An Ontological Framework for Contextualising Information in Hypermedia Systems.

by

Andrew James Bucknell

Thesis submitted for the degree of

Doctor of Philosophy

University of Technology, Sydney

2008

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Student

Production Note:
Signature removed prior to publication.

Dedication

This thesis, and the work that went in to it, is dedicated to Anthony & Margaret Bucknell - Mum & Dad. Your love, support and belief made it possible for me to persevere and succeed. For that, I am eternally grateful.

Acknowledgements

I would like to thank my supervisor, Prof. David Lowe for his advice, support, encouragement and especially his patience as I have undertaken the journey that is a PhD candidature. I would also like to thank the academic and support staff of CSE for creating an environment filled with enthusiasm for applying computer systems to real world problems.

Thank you to Lauren Scott for her assistance with proofreading this thesis. Thank you to Dr. Narelle Smith for her advice on statistical analysis and the staff at the Institute for Interactive Media and Learning for their assistance with Survey Manager.

Thank you to Andrew & Vanessa Watts and Sara & Colin Shorter for being the kind of friends who know me better than I know myself and love me for it.

Thank you to Ed Tobin, Steve Evans, Robyn & Sam Watts, Rodrigo & Marines del Busto, Wen Smallwood, and Alex Thompson for their friendship and for their generosity when I have most needed it.

Thank you to Phillip Kazanis and Sam Beskur who have given me invaluable advice and encouragement along the way, and who's friendship helped put tasks such as the one involved in writing a PhD thesis in perspective.

Finally, and most of all, I would like to thank my family for being all the things one could hope for in a family. My parents, for raising me to be who I am. My brother, Daniel and sisters, Karyn and Deborah, and their families, for being part of who I am. Your love and support has given me the strength to persevere and succeed.

Contents

Table of Tables

Table of Figures

Abstract

The Internet has become part of everyday modern life. A central component of the Internet is the World Wide Web. With hundreds of millions of users trying to find information they need amongst billions of pages, there is an urgent need for tools that help users find the information they need. A key element in assisting users find information is their context. Being able to model and store a user's context provides information about the user that can be used to augment their information-seeking behaviours. This work investigated the hypothesis that it is possible to create an ontology of context that can be used to create tools that users perceive to be useful and easy to use when performing information-seeking behaviours on the World Wide Web.

This hypothesis was investigated through three research stages. First, a concept of context was developed that applies to information-seeking behaviours on the World Wide Web. Next, this concept was modelled using an ontology, and a software framework was created based on this ontology. This framework was used to create tools that augment the information-seeking behaviours of users of the World Wide Web. Finally, an empirical evaluation of these tools was performed to determine if they were perceived to be useful and easy to use. The results of the evaluation indicate that the tools constructed were perceived to be useful and easy to use, providing evidence that supports the validity of the hypothesis. This outcome encourages further research and development into using an ontology of context to develop tools that help people using the World Wide Web to find the information that they need.

Extended Abstract

While context is an integral part of interacting with information, existing approaches to managing contextual information on the World Wide Web are application specific and do not support sharing contextual information. The consequence of this is that the contextual information in each application is stored in a way that is specific to that application, and the reuse of information between tools is not explicitly supported. This lack of explicit support for sharing contextual information between applications limits the effectiveness of tools that contextualise information. This thesis demonstrates that an open model of context can be used across applications to contextualise information, and that users find tools based on this approach to contextualisation to be useful.

This hypothesis for this research states that *it is possible to create an ontology of context that can be used to create tools that users perceive to be useful and easy to use when performing information-seeking behaviours on the World Wide Web.* This hypothesis is investigated through three research stages: development of a concept of context that is application neutral; demonstration that this concept of context can be used to contextualise information on the World Wide Web; an empirical evaluation that shows that it is possible to create useful tools using this model.

The concept of context was developed by undertaking a critical analysis of the literature and using this to explicitly identify the role of the user's context in information-seeking behaviours on the World Wide Web. This concept is developed over two phases of investigation. The first phase reviews hypermedia models and systems, including the World Wide Web, to identify the goals of hypermedia and the approaches to information management that are used to achieve these goals. This review identifies the interaction of a user with web resources in producing information as being fundamental to hypermedia. The next phase of the critical analysis builds on this understanding of hypermedia to develop a concept of context that explicitly includes data about the user as a construct a user's interaction with web resources.

Demonstrating the use of the concept of context to contextualise information on the World Wide Web involved two phases of research. The research in the first phase shows how the concepts expressed in the concept of context can be represented using the Web Ontology Language. The second phase develops a software framework based on the ontology of context that can be used to identify, collect and use contextual information. This framework, the ICU framework, encompasses existing approaches to contextualisation while also providing an open architecture based on web services that can be used to make contextual information available to applications that contextualise information. The utility of this framework is demonstrated by constructing a tool that implements existing contextualisation interfaces in a single tool, using the one collection of contextual information. This tool is called ISeeYou.

The empirical evaluation used the Technology Acceptance Model (TAM) to investigate the usefulness and ease-of-use of ISeeYou for users engaging in their regular information-seeking behaviours on the World Wide Web. TAM has been shown to be effective in evaluating the usefulness and ease-of use of new technologies, and has been successfully applied to evaluating web-based technologies. The results of this evaluation indicate that tools based on the framework and the ontology are useful and easy-to-use when performing information-seeking behaviours on the World Wide Web. This outcome encourages the further development of tools that use the ICU framework and further development of ontologies that represent the context of users of the World Wide Web.

This research investigated an approach to managing contextual information that allows reuse of contextual information by using an open architecture and offers a richer set of contextual information by structuring the information using an ontology. By creating an open shareable model of context, the constraints on using contextual information across different contextualisation tools is removed and richer tools for contextualising information on the World Wide Web can be created. This research is predicated on the belief that contextualisation of information on the World Wide Web is an essential tool for helping users manage information, and the development of tools that perform contextualisation is an ongoing challenge for researchers and developers. The ontology and the framework developed in this work aim to help meet this challenge. The research carried out in this work demonstrates that it is possible to create an ontology of context that can be used to create tools that users perceive to be useful and easy to use when performing information-seeking behaviours on the World Wide Web. This result encourages further research in to an ontology-based model of context that explicitly is focussed on the user.

1 Introduction

The internet has become a core technology supporting human activity - indeed one which is often claimed to have become indispensable to modern society (Hoffman, Novak & Venkatesh 2004). Given that management and utilisation of information is at the core of the internet, a key challenge for users of the internet is how they effectively engage with the massive amount of information that is available. This engagement is, in turn, deeply influenced by users' context. This thesis is fundamentally predicated on the assumption that users' information interactions can be improved by enhancing the contextualisation of the information.

Contextualisation helps a user filter and interpret information by making explicit the context of the information. This is useful in situations where users interact with information, such as when browsing or searching the web, when reading emails, or editing documents. In order to contextualise information, applications use elements of contextual information to filter, process or augment the information being viewed or edited in the application.

All information applications that are used by users can benefit from an understanding of a user's context, and all applications contain information that can contribute to a deeper understanding of a user's context. For example, the list of emails received and sent by a user in their mail client can provide useful contextual information in creating more effective rankings of the search results for that user in a Web browser.

Despite the universality of a user's context, contextual information is not typically shared across applications - at least in part due to the lack of application-independent representations of context. This work presents a novel approach - based on an ontological representation of context - to managing and sharing contextual information across applications. The result is the potential for applications that support substantially improved interaction with information.

1.1 Background

Contextualisation uses contextual information to assist with the interpretation[1] of information. In order to support contextualisation, it is necessary to be able to manage contextual information. This work develops an approach to managing contextual information using ontology. The utility of this approach is demonstrated by applying it to the domain of information-seeking behaviours on the World Wide Web.

Context

The view taken in this work is that context can be represented as a set of contextual information, and that (put simplistically for now) contextual information is any information about the user. It includes things like the history of pages visited, keywords from the users email, and metadata embedded in web pages the user has visited. This view of context is by no means the only, or even a complete, understanding of context. The word context has many meanings even within one application domain. A richer definition of context is developed further in subsequent chapters.

Ontology

An ontology defines a set of representational primitives with which to model a domain of knowledge or discourse (Gruber 2007). The domain

[1] "Interpretation" in this context refers both to providing contextual information directly to the user to assist in their interpretation, as well as use of the contextual information by applications to adapt the information presented to the user.

of knowledge being considered here is that of contextual information. Popular approaches to storing contextual information store it in application specific formats. A core driver of this work is the establishment of an approach to managing contextual information in a new way that improves on existing approaches. The structure and formality of an ontology, coupled with the ability for ontologies to be queried for sets of information, suggests the use of an ontology for managing contextual information is likely to be an effective approach.

Information-Seeking

In order to develop a cross-application approach to contextualisation, the application domain of information-seeking behaviours on the World Wide Web has been chosen. This domain has been chosen for several reasons. First, it is a domain that is widely discussed and of great significance to the general community. Second, it is an area whose technologies are, for the most part, open and extensible. This means that the formats used to structure and exchange information are available, and that the way commonly used information-seeking tools, such as Internet Explorer and Firefox, are described by publicly available specifications. Finally, it is an area that clearly exhibits the problems with closed approaches to contextualisation. A key example is history mechanisms in browsers. Each browser keeps a list of pages the user has visited. However, these lists are not shared between browsers. The approach to contextualisation developed in this work explicitly supports the sharing of such contextual information between browsers by allowing any application to add and query information such as browser history.

The World Wide Web is used to access information. Information seeking can be viewed as a form of problem solving. Two basic behaviours have been identified as being used by people seeking information in electronic environments such as the World Wide Web.

One kind of behaviour is browsing where the user seeks to satisfy an information need by scanning a collection of information (Marchionini 1997). The second behaviour is searching, where the user utilises a query to convey their information need to a search engine (Broder 2002). A description of information-seeking behaviours, shown in table 1-1, is used.

Scanning Mode	**Information Need**	**Information Seeking**	**Information Use**
Undirected Viewing	General areas of interest; specific need to be revealed	"Sweeping" Scan broadly a diversity of sources, taking advantage of what's easily accessible	"Browsing" Serendipitous discovery
Conditioned Viewing	Able to recognize topics of interest	"Discriminating" Browse in pre-selected sources on pre-specified topics of interest	"Learning" Increase knowledge about topics of interest
Informal Search	Able to formulate simple queries	"Satisfying" Search is focused on area or topic, but a good-enough search is satisfactory	"Selecting" Increase knowledge on area within narrow boundaries
Formal Search	Able to specify targets in detail	"Optimizing" Systematic gathering of information about an entity, following some method or procedure	"Retrieving" Formal use of information for decision-, policy-making

Table 1-1 Information Seeking Behaviours on the World Wide Web (Choo, Detlor & Turnbull 2000)

These behaviours provide a categorisation of the kinds of behaviours perform when seeking information on the World Wide Web. Such behaviours are enhanced by contextualisation. This work uses these behaviours as the basis for applying contextualisation to helping users find information on the World Wide Web.

It is important to note that information-seeking behaviours are not the focus of the research outcomes in this work. First and foremost, this work is about contextualisation. Information-seeking behaviours are a natural application for contextualisation, and provide an application domain through which contextualisation can be investigated.

This leads us to propose the following hypothesis that forms the basis for this research.

1.2 Hypothesis

It is possible to create an ontology of context that can be used to create tools that users perceive to be useful and easy to use when performing information-seeking behaviours on the World Wide Web.

This research focuses on establishing the validity of this hypothesis. This thesis investigates this hypothesis by developing and working through a research method.

1.3 Research Method

The research method for this work uses commonly used research techniques to analyse the validity of the hypothesis (Clarke 2001). This work's hypothesis draws together a set of concepts and suggests they can be used to address a particular problem. The concepts are the web, information, context, and ontology. The problem this work is addressing is that of finding information on the World Wide Web. The research method describes how these concepts will be defined and how identified they will be used to address the research problem. A critical analysis of the concepts that inform the hypothesis will be performed. Theoretical modelling is then used to show how the concepts can be applied to help users find information on the World Wide Web. Finally, an empirical evaluation of the outcomes of the modelling to show that they do indeed address the research problem is performed.

1.3.1 Research Techniques

The hypothesis was developed by examining the problems facing web users and considering approaches that will help them find information. The validity of the hypothesis will be established using the standard research techniques of critical analysis, theoretical modelling, and empirical evaluation. These techniques are applied to a set of questions arising from the hypothesis.

In order to analyse the hypothesis, a set of 15 questions have been identified. These questions are investigated to assess the validity of the hypothesis. These questions can be grouped into four categories based on the approach taken to address them. The questions around theoretical modelling have been divided into two subcategories, Ontology Modelling and Software Modelling, to highlight the different modelling techniques involved.

Critical Analysis

The research literature is analysed to develop concepts of the World Wide Web, of information and of context that clearly position this work relative to existing work. In addition, ontology research is analysed and reviewed to assess its usefulness for the problem domain within which this thesis is working. These concepts provide a consistent vocabulary and understanding for the rest of the work.

Theoretical Modelling

This work develops a model of context using ontology. This model demonstrates how the concepts identified in the hypothesis can be applied to the research problem of finding information on the World Wide Web. First, the requirements for a model of context based on the concepts developed earlier are established. Next, it is shown that ontology meets these requirements. Finally, an ontology of context to demonstrate the application of the model is built. The theoretical modelling concludes by applying the concepts and models that have been developed to show how to create software tools that use context to help users find information.

Empirical Evaluation

An empirical evaluation is performed to assess usefulness of the tools that have been modelled in helping users find information on the World Wide Web. Performing this evaluation provides evidence to be used for assessing if the ontology and the tools that have been modelled help users find information on the World Wide Web. The results of this evaluation are used to determine the validity of the hypothesis.

1.3.2 Research Questions

The investigation of the validity of the hypothesis is performed by deconstructing it in to a set of research questions. Each of these questions is investigated in turn, using the appropriate research technique.

1.3.2.1 Critical Analysis

Critical analysis involves a detailed investigation of the concepts that inform the hypothesis. This investigation allows this work to be positioned relative to existing research. It also provides a sound conceptual framework from which to develop the model. The concepts are identified by asking questions about the hypothesis, and the analysis is performed by answering these questions.

Q1. What is the World Wide Web?

The World Wide Web is a collection of protocols and standards that allow information to be shared over networks. It is based on concepts emerging from hypermedia research, in particular the concept of the hyperlink. In order to understand how the World Wide Web can be augmented by an ontology of context it is necessary to have a clear understanding of what the World Wide Web is, what the motivations behind its creation were, and what directions it is heading. Three bodies of work are investigated to develop an understanding of the World Wide Web. The hypermedia literature is reviewed to understand the concepts that have underpinned the World Wide Web. The work of the W3C is reviewed to identify the standards and protocols that implement the World Wide Web.

Q2. Why is the World Wide Web important?

The World Wide Web has emerged from a small research project at CERN to a global infrastructure used daily by hundreds of millions of people. It has transformed the way people shop, work, study and spend their leisure time. The sheer scale of information available on the World Wide Web means that users encounter problems such as being lost in hyperspace and information overload. These problems reduce the usefulness of the World Wide Web for finding information. Understanding these problems, and how they affect the ability of users to find information, is a necessary first step in proposing solutions to them. An understanding of these problems is developed by performing a critical analysis of the literature in which they are discussed. They are not unique to the World Wide Web, and so Human Computer-Interaction (HCI) and Information Retrieval (IR) literature are explored to understand approaches taken to mitigate these problems in other information related fields.

Q3. What are information-seeking behaviours?

This work is interested in helping users carry out information-seeking behaviours. Information-seeking behaviours are the ways that users interact with information in an information management system. There is a significant body of research that investigates these behaviours. A critical analysis of the literature is performed to develop a clear understanding of information-seeking behaviours and how they describe the ways people use the World Wide Web to find information.

Information-seeking behaviours are a way of classifying the approaches that people take to seeking information. Four kinds of information-seeking behaviours have been identified for the World Wide Web. These are directed search, undirected search, directed browsing and undirected browsing. These behaviours are explored to develop a solid understanding of what it is that people do with information on the World Wide Web. By developing an understanding of the behaviours people perform when using the World Wide Web, the kinds of tasks that people perform are identified. This set of tasks informs the choice of what kinds of tools that need to be improved.

Q3. What is context?

Context is a broad term that is used widely to refer to various concepts. To clearly identify what this work means when it discusses context, a survey of the literature in information science and hypermedia research will be performed to clearly establish a concept of context that will be used throughout this research.

Information Science describes concepts of information and concepts of context. These concepts express understandings of context or information in different problem domains. This works draws on concepts of context from information science and also on understandings of context developed in hypermedia research to develop a concept of context for the World Wide Web. This concept provides the basis for the ontological model this work develops.

Q4. What is the role of context in information-seeking?

In the hypothesis is proposed that context can be used to create useful tools. Context can be used in this way because context has shown to be useful in supporting information-seeking behaviours. A critical analysis of the literature is performed to show how context is used in hypermedia systems, IR systems and HCI research to help users find information.

Context helps users find information on the World Wide Web in two ways. First, it helps mitigate the experience of being lost in hypertext by providing the user with familiar reference points and trails that let them easily return to a familiar point. Second, it helps mitigate the experience of information overload by providing the user with summaries and search filtering tools that use knowledge deduced from the ontology to help highlight information that is likely to be of greater relevance to the user.

1.3.2.2 Theoretical Modelling

The theoretical modelling in this work consists of two phases. The first phase of the theoretical modelling is developing an ontology of context. This ontology provides a way of structuring the elements of context and information that have been identified in the critical analysis as being relevant to the research problem. The ontology modelling is performed by working through a set of questions that arise from the hypothesis. This phase is driven by questions 5 through 7. The second phase of the theoretical modelling is the development of a software model that allows the ontological model to be applied to the research problem. The software model builds on the critical analysis and ontology modelling to develop software tools that users can use to find information on the World Wide Web. This investigation shows that the software model has been developed by addressing a set of questions that arise from the hypothesis. This phases covers questions 8 through 11.

Q5. Is ontology an appropriate way to model context?

The hypothesis for this thesis proposes that an ontology of context is useful. When referring to ontology here, this work is referring to Web Ontology as defined by the W3C as part of its Semantic Web initiative.

Web Ontology is a way of modelling knowledge domains for use on the World Wide Web. Having identified context as a knowledge domain in the concept of context, Web Ontology is a good candidate for modelling context.

In order to clearly establish that Web Ontology is suitable for modelling context, the investigation for this question explores the literature around Ontology and Web Ontology and develops a clear understanding of ontology. It then identifies the requirements for a model of context, and shows how ontology satisfies these requirements.

Q6. Can an ontology of context be created?

Having determined that ontology is an appropriate representation of context, the next step is to show that an ontology of context can be constructed. It is important that the knowledge domain of context can be adequately represented by the ontology.

This investigation demonstrates that an ontology of context can be constructed by creating one using the Protege tool from Stanford's Knowledge Systems Laboratory (Knublauch et al. 2004). The approach taken to developing the ontology is based on the principles outlined in the Ontology 101 paper from the Knowledge Systems Laboratory at Stanford (Noy & McGuinness 2001). This approach builds an ontology appropriate to the knowledge domain of interest. In this work, the knowledge domain of interest is context on the World Wide Web.

Q7. How can an ontology of context be used to help contextualize information?

After investigating question 2 about why the World Wide Web is important an understanding of the role of context in helping users find information has been developed. Having constructed an ontology of context, there exists an explicit representation of a domain of knowledge that is made up of contextual information. In order to assess the validity of the hypothesis an investigation is performed examining how this explicit representation can be used to help users find information. This is done by developing queries that are able to use the ontology of context to extract contextual information. This investigation shows that this contextual information is useful by relating it to the kinds of contextual information that have already been shown in previous research to be useful to users when they perform information-seeking behaviours.

Queries are developed that show an ontology of context allows intelligent software tools to identify patterns of behaviour that can be used to assist the user with identifying relevant information. Additionally queries are developed that deduce semantics of links based on knowledge of the user and make these semantics available to the user.

Q8. How can an ontology of context be integrated into existing web infrastructure?

Having a model of context is the first step in proving the hypothesis. It shows that it is possible to represent context in a way that is appropriate for the World Wide Web. The next step is to show that this model can be used with the existing web infrastructure. This is shown by analysing the literature that web services provide an appropriate infrastructure for integrating a new data repository in to the World Wide Web. Following from this, the specifications of a web service that makes the model available on the World Wide Web as a web service are developed. This web service will serve as the foundation for the software framework that is developed next, and will provide a way for web-based tools to manage contextual information.

Q9. Can a software framework based on the ontology of context be created?

After investigating the previous questions, it has been shown that a web service can be created that makes the ontology of context available on the World Wide Web. In order to develop tools that make use of this ontology, it is necessary understand how software tools can utilise the services that have been described. A software framework is created that shows how context can be identified, stored, and used by tools.

Q10. Can tools based on this framework be used with existing web browsers?

The model of context and the software framework allow us to create tools that use context to help users find information on the World Wide Web. Given that people typically use web browsers to access information on the World Wide Web, it is shown how these tools can be integrated with web browsers. The discussion then focuses on the technologies available for extending web browsers, with a focus on Internet Explorer. Next, it is shown how these technologies can be used to support tools that identify and use contextual information. Finally, it is shown how a server that manages contextual information can be integrated with this technology.

Q11. How is the software framework used to construct tools that augment information-seeking behaviours?

The software framework has been developed so that the ontology of context is available as a web service. The investigations have also provided a design for software components that act as clients of this web service, updating and consuming context. This design shows how tools that use context to help users find information can be constructed.

This software framework is used to show how web clients calling into the World Wide Web service form the basis for tools that make contextual information available to people using the World Wide Web. A client-server software architecture is developed that describes the communication protocols that will be available to tools. Additionally, it is shown how these protocols and this architecture are platform independent.

