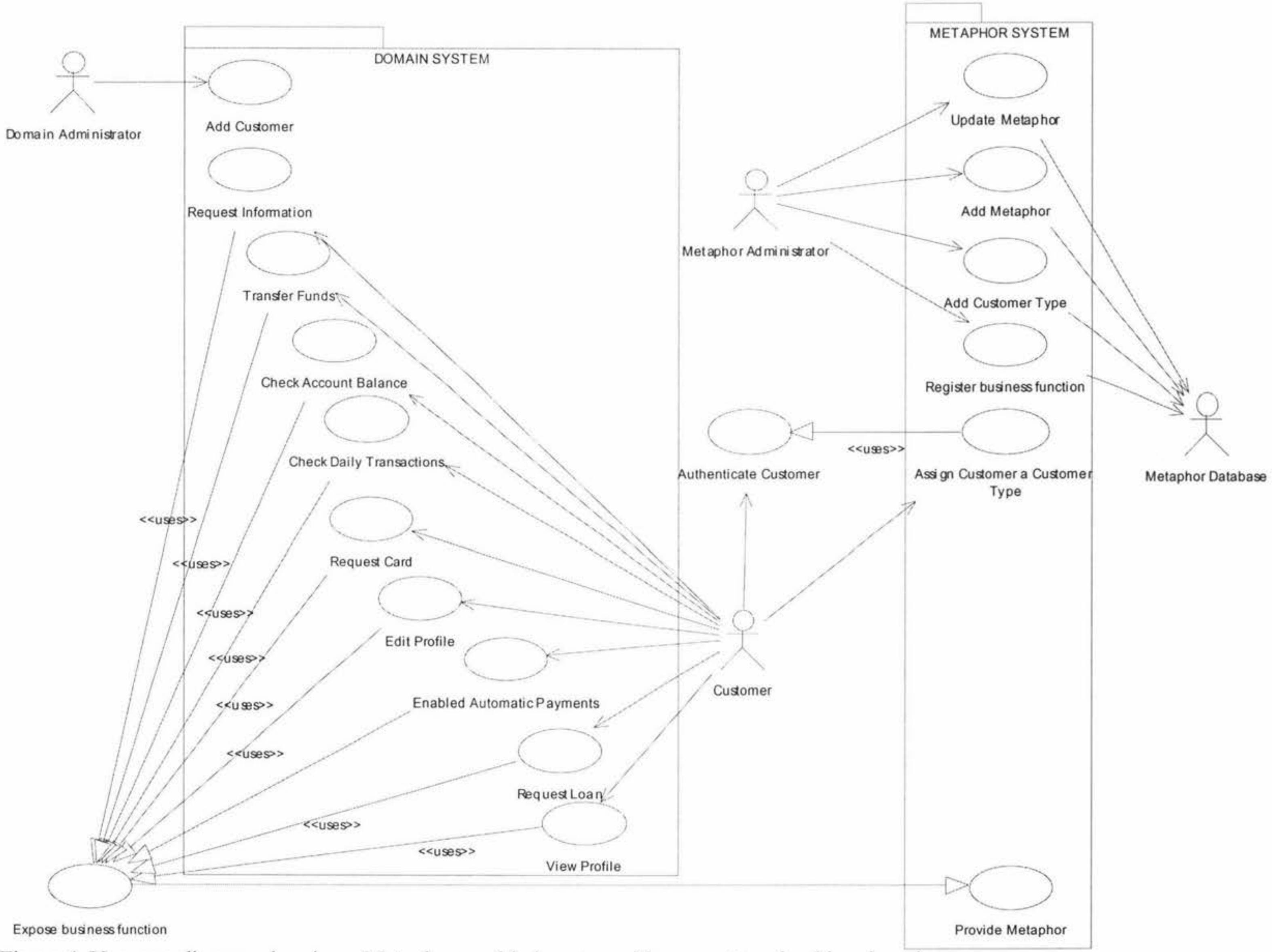


Figure 4. Use-case diagram showing a Metaphor-enabled system with respect to a banking domain.

USE CASE	Update Metaphors.	
Goal in Context	The Metaphors are maintained or kept track of, as they may not be use as effectively as others.	
Scope & Level	Administrator, Primary Task	
Preconditions	Metaphors already exist in the system.	
Success End Condition	Metaphors have been updated.	
Failed End Condition	Metaphors were not updated.	
Primary, Secondary Actors	Administrator, System.	
Trigger	Changes to the existing Metaphor come in.	
DESCRIPTION	**Step**	**Action**
	1	Administrator checks existing Metaphor.
	2	Administrator verifies that change/updating is required.
	3	Administrator changes/updates the existing Metaphor.
	4	Administrator saves changes.
	5	Administrator ends updating.
EXTENSIONS	**Step**	**Branching Action**
	2a	Administrator verifies no change/update required. 2a1. Administrator ends updating.

Figure 5. Example of documented textual use cases.

In order to design the prototype and capture the behaviour of an online banking system hypothetical scenarios were used in the use case exercise.

3.4.3 Internal Behaviour

The key design aspect of the prototype is to separate the Web-based information system from the Metaphor enabled interface system. In this way the Metaphor enabled interface system could act like a plug and play device that could be plugged into any system.

The Metaphor enabled interface system will not interact with the business functions directly, as seen in the use case diagram in figure 4. The metaphors and corresponding business functions are registered in the Metaphor database and mapped together.

The customer comes into the system once he or she is authenticated by the core information system. Once the customer is authenticated they are assigned a customer type.

A customer type will have metaphors associated to it and so will the business functions. Once the customer invokes a business function say for example "Check Account Balance" the business function will be exposed. As the business function is exposed, it contacts the metaphor system and checks if there are any metaphors provided for that particular customer type and business function. If no metaphors are provided, the system may provide a default metaphor or may not provide a metaphor at all. The details of such constraints will be documented in the design documentation of the database.

3.4.4 Configuration Requirements

The processing requirements of data are tightened with the use of constraints applied to the database model. The role of the systems administrator is to keep all the information up to date. This Metaphor database is developed as on-going research, as till now no such Metaphor database has been developed.

This database will help in future research relating to patterns of metaphor usage and customer types. It is also open to further extensions, as research carries on in this field. The creation of customer types and assigning customers to customer types is very complex and is out of the scope of this thesis (Kobsa, Koenemann & Pohl, 2001). Hence user types were created by testing the system in multiple locations (countries) as this is a simpler way of creating user types.

For example users from New Zealand will be assigned the user type "New Zealand". This way it could be seen how different customers react to different metaphors and if really our claim of increase usability prevails. Information storage requirements with estimates of size and growth have not been taken into consideration at this stage, as the prototype is the bases of gathering empirical data about the usability of metaphor enabled interfaces.

This prototype is the beginning of such research and may lead to the development complete systems that would be capable of maintaining metaphors on a larger scale.

3.5 Database Design

In the database the following information should be covered:

1. A catalogue of metaphors used by the system. The metaphor entity will have a name, Metadata and the Metaphor source. The Metadata is the characterization of the metaphor and the Metaphor source is the physical state or actual realization of the metaphor e.g., if the metaphor is an image, the path to the image will be stored here or if it's a line of text (linguistic expression) that will be stored here as well.
2. A catalogue of customers who are identified by their Customer ID.

3. A catalogue of customer types. Each customer is associated with a customer type. Each customer type is characterized by a unique name and description.
4. A catalogue of the Business functions, which is characterized by a name and description.
5. A catalogue of Metaphor Allocations that will satisfy which Metaphor and Business function is applicable for a certain customer type.
6. The catalogue of Metaphor explanations should be present, in order to show how a customer type may enable a metaphor and what the metaphor will do. In other words it is a help option.

3.6 HERM

The Higher-Order Entity Relationship Model (HERM) is used to design the database that will be used in this thesis.

"HERM can be used as a high-level, simple and comprehensive database design model for the complete database information on the structure, operations, static and dynamic semantics. The model has the expressive power of semantic models and possesses the simplicity of the entity-relationship model" (Thalheim, 1993).

HERM is a complete model from end to end in designing a database. In the following section a conceptual model is designed. The conceptual model is transformed into a Relational Data Model (RDM) and then translated into SQL in order to create the physical database. Constraints are defined by the conceptual model can be seen in Figure 6.

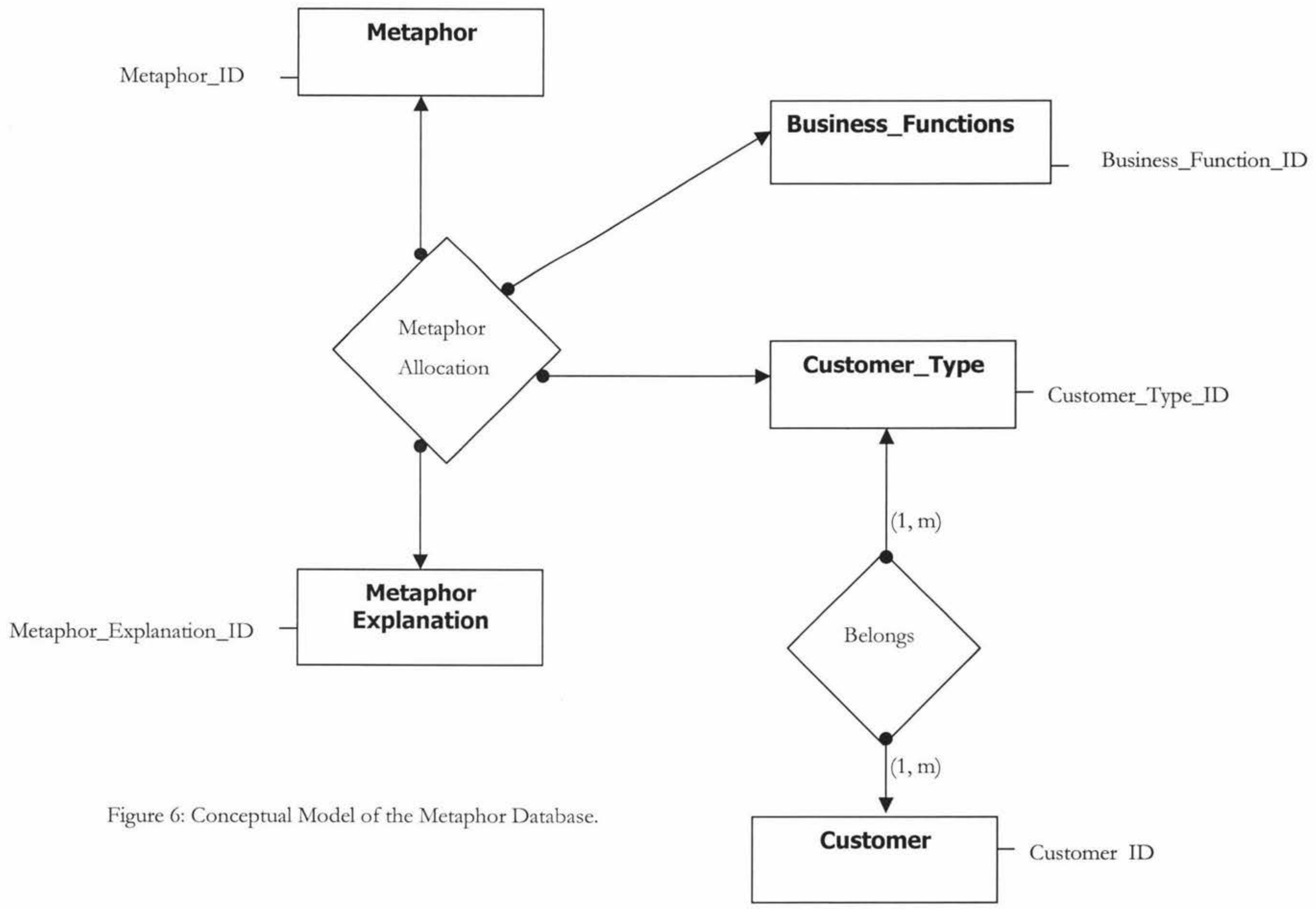


Figure 6: Conceptual Model of the Metaphor Database.

3.6.1 Database Schema:

3.6.1.1 Level 0: (Entities)

Customer = ({Customer_ID, FirstName, LastName, Email, Address, Country, Password}, {Customer_ID})

Metaphor = ({Metaphor_Name, Metaphor_Metadata, Metaphor_Source, Metaphor_ID}, {Metaphor_ID})

Business_Function = ({Business_Function _Name, Business_Function_Description, Business_Function_ID}, {Business_Function_ID})

Customer_Type = ({Customer_Type_Name, Customer_Type_Description, Customer_Type_ID}, {Customer_Type_ID})

Metaphor_Explanation = ({Metaphor_explanation, Metaphor_explanation_ID}, {Metaphor_explanation_ID})

3.6.1.2 Level 1: (Relationships)

Belongs = ({Customer, Customer_Type}, {}, {Customer, Customer_Type})

Metaphor_Allocation = ({Metaphor_ID, Business_Function_ID, Customer_Type_ID, Metaphor_explanation_ID}, {}, {Metaphor_ID, Business_Function_ID, Customer_Type_ID, Metaphor_explanation_ID})

3.6.1.3 Relations:

Customer

Attributes: ID, FirstName, LastName, Email, Address, Country, Password

Primary Key: ID

Customer_Type

Attributes: ID, Name, Description

Primary Key: ID

Metaphor

Attributes: ID, Name, Metadata, source

Primary Key: ID

Business_Function

Attributes: ID, Name, Description

Primary Key: ID

Metaphor_Explanation

Attributes: ID, Explanation

Primary Key: ID

Belongs

Attributes: Customer_Type_Name, Customer_ID

Primary Key: Customer_Type_Name, Customer_ID

Foreign Key: [Customer_Type_ID] $\subseteq$ CUSTOMER_TYPE [ID]

[Customer_ID] $\subseteq$ CUSTOMER [Customer_ID]

Metaphor_Allocation

Attributes: Metaphor_ID, Business_Function_ID, Customer_Type_ID, Metaphor_Explanation_IL

Primary Key: Metaphor_ID, Business_Function_ID, Customer_Type_ID,

Metaphor_Explanation_ID

Foreign Key: [Metaphor_ID] $\subseteq$ METAPHOR [Name]

[Business_Function_ID] $\subseteq$ BUSINESS_FUNCTION [ID]

[Customer_Type_ID] $\subseteq$ CUSTOMER_TYPE [ID]

[Metaphor_Explanation_ID] $\subseteq$ METAPHOR_EXPLANATION [ID]

3.6.2 List of Entity Types

3.6.2.1 Metaphor

A metaphor is a linguistic expression that may be used in an unusual context. We associate semantics or interpretations to metaphors in order to make it usable by different customer types and domains. The entity type Metaphor will have a unique name, with Metadata about it. Here the Metadata is the interpretation of the metaphor; as it can be a line of text describing what the metaphor means in the context of our system. The metaphor source is the physical state or the actual realization (e.g. images, linguistic expressions etc). Metaphors are created and then enabled by the systems administrator in order to be viewed/used by a customer and business function.

3.6.2.2 Customer

The customers that interact with the system are associated to a particular customer type. These customer types are associated to metaphors. However there is a possibility that a customer may not fall into any of the customer types available. In this case it was decided that a default customer type would be associated to the customer by the system.

A customer can also be associated to more than one customer type, as we can see from figure 6 that there is a (1, m) relationship. For example a bank customer can be associated to a home loan and credit card account at the same time. This can be translated as home loan and credit card being the customer types and a customer is associated to both of them. The customer entity type has a unique ID that identifies the customer.