1.3.2.3 Evaluation

In order to show that the analysis and modelling have resulted in tools that help users find information, an empirical evaluation is performed. An empirical evaluation involves a direct measurement of a metric. In order to perform an empirical evaluation, it is necessary to first identify the metric that is to be measured, and then identify a procedure for performing this measurement. It is also necessary to determine how the measurements are to be interpreted.

Q12. How can the perceived usefulness and perceived ease-of-use of context-based tools that augment information-seeking behaviours be evaluated?

The hypothesis states that an ontology of context will allow the creation of tools that are useful and easy-to-use. Usefulness and ease-of-use are metrics defined by the Technology Acceptance Model (TAM). Tools measured by TAM as being useful and easy-to-use have a high rate of adoption by end-users. This indicates that the tools meet the purpose for which they were designed.

Various evaluation methods for software are explored, with particular focus on techniques for evaluating hypermedia systems. The choice of TAM as the approach for evaluating the tools is explained. Finally, a TAM survey is developed that can be used to measure the usefulness and perceived ease-of-use of the tools, and explain how this survey can be analysed to determine the validity of the hypothesis.

Q13. Are the tools useful and easy to use?

The final step in the investigation is to take the TAM evaluation that has been developed is applied to the tools that have been created. The results are collated and analysed to determine whether the tool that has been developed was found to be useful and easy-to-use by the evaluation group..

1.3.3 Approach and Thesis Structure

The approach consists of four steps. First, a detailed analysis of the literature is performed to clearly define the problem and to provide a conceptual framework for its solution. This analysis shows how context can be used to help users find information. Next it is demonstrated how ontology can be used to model context for use on the World Wide Web. Third, it is shown how this model can be used to construct software tools that augment the ability of users to find information. Finally, an empirical evaluation of the tools that have been constructed was performed to show that they help users find information on the World Wide Web.

In chapters two and three, the relevant literature is reviewed to establish clearly the concepts that inform this work. In chapter two, hypermedia literature is reviewed to answer the question 'what is the World Wide Web?' and HCI literature is reviewed to answer the question 'why is the World Wide Web important?'. In chapter three, two questions are answered. First, information science and hypermedia literature is reviewed to answer the question 'what is context?'. Next, the question 'how does context help users find information?' is investigated by exploring how context is used on the World Wide Web.

In chapter four, a set of requirements is developed for a model of context, and it is shown how ontology meets these requirements. An ontology of context is then developed, based on the concepts of context developed in the analysis phase. This chapter answers the questions 'is ontology an appropriate way to model context?' and 'can an ontology of context be created'?

In chapter five, the analysis of information seeking behaviours is built upon to show how an ontology of context be used to help users find information. Context-based tools that are useful for helping people find information on the World Wide Web are modelled.

Chapter six shows how the ontology of context can be integrated into existing web infrastructure. The first part of this demonstration is the creation of a service-based framework that supports the identification, collection and use of contextual information. The second part of this demonstration is the development of a tool based on this framework.

Chapter seven shows how the perceived usefulness and perceived ease-of-use of context-based tools that augment information-seeking behaviours be evaluated using TAM. An evaluation for the tools is developed, and an analysis of the results of the evaluation is performed. Finally, it is shown how these results support the validity of the hypothesis. Table 1-2 outlines the structure of the thesis, showing a question identifier, the research question, and the chapter in which the research question is addressed.

ID	Research Question	Chapter
Q1	*What is the World Wide Web?*	2
Q2	*Why is the World Wide Web important?*	2
Q3	*What is context?*	3
Q4	*What is the role of context in information-seeking?*	3
Q5	*Is ontology an appropriate way to model context?*	4
Q6	*Can an ontology of context be created?*	4
Q7	*How can an ontology of context be used to contextualize information?*	5
Q8	*How can an ontology of context be integrated into existing web infrastructure?*	6
Q9	*Can a software framework based on the ontology of context be created?*	6
Q10	*Can tools based on this framework be used with existing web browsers?*	6
Q11	*How is the software framework used to construct tools that contextualise information?*	6
Q12	*How can the perceived usefulness and perceived ease-of-use of context-based tools that contextualise information be evaluated?*	7
Q13	*Are the tools useful and easy to use?*	7

Table 1-2 Thesis Structure

1.4 Contributions

The key contribution of this work is the demonstration that an ontology of context that can be used to contextualise information. An ontology can be used to express a model of context that integrates existing approaches to contextualization while also allowing the development of new approaches. Such an ontology can be used either stand-alone or in a distributed environment. It provides a central repository for various forms of contextual information that are currently stored in their own data proprietary data structures. Having a central repository for context that can be communicated with using an open protocol means that context can be shared across browsers, and even across multiple applications. For example, a user with two different browsers installed currently has two sets of browsing history. Using the approach, the user could have one set of history stored in a central repository. This means that the contextual information of the user is not tied to any specific tool. Instead, it belongs to the user and can be reused by other applications. Existing approaches to managing contextual information are tied to specific applications. The ontology-based approach developed in this work demonstrates an approach to contextualisation that supports cross-application sharing of contextual information. The contribution here is not the ontology itself (although this is important in its own right), but rather the demonstration that ontology can be applied to the problem of user modelling.

This ontology is intended to contribute to future research in three ways. First, the ontology that has been developed can serve as the basis for developing various tools that augment the user's information-seeking experience. Second, the process that has been used in developing this ontology is intended to be used to extend the ontology to include new classes of contextual information and new ontological queries that can be used to assist with contextualization of information. Finally, it is hoped that the implementation of the ontology will serve as a platform for further development of context-based tools that help users engage in information-seeking behaviours on the World Wide Web.

1.5 Conclusion

The massive amount of electronic information available to users of the internet requires a wide array of tools that help users process and interpret information with which they interact. One such technique is contextualisation, which uses contextual information to assist with processing and interpreting information. Existing approaches to managing contextual information are specific to the application, such as the differing history mechanisms for Internet Explorer and Firefox, or the differing indexes for Google Desktop and MSN Desktop. The hypothesis for this work arises from the idea that contextual information can be organised in a way that supports sharing across applications by using an ontology to structure this information in an application independent way. Different applications can add contextual information to the ontology and also query this ontology for the contextual information they need. Investigating the validity of the hypothesis based on these ideas is the subject of the following chapters.

2 Hypermedia and the World Wide Web

Since its inception (Berners-Lee et al. 1994) the World Wide Web has grown at a rapid rate (Berners-Lee 1999) to become a global information system. This rapid growth has been due to the ability of users to find information they need to complete a wide variety of tasks. This chapter analyses the conceptual and technological frameworks that have underpinned the success of the World Wide Web. An information model of the World Wide Web as a node-link hierarchy that facilitates information management is presented. The problems of information overload, cognitive overload and disorientation are discussed in terms of this information structure. This model-based understanding of the problem highlights avenues that can be taken to mitigate the impact of these problems on the ability of a user to find information on the World Wide Web. This chapter investigates two research questions:

Q1. What is the World Wide Web?

The World Wide Web is a collection of protocols and standards that allow information to be shared over networks. It is based on concepts emerging from hypermedia research, in particular the concept of the hyperlink. In order to understand how the World Wide Web can be augmented by an ontology of context it is necessary to have a clear understanding of what the World Wide Web is, what the motivations behind its creation were, and what directions it is heading. This chapter presents the basic concepts of hypermedia and an overview of the World Wide Web.

Q2. What are information-seeking behaviours?

This work is interested in helping users carry out information-seeking behaviours. Information-seeking behaviours are the ways that users interact with information in an information management system (Spink 1999). There is a significant body of research that investigates these behaviours. A critical analysis of the literature is performed to develop a clear understanding of information-seeking behaviours and how they describe the ways people use the World Wide Web to find information.

Information-seeking behaviours are a way of classifying the approaches which people take to seeking information (Choo et al. 2000). Four kinds of information-seeking behaviours have been identified for the World Wide Web. These are directed search, undirected search, directed browsing and undirected browsing. These behaviours are analysed to develop a solid understanding of what it is that people do with information on the World Wide Web. By developing an understanding of the behaviours people perform when using the World Wide Web the kinds of tasks that people perform are identified. This set of tasks informs the choice of what kinds of tools that can be implemented.

2.1 Introduction

There is a massive amount of information available on the World Wide Web. People all over the world use this information everyday for research, commerce and entertainment. Close to one billion people use the World Wide Web (InternetWorld 2006). This is approximately 15% of the world's population. The World Wide Web makes available information to everyone with internet access, no matter where they are in the world. This freedom to access information is unparalleled in human history, and has led to the present day being referred to as the information age (Turner 2002).

As useful as this information is, at times its sheer volume can lead to users of the World Wide Web feeling lost or overwhelmed as they try to locate the information they need. It is estimated that at the end of

January 2005 there was around 11.5 billion indexable web pages (Gulli & Signorini 2005). The total number of pages on the World Wide Web has been estimated to be between 400 and 550 times higher than the number of indexable pages (Bergman 2001). While exact figures are not available, this means at this time there is likely to be somewhere between 5 trillion and 6 trillion pages of information on the World Wide Web. These pages are hosted on over 100 000 000 web sites (Netcraft 2006). The growth rate of information on the World Wide Web has been observed to be around 30% annually for the period 1999-2002 (Lyman & Varian 2006). If this is consistent, almost 2 trillion pages will be added annually over the next few years. With this amount of information available to users, and new information being added all the time, it is no wonder that it can be a challenge to find a specific piece of information (Nelson 1994).

Helping users find the information they need makes the World Wide Web easier to use and more useful. The Pew Internet & American Life project reports that each day 41% of American internet users use a search engine to find information and 21% search the internet to answer a specific question (Rainie 2005). Additionally, 30% use the internet to get news, 22% use it to check the weather, and 21% surf the World Wide Web for fun. 12% bank online, 6% buy a product, and 8% get financial information. These survey results show us that internet users regularly search for information on the World Wide Web. It also shows us that users of the internet use it to find information on topics such as news, weather, product research, and hobbies. It is reported that the average user spends over 27 hours per month using the internet. Given that current statistics show that 145 million Americans use the internet, it can be seen that a lot of time is being spent on these activities. Many users report that they find information difficult to find (Lawrence & Giles 1998), and that they waste time looking for the information they want. The amount of time spent on these activities, and the number of users involved in them, means that improving the ability of users to find the information they want to find is a significant problem.

This work develops a model of context for hypermedia systems. This model is based on existing approaches to context in hypermedia systems, and on understandings of context drawn from related research disciplines such as information science and information retrieval. The model is expressed as on ontological framework. This framework provides a formalism that can be implemented as a distributed system that provides context services to hypermedia systems. This approach allows the model to be made available to existing hypermedia systems.

The usefulness of the model of context is demonstrated by showing that this approach combines the benefits of existing approaches to context. Existing hypermedia systems only include aspects of context. They use context to improve hypermedia. Integrating these existing approaches into an openly available model that provides context services to any hypermedia system means that any system can realise the benefits of using context, and that contextual information that is useful in one system can be made available to any system.

2.2 Hypermedia

Hypermedia is an approach to information management in which data is stored in a network of nodes connected by links (Smith & Weiss 1988). Hypermedia is also known as hypertext, and grew out of research designed to augment the human intellect by creating tools that mimic the human minds ability to link pieces of information (Engelbart 1963).

2.2.1 Hypermedia Concepts

The recognised starting point for hypermedia research is the Memex system proposed by Vanevar Bush (Bush 1945). Memex was proposed as a 'supplement to memory', a device where an individual could store books, records and communications for fast and flexible recall. In addition to storing information, Bush proposed that the memex be able to associate items. It is this ability to associate information, as well as to store and retrieve it, which is recognised as being a defining characteristic of hypermedia.

The first implemented hypermedia system was Augment/NLS. This system was part of his work to develop systems that would augment human intelligence. Ted Nelson's Xanadu system was another early system that has had a lot of influence on the thinking behind current hypermedia systems (and is itself a current hypermedia system, being an ongoing work in progress). The term hypertext was coined by Ted Nelson to refer to documents that could be distributed online. In particular, it was important to Nelson that documents be reusable. Reusability in this case meaning that documents should be annotatable and quotable without altering the original document or comprising the copyright of the original author.

Work based on the concepts that underpinned these early systems forms the field of hypermedia research. Hypermedia has been applied to various problem domains, such as Navigation, Information Analysis, Hypermedia Literature, Hypermedia Art, and Botanical Taxonomy (Conklin 1997; Nuernberg 1998). The are several operational advantages to using hypertext to manage information (Conklin 1987).

- Ease of tracing references.
- Ease of creating new references.
- Information structuring. Links provide a way of structuring pieces of information.
- Global views. The link structures provide an effective mechanism for summarising hypermedia information.

- Customised documents. The link structure can be used to present different views of information.
- Modularity of information. Information can used in different ways as part of different link structures.
- Consistency of information. Using links allows information to be maintained in one place.
- Task stacking. Several trails of information can be explored simultaneously.
- Collaboration. Link structures can be shared and information can be managed by multiple people.

2.2.2 Hypermedia Systems

Hypertext systems provides its users with the ability to create, manipulate, and/or examine a network of information containing nodes interconnected by relational links. Hypermedia systems have five key characteristics (Ashman 1994; Conklin 1987).

1. A Graphical User Interface, with the help of browsers and overview diagrams, helps the user to navigate through large amounts of information by activating links and reading the contents of nodes.

2. An authoring system with tools to create and manage nodes (of multiple media) and links.

3. Traditional information retrieval (IR) mechanisms such as keyword searches, author searches etc. There are also attempts to incorporate structure queries along with content queries - retrieving a part of the hypertext network based on some user-specified criteria.

4. A hypermedia engine to manage information about nodes and links.

5. A storage system which can be a filesystem or a knowledge base or a relational database management system or an object-oriented

database management system.

The different hypermedia applications that different authors wish to create often require different infrastructure support from the hypermedia system. As well, there remains much scope for investigation into what kind of support can be offered by a hypermedia system, and how this support can best be implemented. As a result, there are numerous hypermedia systems in existence. Some are targeted at supporting specific kinds of applications. Others investigate what kind of support a hypermedia system can provide for applications. Others again investigate how this support can be implemented. These categories are by no means mutually exclusive. In the following discussion, the systems mentioned are included to illustrate properties of a particularly class of hypermedia systems. There is no suggestion that the systems mentioned do not belong in some way also in the other categories.

2.2.2.1 Adaptive and Adaptable Systems

Adaptive systems select, filter and transform information to meet the needs of users in constantly changing environments (Belotti et al. 2005). The information model of adaptive hypermedia systems extends beyond the information that makes up the content of the hypermedia documents. Adaptive hypermedia includes a user model to describe the goals, preferences and knowledge of individual users (Brusilovsky 2001). This information is used by the system to adapt the content and links presented to the user into an appropriate format and level of complexity. There are many techniques for deciding how the content and links are restructured on a page (Brusilovsky 2001; Mobasher, Cooley & Srivastava 1999; Perkowitz & Etzioni 2000). Generally, adaptive hypermedia systems rely on the information having sufficient granularity and/or duplication that components can be assembled as required. The hypermedia information model underlying adaptive hypermedia systems is typically no different from the usual sorts of models discussed previously. What is interesting about adaptive hypermedia information models is the inclusion of user models, and the recognition that the user's context should determine the information that should be presented to the user (Brusilovsky 1996). The importance of user models has led to a focus on personalisation in recent adaptive hypermedia research, where the user model is used to customise the content. In these systems, the user model captures information about the user as a set of properties describing the user. The model of context developed in this work extends this idea further by considering the content the user has viewed and is viewing as part of the user model.

2.2.2.2 AHAM

Much of the work done relating to adaptive hypermedia has been into applications. Typically these applications have been constructed using extensions to a traditional hypermedia system. An example of this is the hypertext course notes made available through the Adaptive Hypermedia Application Model (AHAM) system (De Bra, Houben & Wu 1999). This application is a collection of HTML documents that have associated scripting to allow the tracking of a users progress through the information structure. Users' ability to access some information is restricted by requiring them to view pre requisite knowledge before moving on to new sections. Such applications demonstrate the effectiveness of adaptive hypermedia, but require that each application be constructed essentially from scratch. The AHAM framework addresses this problem by providing a system that supports the functionality typically required in adaptive applications.

2.2.2.3 Microcosm

The Microcosm hypermedia system is based around an open hypermedia model developed at Southampton University. The authors of the model posit that "such systems provide users with richer and more diverse ways to access and integrate information from large and dynamic data sets in distributed, heterogeneous environment" (Fountain et al. 1992). The model is designed to satisfy five essential criteria.

1. A system which does not impose any mark-up upon the data which prevents that data being accessible to other processes that do not belong to the system.
2. A system which can integrate with any tool that runs under the host operating system. Data produced by tools that are not part of the hypermedia system may be used within it without adding any special value to that data and without compromising the continued use of the data outside the system.
3. A system in which data and processes may be distributed across a network, and across hardware platforms.
4. A system in which there is no artificial distinction between readers and authors.
5. A system in which it is easy to add new functionality. i.e. new program modules may simply be inserted.

The key focus of the Microcosm hypermedia model is transparency. The user interacts with the system through viewers, which display the data available to the system. In response to user input, the viewer generates messages that are sent to the Microcosm Document Control System. The MDCS passes the message through the system message filters, which are responsible for processing the message. The filters send the results of their processing to the link dispatcher, which in turn passes a request to the MDCS to display the requested data in the appropriate viewer.

There are two important features of this implementation. Firstly, the media data is not altered by the hypermedia system. The link model is tracked independently of the data so that it can be reused by other applications. This causes some problems in maintaining link consistency, but this issue can be managed through imposing either manual or automated link management strategies.

The second important feature is that the various program components of Microcosm such as the viewers and the filters are not a static part of the Microcosm system. New viewers can be made available by adding media programs to the host system. Filters are created to be dynamically linkable. This means that Microcosm can take advantage of existing media players. This allows the system developer to focus on issues relating to information structuring and management rather than on creating and manipulating proprietary hypermedia data formats.

Adaptive hypermedia research has gained a new significance with the calls for the development of a semantic web (Kravcík & Gaševic 2006). The ability for an information device to modify the content displayed in response to semantic cues is an idea that flows directly from earlier work on systems such as Microcosm and the Amsterdam Hypermedia Model (AHM).

2.2.2.4 Personal Information Management Systems

Personal information management (PIM) is intended to support the activities people perform to order their daily lives through the acquisition, organisation, maintenance, retrieval and sharing of information (Teevan, Jones & Bederson 2006). PIM systems have many features in common with hypermedia systems, in that they provide an interface to a node link hierarchy that supports information management operations. Where they differ is that the information they manage is a set of information belonging to the user.

Various approaches are taken to the creation of PIM systems. The Stuff I've Seen system (Dumais et al. 2003) provides a searchable index of all the documents on a users machine, including word documents, emails and web pages. The index is automatically generated as documents are created or viewed on the users system. Stuff I've Seen provides users the ability to search for documents based on strings from those documents. The user cannot edit the structure. The MyLifeBits system (Gemmell et al. 2002) adds a users documents to a store and then allows the user to create trails and tags to provide access to the information. Because the focus of MyLifeBits is on managing photos and videos and other non-text media, it cannot rely on keyword matching in the same way as Stuff I've Seen. These different approaches to how users interact with information reflect the fact that PIM systems, and hypermedia systems in general, are used for a wide variety of tasks.

PIM recognises the importance of a user centred view of information. The user acts as the author of the information space simply by engaging in information management behaviours, with the information space being dynamically created (Boardman & Sasse 2004). PIM systems are applied to areas such as managing health information management systems (Pratt et al. 2006) and systems that interact with the physical environment (Czerwinski et al. 2006). This approach shifts the focus from the author of information to the user.

2.2.2.5 Contextualisation Systems

Contextualization is the act of placing a piece of information in a context where it can be understood by the person receiving the piece of information (Theodorakis et al. 1999). By making the context of the information explicit and relevant to the receiver, the intended meaning of the information can be retained. In hypermedia systems a similar approach can be taken. By making elements of context relevant to the user visible, the meaning of the information can be more readily grasped, and the relevance of the information to the users information need can be assessed more readily.

Contextualisation is used by the Watson system (Budzik & Hammond 1999) to provide a just-in-time infrastructure in which information is dynamically selected to be made available to the user in response to their information needs. No explicit interaction beyond their regular information seeking behaviour is required. Watson acts an information broker that automatically discovers information on behalf of the user. Several key principles guide the implementation of Watson.

- User behaviour is modelled.
- Search is automatic and distributed.
- Search is directed and context rich.
- Search is performed in real-time.
- Bandwidth is conserved.
- Results are post processed.
- Results are presented unobtrusively.

Contextualization is enabled by the system building a model of the users context. This set of contextual information is combined with the users information need as specified by the information query to create a result set that is grounded in the users context.

Another application of contextualization to information management is the CSAKTive space project (schraefel et al. 2004). An mspace is a semantic view of an information space that allows a user to explore a domain (schraefel, Karam & Zhao 2003). The associations between information components in an mspace based on an ontological structure of the information space. The semantic structuring of an information domain using an mspace supports a richer interaction with the information space than is achieved by traditional search or browsing techniques. An mspace places information in a semantic structure that helps the user better access information by providing structures that are meaningful to the user (schaefel et al. 2005).

2.2.3 Hypermedia Models

Hypermedia models were developed based on early monolithic hypermedia applications. The goal of these models is to provide a principled basis for comparing systems as well as for developing interchange and interoperability standards.

Hypermedia systems structure information by associations between components, much like semantic nets in cognitive science. Because of this similarity hypermedia structures are often shown using a graph notation, with components being represented by nodes and the associations between components being shown as links between nodes. While this graph-based model is useful for representing the structure of hypermedia systems, it is not the only notation available. For example, systems using a set based representation have been proposed. It is important to recognise that hypermedia systems are not about a graph based representation of information; rather they are concerned with expressing associations between information components.

Early research into hypermedia focussed on developing a system that implemented the concepts the researcher thought should be in a hypermedia system. More recent projects have concerned themselves as much with developing models that allow for a common understanding of hypermedia concepts across various systems, as on the systems themselves. Developing formal models of hypermedia serves both as a reference point for comparing systems, and as guidelines for implementing systems.

2.2.3.1 Dexter Model

The Dexter Hypermedia Model was developed as a reference model for evaluating hypermedia systems (Halasz & Schwartz 1994). The Dexter Model "provides a standard hypertext terminology coupled with a formal model of the important abstractions found in wide range of hypertext systems". The Dexter Model describes a three-layer architecture for hypermedia systems. These layers are the run-time layer, the storage layer, and the within-component layer. The definition of these layers provides a model that can be used for implementing and evaluating hypermedia systems (Grønbæk & Trigg 1993).

2.2.3.2 Amsterdam Hypermedia Model

The focus of the Dexter Model was on hypertext applications. Traditional hypermedia systems tended to be text and image based, as this was the limit of the available hardware systems. With the advent of more powerful desktop systems, audio and video are now available to the vast majority of potential hypermedia systems users. Text and image documents typically do not require any time synchronisation, and thus this was not an issue for Dexter. With the addition of time-based media elements such as audio and video, the ability to synchronise elements by time has become desirable. This raises new issues that the Dexter Model does not, and was never intended to, address. The Amsterdam Hypermedia Model is a "general framework that can be used to describe the basic constructs and actions that are common to a wide range of hypermedia systems" (Hardman, Bulterman & van Rossum 1994).

AHM's makes several major departures from Dexter. Firstly, it expands the link model to include temporal relations using properties of AHM components called synchronisation arcs. Secondly, it introduces the idea of link contexts. Link contexts allow the specification of presentation behaviour when a link is followed, thus enabling the author to re-use presentation behaviour, and have components that share a context be presented in a consistent manner. Finally, AHM includes the concept of channels. Channels are an abstract device for presenting the content of a component. Channels have associated with them media related and media independent properties that alter the presentation characteristics. Channels can also be turned on or off to hide or display their content in the current presentation.

2.2.3.3 Open Hypermedia Systems

In early hypermedia systems, the links between documents were embedded in the documents and were managed as part of the information belonging to the document. This approach makes it difficult to create hypermedia applications that interoperate with one another because the link semantics may be different between applications and there is no way of mapping between them. This problem arose in monolithic systems such as NoteCard (Halasz 2001) and KMS (Akscyn, McCracken & Yoder 1987). One solution to this problem is to separate links from content and have them managed separately. Any client application can then recreate the link structure by accessing the link database. Systems built using this approach are referred to as Open Hypermedia Systems (Davis et al. 1993).

This approach of has been applied in hypermedia systems such as Chimera (Anderson, Taylor & Whitehead Jr 1994) and Microcosm (Fountain et al. 1992). In the development of Microcosm several minimal requirements for an OHS were identified. These requirements can be paraphrased to say that an OHS

- should not use inaccessible data formats.
- should not restrict access by external components to the systems data.
- should be distributable across networks.
- should not make a distinction between authors and readers.
- should allow the addition of new functionality.

These requirements can be used to drive the development of Open

Hypermedia Systems. The key benefit of OHS research is the support for interoperability between systems so that components can be reused.

These models provide a common understanding of the features of hypermedia. The fundamental concept in hypermedia is the node and the link. Different semantics can be applied to the links to elicit different behaviours in the hypermedia system. The next section describes how the World Wide Web is constructed based on this node-link hierarchy.

2.3 The World Wide Web

The World Wide Web is based on the node-hierarchy that makes hypermedia an effective information management approach. The World Wide Web can be viewed as an Open Hypermedia System (Anderson 1997). It is built on open protocols and standards. This allows the information on the World Wide Web to be consumed by a variety of agents, and published on a variety of servers. The next section describes some of the protocols and standards that made the World Wide Web so popular. The following section reviews some key technologies in the semantic web, a project which has been proposed to alleviate some of the problems facing existing web users.

2.3.1 Web Technologies

The World Wide Web is made up of many protocols and standards. The key standards that are of interest to this work are the hypertext transfer protocol (HTTP), universal resource locators (URL), the hypertext markup language (HTML) and extensible markup language (XML). Each of these is discussed below.

2.3.1.1 HyperText Transfer Protocol (HTTP)

Web pages are published on web servers. A web server listens for requests sent using HTTP (Fielding et al. 1999). The request contains a Uniform Resource Identifier that specifies the page being requested, and the server sends a reply containing that page. The full set of requests and responses handled by a HTTP server are defined in an open specification defined by the Internet Engineering Task Force (IETF). This specification grew out of a project to share documents at the CERN research lab (Berners-Lee 1999). There are two key features of HTTP that underpin its success.

- Simple. The set of requests and responses handled by HTTP is fairly simple compared to other similar application level protocols. This simplicity makes it easier for developers to build applications that use HTTP as the transport mechanism, and it makes it easier for developers to build servers and clients that comply with the specification.
- Open. The HTTP specification is freely available to developers. It is available to anyone who wants to use it as the transfer layer for their application.

2.3.1.2 URL

A Uniform Resource Locators (URL) is used to `locate' a resource, by providing an abstract identification of the resource location (Berners-Lee, Masinter & McCahill 1994). A resource is either a file or the output of a program. The format for HTTP URL's is shown in figure N.

http://<host>:<port>/<path>?<searchpart>

The host part of the URL is either the fully qualified domain name of a network host, or the IP address of a network host. The optional port number specifies the port on the network host with which the connection is to be established. If no port is specified, the default port of 80 is used for http connections. The path is an HTTP selector (Fielding et al. 1999). The search part is a query string. A URL is used in the HTTP protocol to request documents and other resources from a network host.

2.3.1.3 HTML

The HyperText Markup Language is a collection tags that can embedded in a document (Raggett, Le Hors & Jacobs 1999). More formally, it is an application of the Standard Generalized Markup Language (SGML) application. SGML applications define a set of tags that can be embedded in a text document to associate semantics with sections of text (Goldfarb & Rubinsky 1991). HTML was designed to express the presentation semantics of text elements in a document. It has been extended over time to include scripting elements and other semantic elements such as META tags.

Besides encoding presentation semantics, HTML provided a tag for encoding link semantics. By embedding an A tag in a document, the text enclosed in the tag becomes an anchor to the URL specified in the HREF property of the tag. An anchor tag indicates that the text enclosed by the tag should be rendered using a custom style, and that activating the text should cause an HTTP request for the URL should be sent to the appropriate server. Typically, activation involves a mouse click, but other agents may have different behaviour.

2.3.1.4 XML

The eXtensible Markup Language (XML) describes a class of data objects called XML documents and partially describes the behaviour of computer programs which process them (Graham & Quin 1999). XML is a subset of SGML that allows a set of tags to be described that express the structure of a document. It differs from HTML in that HTML is an application of SGML that describes a set of tags that describe the presentation format of elements within a document. XML can be used to describe such a markup language, as has been done with XHTML. XML is designed specifically for use on the World Wide Web and is also designed to be simple to use and understand. Combining XML with XSL (eXtensible Stylesheet Language) (Clark 1999) and Cascading Style Sheets (CSS) (Bos et al. 1998) allows structured documents to be created and exchanged between web-based applications.

2.3.2 Semantic Web

The semantic web is an initiative of the W3C concerned with defining and linking information so that it can be more effectively searched, browsed, filtered and rendered. It is based on two key principles. First, it is based on common formats for interchanging data. Second, it is based on languages for recording how data on the World Wide Web relates to real world objects.

The semantic web is a collection of standards and protocols, in much the same way that the World Wide Web is a collection of standards and protocols. Significant standards for the semantic web are Uniform Resource Identifiers (URI's), the Resource Description Framework (RDF) and the Web Ontology Language (OWL) (Shadbolt, Hall & Berners-Lee 2006). URI's are used to identify the location of electronic resources. URL's for the World Wide Web are an instance of URI's used to identify web documents. URI's allow for more than just web documents to be identified. A Uniform Resource Identifier (URI) is a compact sequence of characters that identifies an abstract or physical resource (Berners-Lee 2005).

RDF provides a mechanism for describing knowledge in terms of three pieces of information; the subject, the property and the value (Powers 2003). An RDF triple has the following form:

{ http://www.andrewbucknell.com/index.html, title, "Andrew's Home Page" }

This triple encodes the knowledge or fact that the World Wide Webpage located at the URI http://www.andrewbucknell.com/index.html has the title "Andrew's Home Page". A set of knowledge can be created by adding multiple RDF triples to a store known as a triple store. RDF supports the semantic web by providing a way to represent knowledge about web-based resources.

The ability to reason about knowledge is at the core of the semantic web. Reasoning requires that associations between facts be available. This can be achieved by encoding relationships in to an RDF agent. A more interoperable and open approach is to encode these relationships in a data structure so they can be consumed by any agent. This is the purpose of the World Wide Web Ontology Language (OWL) (Heflin 2004). OWL provides support for encoding associations between facts and constructing queries that allow inferences to be made from a set of knowledge.

The semantic web is, at it's a core, a way of structuring knowledge about web-based resources so that agents are able to reason about and make inferences from this knowledge (Berners-Lee & Lassila 2001). This knowledge is used to help users of the World Wide Web find the information for which they are seeking.

2.4 Information Seeking on the World Wide Web

The World Wide Web is used to find information. Information seeking can be viewed as a form of problem solving (Marchionini 1992). Two basic behaviours have been identified as being used by people seeking information in electronic environments such as the World Wide Web. One kind of behaviour is browsing where the user seeks to satisfy an information need by scanning a collection of information (Marchionini

1997). The second behaviour is searching, where the user uses a query to convey their information need to a search engine (Broder 2002). The search engine retrieves a set of documents that are relevant to the query specified. Understanding these modes of behaviour is the first step to understanding the problems web users face. When a people engage in information-seeking behaviour, it's usually because they are hoping to resolve some problem or achieve some goal, for which their current state is inadequate (Belkin 2000). This work seeks to help users achieve these goals and resolve these problems.

2.4.1 Browsing

Browsing is an approach to information seeking that is informal and opportunistic. It depends heavily on the information environment (Marchionini 1997). This kind of behaviour involves scanning a set of information to satisfy an information need. Common sources of sets of information that are scanned are email programs, word processors and web browsers. Any application that displays information to a user can be scanned for information. Browsing in electronic information environments has been categorized as *directed browsing*, *semi-directed browsing*, and *undirected browsing*.

Directed browsing involves a specific information need and a clear set of documents that will be scanned. An example of directed browsing is a user scanning their email inbox for an email containing flight details or reading meeting minutes to see who attended a meeting. Semi-directed browsing involves a less specific information need and a less specific set of documents to scan. Examples include reading through all emails from an associate to see what they have been doing or scanning through the abstracts of a set of documents to identify trends in a research field. Undirected browsing involves an unspecified information goal and no clear set of documents to scan. It is the equivalent of randomly flicking through channels on a television. Examples of undirected browsing include a user scanning through the inbox of their email to see what has come up recently, or reading through a company magazine that is available on a network share. The World Wide Web is used for a wide range of activities. All of the activities on the World Wide Web involve information seeking, also referred to in the literature as information searching. Information-seeking is a process in which humans purposefully engage in order to change their state of knowledge (Marchionini 1997).

2.4.2 Searching

Search engines provide a way for users of the World Wide Web to locate information without having to have a specific URL (Schwartz 1998). Search is used to satisfy a range of information needs on the World Wide Web (Spink 2003).

The user describes their information need using a query string and submits it to the search engine. The search engine replies with a set of links that it considers relevant to the query string. This set of links is called the resultset. The resultset from is retrieved from its index. An index maps query terms to URL's. Indexes for the World Wide Web are typically built automatically to cope with the massive number of documents available (Arasu et al. 2001). Software agents called crawlers visit websites and download all the pages they can access, and create an inverted index. This approach is used by the leading search providers such as Microsoft (MSR), Google (Brin & Page 1998) and Yahoo (Broder 2002). The alternative to automated index construction is to create the index by hand. This approach was used by Yahoo when it first started, but this approach was replaced after a few years by automated indexing to cope with the volume of information available.

2.4.3 Information Seeking

Information seeking is a behaviour people engage in so as to change their state of knowledge. There are various understandings of the information seeking process (Belkin 1993; Ingwersen 1996) This work builds on the an in formation acquisition model developed on an information seeking model developed to explain how users interact with electronic information systems (Ellis 1989).

1	Intention	Mental characterisation of the desired goal
2	Selection	Choosing an appropriate tool and data source
3	Execution	Entering a command to start the desired search.
4	Evaluation	Review of results to direct further actions

Table 2-1 Information acquisition interaction

In terms of this thesis, the process can be viewed as follows. The intention stage is dependent on the user. However, by tracking context, it is possible to build a profile of intentions based on characteristics such as community behaviour. That is, if it is known that the user is a member of a specific community, and that community regular displays a specific intention, there is a good chance that the user will also form this intention. By knowing likely goals of a user, the system can be modified in an appropriate way.

Selection involves limiting the set of information entities that is to be searched for information that satisfies the user's intention. As with selection, tracking context allows a hypermedia system to make assumptions about what information entity sets should be included in the set of information entities that is to be searched. Search engines are one example of information entity sets that a search can be performed on. More specialised entity sets are also available, such as the Microsoft Developer Subscription Network pages. Selecting the appropriate set of information entities to search has a significant impact on quality of search indicators such as recall and precision.

Execution involves the user formulating their intention in a manner that can be processed by the information searching system, entering this representation, known as a command or query, into the system and running it. This process can be assisted by using knowledge about the users context to make available predefined queries that express common intentions. It can also be used to automatically refine the query to include context specific information.

Evaluation of the results of execution phase is often a rather involved process. It can include browsing through the result set, browsing to pages referred to by the result set and their associated site, and it can also involve further refining of the query. As an added complexity, it is quite common for the result set to modify in some way the original intention of the user. A hypermedia system should support all of these behaviours, and by tracking context in the system, the ability to evaluate the result set can be significantly enhanced.

These four activities, intention, selection, execution and evaluation, characterise information seeking tasks. In the earlier paper that explored notions of information management, three of the six categories of information management behaviour can be grouped under the banner of information acquisition, and thus described by the process outlined above. Of the six categories, summarisation, retrieval, navigation and analysis can all be classified as information seeking behaviour, and described by the process outlined above. Information-seeking is a key activity of users of the World Wide Web (Martzoukou

2004). The next chapter explores information seeking on the World Wide Web in more detail.

2.4.4 Issues with Information-Seeking Behaviours on the World Wide Web.

While the World Wide Web is a valuable resource, several key problems reduce its usefulness. These problems are information overload, disorientation and cognitive overload. The first problem arises from the sheer volume of information available on the World Wide Web. The other two are common to hypermedia systems.

Information overload is the inability to extract needed knowledge from an immense collection of information (Nelson 1994). Reasons for this difficulty include a lack of understanding of information, being overwhelmed by the amount of information to be understood, not knowing if information exists, not knowing where to find information, or not having the key to access desired information (Wurman, Leifer & Sume 2001). The success and popularity of the World Wide Web is due to the huge amount of information it makes available, but at the same time this volume of information can reduce the usefulness of the World Wide Web.

As a hypermedia system, the World Wide Web is subject to two problems that are endemic to hypermedia systems. There problems are disorientation and cognitive overload (Conklin 1987). Disorientation is sometimes referred as getting "lost in hyperspace" and is the tendency of a user to lose their sense of location and direction in a non-linear document. Cognitive overhead is the effort and concentration required to maintain several tasks or trails at one time. These problems reduce the usefulness of hypermedia systems, and reduce the usefulness of the World Wide Web.

Kimble et.al. (Kimble, Hildreth & Grimshaw 1998b) suggest that one of the key causes of these problems is the decontextualisation of information in an electronic environment. A significant area of research explores approaches to mitigating these problems by developing methods to contextualise information. In particular, adaptive hypermedia approaches and the semantic web are concerned with providing contextualised information to users. In adaptive hypermedia, the information is filtered and transformed based on the user's context and adaptive rules prior to rendering. In the semantic web, information is filtered and transformed based on the user's model and on semantic information encoded in the page. This work treats the semantic information and the user context as information of equal significance to the document. Rather than transforming the document, the approach taken in this work is to model the semantic information and the user context in a way that allows these pieces of information to be visualised in a way that helps the user contextualise the document they are currently viewing. Such contextualisation helps to reduce the problems of disorientation and cognitive overload (Kirsh 2000) by providing the user with clear reference points about where they are in an information space and what task they are performing (Nilsson & Mayer 2002). It also helps reduce the impact of information overload by providing the user with information that lets them make more informed choices about the links they follow and the content they read. It has been established that approaches to contextualisation do result in tools that mitigate these problems (Kimble, Hildreth & Grimshaw 1998a). Providing useful tools based on contextualisation is an effective approach to mitigating information overload, cognitive disorientation and disorientation.

This work aims to reduce the impact of these problems on the usefulness of the World Wide Web. These problems can be mitigated by providing useful tools that help the user find the information they want (Nelson 1994). This thesis develops a model of context that allows contextualisation techniques from hypermedia systems to be integrated in to the existing web infrastructure. It also develops a model in such a way that existing approaches to contextualising information on the World Wide Web can be integrated into a single model.

2.5 Conclusion

The World Wide Web is a hypermedia system. It organises information as a collection of nodes associated via links. This information structure is implemented using a set of open standards and protocols. The building blocks of the World Wide Web are the HyperText Transfer Protocol and the HyperText Markup Language. Chief among are the reasons for the success of the World Wide Web are the openness of these protocols and standards, their extensibility and their simplicity. The ability to easily create HTML documents and publish them on HTTP servers has led to a massive amount of information being available to users of the World Wide Web. The World Wide Web as it exists now has far outgrown the original designs of its creators.

The shortcomings of the World Wide Web have been recognized with the development of initiatives such as the Semantic Web effort. While the benefits of the World Wide Web are significant, users are encountering problems handling the huge amounts of information available to them. Problems such as information overload and cognitive disorientation have been identified as having a negative impact on the users of the World Wide Web to find information they need. There is a clear need for tools and techniques to be added to the World Wide Web that help to reduce the impact of these problems on the users of the World Wide Web.

The next chapter discusses how the problems of information overload and cognitive disorientation can be viewed as arising from a lack of context on the World Wide Web. Context can help users filter information and also better recognise the associations between information. One class of useful tools on the World Wide Web would be tools that help reduce problems arising from a lack of context. In chapter three, a concept of context is established and the requirements for applying context to these problems are laid out. The remainder of the thesis shows how these concepts of context can be used to create tools that are useful to users engaging in information seeking behaviours.

3 Information and Context

This chapter makes clear the ideas of information and context that are applied in this work. The broad use of these terms across a wide range of research disciplines and even in common speech makes it important to clearly define the scope of any work around these terms. Failing to do so results in a lack of clarity for the work and a lack of focus for the researcher. To make these ideas clear this chapter investigates two research questions:

Q3. What is context?

This thesis proposes modelling context can serve as a basis for creating useful tools for finding information on the World Wide Web. In order to create a model of context, it is necessary to have a clear understanding of what is meant by context. This chapter develops a concept of context that is used as the basis for a model of context.

Q4. What is the role of context in information-seeking?

The notion of context is used widely in hypermedia and World Wide Web research. This question explores some of the approaches that are relevant to helping find information on the World Wide Web. The first part of the exploration looks at how context has been represented in web-based systems. The second part of the exploration identifies a set of information elements that this work views as being context on the World Wide Web. This set forms the basis for the development of a model of context.

Contextualization is defined as a process that makes the context of web-content known to the reader. Context is information about the user and information about the content.

3.1 Introduction

Terms such as information and context are terms with significant semantic overloading. It is important when dealing with terms like information and context to be clear about the meaning of these terms as they relate to the current work.

The previous chapter introduced the idea that the problems of information overload, cognitive overload and disorientation can be mitigated by providing users with tools that are useful. The discussion explored how context is used by adaptive, PIM and contextualisation systems to help users find information on the World Wide Web. The discussion did not explore what this thing called context looked like. This chapter develops a clear understanding of context that serves as the basis for the remainder of this investigation. Before a model of context can be developed, it is necessary to be clear about what is being modelled.

This chapter develops an understanding of information and context by developing concepts of information and context for the domain of the World Wide Web. It then presents a set of contextual elements based on these concepts that describe context in this domain.

3.2 Information Concepts

In seeking to understand what is meant by information in hypermedia systems, it is useful to explore other disciplines relating to information such as information studies. Drawing on the concepts and tools of other fields can make a significant contribution to how information is viewed and discussed in hypermedia. One of the ideas that is drawn on heavily in this work is that of information concepts (Belkin 1978). By focusing on developing an information concept for hypermedia systems, it is possible to avoid getting involved in discussions that are beyond the scope of this work. It is important to establish what is meant when the word information is used. It is worthwhile examining the literature produced by researchers in the field of information science as they have sought to define the focus of their field. A great deal of literature has explored various approaches to defining the term information as it applies to the field. Amongst the various approaches taken, there seems to be a consensus, stated explicitly by Belkin, that "we are not concerned with definitions of information, but rather with concepts of information". As Belkin notes, the distinction is that a definition states what the phenomenon defined is, whereas a concept is a way of looking at, or interpreting, the phenomenon. He then states that "by accepting the idea of a concept one becomes free to look for a useful concept, rather than a universally true definition, of information." Rather than seeking an all-encompassing definition for information, researchers such as Belkin and Ingwersen have focussed on developing a utilitarian understanding of information in their discipline. Given the disparate uses of the term information in reference to hypermedia, the development of a functional information concept for this domain seems more likely to yield results than a search for a specific definition. The remainder of this paper will explore some of the information concepts developed as part of information studies. Following on from this a set of requirements that an information concept for hypermedia systems should address is described, and a concept that fits these requirements is derived.