3.6.2.3 Customer_Type:

Customer type is the grouping of customers into different categories, for instance in a banking scenario the customer types can be determined by the customers' areas of interest in the bank, such as Home Loans, Personal Loans, Credit Cards etc.

A customer may be associated to a customer type by tracking the customers' movement on the website. Throughout this research there has been an issue of categorising users into customer types. During the development of the prototype consideration on handling such categorisation had to be solved in order to evaluate the prototype in a short period of time. Hence, the customer types that came about were based on two distinct geographical locations (e.g., countries). In this way all the users belonged to the two customer types.

3.6.2.4 Business Functions:

The business functions are a sequence of functions that can be executed that may expose the metaphors related to a customer type. For example "Transfer Funds" is a business function that is executed by a customer type. Executing the "Transfer Funds" business function will change the look and feel of the interface according to the different customer types. This means that the interface will place different metaphors for different customer types that execute this business function. A simple example of metaphors in interfaces is illustrated in the figure below:

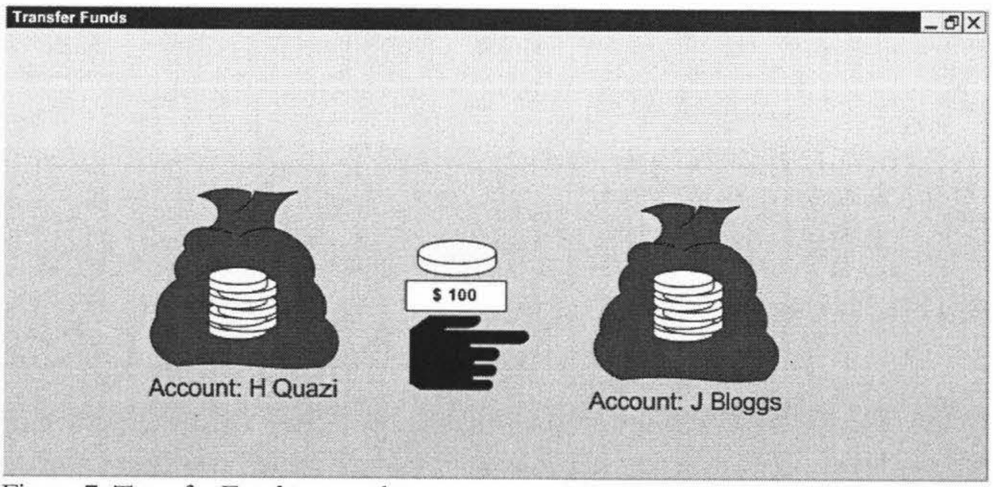


Figure 7: Transfer Funds example

In figure 7 the moneybags are a metaphor used to depict the accounts of H. Quazi and J. Bloggs. The amount to be transferred is entered into a text box and clicking on the pointing hand image will invoke the transfer function. However, this interface can be shown in a different way for a fisherman. The accounts can be represented by two fish tanks while a stream of water with fish flowing across to the other tank as the transfer function. These are just ideas and no implementations of such interfaces are available.

3.6.2.5 Metaphor Explanation

The user must be given an explanation about how a metaphor works and what business functions are activated for a particular customer type. The metaphor explanation is somewhat of a help option in order to have detailed documentation about how the metaphors could be used. The constraint that all metaphors must have an explanation is part of this thesis, however this can be debated.

3.6.3 Relationship Types

3.6.3.1 Metaphor Allocation

Description:

A 'Metaphor Allocation' relates the Metaphor to the Business Functions and the Customer Type. A Customer having a Customer Type can execute a Business Function. The Customer Type may have a Metaphor allocated. A customer without a customer type may also execute a business function, which may have no metaphor allocated to it. In that case a default metaphor or no metaphor will be displayed.

3.6.3.2 Belongs

Description:

A customer may belong to a Customer Type in order to use a Metaphor. If a customer does not fall into a Customer Type, a default Customer Type will be assigned.

3.6.4 Development & Delivery Platforms

The technology used to host this database was Oracle. Oracle *i*SQL*Plus Release 9.2.0.1.0 was used to create the database. The technology used for the server side programming was Java; hence an Oracle JDBC driver was used for the connection between the application and database. Other technologies such as mySQL and PHP were investigated, but eventually the prototype was developed using JavaBeans.

As mentioned in chapter two the implementation of the Intercepting filter pattern was undertaken, but due to technical problems the technology was not used in the implementation of the prototype.

SQL*Plus was the primary interface to the Oracle database server. It provides a powerful yet easy-to-use environment for querying, defining and controlling data. SQL*Plus delivers a full implementation of Oracle SQL and PL/SQL, along with a rich set of extensions. The exceptional scalability of the Oracle database, coupled with the object-relational technology of SQL*Plus, allows the development of complex data types and objects using Oracle's integrated systems solution. *i*SQL*Plus is a browser-based interface to SQL*Plus. *i*SQL*Plus is shipped with Oracle9*i* release 9.0.1 for Windows. *i*SQL*Plus is available on other platforms, including Solaris in the Oracle9*i* Release 9.2.0.1. (Oracle, 2004).

*i*SQL*Plus was used for querying, defining, and controlling data in the database. No specialised web interface has yet been developed to administer the Metaphor database, as this was out of the scope of the research and also a fairly easy task do to if needed later on. The screen shots of the Oracle iSQL*Plus interface used are as follows:

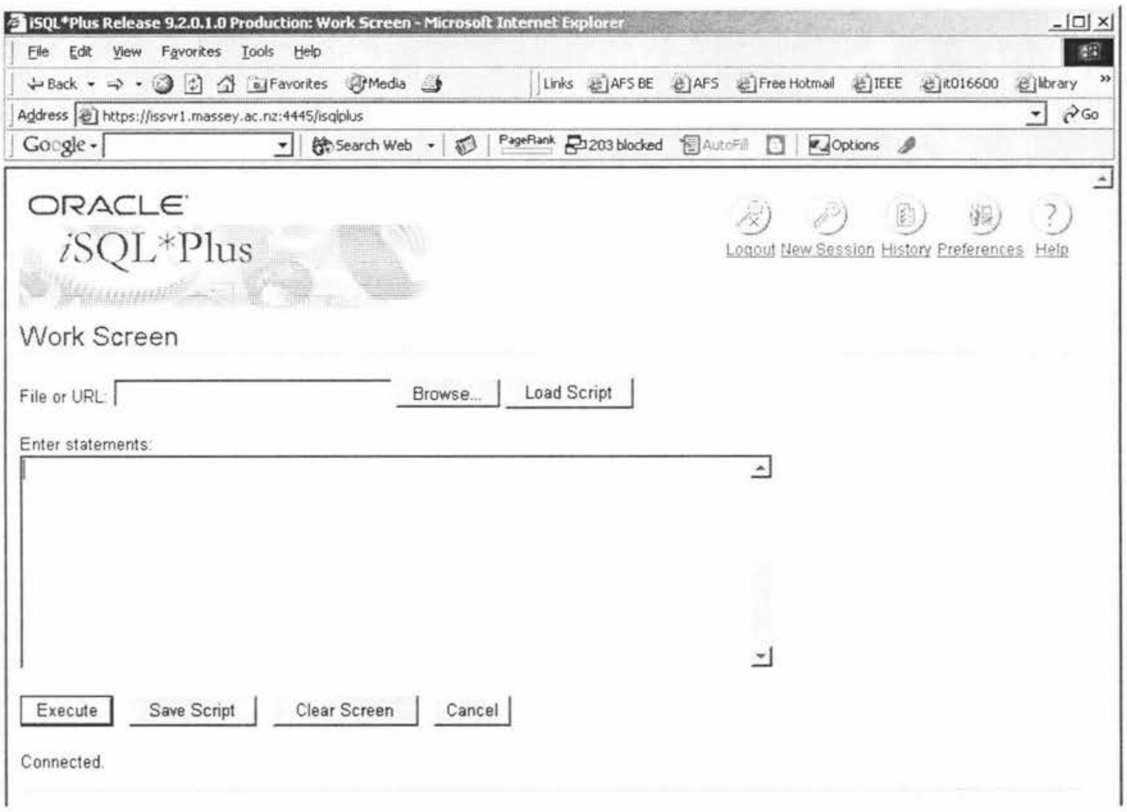
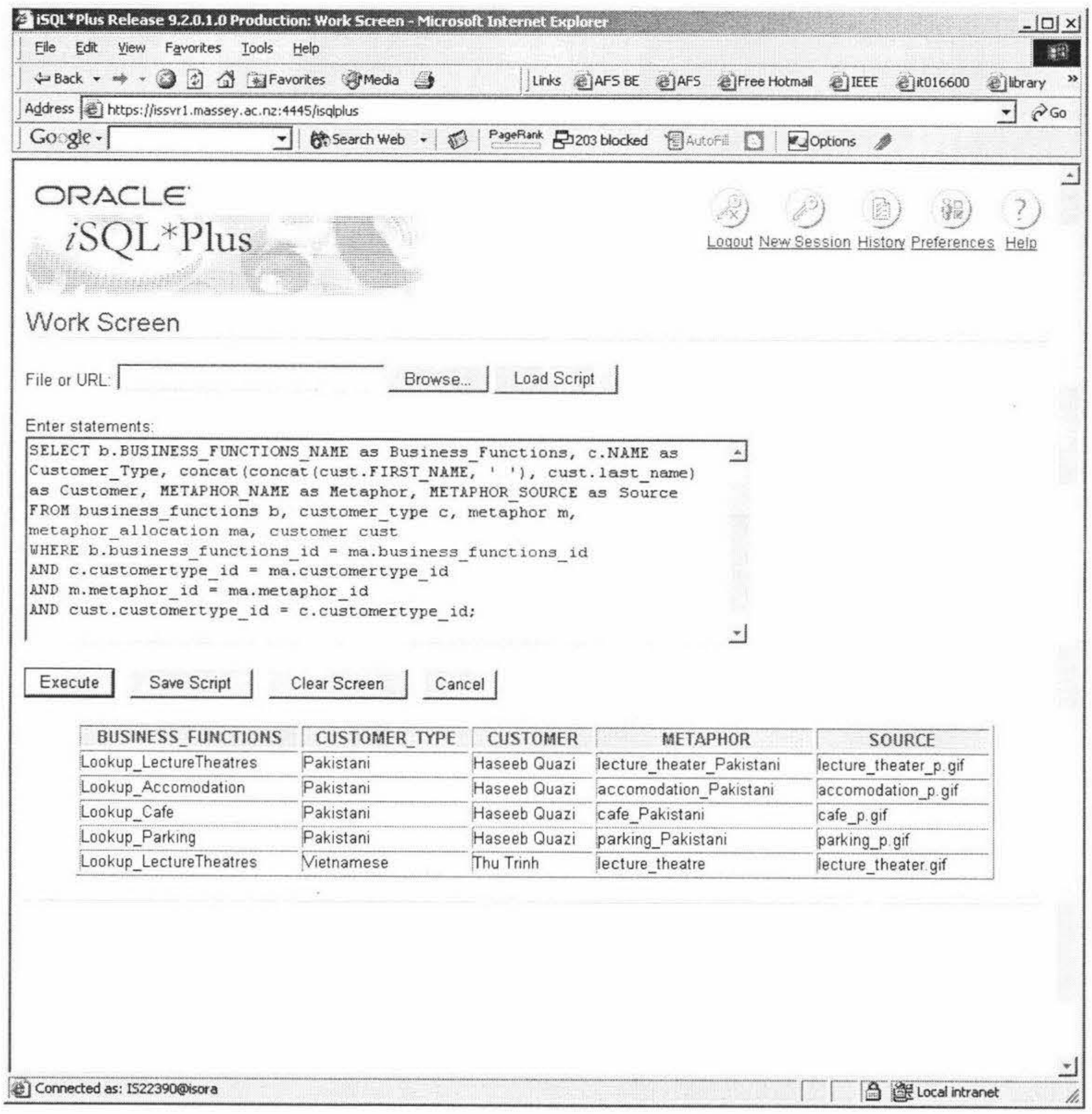

iSQL*Plus Release 9.2.0.1.0 Production: Work Screen - Microsoft Internet Explorer
Address https://issvr1.massey.ac.nz:4445/isqlplus
ORACLE
iSQL*Plus
Logout New Session History Preferences Help
Work Screen
File or URL:
Browse...
Load Script
Enter statements:
Execute
Save Script
Clear Screen
Cancel
Connected.

Figure 8: Oracle iSQL*Plus interface


iSQL*Plus Release 9.2.0.1.0 Production: Work Screen - Microsoft Internet Explorer
Address https://issvr1.massey.ac.nz:4445/isqlplus
ORACLE
iSQL*Plus
Logout New Session History Preferences Help
Work Screen
File or URL:
Browse...
Load Script
Enter statements:
Execute
Save Script
Clear Screen
Cancel
Connected as: IS22390@isora
Local intranet

```
SELECT b.BUSINESS_FUNCTIONS_NAME as Business_Functions, c.NAME as
Customer_Type, concat(concat(cust.FIRST_NAME, ' '), cust.last_name)
as Customer, METAPHOR_NAME as Metaphor, METAPHOR_SOURCE as Source
FROM business_functions b, customer_type c, metaphor m,
metaphor_allocation ma, customer cust
WHERE b.business_functions_id = ma.business_functions_id
AND c.customertype_id = ma.customertype_id
AND m.metaphor_id = ma.metaphor_id
AND cust.customertype_id = c.customertype_id;
```

BUSINESS_FUNCTIONS	CUSTOMER_TYPE	CUSTOMER	METAPHOR	SOURCE
Lookup_LectureTheatres	Pakistani	Haseeb Quazi	lecture_theater_Pakistani	lecture_theater_p.gif
Lookup_Accomodation	Pakistani	Haseeb Quazi	accomodation_Pakistani	accomodation_p.gif
Lookup_Cafe	Pakistani	Haseeb Quazi	cafe_Pakistani	cafe_p.gif
Lookup_Parking	Pakistani	Haseeb Quazi	parking_Pakistani	parking_p.gif
Lookup_LectureTheatres	Vietnamese	Thu Trinh	lecture_theatre	lecture_theater.gif

Figure 9: Querying the Metaphor Database through the Oracle iSQL*Plus Interface.

3.6.5 Metaphor Database Structure Design

Table Structure

Metaphor allocation

Name	Null?	Type
METAPHOR_ID	NOT NULL	NUMBER(38)
BUSINESS_FUNCTIONS_ID	NOT NULL	CHAR(11)
CUSTOMERTYPE_ID	NOT NULL	CHAR(11)
METAPHOR_EXPLANATION_ID	NOT NULL	CHAR(11)

Metaphor

Name	Null?	Type
METAPHOR_ID	NOT NULL	NUMBER(38)
METAPHOR_NAME		CHAR(30)
METAPHOR_METADATA		CHAR(100)
METAPHOR_SOURCE		CHAR(100)

Customer

Name	Null?	Type
CUSTOMER_ID	NOT NULL	CHAR(11)
EMAIL		VARCHAR2(64)
FIRST_NAME	NOT NULL	VARCHAR2(32)
LAST_NAME		VARCHAR2(64)
ADDRESS		VARCHAR2(64)
COUNTRY	NOT NULL	VARCHAR2(64)
PASSWORD	NOT NULL	CHAR(11)
CUSTOMERTYPE_ID		CHAR(11)

Customer Type

Name	Null?	Type
CUSTOMERTYPE_ID	NOT NULL	CHAR(11)
NAME	NOT NULL	VARCHAR2(64)
DESCRIPTION		VARCHAR2(64)

Business Functions

Name	Null?	Type
BUSINESS_FUNCTIONS_ID	NOT NULL	CHAR(11)
BUSINESS_FUNCTIONS_NAME	NOT NULL	VARCHAR2(64)
BUSINESS_FUNCTIONS_DESCRIPTION		VARCHAR2(64)

Metaphor_explanation

Name	Null?	Type
METAPHOR_EXPLANATION_ID	NOT NULL	CHAR(11)
METAPHOR_EXPLANATION	NOT NULL	VARCHAR2(100)

The complete documentation of the SQL statements used to create these tables is available in the Appendix.

3.6.6 Process Definitions

The processes that will take place in the implemented prototype have a very simple structure. The prototype is built using a hypothetical scenario, where a webpage displays information about different cities of two different countries. The users will interact with the website in order to carry out some given tasks. These tasks will be carried out while the metaphors enabled as well as disabled on the website. The website is dynamically created using Java Server Pages. The JSP pages call upon a Java Bean who takes care of getting the metaphors from the Metaphor database and on to the web pages. In the next chapter the Java Bean will be discussed in detail.

Chapter 4

4. IMPLEMENTING THE PROTOTYPE

Once the Metaphor database was completed the next phase was to create a system that would allow the metaphors stored in the database to be used with different web based Information Systems. After reviewing literature and doing further research into this area, an architecture using a number of design patterns was proposed.

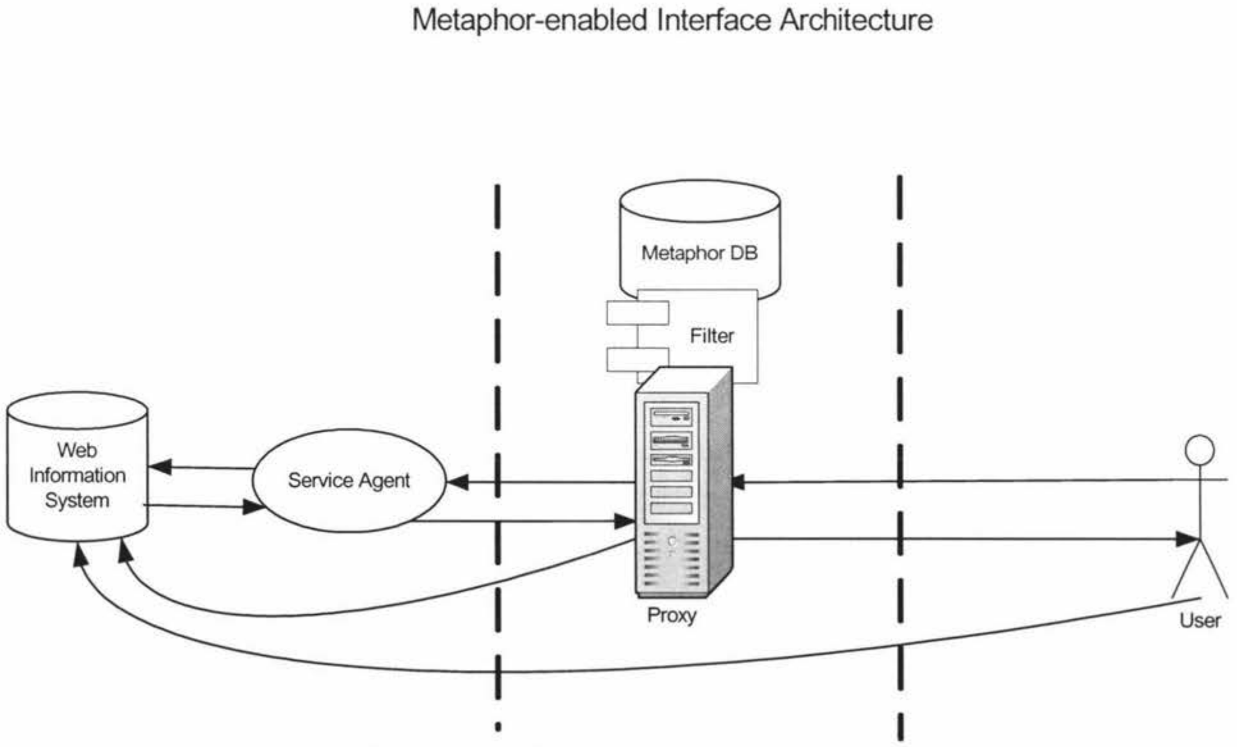


Figure 10: Proposed Metaphor-enabled Interface Architecture

In this architecture the users' request goes through a proxy. The proxy then talks to the Web Information System and processes the users' requests. In-between the Web Information System and the proxy is a service agent. The service agent aggregates the flow of data from the Information system to the proxy. Once the users request comes back from the information system to the proxy, the intercepting filter applies the metaphors to the requested pages. Therefore, the web pages are metaphor enabled at the proxy and the core Information system is not modified.

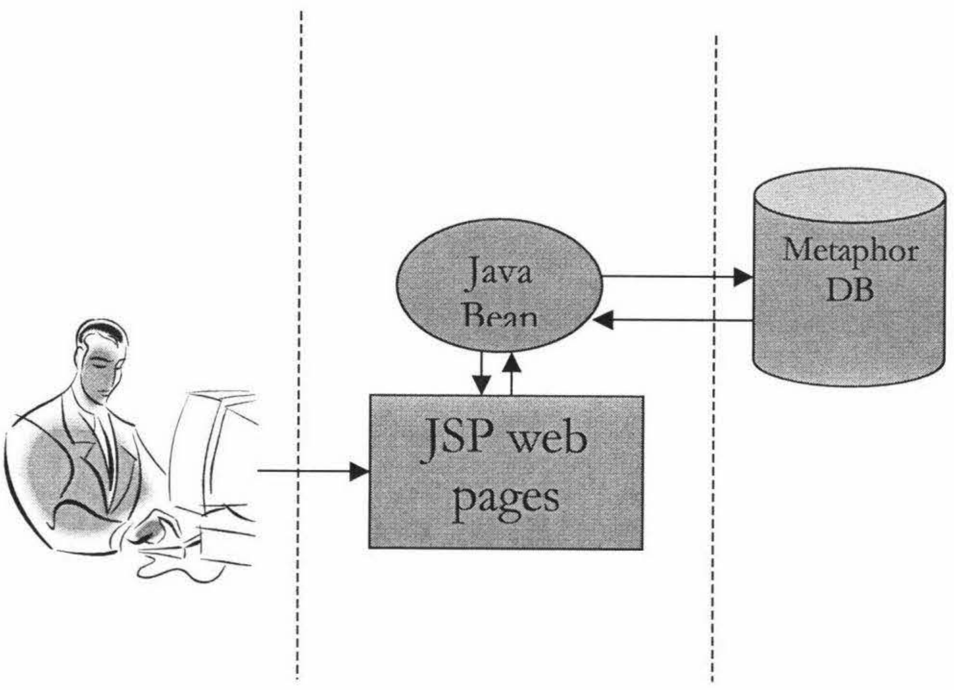
As the transformation of the interface happens at the proxy, the Web Information System does not have to worry about the content being changed permanently. The proxy / intercepting filter investigated was an open source application called Webcleaner. However, due to technical problems the application could not be used. An alternative approach was undertaken which will be discussed in the next section.

The service agent present between the proxy and the information system delegates the tasks to the proxy and web information system in a manner that makes it much easier and efficient to change the interfaces. The service agent may even simulate the metaphor-enabled interface to the users, facilitating testing of the metaphor layer.

4.1 Proposed System

The system developed for the evaluation uses technologies such JSP, JavaBeans and an Oracle database. The user accesses the system, which comprises of static HTML content. The metaphors are present mainly in image format.

The customer type for a particular user can be set by the administrator before they start their session, or the user can directly access the system. The system is created to cater for metaphors regarding users from Pakistan and New Zealand. For details on why these customer types were chosen refer to section 3.5.2.3.



4.2 Database and Java connection

There were basically two problems that needed to be solved before the relational database could be used from within a Java application. Firstly some basic middleware was needed to establish the connection with the database. Secondly manipulating the result sets from the database had to be done.

Sun Microsystems with many software companies defined a large number of APIs for common programming tasks. The API for Java database connectivity (JDBC) was among the first JDK 1.1 APIs to stabilize. There are numerous implementations of JDBC available from various sources (Sun Microsystems, 2000). Some of these are 100 percent pure Java. Others use a mixture of Java and native code, for example the existing ODBC data sources. The Java Soft people have put an extensive overview of available JDBC drivers on their Web site (Sun Microsystems, 2000).

Initially a Java development environment was set up and a JDBC driver was successfully installed and tested. The Java Bean was successfully deployed on the server with the connection to the metaphor database working seamlessly. Two JSP pages were setup in order to get the users input parameters and display the resulting pages. The Java Bean written to accept the users input and process the results can be seem in the sample code as follows:

```
//You need to import the java.sql package to use JDBC
import java.sql.*;
//We import java.io to be able to read from the command line
import java.io.*;
public class MetaphorsController
{
public String userType = "";

public void setUserType(String userType){
  try{
    this.adjustMetaphorToSuitUserType(userType);
  }
  catch(Exception e){
    e.printStackTrace();
  }
}
public String getUserType()
{
 return userType;
}
private void adjustMetaphorToSuitUserType(String userType)
       throws SQLException, IOException
{
   // Load the Oracle JDBC driver
   DriverManager.registerDriver(new oracle.jdbc.OracleDriver());
   System.out.print("Connecting to the database...");
   System.out.flush();
   System.out.println("Connecting...");
   Connection conn = DriverManager.getConnection
("jdbc:oracle:thin:@it020009.massey.ac.nz:1521:isora","is22390","is90");
   System.out.println("connected.");

// Create Oracle DatabaseMetaData object
   DatabaseMetaData meta = conn.getMetaData();

// Create a statement
   Statement stmt = conn.createStatement();
   // Do the SQL "Banner" thing
ResultSet rset = stmt.executeQuery("SELECT Metaphor_source FROM Metaphor " );

ResultSet rset2 = stmt.executeQuery("SELECT b.BUSINESS_FUNCTIONS_NAME as
Business_Functions, c.NAME as Customer_Type, concat(concat(cust.FIRST_NAME, '
'), cust.last_name) as Customer, METAPHOR_NAME as Metaphor, METAPHOR_SOURCE as
Source FROM business_functions b, customer_type c, metaphor m,
metaphor_allocation ma, customer cust WHERE b.business_functions_id =
ma.business_functions_id AND c.customertype_id = ma.customertype_id AND
m.metaphor_id = ma.metaphor_id AND cust.customertype_id = c.customertype_id");

while (rset.next())
   {
     String pixName = rset.getString(1);
     int n = pixName.indexOf(".", 0);
     //System.out.println(pixName + "-");
     String newName = null;
     if (n == -1)
        newName = "Metaphor"  + ".gif";
     else
       newName = pixName.substring(0, n) + "_meta"  + ".gif";

System.out.println("from " + pixName + " to " + newName);
    // doCopy(pixName, newName);
   }

while (rset2.next())
String Allocation = rset2.getString(1);
String Allocation2 = rset2.getString(2);
String Allocation3 = rset2.getString(3);
String Allocation4 = rset2.getString(4);
String Allocation5 = rset2.getString(5);
System.out.println
(Allocation + '\t' + Allocation2 + '\t' + Allocation3 + '\t' + Allocation4 +
'\t' + Allocation5 + '\t');
}
rset.close();
rset2.close();
stmt.close();

// close the result set, the statement and connect
     conn.close();
}
// Utility function to read a line from standard input
static String readEntry(String prompt)
{
   try
   {
      StringBuffer buffer = new StringBuffer();
      System.out.print(prompt);
      System.out.flush();
      int c = System.in.read();
      while (c != '\n' && c != -1)
      {
         buffer.append((char)c);
         c = System.in.read();
      }
      return buffer.toString().trim();
   }
   catch(IOException e)
   {
      return "";
   }
}
```

```
static void doCopy(String from, String to) {
 try {
   InputStream in = new BufferedInputStream(new FileInputStream(from));
   OutputStream out =
       new BufferedOutputStream(new FileOutputStream(to));
   byte[] buffer = new byte[(int)(new File(from)).length()];
   in.read(buffer);
   out.write(buffer);
   in.close();
   out.close();
 }
 catch( IOException e ) {
   e.printStackTrace();
 }
}
public static void main (String x[]){
  MetaphorsController mc = new MetaphorsController();
   mc.setUserType("Eskimo");
}
}
```

Chapter 5

5. EVALUATING THE PROTOTYPE

5.1 Introduction

The evaluation-method chosen for this research is the "Crossover trial". The Crossover trial is undertaken through the form of an informal survey. It relies on people's valuation of changes in their circumstances given a hypothetically constructed scenario. This chapter discusses the benefits to be valued, the development of the evaluation, the chosen population and the evaluation's implementation.

The purpose of the evaluation is to validate the hypothesis we have constructed, using the prototype implemented as a part of the metaphor-enabled interface architecture project.

5.1.1 Hypothesis

The hypothesis states:
Adaptively serving metaphors according to the user type has improved usability of the system, with respect to:-

1. Time taken to complete a task
2. Effect of user types with respect to understanding metaphors.

5.1.2 Crossover Trial Design

In a crossover trial subjects are randomly allocated to undertake a task where each task consists of a sequence of two or more scenarios given consecutively. The simplest model is the AB/BA study. Subjects allocated to the AB study tasks receive scenario A (Metaphors Enabled) first, followed by scenario B (Metaphors disabled), and vice versa in the BA task (Ratkowsky, Evans, Alldredge, 1993).

Crossover trials allow the response of a subject to scenario A to be compared and contrasted with the same subject's response to scenario B. Removing user variation in this way makes crossover trials potentially more efficient than similar sized parallel group trials, where each subject is exposed to only one scenario. In theory scenario effects can be estimated with greater precision given the same number of subjects.

Crossover trials are generally restricted to the study of short-term outcomes because the tasks-at-hand needs to go on long enough for the investigators to expose the subject to each of the hypothetical scenarios and measure the response. Also the scenario must be one that does not permanently alter the system under study. In this case metaphors enabled can be disabled leaving the website at its original state after all, but the users may remember the patterns of the scenarios (Ratkowsky, Evans, Alldredge, 1993).

The principal drawback of the crossover trial is that the effects of one scenario may "carry over" and alter the response to subsequent scenario being applied. The usual approach to preventing this is to introduce a washout (break) period between consecutive scenarios, which should be long enough to allow the effects of a scenario to wear off. In our case, the memory of a user may “carry over” resulting in bias. To eradicate this, we introduced a washout period of a week and change the structure of the task in the latter scenario.

5.1.2.1 Identification of benefits

If the evaluation accepts the hypothesis that adaptively serving metaphors according to the user type has improved usability of the system, then there will two main beneficiaries from such findings:-

- The web using community in general
- User focused web-based businesses

Web Using Community

The people using the Web come from diverse backgrounds, but metaphor being standalone and sometimes universally known will make the users experience much easier and understandable hence, increasing its usability.

User focused Businesses

Businesses where the customers are well–known to the business, customized metaphors will be of interest in order to make specific customers experiences of a web-based system easier, efficient and effective.