3.2.1 An Overview of Information Concepts

Information concepts have been developed in various disciplines, but most notably in information science. In information science, the goal of an information concept is to describe the nature of the phenomena that is the focus of the science. Various concepts have been developed as researchers seek to identify the fundamental basics of what the discipline is about. In seeking an information concept for hypermedia, it is worthwhile revisiting some of the earlier concepts, developed along with current thinking. The different focus of information science means that ideas have developed in a direction that differs from the approach required for considering hypermedia systems.

3.2.2 Shannon's Communication Theory

Shannon developed the first, and one of the most widely used, formal information concept as part of his paper "A Mathematical Theory of Communication" (Shannon & Weaver 1963). The communication theory developed by Shannon is aimed towards solving problems related to the transmission of information over communication lines. As such, it ignores the semantics of the messages being transmitted. The semantics are ignored because the model Shannon develops aims to be able to transmit messages for which the semantics are unknown. The importance of this idea lies in the recognition that the meaning of messages, i.e. their information, is unimportant in the communication system. The information is only recognised when the recipient selects a message from the set of possible messages. In a similar way, the media stored in a hypermedia system can be viewed as sets of messages. How the messages are interpreted is unspecified until a receiver selects a subset of these messages to view.

3.2.3 Information As A Property of Matter

A later concept discussed widely by Soviet researchers but later applied specifically to information science treats information as category and as property of matter. Belkin describes this concept in his paper, "Information Concepts for Information Science". The main point of this concept is that information can be viewed as a property of matter. In this case, matter is "things". This concept describes two types of information, subjective and objective. Objective information is a property of matter, while subjective information is a reflection of an individual's consciousness of objective matter. Information is said to be a fundamental property of matter and of consciousness. The information about things can fall into categories, and what someone sees as the information of an object depends on the categories of information they recognise or are interested in. This builds further on the previous concept by stating that as well as the information resulting from a user selecting a message, the message itself may have some property that is called information. At the time that a message is selected, the recipient also chooses (either explicitly or implicitly) what information properties of the message to recognise.

3.2.4 Information As Structure

More recently, researchers have focussed on information as structure. This concept is developed by several authors, and leads to Ingwersen's concept, which is described next. It is based on the general idea that information is that which is capable of transforming structure. Belkin distinguishes between two levels of an information system, these being the linguistic level and the cognitive level (Belkin 1978). At the linguistic level, text is generated (Belkin deals with text rather than multimedia) by some generator, and this text is then interpreted by a receiver. At the cognitive level, states of knowledge transformed into information which are then processed against anomalous states of knowledge (information needs). Belkin sums this idea up by stating the information associated with a text is the generators modified (by purpose, intent, knowledge of recipients state of knowledge) conceptual structure which underlies the surface structure (e.g. language) of that text.

Ingwersen develops the information as structure concept further (Ingwersen 1996). He states two requirements for an information concept in information science.

"On the one hand information being the result of a transformation of a generator's knowledge structures (by intentionality, model of recipients' states of knowledge, and in the form of signs); and the other hand being something which when perceived, affects and transforms the recipient's state of knowledge." Ingwersen's concept is based heavily on Belkin's information concept and Brookes' information concept. It draws together Belkin's ideas about the dual nature of a concept involving both the relationship of the author to information and the relationship of the reader to information, and Brookes' ideas about the different impact that information will have on different knowledge structures. The recognition of information being associated with both the user and the author parallels well the state of affairs in hypermedia systems. In hypermedia systems media is structured by an author to model an information structure. This media is then accessed by users, with each user generating their own information structure in response to the media.

3.2.5 Information As Process

Some work in information science seeks to develop an information concept that transcends any single application. They seek to unify information concepts used in various disciplines in order to provide a single reference point for research in information science. An example of this is the discipline independent concept of information (Losee 1997). This model aims to encompass all information concepts defined in different areas of research. While the concept raises interesting ideas about information concepts, and indeed is quite useful in a discipline such as information science, it does become quite abstract and would appear to be more suited as a framework for developing application or discipline specific information concepts. (E.g. an information concept for hypermedia.) The information concept states that all processes produce information and it is the value of characteristics in the process's output that are information.

3.2.6 Information As Thing

In his paper "Information as Thing", Michael Buckland distinguishes between three meanings of information, these being information-as-process, information-as-knowledge, and information-as-thing (Buckland 1991). The first two meanings are traditional understandings of information in the information sciences. Information-as-thing is Buckland's link between traditional understandings of information, and information as it is used in information systems. He argues that any representation of information is in itself a thing, and thus information systems deal exclusively with information-as-thing. Information-as-thing does not invalidate the traditional meanings of information. Rather, it provides a way of understanding how these traditional understandings are mapped into information systems such as those that are digital.

Hypermedia systems are examples of systems that Buckland would consider to be managing information-as-thing. There is a subtle containment here, where information in a hypermedia system is actually a representation of other forms of information, while within the hypermedia system, all information is information-as-thing, irrespective of how it may be interpreted by receivers.

Finally, a warning against possible over analysis of the difference between information and data is given to researchers in information-related domains (Machlup 1983). He starts out looking at data. Data means literally (from the Latin) "the givens". Data is nothing more than the things a user is given; in the case of hypermedia, text, audio clips, video clips and images. He then argues that for computer systems that for most purposes there is no difference between data and information and that in fact the two are just different names for the same thing. This second argument ignores the role of the reader and author in describing the information something contains.

Machlup's main aim seems to be to warn against over analysis of the word information; in attempting to clarify the term, researchers wind up ignoring the uses the word is put to by people in general. He suggests that rather than restricting the word information to only mean creation things, it is more appropriate to restrict the meaning temporarily by the use of adjectives to make explicit what is being referred to.

3.2.7 Implications for this Research

There is not one generic, absolute definition of information. In research relating to information, it is necessary to identify a concept of information. From the analysis of context that is outlined above, the following properties of context can be identified as being significant for the World Wide Web.

- Context is data about a user.
- Context is data about a web-resource.
- Context is dynamic.
- Context is emergent.
- Context is incomplete.

These ideas are used as a basis for discussing a concept of context in the following section.

3.3 A Concept of Context

The following sections review the nature of context as it has been considered in information-science research. An understanding of context is developed independent of the problem domain with which this work is concerned.

3.3.1 The Varied Nature of Context

Information in hypermedia systems can be viewed as messages. When a user views a document's contents, the user is receiving a message. The impact this message has on the user's knowledge state is a result of the user's interpretation of the message. However, the change in knowledge state is not purely a function of the message received, otherwise everyone who received a message would interpret it the same way, and there would be no ambiguities or misunderstandings. Obviously, there is more input to the interpretation process than the message. This "something more" is the context of the user. Context itself is a widely ambiguous term (Dervin 1997). Dervin concludes that "there is no term that is more often used, less often defined, and when defined so variously as context.". Dervin's paper asks the question "What is context?" and provides a broad discussion of context in the social sciences. A number of common themes are identified in the discussion of context in the literature. Some of these are more pertinent to a discussion of context in hypermedia systems than others, as the focus in this discussion is not on all modes of communication, but on a particular mode of communication. The following themes are of particular interest.

1) Knowledge is partial and temporary.

When perceiving a message or an event, a receiver will only interpret as much of it as is relevant to his or her current interest. Therefore, what a receiver knows about a message or event depends on the context on which it is being received.

2) The knower and the known are inextricably bound.

Each person has a unique collection of *a priori* knowledge, goals and experiences that affect how a message is interpreted. Thus, what a user knows because of receiving a message depends on the nature of the user. The result of interpretation of a message is not constant. Because it depends on the knowledge, goals and experiences of the user, the interpretation will vary across users, and will even for the same user as their knowledge or goals or experiences change over time.

3) Context is not usefully conceptualised as an independent entity

It is neither meaningful, nor useful to identify an entity independent of the received message and label it as context. Context is more than just a set of related facts that alter perception of a message. Any such entity is an artificial construct that trivialises the impact of external factors on perception.

4) Context is necessary as a source of meaning.

Communication relies on common experiences and understandings in order that the interpretation of messages by sender and receiver are close enough that the ideas understood by the receiver are those generated by the sender.

It is not possible to draw a neat box and put all the things that are context inside it and label the box as context. Context is not easily labelled as set of state variables associated with a message, or as a set of variables describing a user's state. These are possible representations of context in a system, but they are by no means the only possible, or sometimes not even the appropriate, representation.

Contextual information can come in any number of forms. For example, visual context, information context, user context all are forms of contextual information that affect how the user interprets the information component. By altering any of these peripheral inputs, an author can influence the user's interpretation.

3.3.2 Information Concepts that Discuss Context

The issue of context arrives quite regularly in discussions of information concepts. This is not surprising given that context is an important factor in what is perceived to be information, and how that information is interpreted. In this section, some common types of classes of context are identified and discussed in relation to the information concepts they can arise in. By no means is this list exhaustive. Its purpose is to provide some understanding of how some of the more well-known information concepts view context.

3.3.3 Context as *a priori* Knowledge

Information concepts such as those of Belkin (Belkin 1978) and Brookes (Brookes 1977) consider the impact of the reader's state of knowledge in perceiving information. Belkin views the information as the change in the user's knowledge state. Brookes goes further, considering how data received by the information system from the user affects the information delivered to the user. In both cases, the users existing knowledge structure is influencing the perception of the component information.

The existing knowledge of the user (the reader) is a significant factor in how the user interprets a piece of information. A user's pre-existing knowledge can be used to contextualise a piece of information. For example, key words that identify knowledge the user already has can be highlighted in a document to show the user sections that are more likely to be of interest.

3.3.4 Context as situation

In developing his cognitive based information concept for information retrieval, Ingwersen discusses the classification of objects in a social context (Ingwersen 1992). Given a set of objects, similar groups of people in different environments would classify the same objects in different ways. The relationships between objects were influenced by factors external to the users and external to the objects. The relationships identified were explained in terms of environmental factors. Ingwersen uses this to argue that the situation in which a user is interpreting information impacts on the knowledge the user extracts from the information. Situation thus provides a context within which interpretation occurs.

The environment of the user influences how the user interprets information. It also influences the properties of the information that can be made available to a user. For example, a user using a low bandwidth connection to access information on the Internet with a hand held device would be able to handle less information than a user on a broadband connection at their desktop. Information about things such as device capabilities and network connectivity describe the situation of the user, and affect the kind of information that the user would find useful. Information about the capability of the information tool(s) being used also forms part of the situation.

3.3.5 Context as information need

When a user interacts with an information system, they have some motivation that guides their efforts. This can be anything from boredom impelling a series of mindless operations to kill time, to a pressing need to understand something that results in a series of well-structured queries presented to the system interface. In the first case, the user has a quite loose information need, and thus it is likely to be satisfied by anything of mild interest to the user. In the second case, the user has a tight information need, and will view anything that does not address the information need as not being information. Only elements that satisfy the user's information need are accepted as information. The reason a user has for wanting to know something is an important factor in determining what the user will and won't interpret as information (Belkin 1993).

The information need of the user is an important property of an information-seeking task. For a search the information need can be expressed by a query. The ability of the user to formulate a query that represents their information need accurately varies. Nonetheless, searching requires the user to formulate a query, and thus provides an explicit representation of the information need. In browsing tasks, the information need is not so clearly expressible, but this does not make it any less significant. Information about the information need can be derived from a priori knowledge and from the user's situation, as well from the information source.

3.4 Context in Hypermedia Models

Contextual information is information that is peripheral to the component information, but influences how the component information can be perceived and interpreted. A lack of contextual information is a major contributing factor to the problems of cognitive overload, disorientation, and link identification. Various approaches are used to capture contextual information in hypermedia systems.

3.4.1 Metadata

The possible uses of the World Wide Web seem endless, but there the technology is missing a crucial piece. Missing is a part of the World Wide Web which contains information about information - labelling, cataloguing and descriptive information structured in such a way that allows Web pages to be properly searched and processed in particular by computer. In other words, what is now very much needed on the World Wide Web is metadata. W3C's Metadata Activity is concerned with ways to model and encode metadata. A particular priority of W3C was to use the World Wide Web to document the meaning of the metadata (W3C 2000). The work on metadata has been subsumed in to the work on the semantic web (Benjamins et al. 2004).

As noted above by the W3C, there is an urgent need for information about information that is available on the World Wide Web. However, more than this, there is an urgent need for information that is relevant to the current user about information.

In order to meaningfully use metadata, semantics must be assigned to the name/value pairs used. An example of this is work carried out by the Dublin MetaData Workshop. This workshop was concerned with identifying the "metadata elements required for the discovery of what were called document-like objects". The syntax of the elements is not specified, this being left as a detail for implementers. The Core describes the semantics of a small set of elements that can be used to identify document like objects in an online environment.

3.4.2 Dublin Core

The elements included in the Dublin core are concerned with storing information about the document. There is some support for extending the set of metainformation by assigning name/value pairs as the value of one of the elements, but this is an unwieldy mechanism for making any significant additions to the Dublin core elements.

3.4.3 Warwick Framework

Recognising that imposing a predefined set of elements as the only possible metainformation elements limited the potential expressiveness of metadata, the creators of the Dublin core developed the Warwick framework. This metadata framework allows arbitrary sets of metadata to be added to content, and transported with the content over networks. The framework included the quite powerful concept of having filters that would interpret metadata as it was received and take appropriate actions. Thus, the semantics of the metadata can be determined as it received by passing it off to the appropriate filter to be interpreted. The Warwick report also acknowledges that the delineation between data and metadata is not static. In some contexts, metadata may be data, and vice versa. The report gives the example of a film review. It may be viewed as a property of the film, or it may be viewed as a piece of information in its own right. This is an important step in integrating context into hypermedia models, but the framework does not provide a mechanism for specifying context. Contexts are assumed by the filters that process the incoming data.

3.4.4 SHOE

The standard HTML ontology extension (SHOE) project uses an ontological approach to develop standardised taxonomies of metadata that can be added to HTML documents. The work has been extended to also allow metadata to be inserted into XML documents. This approach is concerned with adding information to documents that conforms to defined semantic structure. This defined structure means that agents processing the document can identify how the metadata tags can be processed. This addresses one of the major shortcomings of the common approach of adding metadata in HTML meta tags. Without some way of identifying how the tags should be processed, an agent is forced to make assumptions about the meaning of metadata tags, or to ignore them. In either approach, the usefulness of such tags is minimised. By allowing agents to identify the set of metadata tags being used, the usefulness of metadata is improved. The drawback to this approach is that it is document-centric.

3.4.5 RDF

The Resource Description Framework (RDF) is a metadata application for XML. It allows the definition of a DTD describing the set of metadata elements that can be added to an XML document. In this regard, it is quite similar to SHOE in that it describes how metadata can be added to documents. One of the drawbacks of RDF is that it does not consider how the set of metadata elements that are described in a DTD are decided on. It also does not provide support in the framework to allow metadata to evolve over time without regularly processing and updating the document. RDF is a specification of the W3C (Beckett 2006).

3.4.6 Context in Dexter

The developers of the Dexter model recognised the use of contextual information in hypermedia systems with the description of component information in their model. Component information describes the properties of the component other than its content. Component information falls into three categories. Firstly, anchors that refer to the content are part of the component information. Secondly, the component presentation specification is part of the component information. Finally, the component information consists of arbitrary attribute/value pairs. The Dexter model places no restrictions or requirements on the form of the attribute/value pairs. All three of these categories of component information provide information that is not part of the content, but is necessary to the use or presentation of the content in a hypermedia system.

The component information of the Dexter model is a very open and unstructured form of metadata. No semantics are specified, even for the presentation information and the anchors (the semantics are left to the run time layer and the accessor/resolver functions respectively). More typically, the term metadata applies to the name/value pairs category of component information (Raggett, Le Hors & Jacobs 1999), and in this discussion the term metadata will be used to refer to name/value pairs associated with content.

3.4.7 Context in AHM

The Amsterdam Hypermedia Model (AHM) extends the Dexter model to support multimedia features, in particular timing synchronisation between components. It extends the presentation specification associated with components, the link model is extended to include synchronisation arcs, and the concepts of link contexts and channels are introduced.

It is important to note right from the start that the notion of link contexts introduced as part of the Amsterdam Hypermedia Model is quite different from the notion of context that is being discussed in this paper. In AHM, a link context describes the behaviour of a component with respect to other components when a link or synchronisation arc is activated. Either the display area is cleared to make room for the new component, or only part of the display area is cleared, allowing a constancy of presentation display that improves the ability of the viewer to more readily understand the relationships between the old component and the new component.

Synchronisation arcs extend the basic link model described in Dexter to support temporal relationships. They are used to specifying timing of time dependent components with respect to one another. Synchronisation arcs are not links in the Dexter sense. They only describe synchronisation of components. They do not describe associative relationships or navigation.

While the functionality of a hypermedia system described by the AHM model is richer than the Dexter model, there is still the separation between information and metainformation that is made in Dexter. Synchronisation arcs and channels are more sophisticated forms of metadata that allow more complex, time based systems to be described, but there is no significant change in what is considered to be information or metainformation.

3.4.8 Context in Browsers

Web browsers are the tools most often used by users to access information on the World Wide Web. They use contextual information such as browsing history and favourite pages to assist users as they engage in information-seeking behaviours.

3.4.8.1 History

A common feature in web browsers is a history mechanism. This feature shows the user a list of URL's and page titles that have been visited. The history is typically grouped by date and then by domain. The groupings and orderings of this information are enforced by the UI. History is an example of a trail as described in the Memex system. It shows the users navigation through the information space of the World Wide Web, and allows the user to return to pages previously visited.

3.4.8.2 Favourites

Favourites are a feature of hypermedia and web-based systems that allows the user to add page URLs and titles of pages to a list that can easily be called up at any time. The favourites user interface is intended to make it easy for users to return to pages that are of special interest to the user. While in practice the favourites list often gets cluttered and out-dated, it is a widely used and widely available feature.

3.5 Conclusion

This chapter has applied the idea of an information concept to developing a concept of context. This concept of context was developed by investigating the research questions '*3. What is context?*' and '*4. What is the role of context in information-seeking?*'. The need for a concept of context is a result of the overloaded nature of the term. There are many understandings of context both within the domain of the World Wide Web and in broader domains such as information retrieval and information science. In order to focus research into context it is necessary to be clear about what is meant, and this is achieved by developing a concept of context.

In this work, the concept of context is captured in the answers to research question 3 and research question 4. Context is information about a user or about a web page. This information, referred to as contextual information, emerges as a user interacts with information such as web pages. It is a key contention of this work that this information can be stored in a single repository based on a model of context that can capture the kinds of information currently used by existing contextualization applications such as history and favourites interfaces in browsers. The following chapter uses this concept of context to develop a formal model.

4 Ontology

In this chapter, a set of requirements for a model of context is developed, and it is shown how ontology meets these requirements. Following on from this, an ontology of context is developed, based on the concepts of context developed in the analysis phase. This chapter answers the questions 'is ontology an appropriate way to model context?' and 'can an ontology of context be created'?

This works approach to contextualisation requires a user model. The user model must be extensible so that new information can be added to it over time, it must be flexible so that various kinds of content can be managed, and it must be open and interoperable so it can be used across different applications.

Part of the hypothesis proposes that it is possible to create an ontology of context. One of the research questions asks 'is ontology an appropriate way to model context?' In order to answer this question, this chapter investigate the requirements for modelling context, and then show how ontology satisfies these requirements.

In order to build the model of context three issues need to be addressed. The first issue is to determine the requirements for modelling the concept of context. The second is to show how ontology satisfies these requirements. The third issue is to show how ontology is used to represent the concept of context. Working through these issues is the focus of this chapter.

To represent contextual information it is necessary to make sure that all the ideas captured in the concept of context can be modelled. These ideas are explicitly identified by developing a set of requirements for the model. The requirements provide a basis for choosing an appropriate modelling technique. The requirements are developed by an analysis of the concept of context. Following on from this, the key concepts are analysed, as are approaches to creating a model of these concepts that structures contextual information in a way that can be used to help users find information on the World Wide Web.

The second research question addressed in this chapter is 'can an ontology of context be created?'. It is demonstrated that an ontology of context can be created by describing how a standard ontology design methodology can be applied to the concept of context, and how this methodology yields an ontology of context that can be described by the a standard ontology description language.

In order to apply context to helping users perform information-seeking behaviours, it is necessary to be able to represent context in a way that can be used by web-based software systems. Such a representation is called a model. In software development, modelling involves taking a set of concepts and creating a formal description that can be mapped to a suitable data structure. There are many modelling techniques available. Modelling context requires a technique that can capture the concepts described in the concept of context. The requirements for an appropriate modelling technique are derived from the concept of context and its intended application.

4.1 Understanding Ontology

In the work, ontology is used to model context. This section develops an understanding of ontology so that it can be seen why it is a useful technique for the applications of context that are of interest in this work. The traditional view of ontology comes from philosophy where it is literally 'the study of what is'. It has been applied to the study of systems where it is used to formally describe the entities and relationships that make up a system. From this work, it was then applied to modelling domains of knowledge in knowledge systems work. This knowledge systems work has been drawn on to various projects that use ontology to organise information on web-based systems. These projects have been drawn together in to the Web Ontology specification that is part of the W3C's Semantic Web initiative. Web ontology meets the requirements that have been identified for a modelling technique. This discussion shows how it is appropriate for representing context.