5.1.2.2 Evaluation Population

The evaluation population chosen was the staff present in the Department of Information Systems and a number of international participants from Pakistan. In developing a crossover trial survey it is important to identify the population that is randomly selected. Due to time constraints we undertook the survey with population being the Tutors at the department and a group of computer users from Pakistan and then randomly selecting a sample to carry out the survey. The reason for carrying out the experiment in Pakistan is to clearly identify two user types and compare the interaction of the users with the systems when metaphors are enabled.

5.1.2.3 Sampling Procedure

The total numbers of people surveyed are ten. The evaluation is not large however it is being conducted in order to validate the underlying hypothesis and evaluate the prototype. The process is that two user types with five subjects each, are given tasks to carry out on a website. The tasks take place under two scenarios.

In the first scenario the tasks are conducted with metaphors enabled on the website, while in the second scenario the metaphors are disabled. A wash out period will take place after the first scenario tasks are complete. The samples chosen make a clear distinction between user types, which will help in accepting or rejecting the hypothesis at hand.

5.1.2.4 Evaluation Method

"One of most important aspects when discussing technically enhanced resources for (language) learning is the question of how to evaluate and test the functionality and effectiveness of such materials" (Blin, Chenik & Thompson, 1998). The sample population evaluated the prototype. Their opinions were noted in the form of a questionnaire that took place after the evaluation. The data acquired from the experiment was analysed in order to help us with our final conclusions. We will ask them to evaluate our prototype under the following criteria such as:-

- Usability Aspects *(Aspects regarding the interaction of the user with the prototype)*
- Functional Aspects *(Aspects regarding the way the prototype functioned)*

Results based evaluation means that we seek to create an idea that links performance measures to the ultimate result that the prototype is trying to achieve. The data that will be gathered during the evaluation will let us accept of reject our hypothesis stated above.

The selected sample will carry out tasks:-

- Navigating through sites with metaphors enabled
- Navigating through sites with metaphors disabled

There will be a gap between carrying out the two tasks, in order to minimize the memory bias. A week's gap will be given between the two tasks.

The sample will be divided into two user groups one user group are users from Pakistan while the others are the users in New Zealand. Each group will be given the opposite task to start with:-

- The sample size of 10 will be grouped into two groups of five people.
- The first group will carry out the tasks with metaphors enabled, while the second group will carry out the task with the metaphors disabled first.
- The tasks will be reversed after a week's time.
- During the tasks, time to carry out a task will be measured.

	Group A (5)(NZ)	**Group B (5)(PAK)**
First Week (tasks)	Metaphors enabled	Metaphors disabled
Second Week (tasks)	Metaphors disabled	Metaphors enabled

After tasks are completed a questionnaire will be filled out. The questionnaire will be brief and simple in order to capture the general feel from the users. The questionnaire will be administered in order to take notes for other findings if any.

5.1.2.5 Questionnaire Design

Whether a questionnaire is self administered or completed by an interviewer, it must be well designed. Some the important aspects that have been taken into consideration when designing this questionnaire are:-

- KISS - keep it short and simple. If you present a 20-page questionnaire most potential respondents will give up in horror before even starting.

- Start with an introduction or welcome message. In the case of mail questionnaires, this message can be in a cover letter or on the questionnaire form itself.

- Allow a "Don't Know" or "Not Applicable" response to all questions, except to those in which you are certain that all respondents will have a clear answer.

- For the same reason, include "Other" or "None" whenever either of these are a logically possible answer. When the answer choices are a list of possible opinions, preferences or behaviours you should usually allow these answers.

The professional image of the questionnaire is highly influential in enhancing the importance of the survey to the respondent. The end result must be aesthetically pleasing while maintaining question and page structure to keep respondents from skipping individual items or whole sections of the questionnaire.

5.1.2.6 Evaluation-Questionnaire Implementation

After evaluating the prototype, a questionnaire is put forward to the users. The questionnaire includes questions as the following:

Task Related Questions

1. How did you find the task?
    1. Very Easy
    2. Easy
    3. Hard
    4. Very Hard
    5. Don't know ________

2. Did you complete the task?
    1. Yes
    2. No

3. How long did it take to complete the task?
    Time______

User Related Questions

4. How often do you use the Internet?
    1. Daily
    2. Weekly
    3. Monthly
    4. Never
    5. Don't know ________

5. What is your preferred style to view a website?

1. Graphical
2. Textual
3. Other __________

5.2 Expected Results from the questionnaire.

Question 1:
Percentage values of who found the tasks
1. Very Easy
2. Easy
3. Hard
4. Very Hard
5. Don't know

This will give us information on the understanding of the task given to user, and how they dealt with it with enabling metaphors.

Question 2:
Percentage of the users completed the task or not.

Question 3:
Average time taken to complete the tasks A and B.

Time taken by users, who found the tasks
1. Very Easy
2. Easy
3. Hard
4. Very Hard
5. Don't know

Graph displaying the interaction between time-taken and how the users found the tasks. This will give a idea of how these factors are related.

Question 4:

Percentage value showing how often the users access the Internet, as in
1. Daily
2. Weekly
3. Monthly
4. Never.
5. None

This will give us an idea of the user navigational skills over the Internet and we can compare its interaction with the findings from the above questions, using three way interaction graphs.

Question 5:

Percentage value showing the users preferred way to view websites, such as

1. Graphical
2. Textual
3. None.

As the metaphors are graphically displayed as images in the prototype, it is important for us know if the users preference of viewing website. This will clarify any preference bias.

The complete results from the evaluation can be found in the appendix.

Chapter 6

6. CONCLUSIONS

The tasks were carried out, on different dates. Three basic tasks were asked to be carried out. The tasks involved browsing through a website with information around the cities of New Zealand and Pakistan. Users were to do the following:-

1. Find the capital cities of both countries, i.e. Pakistan and New Zealand.

2. Find the educational institutions in the largest cities of New Zealand and Pakistan.

3. Find the educational facilities of Lahore and Christchurch.

These tasks were carried out on the websites with metaphors enabled and disabled, as specified in section 5.1.2.4. Once all tasks were completed the questionnaire had to be completed. The Analysis is Semi-qualitative as observations have also been made while collecting the data. The following table shows the summary of the results attained from the questionnaire.

Figure 11: Table of data from the evaluation exercise.

	NZ Metaphor	**User Type**	**NZ No Metaphor**		**PAK Metaphor**	**User Type**	**PAK No Metaphor**
User1	3.10	graphical	4.00	**User1**	1.10	graphical	1.50
User2	3.30	both	1.30	**User2**	1.30	graphical	2.20
User3	2.00	textual	1.40	**User3**	4.15	graphical	4.50
User4	3.10	both	1.50	**User4**	3.10	graphical	5.10
User5	3.50	graphical	4.10	**User5**	2.10	both	1.05
Avg Time	**3.00**		**2.46**		**2.35**		**2.87**

In the above table the Users are the people who completed the questionnaire and the tasks. The time taken by each user to complete each of the tasks are recorded along with the User-Type the person chose, which was asked of them in question 5 of the questionnaire.

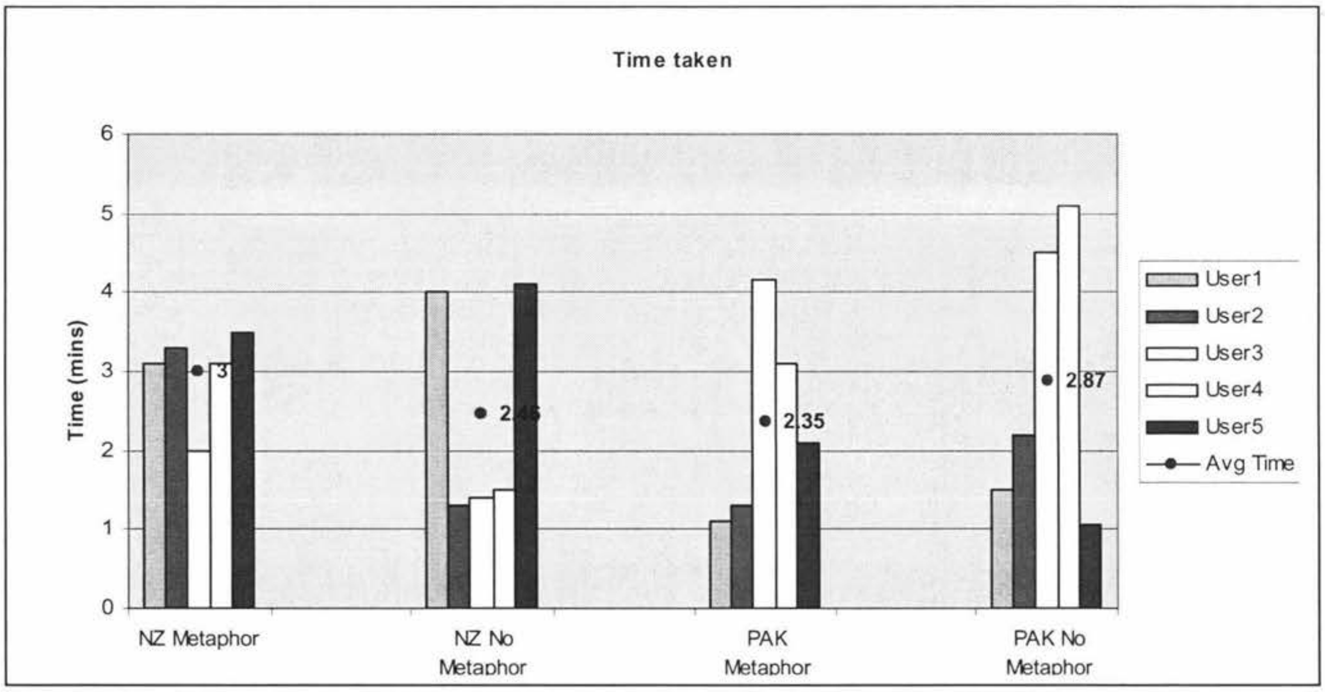
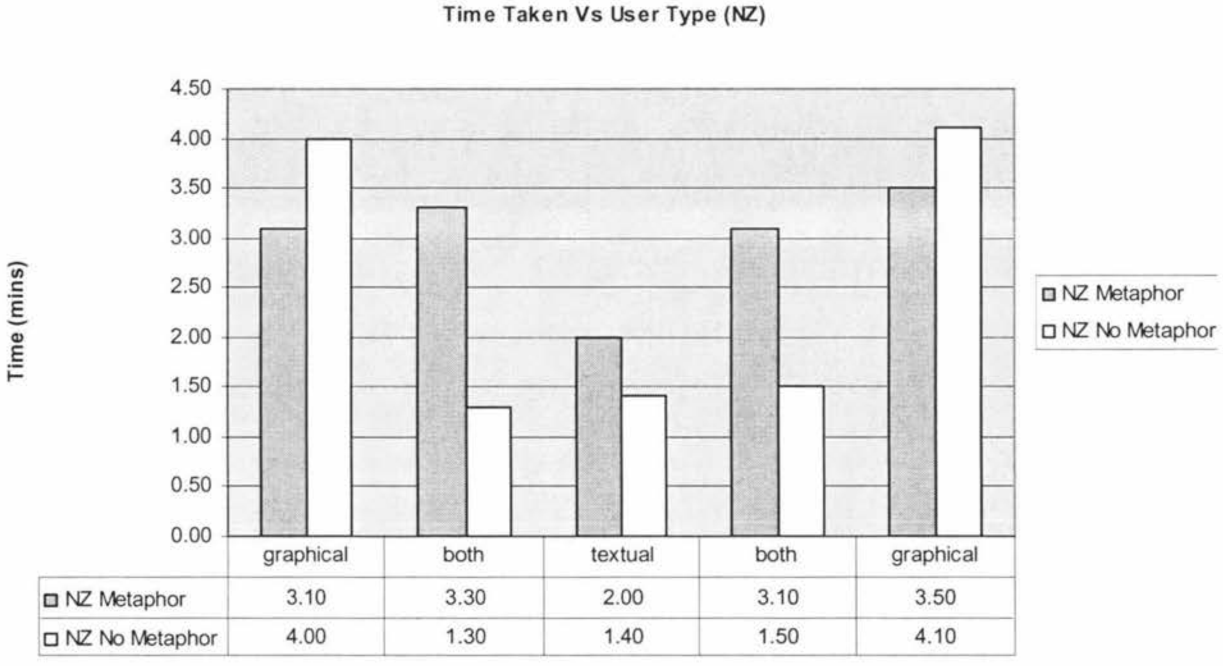


Figure 12: Graph showing the individual and average time spent doing the tasks.

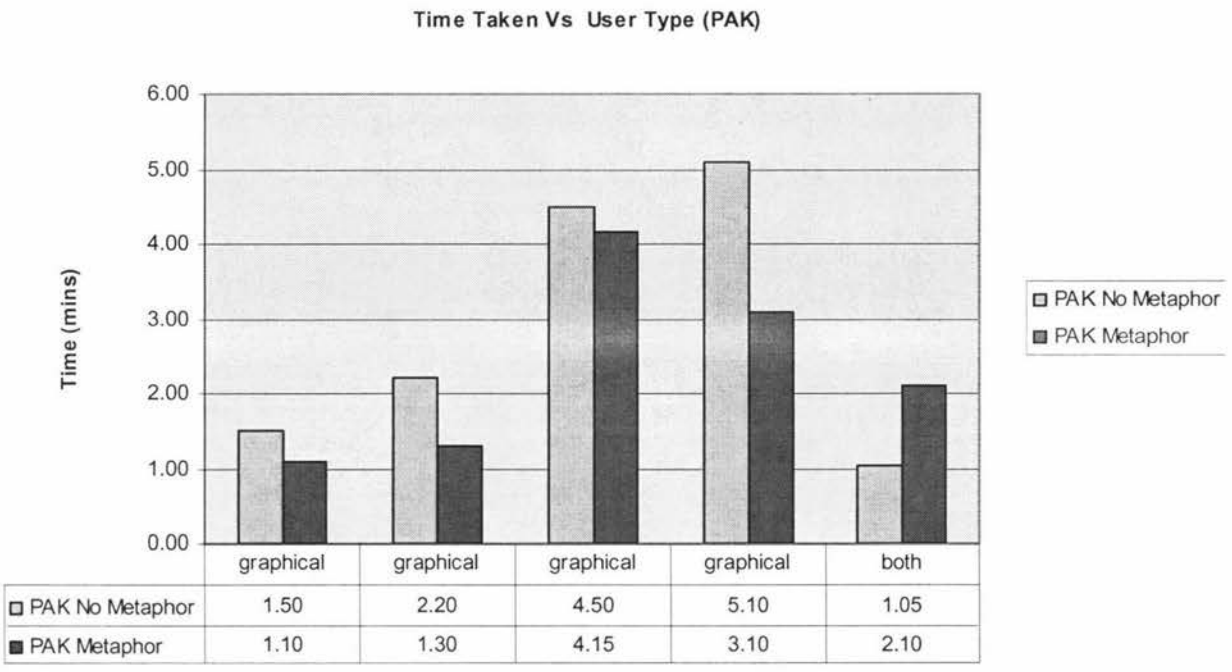
In the Graph above we can see the time taken by each user to complete the tasks. The average time taken to complete the tasks with Metaphors enabled and with Metaphors disabled are also recorded. The average time taken by the New Zealand User with Metaphors enabled was 3 minutes, while with Metaphors disabled their average time was 2 minutes and 46 seconds, which tell us that on the New Zealand Users took longer to complete the task when the metaphors were present. In the case of the Pakistan Users the average time taken to complete the tasks with metaphors enabled was 2 minutes 35 seconds, while doing the task with Metaphors disabled it was 3 minutes 27 seconds. This tells us the Pakistan Users took less time to complete the task with the metaphors enabled.



	graphical	both	textual	both	graphical
▨ NZ Metaphor	3.10	3.30	2.00	3.10	3.50
□ NZ No Metaphor	4.00	1.30	1.40	1.50	4.10

Figure 13: Graph showing the time taken to do the tasks for the different User types in the New Zealand lot.