4.1.1 What is Ontology?

Ontologies are widely used in knowledge-based systems (Guarino 1998; van Heijst, Schreiber & Wielinga 1997), and are increasingly being applied to web-based systems (Heflin, Hendler & Luke 1999). This section begins by introducing the idea of ontology, and describes how they are used to represent knowledge. It then outlines the benefits of ontology in general, and describes the specific benefits of taking an ontology-based approach to modelling context. The section concludes by showing the approaches used in the construction of the model.

In knowledge-based systems, ontologies provide a conceptualisation of a domain of knowledge. This work is interested in being able to express a conceptualisation of context in hypermedia systems. Ontologies provide a way of modelling the concepts of context in a formal, structured way that is also readily implementable for integration with existing hypermedia systems.

Ontology is used across various diverse domains, such as A, B and C. The following sections explore some of the key ideas of ontology as it is used in disciplines that form the basis for the approach taken to the application of ontology to this work.

4.1.1.1 Traditional Ontology

The word ontology translates as *the study of what exists*. Ontology can be described as *the study of being in so far as this is shared in common by all entities, both material and immaterial. It deals with the most general properties of beings in all their different varieties* (Bunnin & Tsui-James 2003). As part of metaphysics, ontology is variously concerned with substances, properties, relations, events, times, places and states of affairs.

4.1.1.2 Systems Ontology

The traditional view of ontology as the study of that which exists provides an important jumping off point in the work. Drilling down on this idea in the domain of context on the World Wide Web, it is possible to start asking such questions as 'what are the things that are context in a hypermedia system?' and 'what are the relationships between the things that are context in a hypermedia system?'. These are important questions in the research, but it is not immediately apparent how these relate to the objective of explicitly representing context in hypermedia system. It is necessary to move from broad questions about the nature of context in hypermedia systems to a well-defined formal representation of context that can be applied to actual hypermedia systems.

This link from the traditional philosophical approach to a formal systems approach to ontology can be found in the work of Mario Bunge. Bunge states that *ontological frameworks are constructs intermediate in structure between shapeless views and closed hypothetical-deductive systems.* (Bunge 1977). The construction of an ontological framework of context is used to allow the structuring in a formal way of the various contextual elements that have been considered in hypermedia systems. In additional, the framework should be extensible to allow new ideas on context to be included. To cite Bunge again, *an ontological framework serves as a matrix for any number of ontological systems or theories* (Bunge 1977). While using an ontological approach to order existing approaches to context in hypermedia, it is recognised that new approaches may emerge over time. By constructing an ontological framework, it is hoped that these new approaches can be formulated into their own ontological system that, through the support provided by the ontological framework, can be incorporated into existing hypermedia systems.

Bunge defines an ontological framework as follows. *We shall agree to call an ordered triple C=<S,P,D> an ontological framework iff S is a set of statements in which occur only the predicate constants in the predicate family P, which includes a non empty set O of basic ontological concepts (i.e. categories) and the reference class of every p in P is included in the universe or domain D of hypothesised entities, or objects assumed to exist.* (Bunge 1977)

4.1.1.3 KBS Ontology

In Knowledge Based Systems research, ontologies are intensional descriptions of the domain knowledge in some field (van Heijst, Schreiber & Wielinga 1997). According to Gruber, an ontology is an explicit specification of a conceptualisation (Gruber 1995). The term is borrowed from philosophy, where Ontology is a systematic account of existence. For AI systems, what 'exists' is that which can be represented. When the knowledge of a domain is represented by a declarative formalism, the set of objects that can be represented is called the universe of discourse. Pragmatically, a common ontology defines the vocabulary with which queries and assertions are exchanged among agents.

4.1.1.4 E-Commerce Ontology

In business communication, especially e-commerce, it is important to ensure that the vocabulary used for communication is consistent. It is not uncommon for entities and processes that are equivalent are referred to with different names across different organisations. As a simple example, an invoicing system in one company may keep track of clients; another may keep track of customers. Logically, the two are equivalent. However, if a client record is sent from the first organisation to the second organisation, and the record is an XML document of type client, the second organisation will not recognise the semantics of the document. The business ontology approach being developed and promoted by Computer Sciences Corporation seeks to address this breakdown in communication by developing standard taxonomies for describing business entities. These standard taxonomies are labelled ontologies.

It is useful to note the approach taken to ontology here, as such standard taxonomies for information that are utilised in specific situations are quite similar to some of the ideas being developed in this work in regard to how context and information can be represented and utilised.

Ontology is also often used in e-commerce to construct exchanges in a way that allows semantic information about the contents of the exchange to be available to both users (Guarino et al. 1999) and to automated buying and selling agents (Durfee et al. 1998).

4.1.1.5 FOIS

There has been much attention from the knowledge engineering community to the use of ontology in the building of knowledge-based systems. Guarino and Giaretta summarise the various interpretations of ontology as follows (Guarino & Giaretta 1995):

1. Ontology as a philosophical discipline.
2. Ontology as an informal conceptual system.
3. Ontology as a formal semantic account.
4. Ontology as a specification of a conceptualisation.
5. Ontology as a representation of a conceptual system via a logical theory
    a. Characterised by specific formal properties.
    b. Characterised only by its specific purposes.
6. Ontology as the vocabulary used by a logical theory.
7. Ontology as a (meta-level) specification of a logical theory.

As Guarino and Giaretta note, definition four has been stated as a definition of what ontology is for the AI community. That is, that ontology is a specification of a conceptualisation.

In the same paper, Guarino and Giaretta describe a glossary of ontological terms. Given the varied interpretations of ontology applied to various areas of research, there is a clear imperative to be consistent in how the term is applied. The glossary of Guarino and Giaretta is as follows (Guarino & Giaretta 1995) :

conceptualisation: an intensional semantic structure which encodes the implicit rules constraining the structure of a piece of reality.

Formal Ontology: the systematic, formal, axiomatic development of the logic of all forms and modes of being.

ontological commitment: a partial semantic account of the intended conceptualisation of a logical theory.

ontological engineering: the branch of knowledge engineering which exploits the principles of (formal) Ontology to build ontologies.

ontological theory: a set of formulas intended to be always true according to a certain conceptualisation.

Ontology: that branch of philosophy which deals with the nature and organisation of reality.

ontology: (sense1) a logical theory which gives an explicit, partial account of a conceptualisation; (sense2) synonym of conceptualisation.

While the ontologies used in FOIS and KBS work are similar in construction and concept to the contextual ontologies this work considers, the focus in the domains tends to be on task analysis. Therefore, while seeking to draw on existing work as much as possible, it is important to be careful to ensure that the focus in this chapter is on modelling contextual elements in a task independent fashion. This means that the methodologies of KBS and FOIS ontology cannot be applied directly.

4.1.2 Goals of Ontology

The use of formal ontology in information systems is an area of growing interest. Five key benefits arise from representing information using a formal knowledge. While in this work, ontology is being used to represent information about the information entities that exist in an information system, these benefits still apply to the representation. Each of these benefits is discussed in turn below.

4.1.2.1 Completeness

Since an ontology will be developed to represent context in a domain, rather than just for a single application, the set of entities specified by the contextual ontology provided a useful reference for application developers working within that domain.

4.1.2.2 Consistency

The elements of an ontology allow us to refer to domain elements in a consistent way, reducing ambiguity and allowing for the construction of thesauri that allow the mapping of vocabularies between groups of specialists operating in the domain.

4.1.2.3 Formality

The rigour and structure imposed on the domain elements being described in a contextual model mean that assumptions about structure are minimised, and allow for the creation of standardised interfaces that work with the formal structure.

4.1.2.4 Reusability

Since the contextual models are well defined and can be mapped to different vocabularies, a model created for one application can be used by other applications that are aware of the ontological structure. It is also possible to provide mappings between ontological structures to support this kind of reuse.

4.1.2.5 Interoperability

Ontologies can be implemented as interfaces rather than as just data elements, so they can be made available as standalone components that can be readily plugged in to other systems or applications. It is also possible to create various tools that utilise ontologies without having to impact on existing tools or systems that are in place.

4.2 Modelling a Domain with Ontology

Web Ontology provides technique for modelling context. In order to develop the model this investigation uses a methodology that describes how to map the ideas from the concept of context in to a formal description. The discussion begins by describing the methodology that is used for creating the ontology. Next, it is shown how this methodology is applied to the concept of context. The discussion concludes with a description of the ontology.

4.2.1 Modelling Ontology on the World Wide Web

Various approaches have been taken to applying KBS approaches to ontology to problems of information management on the web. These are discussed below.

4.2.1.1 Ontobroker

Ontobroker is a broker architecture that has three core elements (Fensel et al. 1998). The first is a a query interface for formulating queries. The second is an interface engine to derive answers. The third is a web crawler used to collect the required knowledge from the World Wide Web. Ontobrokers ontological information is added to web pages as annotations that are defined as ontobroker specific extensions to HTML. In ontobroker, the ontologies used to mark up documents are fixed. This means the ontobroker tools will always be aware of the elements that are included in an annotated ontological description, and will thus be able to process them.

4.2.1.2 Ontolingua

Ontologies as defined by the Ontolingua project provide explicit specifications of domain conceptualisations (Farquhar, Fikes & Rice 1997). By providing a common vocabulary for the domain, they support the system developer, the system user, and system agents in sharing domain specific knowledge that lets them manage the system information better. The Ontolingua server provides a shared repository for storing ontologies, and various tools for creating and manipulating these ontologies. One of the reasons this support is provided in a distributed fashion is so that communities can build up ontologies describing domain conceptualisations quickly by reusing existing ontologies.

4.2.1.3 OntoSeek

OntoSeek is a system designed for content-based information retrieval from online yellow pages and product catalogs (Guarino et al. 1999). It uses a large linguistic database that encompasses both ontological and lexical information to structure the information in the catalog in a hierarchy, and to support the formulation of queries by end users. OntoSeek achieves its increases in effectiveness as an information retrieval system by

1. Decoupling the user vocabulary from the data vocabulary,
2. Exploiting the hierarchy to make generic queries and recognizing synonyms,
3. Providing the ability to navigate the hierarchy to select specific queries,
4. Considering the structure of queries and descriptions.

OntoSeek works by organising the information that is to be made available into an ontology-based hierarchy. This hierarchy allows the user to more effectively structure queries, and also allows the system to retrieve the information relevant to the query with both greater recall and greater precision.

4.2.1.4 SHOE

SHOE is an HTML-based knowledge representation language. SHOE is a superset of HTML which adds the tags necessary to embed arbitrary semantic data into web pages (Heflin & Hendler 2000). SHOE tags are used both to construct ontologies that specify assertions and their semantics, and to annotate web documents to define which ontology applies to the document.

A SHOE ontology declares categories for data entities, relations between data entities, inferences, inheritance from other ontologies, and versioning. These ontologies are then used to capture the semantics of web pages by embedding tags into the page that declare arbitrary data entities, declare the ontology being used, categorise entities, and declare relationships. This embedded markup allows for information about the document and its data entities to be stored in the page.

One of the criticisms levelled against embedded tags for representing semantics in a web page is that authors of existing pages will not add these new tags to their pages. The creators of SHOE counter this criticism by observing that an increasing percentage of pages on the World Wide Web are automatically generated and that by providing tools that support semantic annotation, this issue ceases to be a negative factor in the design of SHOE. While this argument has merit, the use of ontology in this work does not require any embedded content. Rather, the ontology is maintained in a separate data structure independent of the web content being viewed.

4.2.2 Creating an Ontology

Creating an ontology involves formally describing the concepts relating to a domain of knowledge. In order to ensure the concepts are all covered, and to ensure the process is clear and reusable, a defined methodology is followed to create the ontology. The methodology is described by an approach, which is a set of steps to be followed, and a set of principles that guide the decisions made by the ontology designer.

4.2.2.1 Approach

There are many approaches to constructing an ontology (Farquhar, Fikes & Rice 1997; Heflin & Hendler 2000). The methodology adopted in this work is based on the Simple Knowledge-Engineering Methodology described in Ontology Development 101 : A Guide to Creating Your First Ontology (Noy & McGuinness 2001). This methodology was chosen for three reasons. First, it is developed by the Knowledge Systems Laboratory group at Stanford who have been influential in the development of the Web Ontology W3C standard. Second, it is supported by a modelling tool and database implementation for Web Ontology. Third, it is based on principles that support the requirements for a model of context.

The methodology involves seven steps.

Step 1. Determine the domain and scope of the ontology. The domain of the ontology is the area of knowledge the ontology will represent. In the case, the domain is context in web-based systems. The scope of the ontology is concerned with how it will be used and maintained.

Step 2. Consider re-using existing ontologies. One of the benefits of ontologies is the reusability. There may already be an ontology that represents the domain of interest, or it there may be ontologies that represent part of the domain that can be integrated in to the ontology being developed.

Step 3. Enumerate important terms in the Ontology. Drawing on the domain and scope, the concepts the ontology is concerned with are identified. The concept of context informs this, along with the knowledge of web-based systems and information-seeking behaviours.

Step 4. Define the classes and class hierarchy. Using the concept of context, a top-down approach to defining the class hierarchy is applied. The terms identified in step 3 are taken from the concept of context and classified into a class hierarchy. This classification starts with the most general terms and finish with the most specific ones.

Step 5. Define the properties of classes – slots. Slots are the properties of a class. They are akin to the concept of attributes in OO design. In the approach, all the slots come from the concept of context. They will be terms that describe properties of the classes that have already been identified.

Step 6. Define the facets of the slots. The facets of a slot are constraints on the kinds of values the slot can contain. The most common facets are the type and the cardinality of values that can be assigned to the slot. The concept of context and the knowledge of web protocols are used to identify the facets that apply to the slots created in the ontology.

Step 7. Create instances. The ontology is intended to have its instances created automatically. Most of them will be added as the user carries out information-seeking behaviours. There are, however, a small set of instances that need to be in place to allow the system to begin operation. Step 7 is used to identify these initial instances and talk about how they will be managed by the context store system.

4.2.2.2 Principles

The Simple Knowledge-Engineering Methodology emphasises three fundamental rules. These rules act as principles that inform the development of the ontology.

> *1. There is no correct way to model a domain – there are always viable alternatives. The best solution almost always depends on the application that you have in mind and the extensions that you anticipate.*

> The application of this ontology is to help users perform information-seeking behaviours. When decisions are made about how to represent the elements that make up the concept of context it is necessary to look at how these conceptual elements might be used to help users find information. The design of the

ontology also aims to construct the class hierarchy in a way that will allow new elements to be added. This allows the ontology to be extended and re-used beyond the initial set of applications.

2. Ontology development is necessarily an iterative development.

The development of the ontology has been an iterative process. While the argument here is presented linearly, the process for arriving at the concepts presented here was iterative. In particular, the development of the concept of context was an iterative process going back and forward between hypermedia research, web protocol and information concepts. The development of the ontology was also iterative, with decisions about whether objects were classes or slots, and the structure of the class-hierarchy being refined numerous times by trial and error as attempts were made to construct instances that mirrored the application of the ontology.

3. Concepts in the ontology should be close to objects (physical or logical) and relationships in your domain of interest. These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe your domain.

The domain of interest is described by the concept of context. The concept of context describes the objects and relationships in which this work is interested. The concept of context has been developed to describe the objects and verbs relating to context that are also relevant to the intended application of the ontology. The concept of context is an informal description of these entities. The ontology is formal representation of these same concepts.

4.3 Modelling Context with Ontology

An ontology is used to represent knowledge in a domain. It does this by describing key concepts in the domain and relationships between these concepts, and how instances can be added to the ontology. This work applies the Simple Knowledge Engineering Methodology, as discussed in the previous section, to develop an ontology of context that captures contextual information about information-seeking behaviours on the World Wide Web. The ontology created here acts as a proof-of-concept that such an ontology can be created. It reflects one view of context in information-seeking behaviours, and is no way claimed to be authoritative or complete. Rather, it is intended to be sufficient to support further investigation of the hypothesis, and is also intended to spark further investigation into appropriate and useful ontologies that represent context.

4.3.1 Step 1 - Domain and Scope of the Ontology

Understanding and defining the domain and the scope of the ontology is the first step. The domain is the problem domain to which it is being applied and the knowledge domain that is being modelled. The scope defines which problems in the problem domain to which the ontology will be applied. As has been seen when creating the concept of context, it is not possible, nor desirable, to solve everything. Decisions need to be made that make the scope of the ontology finite.

The domain and scope of the ontology is described in the concept of context. The SKEM describes several questions that can be used to determine the domain and scope of an ontology.

What is the domain the ontology will cover?

The domain of this ontology is described in the concept of context. Essentially, the domain is contextual information in information systems on the World Wide Web.

For what are we going to use the ontology?

The ontology is going to be used to help users find information on the World Wide Web. This will be achieved by providing contextual information that helps the user make decisions about links and content while engaging in information-seeking behaviours. The ontology will be used by navigation aids that make contextual information visible to a user while they are browsing.

For what types of questions the information in the ontology should provide information?

Most of the questions that the ontology should answer are discussed in the concept of context. Some key examples are the following :

- What pages does the user visit most frequently?
- What pages does the group visit most frequently?
- What is the current resultset for the users query?
- What is the metadata for page X?
- What are the keywords for page X?
- What pages has the group visited that contain the word Y?
- What pages does the group frequently visit that contain the word Y?

Who will use and maintain the ontology?

The ontology will be used and maintained by the context store. The ontology should be constructed in a way that allows instances to be added without user input being required.

4.3.2 Step 2- Consider re-using ontologies.

Existing ontologies will not be re-used in this work due a lack of appropriate ontologies. Ontology re-use is an important benefit of modelling domains with ontology. There are a number of existing ontologies. These are documented in repositories such as SchemaWeb (Firefox), the DAML Ontology Library (Firefox) and Swoogle (Firefox). A review of these repositories did not find any ontologies that covered the domain that is being modelling in this work.

Where appropriate ontologies are available, two main benefits can be obtained by reusing them. The first is that the amount of design work can be reduced if part of the domain to be represented has already been modelled. Given the concepts that are being modelled, there is no real need to re-use existing ontologies. Second, re-using existing ontologies supports interoperability between applications that share ontologies. In this instance, there exists a specific application that this work wishes to support. Support for a wider range of applications could be added if it were thought it would be worth extending the ontology to be interoperable with other domain ontologies.

4.3.3 Step 3 - Enumerate important terms in the ontology

The list of important terms captures all the concepts that the ontology will model. For this list, it is possible to draw on the concept of context, as it already describes the concepts to be modelled. As this ontology is being developed as a proof of concept to investigate the hypothesis, the list of terms here is not claimed to be exhaustive or authoritative. It is one of many possible lists that could be constructed, each of which would lead to a different ontology. It is intended that further research into ontologies of context and their construction will flow from this work, but as ontology construction is not the focus of this work, an ad-hoc and informal process has been used to produce the list of terms below.

The following list contains the important terms from the concept of context, along with a brief description of each one. This list was obtained by reading through the concept of context and selecting nouns and verbs that relate to the application of finding information on the World Wide Web. The terms identified in this work are shown in Table 4-1 Important Domain Terms.

User	A person who uses the World Wide Web.
Group	A collection of users with some common interest.
Account	A specific computer system account from which a User accesses the World Wide Web.
Page	A web page to which the user of an account navigated.
Host	The hostname of a server from which a web page was loaded.
Query	A query submitted to a search engine
Query Term	A term submitted to a search engine as part of a query.
QueryEngine	A search engine to which a query was submitted.
QueryResult	An page reference that has been returned by a search engine in response to a query.
QueryResultSet	A collection of page references that have been returned in response to a query.
Contextual Element	A piece of information that describes a page or a query result set.
MetaTags	The contents of an HTML <meta> tag.
Keywords	Keywords for a page that were identified as such by the pages author using <meta> tags.
Title	The title of a web page or a query result set.
URI	The address of a page, a query result set, a query result, or a host.
Timestamp	The time at which an event occurred.
Host	The hostname for a page or a query result set.
Engine	The search engine to which a query was submitted.
ContentWord	A word contained as text inside the <body> tag of an html *page.*
Entity	Any web based piece of information for which context is being tracked.
Event	An activity performed by a user while engaged in an information seeking behaviour.
PageBrowse	Any activity where user browses to a web page
DirectedBrowse	An activity where a user browses to find a specific piece of information.
UndirectedBrowse	An activity where a user browses without a specific information need.
Search	Any activity where a user uses a search engine to look for information.
DirectedSearch	An activity where a user performs a directed search as defined by Choo.
UndirectedSearch	An activity where a user performs an undirected search as defined by Choo.
Notation	An annotation added to an entity by a user.
Favourite	Mark the page as a favourite.
Review	Mark the page for review
Tracking	Mark the page or host for tracking in the context store.

Table 4-1 Important Domain Terms

4.3.4 Step 4 – Define the classes and class hierarchy

This step involves the analysis of the terms from step 3 and the identification of which of these terms should be represented by a class. Additionally, this step involves the definition of the hierarchy for the classes included in the ontology.

In OWL every class is a specialisation of the base class owl:Thing. It serves as the root of any class hierarchy defined in OWL. Specialisations are understood as "is-a" relationships, similar to object-oriented software design. In the SKEM specialisations are referred to as sub-classes, and the relationship between a class and its sub-class is "kind-of". It is important to remember though that this process is creating a model of a domain of knowledge, not of a software system.

The classes are chosen from the terms based on principles derived from some of those described in SKEM (Noy & McGuinness 2001).

1. *Siblings should be at a similar level of generality.*

   Sibling classes act as a way of grouping different subclasses. The class hierarchy has been organised so that siblings are at the same level of abstraction. The sub-classes of each class make the ideas of that class more concrete. In particular, the Service class was added as a superclass of the User, Account and Group concepts because they were not at a more specific level of abstraction than Element, Entity and Event.

2. *Subclasses have additional properties and participate in different relationships than the superclass.*

   The hierarchy becomes more specialised its nodes are traversed top-down. This specialisation is reflected by the additional properties that indicate there is more to know about an object of this class.

3. *If concepts with different slot values become restrictions for different slots in other classes then w---e should create a new class for the distinction.*

   This principle is applied when trying to determine whether a concept from the knowledge domain should be modelled as a class or as a property of a class. For example, this approach has chosen to model concepts such as Title and Host as classes rather than as slots. This is because the intended application of the ontology wants to be able to use these concepts as restrictions on other classes.

Starting from owl:Thing it is possible to identify the following classes and hierarchical relationships for the terms identified in step 3. The concepts are listed in table 4-2. The domain concepts can be organised into a class hierarchy similar to those used in software engineering, and represented using a UML 2.0 diagram. This has been done in Figure 1.

Element	*is-a* owl:Thing that represents an object based property of an event or an entity.
MetaTag.	*is-a* Element that belongs to an event or an entity.
KeyWord	*is-a* MetaTag that has the name "keyword".
Title	*is-a* Element that belongs to an entity.
Host	*is-a* Element that belongs to an entity.
Engine	*is-a* Element that belongs to an entity.
ContentWord	*is-a* Element that belongs to an entity.
Entity	*is-a* owl-Thing that represents the kinds of the ontology is aware of.
Query	*is-a* Entity.
ResultSet	*is-a* Entity.
WebPage	*is-a* Entity.
Event	*is-a* owl-Thing that represents the user activities of which the ontology is aware.
PageBrowse	*is-a* kind of user activity.
DirectedBrowse	*is-a* kind of PageBrowse
UndirectedBrowse	*is-a* kind of PageBrowse
Search	*is-a* kind of user activity.
DirectedSearch	*is-a* kind of Search
UndirectedSearch	*is-a* kind of Search
Notation	*is-a* kind of user activity.
Favourite	*is-a* kind of Annotation.
Review	*is-a* kind of Annotation.
Tracking	*is-a* kind of Annotation.
Service	*is-a* kind-of owl-Thing that can produce or consume contextual information.
Account	*is-a* kind of service.
User	*is-a* kind of service.
Group	*is-a* kind of service.

Table 4-2 Domain Concepts

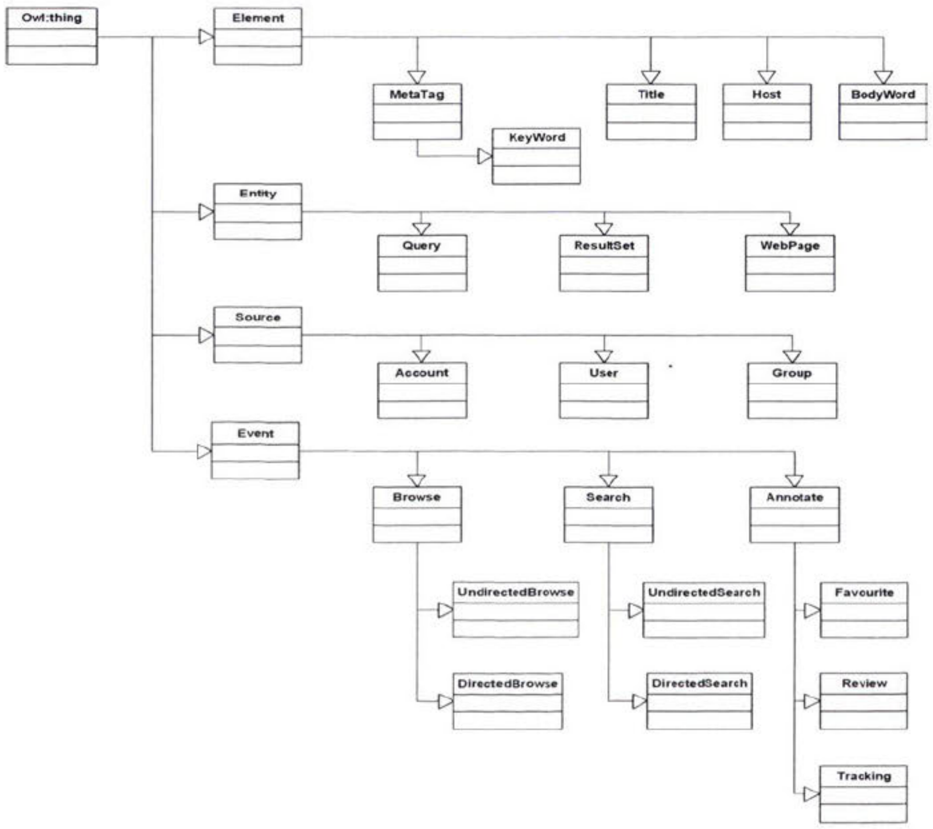


Figure 4-1 Representation of Concept Associations (UML 2.0)

4.3.5 Step 5 - Define the properties of classes – slots

Properties describe the internal structure of the concepts represented by classes. In OWL, properties are also referred to as slots. Properties of concepts in OWL play a role similar to the role of class attributes in OO software design.

The primary candidates for properties are the terms from step three that were not represented as classes. As an example, in the case, 'tagname' and 'tagvalue' are terms from the concept of context that are not meaningful as concepts in their own right, but are meaningful as part of the metatag concept. They are properties of the metatag because they define the structure of the concept. A metatag has a tagname and a metatag has a tagvalue. All terms from the domain of knowledge that were not meaningful as classes are analysed to determine the classes of which they should be made properties.

4.3.6 Step 6 – Define the facets of slots

Facets are constraints on the kinds of values that can be assigned to a slot. In the ontology, the following facets are used:

1. *Slot Cardinality.* This is the number of values that can be assigned to a slot. OWL supports the specification of MaxCardinality and MinCardinality. MaxCardinality is the upper limit on the number of values, while MinCardinality is the lower limit. It is also possible specify multiple-cardinality, which means the slot can have any number of values.

   In the ontology of context the choice has been made to either restrict the cardinality of the slot to 1, or allow the slot to have multiple-cardinality.

2. *Slot value-type.* The value-type of a property is the kind of information that it contains. In OWL properties can either be object properties or datatype properties. Object properties contain an instance of one of the classes defined for the ontology. Datatype properties contain a value those type is one of the basic OWL datatypes, like a string or an integer.

3. Domain and range of a slot. The domain of a slot is the set of class to which it is attached. The range of a slot is the set of classes that can be assigned to the slot as a value. Web Ontology treats slots as entities in their own right that are mapped to classes, rather than being

4.3.7 Step 7 – Create instances

In the application, the instances of the ontology will be created dynamically by the context store application. A users context evolves over time as they engage in information related behaviours, and so the ontology instance will be populated over time. How this contextual information is identified and added over time is discussed in chapter 6.

4.3.8 Summary

The process that has been used to construct an ontology of context based on the concept of context has been described. The complete ontology is documented in Appendix A. The construction of such an ontology using a re-usable approach shows that it is possible to build an ontology of context, satisfying one of the research questions.

4.4 Conclusion

One of the key goals of this work is to use context to help people find information on the World Wide Web. In order to do this, it is necessary to be able to represent context in a way that allows it to be used with web-based tools. An ontology developed using OWL provides a reusable, extensible way of modelling context that meets the requirements. In this chapter, it has been shown how such the approach used to develop an ontology representing the concept of context that was developed in chapter three. The full ontology is shown in Appendix A.

The ontology that has been developed allows us to store context for a user or a group of users. It has been shown that ontology is appropriate for modelling context by establishing a set of requirements for modelling context and then showing that the features of web ontology meet these requirements by developing an understanding of ontology. It has then been shown how an ontology of context can be created by working through the principles of developing web ontologies and showing how these can be applied to the concept of context. The result is an ontology that can be used to represent context in web-based systems.

The next step in exploring the hypothesis is to show how this ontology of context that has been developed can be used to help users find information on the World Wide Web. This is the subject of the next chapter.

5 Contextualisation

Contextualisation uses contextual information to augment the interpretation of information. The ontology of context developed in the previous chapter provides a structured way in which to store contextual information relating to information-seeking behaviours on the World Wide Web. This chapter investigates how this contextual information can be used to contextualise information viewed while browsing and/or searching the World Wide Web.

The approaches to contextualisation in this chapter are by no means the only approaches to contextualisation. The purpose of the discussion in this chapter is to show that the ontology of context that has been developed can be used to contextualise information. This chapter does not claim to be developing new approaches to contextualisation. Rather, the purpose of this chapter is to show how the ontology of context can be applied to contextualising information on the World Wide Web. It is expected the ontology of context and the software framework developed later will serve as a platform for further research into approaches to contextualisation.

The approaches to contextualisation are described in terms of scenarios. Scenarios are used to describe the information-seeking behaviours that will be supported using contextual information from the ontology. The scenarios used in this work are constructed using Scenario Based Design (SBD). SBD is used to describe the activities that a software system is designed to support (Rosson & Carroll 2001). Developing scenarios based on information seeking behaviours provides a basis for developing a software framework that uses the ontology to support contextualization.

5.1 Introduction

Having developed an ontology of context, this chapter looks at how user interfaces that use this ontology to contextualize web-based information can be implemented. It begins by identifying a set of scenarios that inform the design of appropriate context-based user interfaces. These scenarios are based on example instances of the information-seeking behaviours. It then describes how these scenarios are constructed. Next, it describes a set of scenarios that express the kinds of behaviours to which this work will apply contextualisation. It then shows how these scenarios can be supported by querying contextual information from the ontology. Finally, this chapter describes the user interfaces that provide contextualization in the scenarios described using a task-based analysis methodology.

The application of context to helping users find information on the World Wide Web by investigating the following research question is described.

Q7. How can ontology of context be used to contextualise information?

The ontology of context can be queried to retrieve information that contextualizes a web page or a search result. Ontology of context provides a way of storing contextual information and reasoning about this information. This information and inferences made from it can be made visible to users and help to contextualize a web page or a query result set. Contextual information is presented to a user through a user interface.

5.2 User Interfaces

Shneiderman (1992) describes user interfaces (UI) as software components that help a user perform a task. The phrase 'user interface' is used to be consistent with information retrieval literature which discusses similar components (Hearst 2000). Other literature at times refers to such components as tools. In this work, a choice has been made not to refer to these components as tools because they do not generate an output. In software engineering and computer science tools refer to programs that produce an output, such as software development tools that produce other programs, or text processing tools. User interface more accurately reflects that the components help a user perform a task, rather than perform a task themselves.

User interfaces in this work are software components that make contextual information available to the user. The contextual information is retrieved from the ontology of context while the user is performing an information-seeking behaviour. The contextual information is retrieved from the ontology using a query to support the user during one of the scenarios that has been described. The user interface presents the contextual information to the user in a way that is useful and easy-to-use.

In this section existing user interfaces that exist in web-based systems are reviewed. This review focuses on user interfaces integrated with browsers but also looks at desktop tools that operate stand-alone. Next, user interfaces from other information-related domains such as information retrieval to identify alternative approaches to their design and construction are reviewed. Based on these reviews an approach to designing user interfaces that use an ontology of context to help users find information on the World Wide Web is presented. Finally, this approach is applied to developing user interfaces for each of the scenarios that have been presented.

5.2.1 Side Bars

Sidebars are UI components that provide a display area inside the UI frame of a browser, next to the content pane. Side bar is the name used in Firefox. In Internet Explorer these components are called Explorer Bands. They are commonly used to provide a navigation view alongside the content currently being viewed, similar to the common split screen mode used in Windows Explorer or MS Outlook. The most common of these sidebars are the history and the favourites side bars, which are available in both IE and Firefox. Additionally IE has sidebars for search, researching a topic, and accessing media files through the browser. Developers can implement their own sidebars for either browser. In Internet Explorer, explorer bands are registered COM objects (Microsoft 2001). In Firefox sidebars are implemented using XUL (Oeschger et al. 2002).

5.2.2 Toolbars

Toolbars are a popular user interface for extending the functionality of browsers. Popular examples of toolbars are the Google toolbar, the Yahoo toolbar, and the A9 toolbar from Amazon. They provide easy access to functionality provided by these sites. They are software components loaded by the browser at run time that are anchored at the top of the browser in the UI frame with the built-in toolbars. They are outside of the content rendering region and thus do not overlay any content. They are a familiar and useful way to provide new functionality to users. One important caveat with the use of toolbars is to be careful of overuse. They extend the size of the UI frame area at the expense of the content region, and so a user with many toolbars loaded can find themselves with a much reduced region in which to view content. A secondary issue is browser performance. Poorly constructed toolbars can have a major impact on the performance and stability of a browser. In Internet Explorer, a COM-based API is exposed that supports the implementation of toolbars and other extensions. In Firefox, toolbars are implemented using the XML User-interface Language (XUL).

5.2.3 Browser Extensions

Firefox supports the development of UI extensions. These extensions allow developers to add new functionality to the browser. There is a large number of extensions available for Firefox (Firefox 2006a). The kinds of functionality these extensions implement falls into many categories. Categories of particular interest in this work are bookmark extensions, navigation extensions and search extensions. Internet Explorer also has a wide array of extensions available.

5.2.4 Desktop Search

The Google desktop provides a search service on the users computer desktop. All the files, emails and chat sessions are indexed locally and can be searched by the user. There are several entry points to the Google desktop search service. One is a locally hosted form that is displayed in the default browser. Another is a floating deskbar. Another is a deskbar hosted in the taskbar of the machine. A final entry point is the Google sidebar. This sidebar is another deskbar that docks to the closet side of the desktops display area. It is always visible over other applications. It provides access to local search, web-based search, and various server-based information sources pushing information like weather and news down to the sidebar for display.

5.3 Scenario Based Design

A scenario describes a user activity (Winckler, Palanque & Freitas 2004). In software modelling using UML, scenarios are often referred to as use-cases (Bittner, Spence & Jacobson 2002). This work chooses to label them scenarios because at this stage they are not being created as part of the software model. Rather, they will serve to inform the software modelling process.

Scenario Based Design is a methodology for designing user interfaces

(Carroll 1995). It is based around identifying scenarios that describe the user behaviour that the software being designed supports. Scenarios are an effective way of describing instances of information seeking behaviours.

The design of the user interfaces that will use the ontology of context follows an approach based on SBD. There are five phases of Scenario-Based Design as shown in Table 5-1 Scenario-Based Design Phases. These phases are applied to the domain of contextualisation in the following sections.

Scenario-Based Design Phases
Identify Scenarios
Activity Design
Information Design
Interaction Design
Prototyping & Evaluation

Table 5-1 Scenario-Based Design Phases

5.3.1 Identify Scenarios

The first step of scenario-based design is to identify the scenarios that are of interest in the problem domain (Rosson & Carroll 2001). A user interaction scenario is a story about people and their activities. In SBD the problem scenarios are the outcome of analyzing the problem domain. Factors considered are the activities the user engages in, the artefacts that are involved in the activities, and the social context of the user. The first step in developing scenarios is determining the key problems in the domain of interest. The next step is to develop scenarios that describe how users currently approach the solving of these problems. Finally, claims are made about how these problem-solving approaches can be improved with software.

Scenarios are identified that describe the ways in which users will use the system. In this case, the scenarios described are those that benefit from having the user being able to view contextual information.

The scenarios are created as instances of the information behaviours that were identified in chapter three. Table 5-2 recaps the information behaviours discussed in chapter three. Table 5-3 lists some scenarios that were identified as being examples of these behaviours.

Behaviour	**Description**
Directed Search	In directed search, a specific set of data is being searched to satisfy a specific information need.
Directed Browsing	The user browses a specific information store to locate information that satisfies a specific information need.
Undirected Browsing	The user browses unspecified sources for an unspecified information need.
Undirected Searching	The user searches an unspecified set of information sources for a specified information need.

Table 5-2 Information Seeking Behaviours to support

Behaviour	**Scenario**
Undirected Browsing	View Page Info
	View Related Pages
Directed Browsing	View Hotlist
	View Dynamic Favourites
Undirected Search	Browse Result Set
	Search Result Set
Directed Search	Browse Context Store
	Search Context Store

Table 5-3 Information Seeking Scenarios

The following scenarios describe interaction behaviours that can be implemented using the ICU context model. These interactions are not necessarily unique to this system, but rather serve to show how the context model for implementing a wide range of commonly used interactions. The fact that a variety of interactions can be supported illustrates the usefulness of the contextual model.

Problem Scenarios

These scenarios describe the user interactions implemented by the useApp applications. The problem scenarios are described as use cases (Bittner, Spence & Jacobson 2002). Table 5-4 Problem Scenarios for Behavioursshows a list of problem scenarios that illustrate the kinds of information-seeking behaviours being discussed in this work. These scenarios are the targets for implementation. Two scenarios were not chosen for further investigation due to the complexities of their likely implementation. View Dynamic Favourites was omitted because the intended implementation platform (Internet Explorer) did not have an API for managing favourites or trapping favourite related events. Browse Context Store was omitted because a clear interaction paradigm was not readily available, and creating one was considered outside the required scope of this work. However, designing such an interaction is an area that has been flagged for further investigation. Further detail about these scenarios can be found in Appendix D.

Behaviour	**Scenario**
Undirected Browsing	View Page Info
	View Related Pages
Directed Browsing	View Hotlist
Undirected Search	Browse Result Set
	Search Result Set
Directed Search	Search Context Store

Table 5-4 Problem Scenarios for Behaviours

5.3.2 Activity Design

Activity design involves describing the functionality of the software being developed. Scenario Based Design defines activity scenarios as *"narratives of typical or critical services that people will seek from the system."* (Rosson & Carroll 2001). These scenarios focus on the functionality of the software and not on the user interface or the implementation details. The activity scenarios provide the basis of what the software will do. Each scenario is implemented by one user interface component.

When designing activity scenarios in Scenario Based Design the following key principles are adhered to.

> *Effective*. The scenarios should directly meet the need and goals of a user.
>
> *Comprehensible.* The scenarios should be clear. Their application to the problem domain and their usefulness should be readily apparent.
>
> *Satisfying.* The scenarios should improve the experience the user has when using the system to carry out tasks.

An activity in SBD describes the steps that occur when a particular scenario occurs (Rosson & Carroll 2001). Activities are similar to the idea of use cases in UML (Bittner, Spence & Jacobson 2002).The following activities can be described for the scenarios that were identified in the previous section. The activities associated with the problem scenarios shown in Table 5-4 Problem Scenarios for Behaviours are presented in appendix D.

5.3.3 Information Design

Information Design develops details about the information that the system will provide to the user during a scenario. The focus is on how the information is organized and displayed to the user. There are three principles that guide information design.

Perception. What are the graphical structures that will be meaningful to the user in the scenario.

Interpretation. What are the graphical elements that make up the vocabulary that will be used to present information to the user.

Making Sense. Ensure that the vocabulary and structures used are meaningful and useful to the user in the scenario.

Information design is concerned with how information will be presented to the user during the scenario. Three perspectives are applied to information design. Perception looks at the graphical elements that will be used to display information. This can be viewed as the vocabulary of the scenario. Interpretation looks at how the graphical elements can be organized on the display to convey meaning. This can be thought of as the grammar of the display. The final stage, making sense, is concerned with how meaning is conveyed using the artefacts produced from perception and interpretation analysis. It can be thought of as the semantics of the display.

5.3.3.1 Perception

The information elements used are common HTML controls such as Headings, URL's, Input Boxes and Buttons. The behaviour of the HTML elements is described by the HTML specification (Raggett, Le Hors & Jacobs 1999).

5.3.3.2 Interpretation

The following structures were identified based on the interpretation analysis. The activities of the scenarios were analysed, and also the elements used by Mozilla and IE navigation aids and extensions.

Lists	A list is a collection of URL's with an optional border and an optional header.
Text Boxes	A text box is a collection of labels with an optional border and an optional header.
Toolbars	A toolbar is a collection of buttons, select boxes and input boxes.
Sidebars	A sidebar is a collection of any elements, and usually has at least one list.

Table 5-5 Graphical Structures for Browsing

5.3.3.3 Making Sense

Making sense of information design means applying a set of principles to how information layouts for the scenario are constructed. In general, the goal of this stage is to ensure that information displays visible to the user are organized in such a way that the meaning will be readily apparent to the user. This stage draws heavily on existing browser user interfaces to inform the kinds of layouts that can be applied to contextual information. The principles applied from SBD are as follows.

1. Consistency. Consistency means using elements and items in a way such their meaning is the same regardless of the scenario. A URL should always look and behave the same way no matter what it is being used for. If different functionality or behaviour is required, a new element should be created rather than overloading the existing element. In

addition to consistency across scenarios, an aim in this work is consistency with existing applications. So, for example, the text of a URL is highlighted when a mouse moves over it and change the mouse cursor. This is instantly recognized by users of browsers as signifying that clicking on this element will cause a new page to load.

2. Models. The way information is organized conveys meaning to the user. In design of user interfaces, the information model is restricted to lists. This is to minimize the about of cognitive effort required by the user to interpret models. It also means that the user interface only needs to be able to render one kind of data model. In a richer commercial-based system, a wider set of data models would be supported. For the purposes of this work, the list model is sufficient.
3. Dynamic Displays. the use of dynamic displays in the design effort is limited to tooltips and select boxes popping up information over the current display, and also having text boxes that update based on which URL currently is under the mouse cursor.
4. Metaphors. Metaphors use visual similarity to real world objects to convey meaning of information. For example, tooltips are similar to post-it notes or scraps of paper stuck to a page to add extra information to that page.

5.3.4 Interaction Design

Interaction design is concerned with how the user will interact with the software while performing the task. This design phase specifies the mechanisms for accessing and manipulating tasks information. In information design, the mechanisms for presenting information the user are described. In interaction design, the mechanisms for accepting input from the user are specified. These mechanisms can be as simple as selecting options for the UI or as complex as filling out forms. These interactions are described using an action sequence. The interactions for the scenarios in Table 5-3 Information Seeking Scenarios are included in Appendix E.