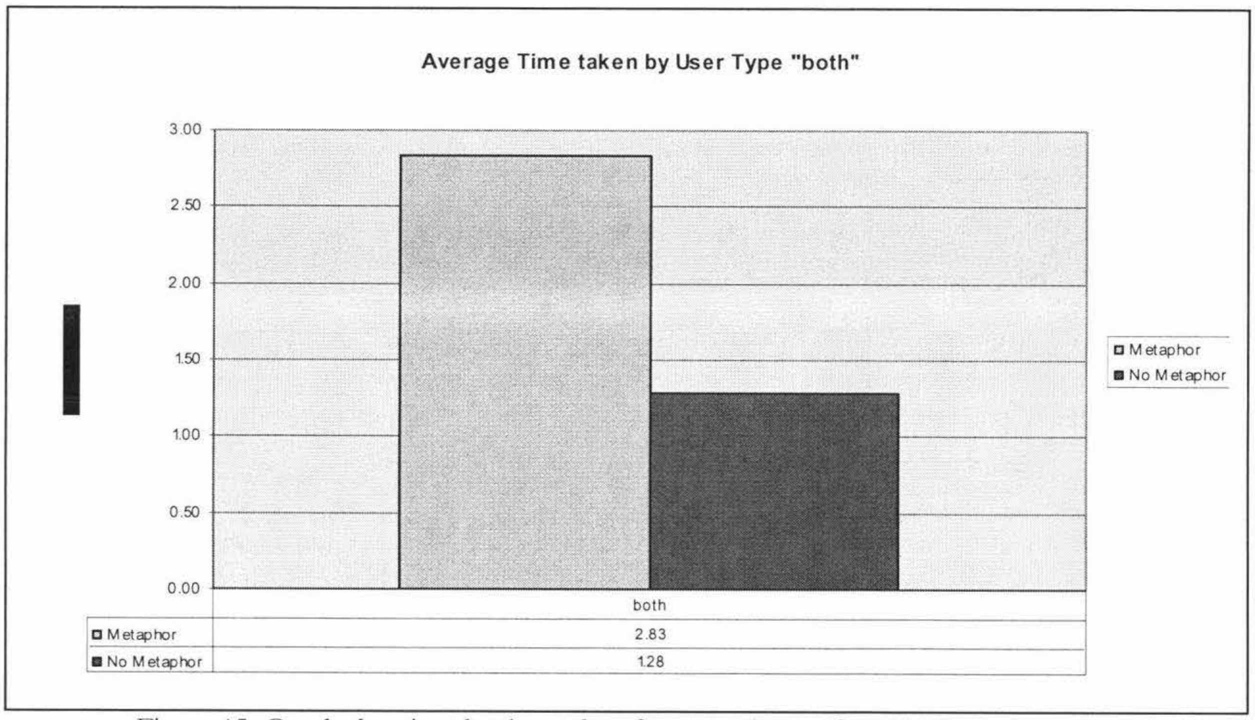
In the above graph each New Zealand User is classified by the User Types answered in question 5 of the questionnaire. From this graph we can see the time taken by the graphical, textual and mixed users (prefer both) to complete the tasks. We can see that the graphical users took relatively longer to complete the task with Metaphors disabled, while the textual user took longer to complete the task with Metaphors enabled. The 'Both' user took relatively longer on the Metaphor enabled tasks compared to the Metaphor disabled task. The time taken by the 'Both' users on the Metaphor disabled task is considerable less.



	graphical	graphical	graphical	graphical	both
▫ PAK No Metaphor	1.50	2.20	4.50	5.10	1.05
▪ PAK Metaphor	1.10	1.30	4.15	3.10	2.10

Figure 14: Graph showing the time taken to do the tasks for the different User types in the Pakistan lot.

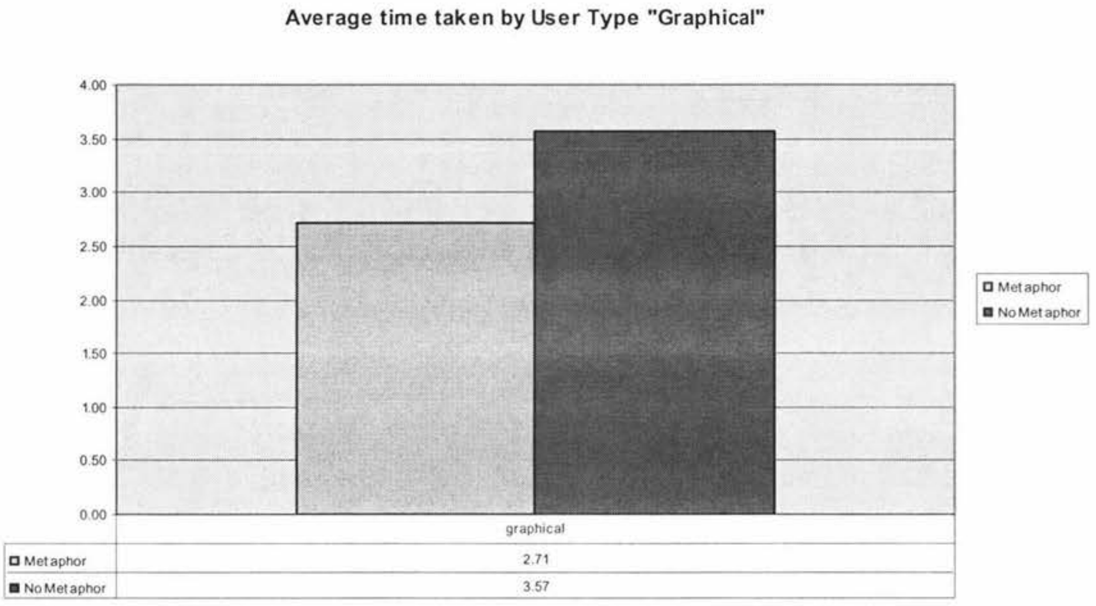
In the graph above each Pakistan User is classified by the User Types answered in question 5 of the questionnaire. Majority were graphical users and it can be seen the time taken to complete the Metaphor enabled tasks was less then that taken to complete the tasks with Metaphors disabled. However seeing the trend that Metaphors are suitable for graphical users, the 'both' user in both groups that is New Zealand and Pakistan have taken longer to complete the task with Metaphors enabled. The average time taken by the user type 'both' can be seen from the graph below. The average time taken to complete the task with Metaphors enabled is significantly greater (2.83) than the time taken for Metaphors disabled (1.28). It can be seen in the graph below.



	both
☐ Metaphor	2.83
■ No Metaphor	1.28

Figure 15: Graph showing the time taken for user who preferred both visual and textual websites.

From the graph below we can see a positive trend of the average time taken by graphical users who completed the tasks with Metaphors enabled was less then that of when Metaphors were disabled. This shows that the graphical user prefers to interact with Metaphors compared to the 'Both' and 'Textual' user types. Interface can be manipulated according to the user types and in this instance we can see that for graphical user type it would be prefer to enable metaphors as the time taken to achieve the task at hand were affected.



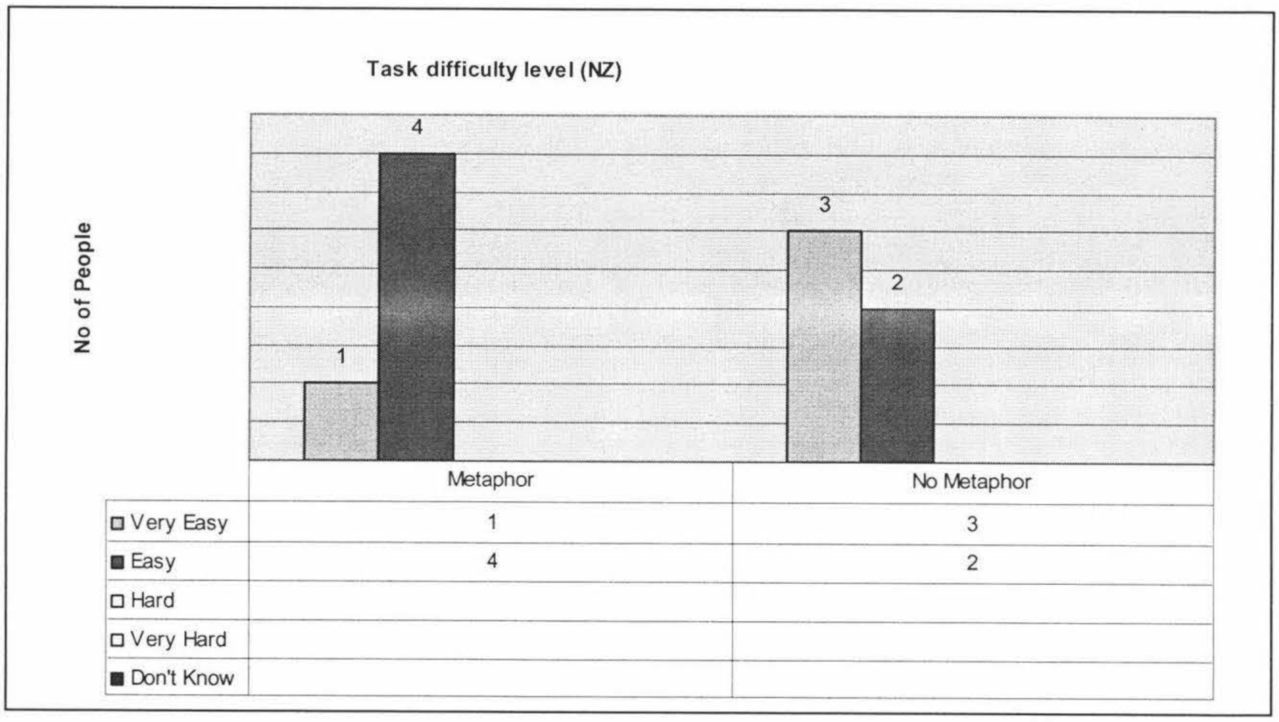
	graphical
□ Metaphor	2.71
■ No Metaphor	3.57

Figure 16: Graph showing the time taken for user who preferred graphical style websites.

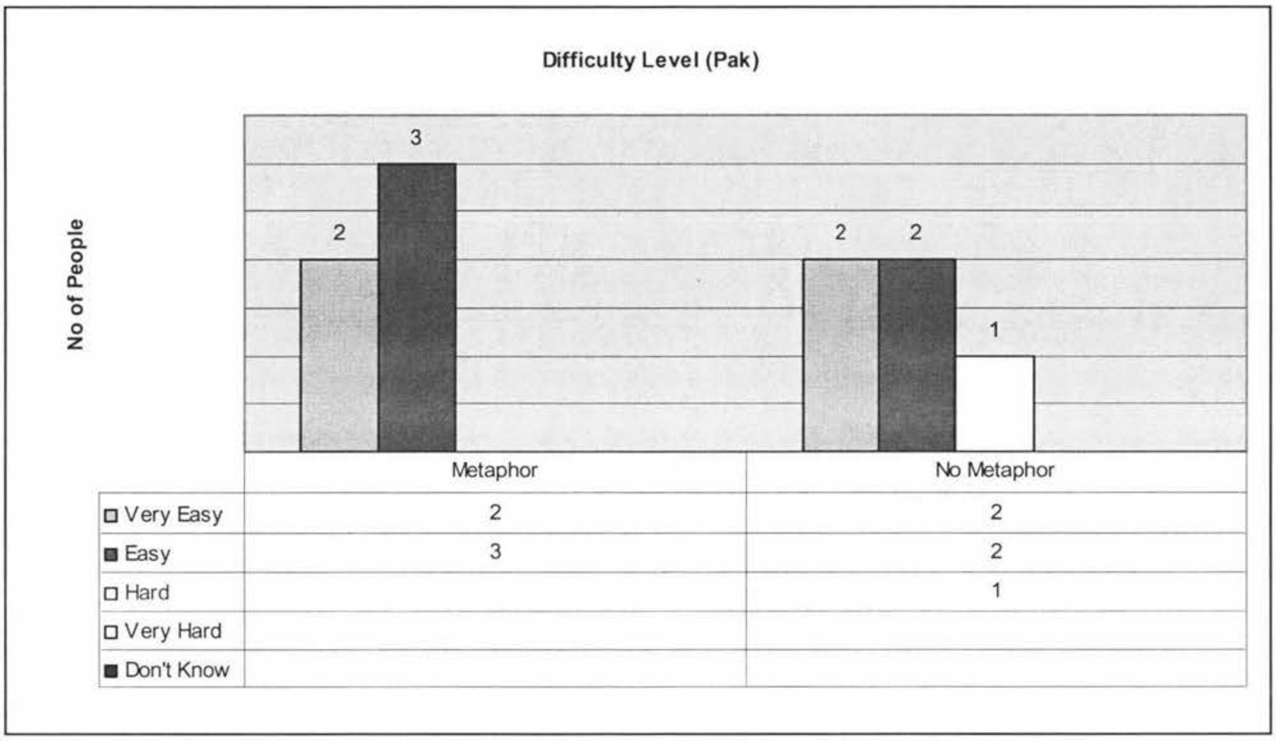
Such empirical data can be taken into consideration when developing metaphor enabled websites that should have the different options for different user types. The idea here is to get the optimum understanding of the subject matter in less time. Reducing the time spent on a webpage and increasing the knowledge intake.

By the above data it can be seen that the for graphical user type the time to complete a task was less when Metaphors were enabled, hence having metaphors on the website suited the learning/working style of those users and was effective in the sense that the learning/working time was reduced. However further research can be done into creating better and more effectively metaphors for different user types, as the graphical user type can be further broken down e.g. graphical user with science background, maths background, hence providing metaphors that are visual and may suite students/users from science or maths backgrounds.

Figure 17: Graph showing the difficulty levels for the New Zealand lot with Metaphors enabled and disabled.



	Metaphor	No Metaphor
▫ Very Easy	1	3
■ Easy	4	2
▫ Hard		
▫ Very Hard		
■ Don't Know		



	Metaphor	No Metaphor
☐ Very Easy	2	2
■ Easy	3	2
☐ Hard		1
☐ Very Hard		
■ Don't Know		

Figure 18: Graph showing the difficulty levels for the Pakistan lot with Metaphors enabled and disabled.
User Acceptance:

However, the difficulty levels were not that of concern as the tasks were fairly simple and most users found it very easy or easy to complete the task. Every user was able to complete the task. From the graphs it can be seen that only one user found it hard, however that user was the only one that did not use the internet/computer daily.

Therefore, in light of this knowledge; less internet usage or even not knowing how to use a computer influenced the time taken to complete the tasks; however the understanding of metaphors was not affected in this case. The tasks seemed to be harder once the metaphors were disabled. The Metaphors were accepted well by both the users but looking at the time taken, the Pakistani users seem to have a completed the task sooner.

Based on knowledge gained from the study above, it will be possible say that metaphor-enabled websites will reduce the interaction time of the user depending on the user type. Metaphors can help users browse through websites efficiently and more effectively and even help understand concepts that may be out of the domain one comes from.

Websites for university courses can be setup to be metaphor-enabled so that different students, from different backgrounds can efficiently and more effectively pick up concepts. A practical example of this concept can be implemented by using an existing university course website such as, an Information Systems course website at Massey University (Tretiakov & Kaschek, 2005).

During the course of this thesis conceptual data model has been introduced allowing us to enable metaphors on a website, according to a specific domain or User Type. The model defines a database for a metaphor enabler forming part of the architecture that we propose for the Metaphor-Enabled Interface Architecture. The architecture suggests decorating a web-based system with metaphors by adding a separate architectural layer, so that the base service does not need to be changed.

A limited prototype system was developed, and conducted a user evaluation. The results of the evaluation indicate a good user acceptance when the metaphors where enabled when compared to not having metaphors enabled.

Banerjee, A. & Ghosh, J. (2001). *Clickstream clustering using weighted longest common subsequences.* In Proceedings of the Web Mining Workshop at the 1st SIAM Conference on Data Mining, Chicago, April 2001.

Raskin, J. (1997). Looking for a Humane Interface: *Will Computers Ever Become Easy to Use?* Communications of the ACM. February 1997/vol. 40, no. 2.

Nardi, B.A & Zarmer, C.L. (1993). Beyond models and metaphors: *Visual Formalisms in user interface design.* Journal of Visual Languages and Computing, 4, 1993, pp. 5--33.

Johnson, G.J. (1994). *Of Metaphor and the Difficulty of Computer Discourse.* December 1994/vol. 37, no. 12. Communications of the ACM.

Celentano, A. (1999). Virtual Worlds as Metaphors for Websites Exploration: *Are they Effective?* IEEE Symposium on Visual Languages, Sept 13 16, 1999, Tokyo, Japan.

Meyer, R., Chalon, C., & David, B. (2003). *Adaptive Hypermedia based on Metaphors as an Instructional Strategy.* The 3rd IEEE International Conference on Advanced Learning Technologies (ICALT'03). July 09 - 11, 2003 Athens, Greece .

Chomsky, N. & Herman, E.S. (1988). *Manufacturing Consent* (2nd ed. 2002). New York: Pantheon Books.

Chomsky, N. (1989). *Necessary Illusions: Thought Control in Democratic Societies.* Boston, MA: South End Press.

Thalheim, B. & Dusterhoft, A. (2000). *The use of metaphorical structures for internet sites.* Data and knowledge Engineering, 35. (2000). p 161 –180.

Carroll, J. & Thomas, J.C. (1982). *Metaphor and the cognitive representation of computer systems.* IEEE Transactions on Man, Systems, and Cybernetics., SMC-12 (2), pp. 107-116.

Trausan-Matu, S., Maraschi, S., & Cerri, S. (2002). *Ontology-Centered Personalized Presentation of Knowledge Extracted From the Web,* in S.Cerri, G.Gouarderes (eds.), Intelligent Tutoring Systems 2002, Springer, Lecture Notes in Computer Science number 2363, pp 259-269.

Trausan-Matu, S. (2000). *Metaphor Processing for Learning Terminology on the Web*, in S.A.Cerri (ed.), Artificial Intelligence, Methodology, Systems, Applications 2000, Springer-Verlag, ISBN 3-540-41044-9, 2000, pp.232-241.

Kobsa, A., Koenemann, J., & Pohl, W. (2001). *Personalised hypermedia presentation techniques for improving online customer relationships*, The Knowledge Engineering Review, Vol. 16:2, 111–155. Cambridge University Press.

Kobsa, A., & Wahlster, W. (1989). *User Models in Dialog Systems.* In A. Kobsa and W. Wahlster, eds., User Models in Dialog Systems, p 4-34. Springer, Berlin, Heidelberg.

Noble, J., Biddle, R., & Tempero, E. (2002). *Metaphor and Metonymy in Object-Oriented Design Patterns.* Twenty-Fifth Australasian Computer Science Conference (ACSC2002). Melbourne.

Blumenthal, B. (1990). *Strategies for Automatically Incorporating Metaphoric Attributes in Interface Designs.* Symposium on User Interface Software and Technology.

Proceedings of the 3rd annual ACM SIGGRAPH symposium on User interface software and technology. Snowbird, Utah, United States. p 66–75.

Kuhn, W., & Frank, A.U. (1991). *A Formalization of Metaphors and ImageSchemas in User Interfaces.* In Cognitive and Linguistic Aspects of Geographic Space. Edited by D. M. Mark and A. U. Frank. NATO ASI Series. Dordrecht, The Netherlands: Kluwer Academic Press.

Barbosa, S.D.J., & de Souza, C.S. (2000). *Extending software through metaphors and metonymies.* Intelligent User Interfaces, p 13-20.

Stathis, K., & Sergot, M. J. (1996). *Games as a metaphor for interactive systems.* In M. A. Sasse, R. J. Cunningham, and R. L. Winder, editors, People and Computers XI (Proceedings of HCI'96), BCS Conference Series, London UK, Springer-Verlag, p 19-33.

Farrell, S., Maglio, P.P., & Campbell, C.S. (2001) *How to Teach a Fish to Swim.* IBM Almaden Research Centre, 650 Harry Rd. San Jose, California 95120.

Cunningham, W. (2003). *Enterprise Solution Patterns Using Microsoft .NET.* Version 1.0. Patterns & practices. © 2003 Microsoft Corporation.

Lakoff, G., & Johnson, M. (1980). *Metaphors We Live By.* Chicago: University of Chicago Press.

Lakoff, G. (1995). *Moral Politics.* Chicago: University of Chicago Press. (2nd ed. 2001)

Reddy, M.J. (1993). The conduit metaphor: *A case of frame conflict in our language about language.* In Andrew Ortony (Ed.). Metaphor and thought (2nd ed.) (pp. 164-201). Cambridge. UK: Cambridge University Press.

Ratkowsky, D.A., Evans, M.A.., & Alldredge, J.R. (1993). Cross-over experiments: *design, analysis and application.* Senn SJ. Cross-over trials in clinical research. Chichester: John Wiley. Marcel Dekker, Inc.

Freeman, P.R. (1989). *The performance of the two-stage analysis of two-treatment, two-period crossover trials.* Stats Med 1989; 8: p 1421-1432.

Jacko, J., & Stephanidis, C. (2003). Human Computer Interaction: Theory and Practice (Part I). Proceedings of the 10th International Conference on Human-Computer Interaction, Crete, Greece, 22-27 June. Mahwah, New Jersey: Lawrence Erlbaum Associates (ISBN: 0-8058-4930-0).

Wells, J.D., & Fuerst, W.L. (2000). Domain-Oriented Interface Metaphors: *Designing Web Interfaces for Effective Customer Interaction.* HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 6, 0-7695-0493-0, IEEE Computer Society, Washington, DC, USA.

Thalheim, B. (1993). *Foundations of Entity-Relationship Modelling.* Annuals of Mathematics and Artificial Intelligence, 7, 1993, pp. 197-256.

Tretiakov, A. & Kaschek, R. (2005). *Towards Non-Invasive Adaptation of Metaphors in Content.* Chapter 6. Cognitively Informed Systems: Utilizing Practical Approaches to Enrich Information Presentation and Transfer. IGI Idea Group.

Madsen, K.H. (1994). *A Guide to Metaphorical Design.* Communications of the ACM, (37:12), p 57-62.

Erickson, T.D. (1990). *Working with Interface Metaphors.* In The Art of Human-Computer Interface Design, B. Laurel (ed.), Addison-Wesley, Reading, Massachusetts, p 65-73.

Gentner, D.R. (1988). Metaphors as structure mapping: *The relational shift.* Child Development, (59), p 47-59.

Gillan, D.J., Fogas, B.S., Aberasturi, S., & Richards, S. (1995). *Cognitive ability and computing experience influence interpretation of computer metaphors.* In Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting, p 243-247.

Coffman, K.G., & Odlyzko, A.M. (2002). *Growth of the Internet:* In Optical Fiber Telecommunications IV B: Systems and Impairments, I. P. Kaminow and T. Li, eds. Academic Press, p 17-56.

Odlyzko, A.M. (2003). *Internet traffic growth:* Sources and implications: Optical Transmission Systems and Equipment for WDM Networking II, B. B. Dingel, W. Weiershausen, A. K. Dutta, and K.-I. Sato, eds., Proc. SPIE, vol. 5247, p 1-15.

Silva, A., & Delgado, J. (1998). The Agent Pattern: *A Design Pattern for Dynamic and Distributed Applications.* Proceedings of the EuroPLoP'98, Third European Conference on Pattern Languages of Programming and Computing, Irsee, Germany.

Guttman, E., Perkins, C., Veizades, J., & Day, M. (1999). *Service Location Protocol.* Ver. 2. IETF RFC 2608.

Sun Microsystems. (2000). *JDK 1.1 for Solaris Developer's Guide.* 806-3461-10 Sun Microsystems, Inc. 901 San Antonio Road Palo Alto, CA 94303-4900 U.S.A.

Sun Microsystems. (2002). Retrieved May 13, 2006 from http://java.sun.com/blueprints/patterns/InterceptingFilter.html

Webcleaner Soureforge.(2003). Retrieved January 3, 2003 from http://freshmeat.net/projects/webcleaner and http://webcleaner.sourceforge.net

Oracle (2004). Retrieved May 19, 2006 from http://www.oracle.com/technology/tech/sql_plus/index.html

Brewster, S. (2001) *User interface architectures*. Lecture notes Retrieved May 19, 2006 from http://www.dcs.gla.ac.uk/~stephen/lectures/IS3/

Cavaness C. (2004). *Programming Jakarta Struts* (Second Edition). O'Reily Media Inc.

Schnitzer, J., Hernandez, S., & Moore, J. (2001). *Maverick*. Retrieved May 19, 2006 from http://mav.sourceforge.net/

Opensymphony (2003). *Webwork* Retrieved May 19, 2006 from http://opensymphony.com/webwork/ .

Relevant sections of the documented Use cases

USE CASE	Add Metaphor.	
Goal in Context	New Metaphors are added to the systems once administrator has completed creating them.	
Scope & Level	Administrator, Primary Task	
Preconditions	System has Metaphors.	
Success End Condition	Metaphor has been added.	
Failed End Condition	Metaphor was not added.	
Primary, Secondary Actors	Administrator, System.	
Trigger	New Metaphor comes in.	
DESCRIPTION	**Step**	**Action**
	1	Administrator adds a Metaphor.
	2	Administrator tests the Metaphor.
	3	Administrator makes Metaphor useable to the customers.
	4	Administrator saves changes.
	5	Administrator ends adding Metaphor.
EXTENSIONS	**Step**	**Branching Action**
	3a	Administrator may not make Metaphor useable for customers, for research/political reasons.

		2a1. Administrator saves and ends adding.
USE CASE	Request Information.	
Goal in Context	The Customer asks for certain information, without enabling Metaphors.	
Scope & Level	Customer, Primary Task	
Preconditions	No Metaphors have been enabled, and no profile entered.	
Success End Condition	Information requested by the customer is displayed without metaphors enabled.	
Failed End Condition	Information requested not available.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, bank, System.	
Trigger	Customers information request comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer requests information.
	2	System captures request.
	3	System checks for customer profile.
	4	System displays information requested.
	5	Customers' leaves the website.
EXTENSIONS	**Step**	**Branching Action**
	2a	System does not find a customer profile. 2a1. Displays no metaphors.
	4a	Information requested not available. 4b1. System displays a message telling the customer that the information is not in the system at the moment.

USE CASE	Check Definition.	
Goal in Context	Once the Customer requests Information after entering their profile or login, the system checks for the compatible Metaphors, customer types for the corresponding Core_IS_Function activated.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered.	
Success End Condition	Customer satisfies a definition.	
Failed End Condition	Customer does not satisfy a definition.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System, Core_IS_function.	
Trigger	Customers Information request or Profile comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer enters profile or logs in.
	2	Customer requests information.
	3	System maps for customer profile with Metaphor and Customer Type.
	4	System displays information requested with Metaphors.
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer enters incorrect profile or Log in details. 1a1. System requests those details to be entered again.
	2a	Customer profile does not map with any definition. 2a1. System displays no metaphors.

USE CASE	Transfer Funds	
Goal in Context	A Customer is able to transfer funds from one account to another, by providing the correct information about his account and the account that the customer would like the funds to be transferred to.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Fund will be Transferred from the customers account.	
Failed End Condition	Customers' funds were not transferred.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers Request to transfer funds comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer enters amount of funds to be transferred.
	2	Customer enters receivers account information.
	3	Customer submits the Information.
	4	System processes the Transaction.
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer enters incorrect amount. 1a1. System requests amount to be entered again.
	2a	Customer enters incorrect receiver account information. 2a1. System requests information to be entered again.

	4a	System could not process transaction. 4a1. System gives the customer an error and asks to try later.

USE CASE	Check Account Balance	
Goal in Context	A customer is able to view his account balance and information.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customer views account balance.	
Failed End Condition	Customer does not view account balance.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request to view account balance comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer selects account for balance to be viewed.
	2	System displays accounts' balance requested.
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer does not select an account to be viewed. 1a1. No balance will be shown.
	2a	System could not display balance. 2a1. System displays error and prompts to try later.

USE CASE	Check Daily Transactions	
Goal in Context	A customer is able to view the transactions done on a particular account in specified time	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customer views daily transactions.	
Failed End Condition	Customer does not view daily transaction.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request to Check Daily Transactions comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer selects a bank account for the transactions to be viewed.
	2	Customer specifies a time interval for transaction to be viewed.
	3	System displays the transactions requested for the particular time interval.
	4	Customer logs out
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer does not select an account for transactions to be viewed. 1a1. No Transactions will be displayed.
	2a	Customer selects incorrect time interval. 2a1. No Transactions will be displayed.

USE CASE	Request Card	
Goal in Context	A customer is able to request a new card in the event of losing it or expiry.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customers request accept and card sent to the customer.	
Failed End Condition	Customers request rejected to get a new card.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request for a new card comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer selects a card type to be requested.
	2	Customer selects a reason for requesting a card.
	3	System accepts the customers' request.
	4	Card is sent to the customer.
	5	Customer logs out
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer does not select a card type. 1a1. Request rejected.
	2a	Customer does not select a reason for request. 2a1. Request rejected.

USE CASE	Request Loan	
Goal in Context	A customer is able to request a loan. The loan may be of any type, as the customer will have to fill in an online form, specifying their requirements.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customers request accepted and sent to the responsible authority for feedback.	
Failed End Condition	Customers request rejected due to not reaching the minimum qualification criteria.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request for a loan comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer fills in a loan request form.
	2	Customer submits the form.
	3	System accepts the customers' loan request.
	4	Systems send the information to the responsible authority.
	5	Customer logs out
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer fills in the loan request form incorrectly. 1a1. Request rejected.

USE CASE	Enable Automatic payments	
Goal in Context	A customer is able to set up automatic payments to other accounts in order to pay bills on time.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customer enables an automatic payment.	
Failed End Condition	Automatic payment not enabled.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request to enable automatic payment comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer selects the account from which the payment will be made.
	2	Customer selects the account to which the payment will be made.
	3	Customer selects the start date for the payments.
	4	Customer selects the end date for the payments.
	5	Customer selects the interval of payments. (Weekly, fortnightly, monthly etc)
	6	Customer submits information.
	7	System enables Automatic payments.
EXTENSIONS	**Step**	**Branching Action**
	6a	Customer submits incorrect information.