5.3.5 Prototyping & Evaluation

The final phase of Scenario Based Design is prototyping of the software and evaluating whether the prototype achieves the goals for which it was designed. A prototype is a concrete but partial implementation of a system design. Prototypes may be created to explore many questions during system development—for example, system reliability, bandwidth consumption, or hardware (Sommerville 1996). In scenario-based design prototyping and formative evaluation are performed iteratively to refine the system. Once the system is thought to be complete, summative evaluation is performed to determine the results of the system (Rosson & Carroll 2001).

In this approach, an informal prototyping and evaluation cycle has been followed. The software components were developed and evaluated by analytic inspection. Modifications were made in an iterative fashion until components had been developed that were believed to be useful for contextualising information on the World Wide Web. These components are described in more detail in later chapters.

5.4 Conclusion

There are many ways to contextualise information. One approach is to make contextual information available to a user through a user interface. This chapter has discussed how contextual information from an ontology of context can be displayed using user interfaces to support users engaging in information-seeking behaviours.

Context helps users find information on the World Wide Web. The discussion has focussed on scenarios that describe information-seeking behaviours that benefit from contextualisation. The scenarios are based on existing approaches to contextualisation, and the user interfaces described are based on commonly used user interfaces such as history mechanisms for browsers. The goal of this discussion is not to develop new user interfaces or interactions to support information-seeking behaviours. Rather, the goal is to show how the ontology of context can be used to contextualise information.

Contextual information is presented to the user through a user interface. To demonstrate how the ontology of context can be used to contextualise information during information-seeking behaviours on the World Wide Web, this work develops a small set of user interfaces based on existing approaches to contextualisation in web browsers. The purpose of these interfaces is to demonstrate the utility of the ontology of context. The research question underpinning the discussion in this chapter was Q7, which asks *'How can ontology of context be used to contextualise information?'*. The approach taken in this work is to contextualise information by presenting relevant contextual information in user interfaces.

The following chapter develops a software framework based on the ontology of context that supports approaches to contextualisation such as the ones described in this chapter.

6 Framework and Implementation

This chapter discusses how the ontology of context can be used to contextualise information on the World Wide Web in conjunction with existing web technologies. The discussion consists of two parts. The first part of the discussion describes a software framework that supports an ontology of context such as the one developed in chapter four. The second part of the discussion describes an implementation of an application based on this software framework. This implementation uses the ontology of context to make available contextual information that, using the concepts developed in chapter five, contextualises information on the World Wide Web.

6.1 Framework

One of the key motivations of this work is the belief that it should be possible to share contextual information between applications. There are many approaches to contextualisation in existing applications, and this work does not claim the idea of contextualisation. What is claimed to be new in this work is the ability to share contextual information between applications. In order to support this sharing of contextual information it is necessary to move beyond approaches that tie contextual information to specific applications, and describe an approach to managing contextual information that is decoupled from the end application. This kind of decoupling can be achieved in software through the use of software frameworks. The focus in this section is to show how a software framework that supports contextualisation using an ontology of software can be created. The ability to create such a framework is demonstrated by creating a software framework that supports information-seeking behaviours by interoperating with existing web infrastructure.

6.1.1 Application Frameworks

Frameworks are an object-oriented reuse technique. An application framework is a way of creating reusable design elements that can be used to create applications that solve a problem in a specific domain. In this work, the domain of interest is contextualising information in web-based systems. The purpose of creating a framework is to allow the ideas that are developed in this work to be applied to problems beyond the scope of this work. A key principle in developing ontologies is that they be reusable. To fully realise the reusability of the ontology the framework that is developed allows the ontology and the software associated with its use to be reused for applications beyond those developed here. This section examines what frameworks are, why they are useful, how they can be developed, and how they can be applied to developing applications.

6.1.1.1 Understanding Application Frameworks

Frameworks are a software engineering technique whose purpose is to decrease the cost and increase the quality of software. They achieve this purpose by capturing the structure and behaviour of applications in a specific problem domain for reuse in other applications in the same domain (Gamma et al. 2002). Frameworks have been applied to a wide range of software across domains as diverse as Operating Systems, Communication Systems, Financial Systems, User Interfaces, and Knowledge Based Systems (Fayad & Johnson 1999). A framework is a reusable, "semi-complete" application that can be specialized to produce custom applications (Fayad, Schmidt & Johnson 1999). A model framework is a collection of reusable designs and specifications. A framework is a generic package; it is applied by importing its package and substituting problem specific elements for the generic model elements as appropriate. There are various understandings of what a framework is. For example, the following are definitions of the concept of a framework.

- *A framework is a reusable design of all or part of a system that is represented by a set of abstract classes and the way their instances interact.*
- *A framework is the skeleton of an application that can be customised by an application developer.*
- Framework. *A set of cooperating classes that makes up a reusable design for a specific class of software. A framework provides architectural guidance by partitioning the design into abstract classes and defining their responsibilities and collaborations. A developer customises the framework to a particular application by subclassing and composing instances of framework classes.(Gamma et al. 2002)*

Developing a framework for contextualising is a way of making explicit the concepts and behaviours that have been modelled so far in this work. The framework serves two purposes. First, it serves as a way of documenting how the concepts of context, ontology and user interfaces can operate together. Second, it shows how software systems that implement these concepts can be constructed.

6.1.1.2 Using Application Frameworks

An application framework is, basically, a design reuse technique. It draws on the common design elements of a set of applications and shows the common structures and interactions that need to be supported for these applications to function. Frameworks have been described as recipes for creating applications. For example, when a developer wants an application that presents context to a user, they can either start from scratch and figure out how to build one, or they can follow a recipe that tells us which components need to be present and what interfaces they should support. The developer still has to implement these components, but the amount of effort required overall is reduced by using the framework.

In this work, the benefits of Modularity, Reusability, Extensibility and Inversion of Control that result from using frameworks (Fayad, Schmidt & Johnson 1999) have influenced the decision to develop an application framework for contextualisation.

Modularity. A framework is built using a set of abstract classes to define interfaces. The implementation of these classes is independent of the framework. A framework decomposes the system into a set of components that interact with one another through well-defined interfaces. The implementations of the components are independent of one another. This allows different implementations to be provided and even different behaviours within components to be supported so long as the interfaces are honoured.

Reusability. The use of interfaces and abstract classes in the description of a framework encourages reusability. It identifies the design elements common to all instances of an application and defines their structure and how they work together. Additionally, a useful side effect is code reuse where a well-designed framework will mean that an implementation of an abstract class for an instance of a framework application will be able to be reused in other instances of the framework where the same behaviour from the component is required.

Extensibility. Instances of a framework application can extend the abstract classes in their implementation to provide functionality beyond that defined by the framework. The framework provides the template for an application, but the scope of the application is not restricted to the template. Mechanisms for extending the framework include the use of hook methods, and subclassing abstract classes.

Inversion of Control. Inversion of Control means that the framework acts as an event dispatcher. Rather than each application determining which methods of its components to call and which behaviours to execute, these decisions are made by the framework. This inversion of control away from the application components enforces the independence of the components as the framework takes responsibility for maintaining information that must be shared between components. Inversion of Control is enabled by having the framework call methods of implementation classes in response to events that occur.

6.1.2 Designing the ICU Framework

This framework divides the problem domain of contextualisation into three distinct activities. The key concepts in this framework are that context must be Identified, it must be Collected, and it must be Used. This work refers to the application framework based on this model the ICU Framework. In order to describe these activities in terms of the problem domain the ICU paradigm is developed. The ICU paradigm describes the problem domain in terms of a model that can serve as the basis for a software-based application framework.

6.1.2.1 The ICU Paradigm

The ICU Paradigm describes a way of performing contextualisation. It describes in a generic way a set of entities that participate in contextualisation and explains how these entities interact with one another. The paradigm does not describe how a software system can be constructed. Rather, it describes the entities that a software system must implement and the interactions between the entities that the system must support.

The ICU Paradigm breaks the process of contextualisation into three distinct sub-processes. These activities can be performed independently of one another. The process of identifying context is separate from those of storing and using context. The process of storing context is independent of the processes of identifying and using context. The process of using context is independent of identifying and storing context.

When identifying these processes as being independent of one another, they are independent in the sense that in order to identify context it is not necessary to know how it is going to be stored or how it is going to be used. There will be system level dependencies between these processes, as they need to be able to pass data between one other, and the data structures to be exchanged need to be known. Each process of any type acts as a black box, executing the appropriate process and passing data between processes as necessary. An overview of the interaction between these processes is shown in Figure 6-1.

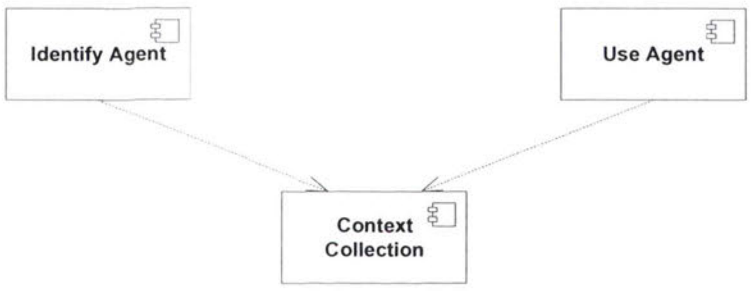


Figure 6-1 ICU Architecture (UML 2.0)

In the ICU Architecture there a couple of significant points worth noting. First, it shows that context is identified by one or more sub-processes. Context can be extracted from various sources, and the one source can yield different kinds of contextual in formation. For example a browser can yield contextual information about page visits and also contextual information about the pages a user has identified as being favourites. By using separate processes for each kind of contextual information that is being identified, it is possible to incorporate new kinds of contextual information into the identification activity.

Second, ICU Architecture shows that context is used by one or more sub processes. As with the identify activity, this is because context can be used in various ways. It can be presented in different ways, and its presentation UI's can be either stand-alone or integrated with various applications.

Finally, the ICU Architecture shows a linear flow from Identify activities to Store activities to Use activities. There is no direct interaction between the identify activity and the use activity. Contextual information is identified, stored, and then used. Each activity has a distinct role to play in the process of contextualisation. These distinct boundaries between the activities arise from the concept of context that has been developed previously, where the use of contextual information depends only on the end-user and not on the source of the information.

In order to develop the ICU Framework, each of the activities identified in the ICU paradigm can be broken down further.

6.1.2.2 Identify

The identify component consists of two key sub-activities. The first is the IdentifyMonitor. The IdentifyMonitor registers with appropriate dispatch interfaces of the operating system and with other applications to receive notification of events. When these events occur the IdentifyMonitor retrieves information about the event and dispatches this information to the StoreConnector. The StoreConnector assembles the information into a message and dispatches it to the Context Collector.

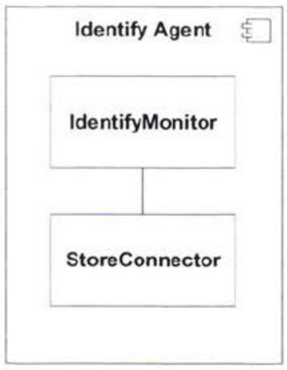


Figure 6-2 Identify Component (UML 2.0)

6.1.2.3 Collect

The collect activity consists of two main sub-activities, as shown in figure 6-3. It handles incoming messages containing updates and requests, and it manages the ontology containing the contextual information. Incoming messages are processed by the Message Service. Updates are validated and routed to the Context Store. The Context Store converts requests and updates to the ontology into queries that are executed against the ontology. It then assembles responses before sending them back to the Message Service to be handled appropriately.

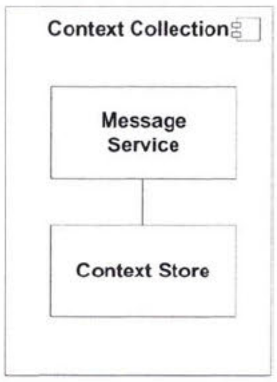


Figure 6-3 Collect Component (UML 2.0)

6.1.2.4 Use

The use activity is the most complex activity for two reasons. First, it has to display contextual information to the user, and it has to be able to respond to user input. Second, unlike the identify activity, the use activity relies on the responses from the store for its function. The structure for the Use activity is shown in figure 6-4.

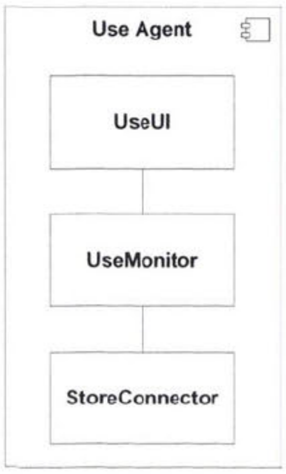


Figure 6-4 Use Component (UML 2.0)

The UseUI provides both the mechanism for presenting contextual information to the user, and the mechanism for accepting input from the user. Input typically takes the form of links being clicked or query strings being entered and an action being selected. Either the UseUI can run as a standalone component, or it can be hosted by a third party. This could be a browser such as Internet Explorer, or a standalone application. Other hosts are also possible.

The UseMonitor identifies events occurring on the system or in applications that require the UI to be updated. It can register with other applications to receive events, as well as being aware of internal events. The kinds of events will typically be new pages being loaded or the user clicking a link in a page. Events can be user generated or they can be application generated.

The StoreConnector provides an API that can be used to wrap messages in a SOAP envelope and send these messages to the Collection Service. Abstracting the StoreConnector allows it to be reused as well as allowing the UI and Monitor to be developed with minimal dependency on the communication mechanism.

The ICU paradigm is a model of the contextualisation process. It provides the conceptual basis for the construction of a software-based application framework that supports the development of applications that use context to help users find information on the World Wide Web.

6.1.3 Supporting the ICU Paradigm with Web Services

The ICU paradigm describes a collection of components that communicate with one another. The ICU paradigm can be described using a Service Oriented Architecture. A service oriented architecture describes a software system as a collection of loosely coupled software agents that interact with one another (Erl 2004). The interaction is between services and client applications. Another way to describe an SOA is simply as a collection of services and agents that consume these services. The communication between these entities occurs via messages. A service-oriented architecture has four key characteristics.

1. The messages exchanged are descriptive rather than instructive. Agents send messages to services specifying what information they want, but not how to generate it. A practical example is that a message should contain the parameters for a database query, but not the query itself. Sending an SQL query from an agent to a service breaks this principle and breaks the loose coupling between the two. If the implementation of the data layer changed from an SQL database to an OO database, a message containing parameters will still be handled, but a message containing SQL may not be appropriate.

2. The messages are written using a well-defined vocabulary. The

interface the service publishes defines what functions are available to calling agents. The messages sent between agents and services must be clear and unambiguous so that the receiver of the message is able to correctly process it.

3. Messages and services are extensible. The ability to easily add new services and messages to a system is an important reason for using an SOA. Message extensibility covers both being able to add new classes of messages, and the ability to extend existing messages. The use of XML based protocols for messages means that the key to extensible messages is to design the vocabulary and the semantics of the message in a way that allows additions to be made. Service extensibility means being able to add new services to the system without affecting existing services, and being able to add functionality to existing services without breaking existing agents.

4. Services are discoverable. A feature of service-oriented architectures is that the services provided by the system are available to be consumed by any agent that invokes the published interfaces using messages that match the vocabulary of the architecture. The architecture does not constrain how agents behave nor how services behave, only the interfaces and messages they use to communicate. This means agents that have different behaviours and purposes can be created independent of the services being specified.

An SOA is developed by identifying the services to be provided by the system and a vocabulary of messages that will form the communication between the services and agents. The Store can readily be described as a collection of services that are consumed by Identify agents and Use agents.

6.1.4 Creating the ICU Framework

This section describes the ICU Framework in terms of several standard software views. First, it describes the architecture of the framework. Then it describes the subsystems that exist in the framework. Next, it describes the classes that make up the framework. Once it has developed these elements, it describes the interfaces that allow the subsystems of the ICU framework to communicate with one another. Finally, it describes the behaviours that these interfaces support.

6.1.4.1 Architecture

An architecture acts as a blueprint for mapping the concepts of a model to a software system. In order to create the architecture the components that model the entities described in the ICU paradigm are identified, and then describe the interactions between these components.

6.1.4.2 Subsystems

The next step is to identify the loosely coupled subsystems in the SOA. In this case, there are three clear subsystems, these being the Identify subsystem, the Use subsystem and the Store subsystem. The Store subsystem is a collection of web services that encapsulate the ontology of context and support operations being performed on the ontology, in particular adding contextual information and retrieving contextual information. The Identify and Use subsystems are collections of agents that interact with the Store subsystem.

The Store is intended to be run as a web service. Web Services can be run either as stand-alone services by having the service instantiate its own http listener and manage its own lifecycle, or they can be included in application server which manages their lifecycle and routes http requests to the service. The service itself should be constructed to be agnostic of which approach is taken. In particular, its design should not be dependent on mechanisms specific to a particular application server or web service API. Additionally it is necessary to avoid platform dependencies in any of the Stores subsystems. The Store subsystem can itself be further decomposed into two key subsystems.

- Context Store
- Message Service

The Identify subsystem and the Use subsystem are to be used to build components that are either standalone or run as extensions to other applications.

6.1.4.3 Interfaces

An interface defines the communication between two subsystems. Interfaces are described using the Interface Definition Language (IDL). IDL is a language independent notation for describing interfaces (Vinoski 2000). The IDL descriptions can be mapped to implementation languages such as Java and C++. Using IDL allows us to define the interface without having to know anything about its implementation.

In the ICU architecture, there are two interfaces that must be supported. The first is the interface between the Identify subsystem and the Store subsystem. The second is the interface between the Use subsystem and the Identify subsystem. Both interfaces are implemented by the Store subsystem.

IUseContext. This interface defines how Use agents interact with the Store subsystem. It allows agents to register with the Store, authenticate against the store when required, and to send requests for contextual information to the store.

IIdentifyContext. This interface defines how Identify agents interact with the Store subsystem. It allows agents to register with the Store, to authenticate against the store when required, and to add contextual information to the store by submitting updates.

6.1.5 Summary

A software framework describes how to construct software systems that solve problems in a specific domain. The ICU framework describes how to construct software systems based on an ontology of context that use contextualization to help users find information on the World Wide Web. This framework is based on the ICU paradigm, which describes how such a system can identify, store and use contextual information. The framework developed is built on a service-oriented architecture that allows it to be deployed as an extension along side existing web infrastructure.

By developing the ICU Framework using standard software engineering methodologies it is possible answer research question Q8 which asks "Can a software framework based on the ontology of context be developed?" By applying the Catalysis approach to developing a framework it has clearly been possible to develop a framework based on the ICU paradigm. This paradigm is based on the flow of information in to and out of the ontology of context.

The use of Web Services to support the ICU paradigm answers research question Q9 that asks, "How can an ontology of context be integrated in to existing web infrastructure". By developing web services it is possible to take advantage of web 2.0 based mechanisms that support the development and deployment of new web services. This means that the model does not require the modification of existing web infrastructure.

With a software framework in place and a service-oriented architecture established it has been shown that the ontology of context can be applied to developing software systems that support contextualization. This framework is designed to be reusable and can serve as the basis for various implementations of agents and user interfaces that support contextualization. It solves the problem of contextualization in the general case. The next section shows how this framework can be applied to develop contextualising agents and user interfaces for specific applications of contextualization of web-based information to support information seeking behaviours. These applications serve as a basis for evaluating the usefulness and ease-of-use of the approach.

6.2 Implementation

The ability to be able to build software tools that will help people find information on the World Wide Web is a motivation for investigating the role of contextulisation of web-based content. A prototype of such a tool has been developed, called ISeeYou. This tool was developed using the ontology of context and the ICU framework developed earlier in this work. In the following, the technologies used to implement ISeeYou are described and the role of the ICU framework in the development of this tool are also described.

The ICU framework is based on a service-oriented architecture. An alternative way to view this architecture is as a client-server architecture. The client layer consists of the *Identify* and *Use* agents. The server layer consists of the context collection.

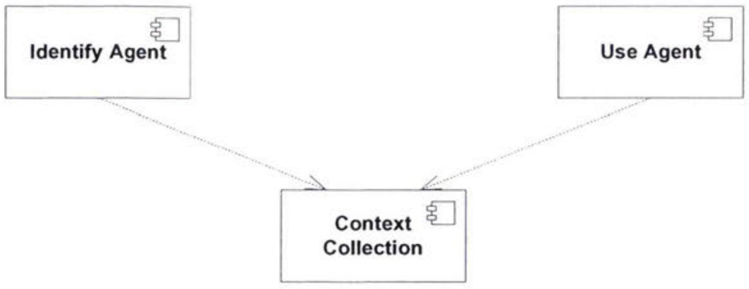


Figure 6-5 ICU Framework (UML 2.0)

In this section the functionality of the components that were implemented for the evaluation is described, and describe the technologies used to implement this functionality. The description of the technologies includes discussion of the range of available technologies for the components, and the basis for the decisions that were made to focus on specific technologies. The evaluation implementation has been developed for Internet Explorer but designed in a way that it can be ported easily to other browsers or to desktop-based applications. This is considered an acceptable limitation given that this is a prototype intended. The discussion is broken into two sections. The first section considers implementation issues for the client components while the second section considers issues for the server components.

6.2.1 Client Implementation

The client layer encompasses the Identify and Use components. The following sections consider technological and design issues considered when implementing these components. Further discussion of these issues, particularly technical issues, can be found in Appendix F.

6.2.1.1 Technology Overview

To answer Q10 it needs to be shown that the client layer of the ICU framework can be implemented using existing web technologies. This is done by analysing the client technologies used on the World Wide Web. In particular this chapter focuses on browsers because these are the clients most users use when searching or browsing the World Wide Web (Berners-Lee 1999). Other technologies are briefly considered, but to answer the research question it is sufficient to show that *a* way of integrating the framework with existing client technologies exists.