		1a1. Automatic Payments not enabled.

USE CASE	View Profile	
Goal in Context	A customer is able to view their details stored by the system.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customer views profile.	
Failed End Condition	Customer not able to view profile.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request to view profile comes in.	
DESCRIPTION	**Step**	**Action**
	1	System displays the customers profile..
	2	Customer logs out

USE CASE	Edit Profile	
Goal in Context	A customer is able to edit their profile if parameters change in due course.	
Scope & Level	System, Sub Function.	
Preconditions	A customers profile is entered and the customer is Authenticated.	
Success End Condition	Customer profile is edited and updated.	
Failed End Condition	Customers profile not updated.	
Primary, Secondary Actors	Customer, any agent (or computer) acting for the customer, System.	
Trigger	Customers request to edit customers profile comes in.	
DESCRIPTION	**Step**	**Action**
	1	Customer edits profile parameters.
	2	Customer saves editing.
	3	System updates the existing profile with the edited profile.
	4	Customer views the updated profile.
	5	Customers logs out.
EXTENSIONS	**Step**	**Branching Action**
	1a	Customer edits profile parameter incorrectly. 1a1. Editing will not be saved.
USE CASE #	Provide Metaphor	

Goal in Context	Provides metaphors that are allowed to be used, as the database may contains metaphors that may not be used.	
Scope & Level	System, secondary	
Preconditions	User request is actioned.	
Success End Condition	Metaphors are provided in order to be exposed.	
Failed End Condition	Metaphors are not available, hence default metaphors or no metaphors are exposed.	
Primary, Secondary Actors	System, Metaphor Administrator	
Trigger	Business function is requested.	
DESCRIPTION	Step	Action
	1	Customer request is actioned.
	2	Metaphors are provided by the system.
	3	Business function is executed with metaphors.
EXTENSIONS	Step	
	2a	Metaphors are not provided. 2a1. No Metaphors were available for the customer or business function.

USE CASE #	Add Customer Type
Goal in Context	New Customer Types are added to the Metaphor System once administrator has created them.
Scope & Level	Metaphor Administrator, Primary Task

Preconditions	Metaphor database is running.	
Success End Condition	Customer type was added.	
Failed End Condition	Customer type was not added.	
Primary, Secondary Actors	Metaphor Administrator, System, Metaphor database	
Trigger	New Customer type come in.	
DESCRIPTION	Step	Action
	1	Metaphor Administrator adds a customer type.
	2	Metaphor Administrator tests the customer type.
	3	Metaphor Administrator assigns the customer type.
	4	Metaphor Administrator saves changes.
EXTENSIONS	Step	
	3a	Metaphor Administrator does not assign the customer type. 3a1. Metaphor Administrator saves the customer type.

USE CASE #	Register business function
Goal in Context	New Business functions are registered into the Metaphor database in order to map metaphors to them.
Scope & Level	Metaphor Administrator, Primary Task, Domain Administrator

Preconditions	Metaphor database is running	
Success End Condition	Business function registered	
Failed End Condition	Business function did not register	
Primary, Secondary Actors	Metaphor Administrator, Metaphor Database.	
Trigger	New business Function comes in.	
DESCRIPTION	Step	
	1	Metaphor Administrator registers business function into the Metaphor database.
	2	Metaphor Administrator test the business function
	3	Metaphor Administrator assigns metaphors to business functions.
	4	Metaphor Administrator saves the changes.
EXTENSIONS	Step	
	3a	Metaphor Administrator does not assign metaphors to business functions. 3a1. Business functions are saves in database.

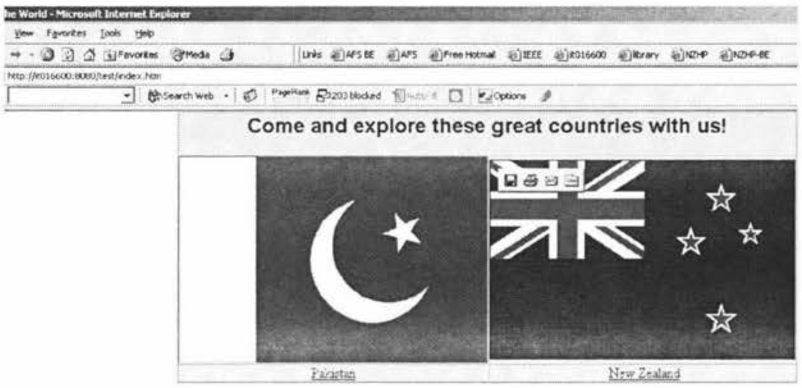
SQL Statements for creating the database.

```
create Table Metaphor (
Metaphor_ID Integer,
Metaphor_Name varchar(30),
Metaphor_Metadata varchar(100),
Metaphor_source varchar(100),
PRIMARY KEY (Metaphor_ID)
);

CREATE TABLE Customer(
Customer_ID    CHAR(11)   NOT NULL,
Email          VARCHAR(64),
First_name     VARCHAR(32) NOT NULL,
Last_name      VARCHAR(64) ,
Address        VARCHAR(64) ,
Country        VARCHAR(64) NOT NULL,
Password       CHAR(11) NOT NULL,
CustomerType_ID                                      CHAR(11)   NOT NULL,
CONSTRAINT Customer_PK                          PRIMARY KEY (Customer_ID)
);

CREATE TABLE Business_Functions(
Business_Functions_ID              CHAR(11)   NOT NULL,
Business_Functions_Name              VARCHAR(64) NOT NULL,
Business_Functions_Description         VARCHAR(64) ,
Customer_ID       CHAR(11),
CONSTRAINT Business_Functions_PK PRIMARY KEY
(Business_Functions_ID),
CONSTRAINT Business_Functions_FK FOREIGN KEY (Customer_ID)
REFERENCES Customer(Customer_ID)
);


CREATE TABLE Customer_type(
CustomerType_ID                                      CHAR(11)   NOT NULL,
name                                               VARCHAR(64) NOT NULL,
Description                                        VARCHAR(64),
Customer_ID                                          CHAR(11)   NOT NULL,
CONSTRAINT CustomerType_PK PRIMARY KEY (CustomerType_ID),
CONSTRAINT CustomerType_FK FOREIGN KEY(Customer_ID)
REFERENCES Customer(Customer_ID)
);
```

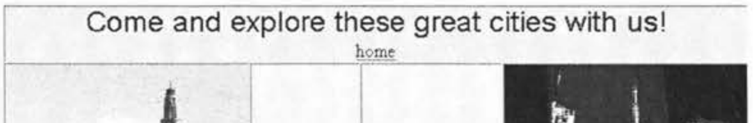
```
CREATE TABLE Metaphor_explanation(
Metaphor_ID              NUMBER(30)                              NOT NULL,
Metaphor_explanation       VARCHAR2(100)   NOT NULL,
BUSINESS_FUNCTIONS_ID CHAR(11)                                NOT NULL,
FOREIGN KEY (Metaphor_ID) REFERENCES metaphor(Metaphor_ID),
FOREIGN KEY (BUSINESS_FUNCTIONS_ID)
REFERENCES BUSINESS_FUNCTIONS(BUSINESS_FUNCTIONS_ID),
CONSTRAINT Metaphor_explanation_PK
PRIMARY KEY (Metaphor_ID, BUSINESS_FUNCTIONS_ID)
);

CREATE TABLE Metaphor_allocation(
Metaphor_ID
Business_Functions_ID      CHAR(11)    NOT NULL,
customerType_ID            CHAR(11)    NOT NULL,
FOREIGN KEY (Metaphor_ID) REFERENCES metaphor(Metaphor_ID),
FOREIGN KEY (Business_Functions_ID) REFERENCES
Business_Functions(Business_Functions_ID),
 FOREIGN KEY (CustomerType_ID) REFERENCES Customer_type(CustomerType_ID),
CONSTRAINT Metaphor_allocation_PK
PRIMARY KEY (Metaphor_ID, Business_Functions_ID, CustomerType_ID)
);
```

The website used for the Evaluation. *(This version shows both graphical and textual in order to reduce size of this document)*

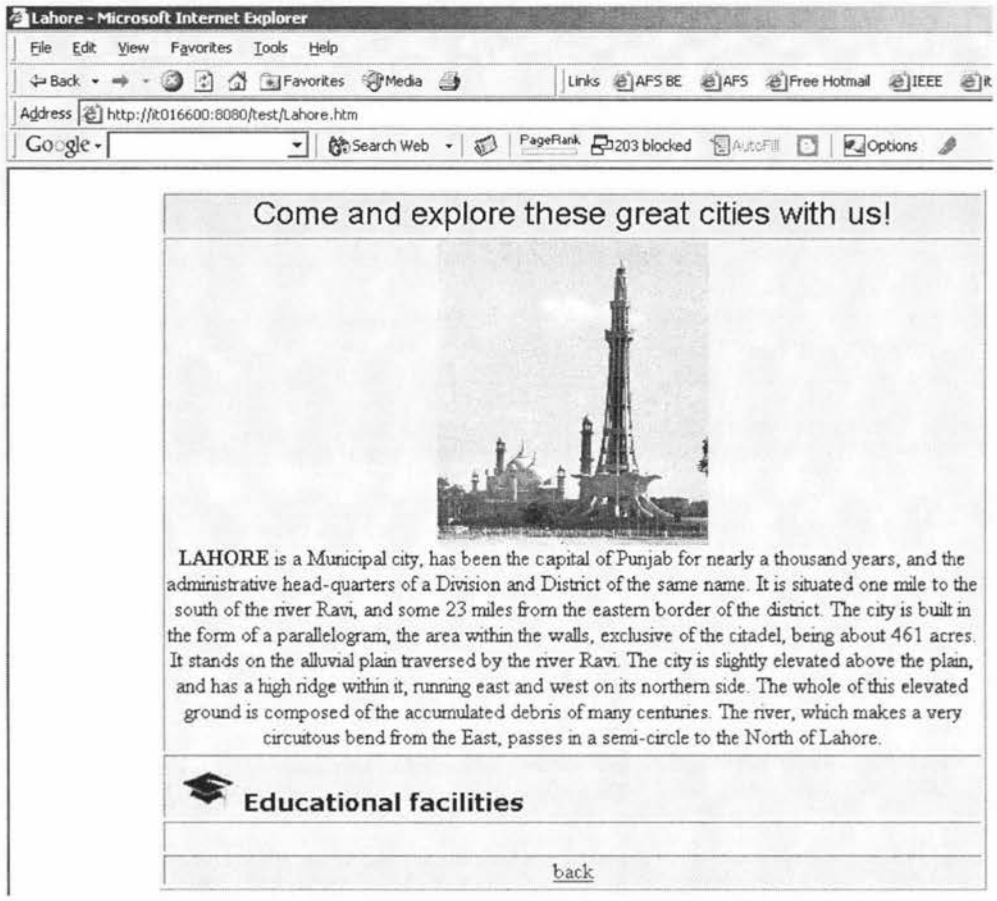
Index Page



Second Page



Lahore's page – with the educational facilities hidden, but metaphor present.


Lahore - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Favorites Media Links AFS BE AFS Free Hotmail IEEE
Address http://it016600:8080/test/Lahore.htm
Google Search Web PageRank 203 blocked AutoFill Options

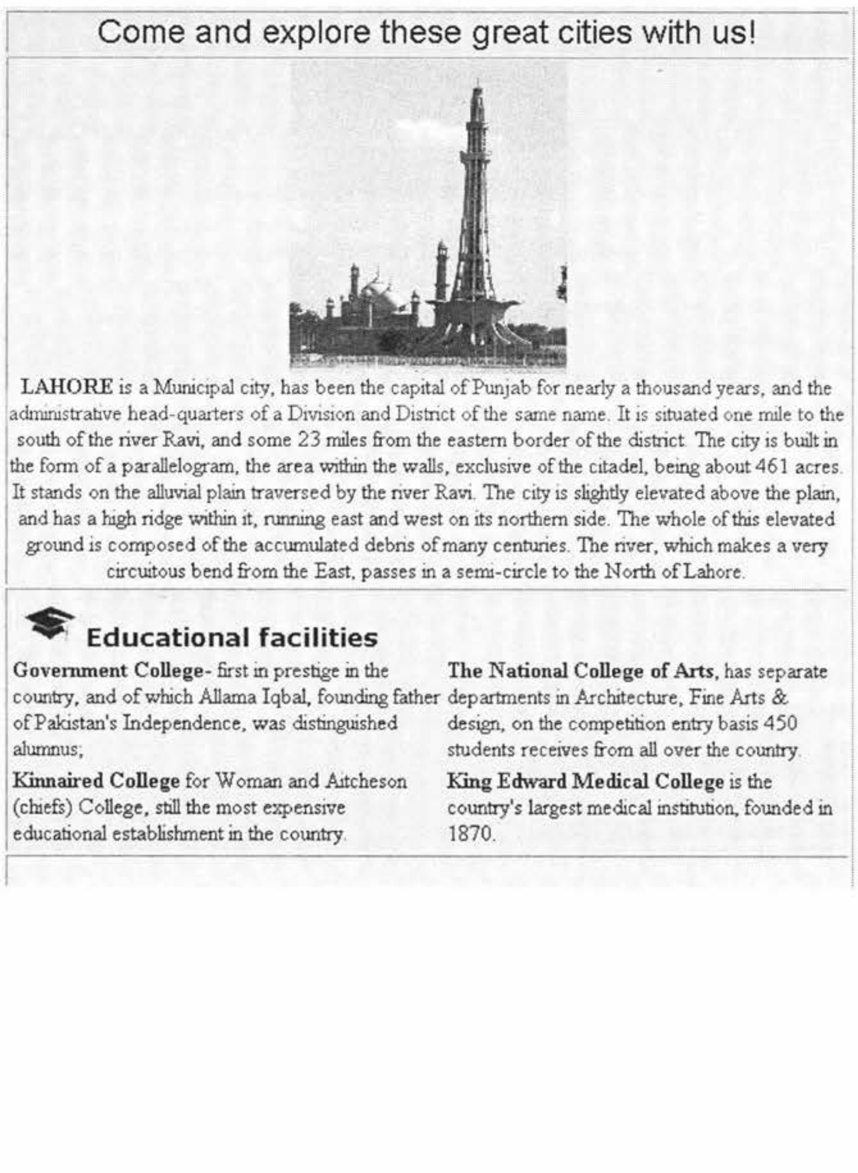
Come and explore these great cities with us!

LAHORE is a Municipal city, has been the capital of Punjab for nearly a thousand years, and the administrative head-quarters of a Division and District of the same name. It is situated one mile to the south of the river Ravi, and some 23 miles from the eastern border of the district. The city is built in the form of a parallelogram, the area within the walls, exclusive of the citadel, being about 461 acres. It stands on the alluvial plain traversed by the river Ravi. The city is slightly elevated above the plain, and has a high ridge within it, running east and west on its northern side. The whole of this elevated ground is composed of the accumulated debris of many centuries. The river, which makes a very circuitous bend from the East, passes in a semi-circle to the North of Lahore.

Educational facilities

back

Lahore's page – with the educational facilities shown, clicking of metaphor shows the information.



Come and explore these great cities with us!

LAHORE is a Municipal city, has been the capital of Punjab for nearly a thousand years, and the administrative head-quarters of a Division and District of the same name. It is situated one mile to the south of the river Ravi, and some 23 miles from the eastern border of the district. The city is built in the form of a parallelogram, the area within the walls, exclusive of the citadel, being about 461 acres. It stands on the alluvial plain traversed by the river Ravi. The city is slightly elevated above the plain, and has a high ridge within it, running east and west on its northern side. The whole of this elevated ground is composed of the accumulated debris of many centuries. The river, which makes a very circuitous bend from the East, passes in a semi-circle to the North of Lahore.

Educational facilities

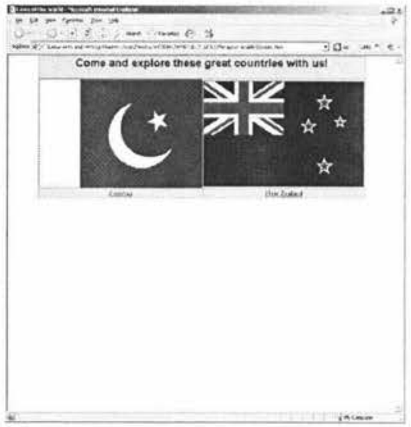
Government College- first in prestige in the country, and of which Allama Iqbal, founding father of Pakistan's Independence, was distinguished alumnus;

Kinnaired College for Woman and Aitcheson (chiefs) College, still the most expensive educational establishment in the country.

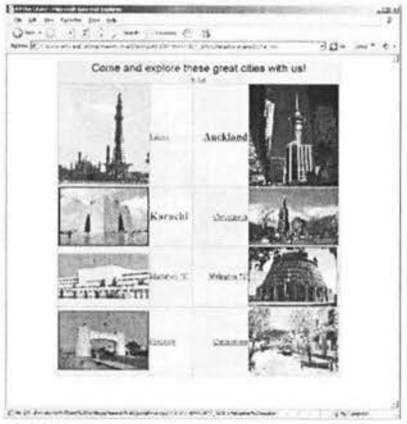
The National College of Arts, has separate departments in Architecture, Fine Arts & design, on the competition entry basis 450 students receives from all over the country.

King Edward Medical College is the country's largest medical institution, founded in 1870.

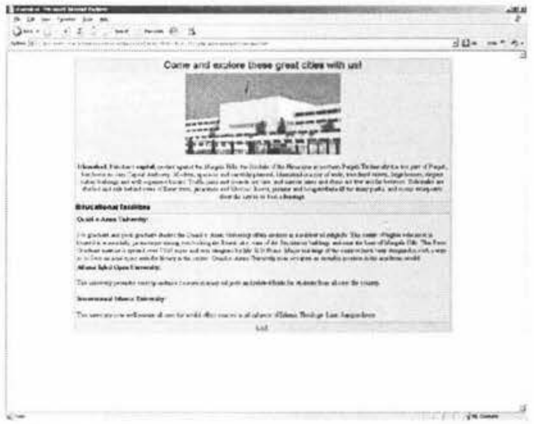
Storyboard:



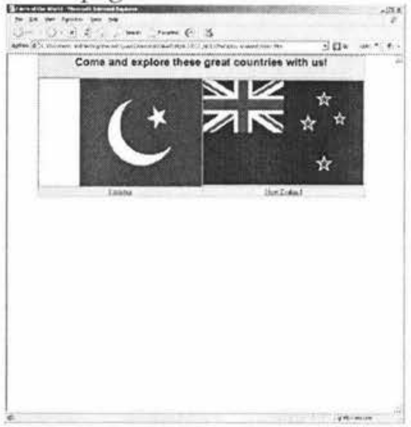
Homepage



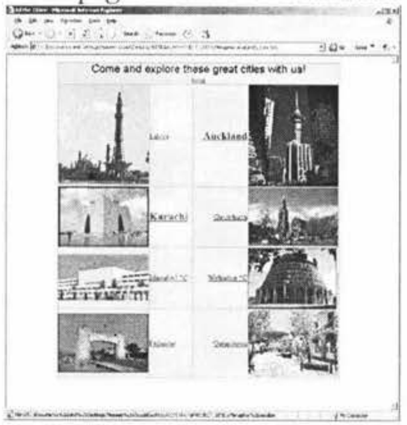
Cities pages – Click on Islamabad



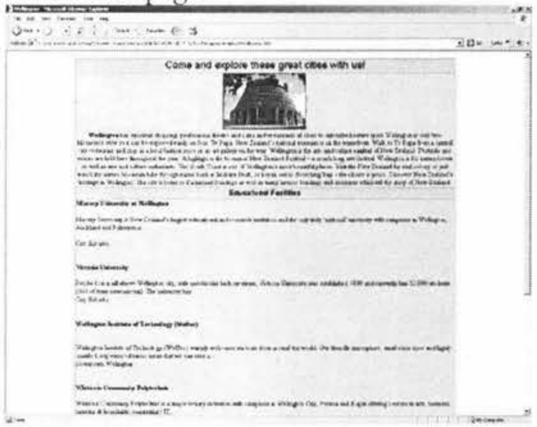
Islamabads page.



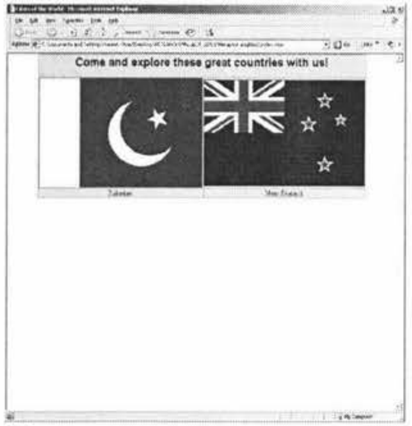
Homepage



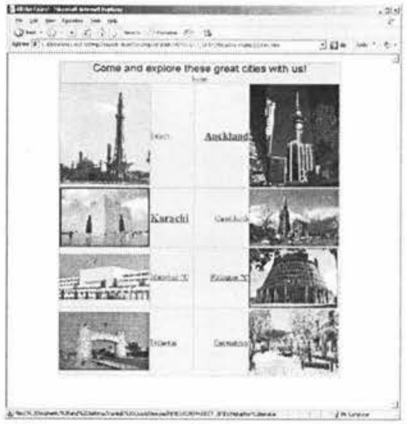
Cities pages – Click on Wellington.



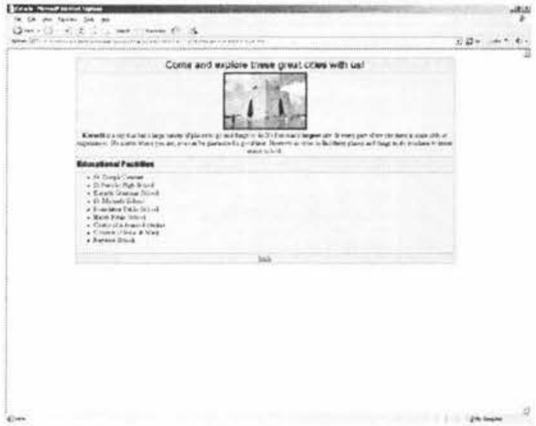
Wellingtons page.



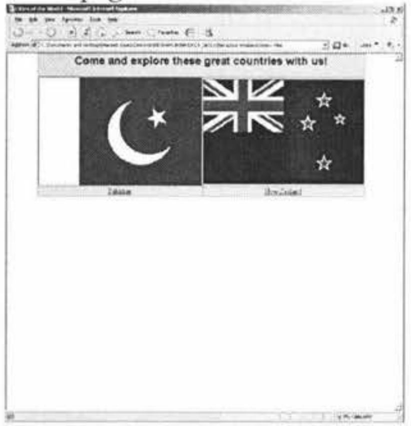
Homepage



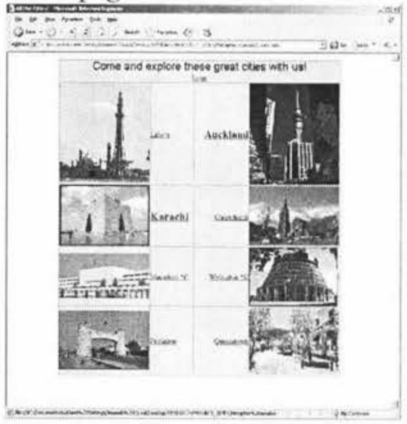
Cities pages – Click on Islamabad



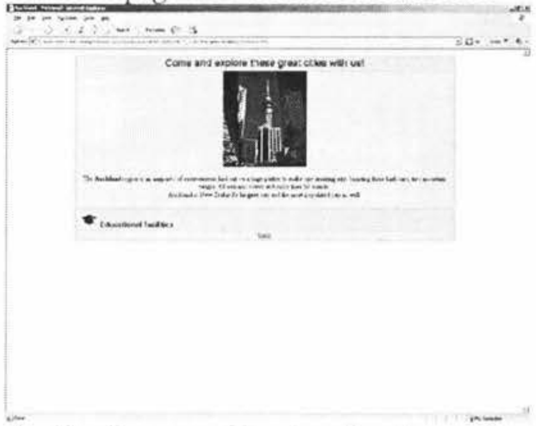
Karachi's page with educational facilities.



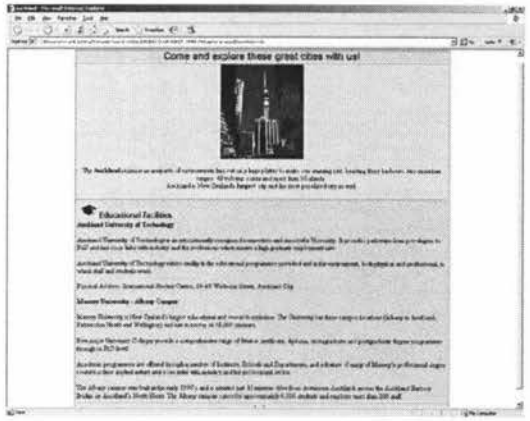
Homepage



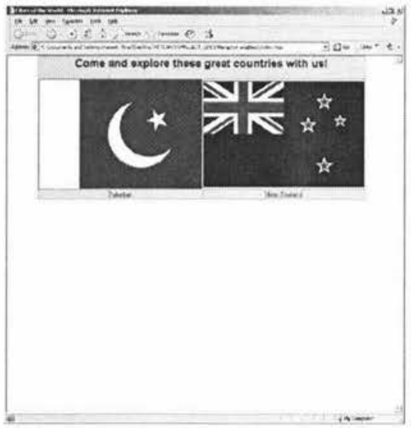
Cities pages – Click on Wellington.



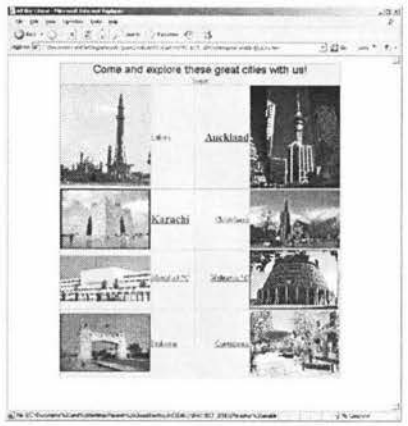
Aucklands page with educational metaphor – content no showing yet.



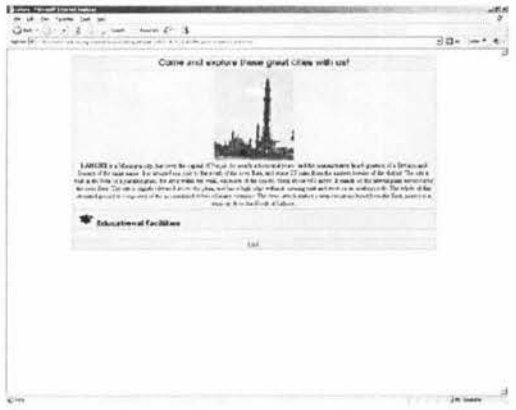
Aucklands page with educational metaphor – content showing once clicked on metaphor.



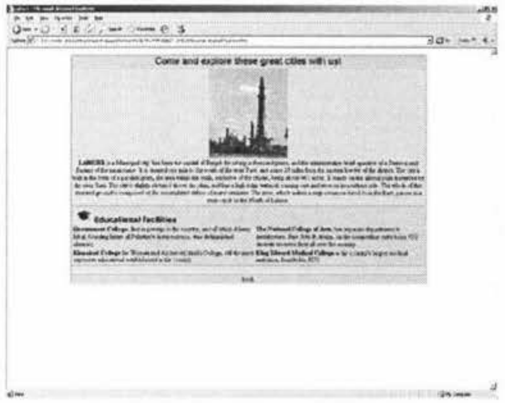
Homepage



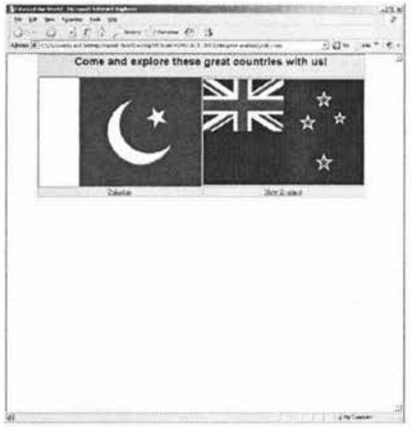
Cities pages – Click on Islamabad



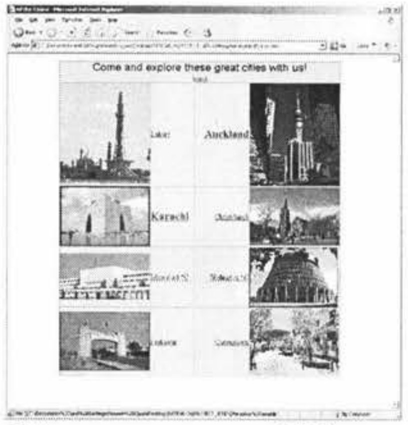
Lahore's page with educational facilities.



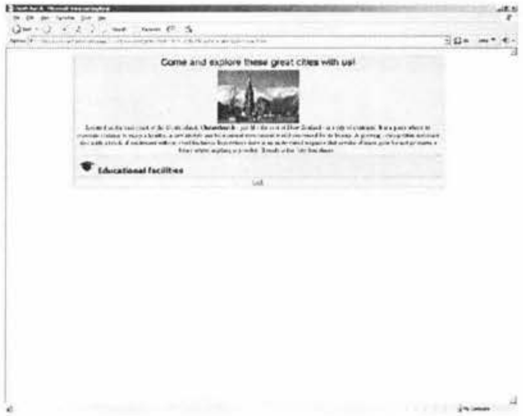
Lahore's page with educational metaphor – content showing once clicked on metaphor.



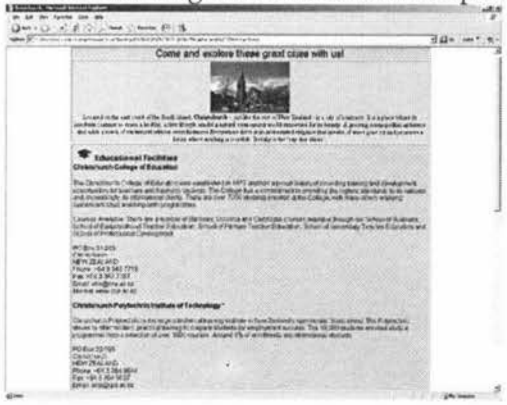
Homepage



Cities pages – Click on Wellington.



Christchurch's page with educational metaphor – content no showing yet.


Come and explore these great cities with us!
Educational facilities
Christchurch College of Education
Christchurch Polytechnic Institute of Technology
Christchurch's page with educational metaphor – content showing once clicked on metaphor.