When developing ISeeYou the ICU framework to has been used to inform the design decisions. There were three key decisions to be made.

1. Which browser to support?

2. Which extensibility technology to use?
3. How does the extensibility technology map to ICU Client agents?

The client layer of the ICU framework consists of Use agents and Identify agents. These agents are used to identify contextual information relevant to the user and to then adapt the information that is presented to the user. This implies that these agents must interact closely with existing user applications. A key initial decision in the design of these agents was whether to implement them as stand-along applications or as plug-ins to existing user-oriented information management tools (such as Web browsers and media players).

The ICU framework places no constraint on whether these agents are stand-alone applications or plug-ins to existing applications. Both stand-alone application and application plug-ins are useful ways of augmenting tools used for searching and browsing the World Wide Web (Card, Robertson & York 1996) (Qu 2003).

Stand-alone applications are applications that have their own process. Stand-alone applications may be installed as part of an application suite, in which case the applications share resources with other members of the suite, but typically stand-alone applications are independent. Microsoft Word is an example of an application installed as part of a suite. Microsoft Windows Media Player and Nullsoft's Winamp are examples of applications that are stand-alone. Internet Explorer can be considered a stand-alone application because it runs in its own process even though it's components are integrated with the operating system.

Plug-ins are software components whose operating lifecycle is managed by an application that is already running in the system. The component implements a plug-in API and is then registered with the application. When the application runs it can, when appropriate, access the component through the plug-in API. The component then has access to a set of the applications resources that have been exposed through API's. Internet Explorer, Firefox and Microsoft Windows Media Player are examples of applications that have many plug-ins developed for them.

In order to illustrate the effectiveness of ISeeYou several Use agents and Identify agents were developed. The decision was taken to develop these for a Web browser given that Web browsers are currently one of the dominant tool for users to access complex information, and are well-understood by almost all computer users.

Given that the current implementation of ISeeYou is intended only for evaluation purposes, it is sufficient to adapt a relatively narrow implementation (i.e. restricted to specific browser support). The implementation of ISeeYou agents will be carried out by creating COM-based extensions to IE. To implement ISeeYou it is necessary to implement Identify agents and Use agents. The following discussion describes the specific extensions to IE that were used to implement these ICU components.

Internet Explorer is a Windows application that is available on all Windows machines. It provides a high level of customisation and reuse by exposing a number of COM objects that provide access to its internal operations and data structures. Extensibility for IE is based on a COM object called a Browser Helper Object. The most common ways to extend IE are Toolbars such as the Google Toolbar and Explorer Bands such as the Google desktop search band.

It is possible to implement the client layer of the ICU framework using existing web-based technology. For the ISeeYou implementation, it was decided to extend Internet Explorer. Internet Explorer was chosen because it is the most widely used browser. The Identify agent for ISeeYou is implemented using a Browser Helper Object. Using a BHO gives us access to the event model for the browser and the DOM of the page within IE. The Use agent for ISeeYou is implemented using an Explorer Band. By reusing the web browser control inside the explorer band it is possible to implement an HTML-based user interface that can be reused in other implementations in different browsers and in different platforms.

6.2.1.2 IdentifyApp Design

IdentifyApp is an implementation of an identify agent as a Browser Helper Object. In this current implementation, there are three scenarios where context is identified. These scenarios are described in chapter five.

1. When the user clicks on a link embedded in a page, it identifies the anchor text of the link, the URL of the current page and the URL of the link destination as context. The set of elements identified is listed below.
    - AnchorText. The text data belonging to the a tag being clicked.
    - AnchorDest. The URL contained in the href attribute of the anchor being clicked. If the protocol of the URL is not http, then the event is ignored. If the link uses dhtml scripting to perform its navigation it is ignored.
    - AnchorSrc. The URL of the page for which the event has occurred.

2. When the browser finishes downloading a page that is recognised as being part of a Google result set, the set of links in the page that are recognised as being search results are identified as context, along with the link to the next page in the result set.

- Links. A Google result page has 10 items that are not ads. The links are scraped from the current page. Only the URL is scraped.

3. When the browser finishes downloading a page that is not part of a Google result set, the page is parsed and a set of contextual elements is identified. This set consists of the entities listed below.

- Metadata. The name and value attributes from any meta tags in the head section of the document.
- Title. The contents of the title tag in the head section of the document.
- Anchors. The href attribute and the anchor text of every a tag in the body of the document.
- Keywords. A count of all words that occur as content in the page. Words in a stoplist are excluded. A stoplist is a list of words that occur so commonly in normal usage that they are not considered meaningful when generating an index of a page (Fox 1992). At this time the stoplist used is English only.
- Timestamp. The time the page finished loading.
- PageURL. The URL of the current page.
- Referer. The http-referrer of the page. This is the URL of the page from which the request for the current page originated.

The IdentifyApp is written in C++ using the ATL library to simplify the interaction with COM objects. The BHO uses the DWebBrowserEvents2 interface (MSDNa) to track events that occur in the browser. It recognises the OnClick event to trigger scenario 1. It uses the OnDocumentComplete and OnBeforeNavigate2 events to determine when a page has finished downloading. When a page finishes downloading the URL is analysed to determine whether to trigger scenario 2 or scenario 3. The DOM is accessed through the IWebBrowser2 interface by retrieving the IHTMLDocument2 interface for the current page.

6.2.1.3 UseApp Design

UseApp is an implementation of a Use component as described by the ICU framework. It is used to display contextual information to the user. The ISeeYou system has been implemented as an Explorer Bar that shows contextual information relevant to the users' current activity.

The Explorer Bar has seven options in a tool bar at the top. When a new page loads in the main window, the ISeeYou bar is automatically updated. The feature mode can be changed by the user. These feature modes are described below.

Info

Displays keyword information and an ordered list of links from the World Wide Web page currently displayed by the browsers main window. The links are ordered based on your browsing patterns to put relevant links first. Relevancy is determined by counting the frequency of anchor terms the users has clicked on in the anchor text of the link.

Local?

Retrieves a list of pages you have visited that contain the words specified in the query. Short and common words may be ignored. The list is ordered using a simple heuristic. The heuristic matches the terms in the query against the keywords in the page. The greater the frequency with which terms in the query occur in the word list for the page, the higher the ranking. The list is then ordered on descending rank.

Page*

Displays a list of pages you visit most regularly. The pages are ranked on how often the user has visited the page. A count of the number of times a page has been visited is stored as part of the contextual information for each page. The top 13 entries are returned.

Page

Displays a list of pages the user has visited most recently. The context store orders the pages visited in the users store by the timestamp that indicates when the page was last visited and returns the 13 most recent results. Only the most recent visit to each page is shown.

Host*

Displays a list of hosts you visit most regularly. The hosts are ranked on how often the user has visited the host. A count of the number of times a host has been visited is stored as part of the contextual information for each host. The top 13 entries are returned.

Host

Displays a list of hosts the user has visited most recently. The context store orders the hosts visited in the users store by the timestamp that indicates when the host was last visited and returns the 13 most recent results. Only the most recent visit to each page is shown.

Web?

Navigates to the most recently requested Google search result page. Shows all the result links for the current search query, along with a summary popup for each link. This view persists if result links in the list are clicked so you can browse the result set.

Information Seeking Behaviour	**ISeeYou function.**
Directed Browsing	Info
Directed Search	Local?
Undirected Search	Web?
Undirected Browsing	Page*
	Page
	Host*
	Host

Table 6-1 ISeeYou Functions as Behaviours

6.2.2 Server Implementation

The server layer encompasses the Store component. The following discussion describes technical and design considerations faced while implementing this layer. Further detail can be found in Appendix F.

6.2.2.1 StoreApp

The StoreApp consists of the context collection service and the context store. The context collection service exposes an interface that can be used to add and retrieve context from a collection of context as a WebService. The context store provides a data structure that stores and organises contextual information in a way that is persistent and is also efficient for the kinds of queries to be submitted.

6.2.2.2 Context Collection Service

The Context Collection Service publishes an interface as a web service. This interface allows identify agents to add contextual to the context store and it allows use agents to retrieve contextual information from the context store. The methods included in this interface are described in appendix F.

6.2.2.3 Context Store

The Context Store manages contextual information. It is part of Collection layer of the ICU Architecture. The ICU framework describes a set of constraints and requirements on the implementation of the context store to ensure that it manages context in a way that supports the development of tools that help users carry out information-seeking behaviours on the World Wide Web. The ontology encapsulated by the framework describes the entities that the Context Store must manage and the relationships between these entities that must be supported.

The framework requires that the store be persistent and that it be responsive to requests. These constraints on any implementation of the framework drive the decision to use prevayler (Prevayler 2006) as the persistence layer and XFire (codehaus) as the SOAP implementation. The framework also requires that the collection and store not be tied to a specific platform.

The Context Store is not accessed directly by the client layer. It is only accessed by the collection message handler. Having the Context Store decoupled from the identify agents and the use agents means the store can be constructed to be independent of any specific browser. This decoupling means that the contextual information in the store can be used by any agents that use the interfaces published by the collection. These agents do not need to know how the underlying data structure is organised. Agents using the collection interfaces can run in any browser that supports the invocation of SOAP services from a plug-in, and from any stand-alone application that supports SOAP calls.

The responsiveness constraint means that the data structures used in the context store have been chosen for cheap lookups. In particular, extensive use has been made of hash maps. Hash maps have a cost associated with insertion but are O(1) for lookup (Kernighan & Pike

1999). Because responsiveness is most affected by lookup and minimally impacted by insertion, the overhead of insertion maintaining the hash map is acceptable given the gain in performance on lookups over lists or arrays.

The ICU Framework describes the entities and relationships that the store must manage. These entities and relationships form the basis for designing the class hierarchy and the data structures that implement the context store. Additionally the framework describes constraints on the context store that inform decisions about the technology used to implement the store and its data structures. Finally, the Context Store is designed so its implementation details are hidden from agents that use contextual information, allowing the information to be used by any agent that communicates via the appropriate interface.

6.2.3 Summary

The implementation of ISeeYou was performed to investigate research questions Q10 and Q11. ISeeYou is an implementation of a contextualisation application based on the ICU Framework. The implementation integrates with Internet Explorer to contextualise information encountered when browsing and searching the World Wide Web, showing that the framework can be used to create contextualisation tools that integrate with existing web technologies. This integration is possible because the ontology of context has been designed to be independent of specific applications. Rather, it is based on domain knowledge. The discussion in this chapter, along with the technical notes in appendix F, shows how the framework can be used to create tools that contextualisation.

6.3 Conclusion

This chapter has shown that the approach to contextualisation that uses an ontology of context to manage contextual information can be realised as a software system using existing web technologies. The management of the ontology of context, and thus also the contextual information, can be exposed as web-service. A fundamental property of web-services is they support cross-application access to the functionality published by the service. The ICU framework is a software framework that supports the contextualisation through the ability to manage contextual information.

7 Evaluation

This chapter shows how the perceived usefulness and perceived ease-of-use of context-based tools that augment information-seeking behaviours be evaluated using the Technology Acceptance Model (TAM). An evaluation for the tools is developed, and an analysis of the results of the evaluation is performed. The implications of these results for the validity of the hypothesis are discussed.

This chapter focuses on investigating two research questions, Q12 and Q13. These questions are investigated by developing and carrying out an empirical evaluation of the ISeeYou tool developed in chapter seven.

The first section of this chapter describes the concept and purpose of software evaluation, and discusses possible approaches that could have been taken to evaluating ISeeYou. The reasons for choosing TAM for the evaluation are discussed, and a basic understanding of how TAM is used to evaluate software is developed.

The second section of this chapter discusses how TAM can be used to evaluate ISeeYou. The key factors measured using TAM, Perceived Usefulness and Perceived Ease-of-Use are discussed. It also describes how TAM is used to create an appropriate survey instrument. Finally it describes how the evaluation was carried out, focusing on ethical and operational issues. This section addresses Q12, *How can the perceived usefulness and perceived ease-of-use of context-based tools that augment information-seeking behaviours be evaluated?*

In the third section of this chapter, the results of the evaluation are presented. These results are analysed and their implications for the validity of the hypothesis are discussed. This section addresses Q13, *Is ISeeYou Useful and Easy-to-Use?*

7.1 Evaluation

Once an operational user interface prototype has been created, it must be evaluated to determine whether it meets the needs of the user. (Pressman 2000) User interface evaluation is the process of assessing the usability of an interface and checking that it meets user requirements. (Sommerville 1996) The key user requirement for this work is identified in the hypothesis, which states that tools built using the ICU framework are useful.

This section presents an approach to evaluating tools developed using the ICU framework. The approach is based on the Technology Acceptance Model. The section begins by discussing the reasons for evaluation and what kind of evaluation is appropriate for tools such as ISeeYou. It then describes the constructs of TAM and explores the validity of the model. Finally, a method for applying TAM to evaluation is presented.

7.1.1 Evaluating Software

Software evaluation is concerned with assessing the quality of software. The quality of software can be determined by six factors, namely Reliability, Usability, Functionality, Maintainability, Portability and Efficiency (Boegh et al. 1999). When designing software, requirements around these factors are identified. This work is most concerned with usability of software, and in particular the usefulness. In order to perform a successful evaluation, it is important to identify when the evaluation will be performed and the purpose of the evaluation.

7.1.1.1 Summative Evaluation

The quality of software can be assessed at various points in the development cycle. Assessing the quality of software at the end of the development cycle is summative evaluation. Evaluation can be used at the end of a development cycle to ensure that the system meets the requirements it was intended to satisfy. Such an evaluation is called a summative evaluation, and is typically used as a gateway check before moving to the next phase of the development cycle or before releasing the product. Another approach to evaluation is to assess the quality of the product during the design and development process, and use the feedback from the evaluation to refine the product. Such an approach is called formative evaluation. This work uses a summative evaluation to assess whether the goal of usefulness established by the hypothesis is met. The technique developed here is, however, also applicable to formative evaluation.

7.1.1.2 Usability Evaluation

In this work, the goal of the evaluation is to assess the usefulness of a software tool developed using the ICU framework. A tool that is useful helps a user perform a task. This work is interested in creating tools that are useful for users who are performing information seeking behaviours on the World Wide Web. Usefulness is the extent to which software actually helps to solve users' practical problems. Usefulness is one determinant of the usability of a software tool. The usefulness of a software tool can be assessed by performing a usability evaluation.

Usability is quality of use (Bevan 1995). Usability evaluation is concerned with gathering data about the usability of a design or product by a specified group of users for a particular activity within a specified environment or work context (Preece et al. 1994). The most effective evaluations are systematic evaluations performed with large groups of users. These involve users interactions with the tools being analysed by designers, cognitive experts and developers, and often involve sophisticated labs (Wichansky 2000). Such techniques are outside the scope of resources available to us. Inexpensive techniques include questionnaires, observation, snapshots, and instrumentation (Nielsen 1994).

1. *Questionnaires*. The user interacts with the system and then answers a series of questions about their interaction. These responses are then analysed and conclusions from that analysis are used to refine the design of the interface. It is important that the questions be precise and easy to answer. Asking users to rate attributes on a scale is usually more effective than asking an open question asking them to explain their experience. Questionnaires are a cheap way of evaluating a users interaction with a system.

2. *Observation.* The evaluator observes the user as they use the system and notes their interaction with the system. The evaluator also prompts the user to think aloud by explaining their thought process as they use the system. This is an effective approach to understanding how people will use the system, but requires significant time as well as an appropriately equipped laboratory for the evaluation to avoid the user being distracted. The overhead of this approach made it inappropriate for early evaluation of ISeeYou.

3. *Video snapshots*. In this approach the users interaction with the system is videotaped and then later analysed for signs of difficulty using the interface, such as frustrated body language, excessive mouse movement, or unnatural eye movement. These kind of attributes are not useful for evaluating the model as this work is more concerned with the usefulness of the model rather than fine details about the users interaction with the interface.

4. *Instrumentation.* Usage statistics for the functions of the system are collected by the tool itself using instrumentation code. The instrumentation data is then sent to the developer for analysis. Instrumentation can be very effective, but can also raise significant privacy concerns for users. Given the strict requirement for survey

participation to be anonymous, and that concerns about privacy regarding web-usage are significant (Hoffman 1999), it was decided instrumentation of the code was not appropriate.

For ISeeYou, the resources available favour either instrumentation or questionnaires as appropriate evaluation techniques. Both can be performed with minimal resource requirements and without require the users participate in lengthy lab based sessions. Instrumentation would be an easy to measure whether people regularly use the software. However, concerns about privacy of data and about the anonymity of users led us to adopting a questionnaire-based approach. The next section describes how an appropriate questionnaire for ISeeYou was developed.

7.1.1.3 Evaluating ISeeYou

An evaluation method for IseeYou must meet several criteria for the evaluation to be useful. These criteria are as follows.

1. Summative. The evaluation of ISeeYou is part of investigating the validity of the hypothesis. At the end of development of ISeeYou, it is evaluated to see if it is useful.
2. Inexpensive. The evaluation must be able to be performed given the resources available for this work. The evaluation should be able to be repeated for new research projects based on the ICU framework. An effective but inexpensive evaluation makes the ICU framework a more attractive candidate for future work.
3. Qualitative. Qualitative metrics have been used to measure the usability of tools that are used for information seeking and for personal information management. There is a clear absence of metrics for quantitatively measuring usefulness of software tools that manage personal information, as ISeeYou does. Therefore, a qualitative evaluation of ISeeYou is performed in this work.

4. Dynamic. ISeeYou does not place constraints on the kinds of information-seeking behaviours a user can perform, and it would be inappropriate for an evaluation to place any such restrictions. Therefore, an evaluation of ISeeYou should not be limited to single properties of the user's interaction of the tool.

These goals are all satisfied by a evaluation method called the Technology Acceptance Model (TAM). The following section describes TAM in more detail.

7.1.2 TAM

Since its development, TAM has become a widely used tool for evaluating software and other technology based system. Some key areas it has been applied to include operating systems, banking systems, e-commerce sites on the World Wide Web, and mobile devices. Its broad application means that it has been reviewed and verified several times and found to be a reliable indicator of the adoption of a technology. One of the reasons for its wide adoption is that while it has been shown to be very accurate, it is also very simple. It is based on a survey and measures two factors. It is this simplicity, along with its effectiveness, that makes it ideal for the evaluation.

TAM is used to explain and predict the user acceptance of an information system (Davis 1989). It is based on the theory that system use is predicated by two constructs, perceived usefulness (PU) and perceived ease-of-use (PEOU). The validity and reliability of these constructs has been rigourously reviewed and shown to be strong.

7.1.2.1 Constructs

The technology acceptance model is an adaption of the Theory of Reasoned Action (TRA) (Fishbein & Ajzen 1975) which is specifically meant to explain computer usage behaviour (Davis 1989). TAM has shown that the intention to use software for a given task in a given context is influenced by perceived usefulness and perceived ease-of-use. The relationship between these constructs as used in the development of TAM is shown in the block diagram in Figure 8-1.

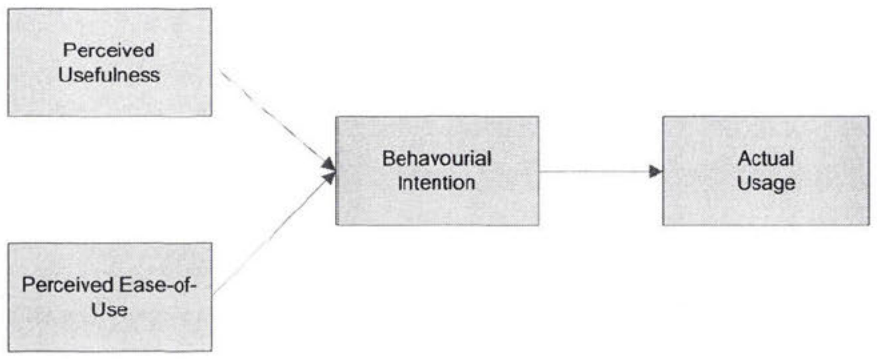


Figure 7-1 Technology Acceptance Model (from Davis 1989)

TAM has shown that the intention to use software for a given task in a given context is influenced by perceived usefulness and perceived ease-of-use.

Perceived usefulness is defined by Davis as 'the degree to which a person believes that using a particular system would enhance his or her job.' It is a measure of the extent to which a user believes a system would help them with the task they are trying to achieve.

Perceived ease-of-use is defined by Davis to be 'the degree to which a person believes that using a particular system would be free of effort.' It is a measure of how easy a user finds a system to use for the task they are carrying out.

These constructs are measured by a survey that uses a set of established questions to collect user's responses.

7.1.2.2 Validity

The validity of TAM's claim that PU and PEOU accurately predict user acceptance of a technology has been tested many times (King & He 2006; Ma & Liu 2004), (Legris, Ingham & Collerette 2003). The construct validity, internal validity and external validity of TAM have repeatedly shown to be high. It has also been shown to be generalisable for users operating in the same context. Part of the reason for the popularity of TAM is this rigorously checked accuracy. TAM has also shown to be reliable for groups of users as low as 12 (Laitenberger & Dreyer 1998; Zettel 2001), and has repeatedly been shown to be a reliable indicator of usefulness across a range of web-based technologies such as websites, e-commerce, and chat tools. It has also been shown to be applicable to browsers, with an evaluation of Netscape having been performed (Morris & Dillon 1997) .

The technology acceptance model provides an inexpensive approach to performing a usability evaluation that assesses the usefulness of a software tool. TAM provides a subjective measure of whether a technology is useful and easy-to-use for a specified task in a specified context. The following section shows how TAM can be applied to evaluating the usefulness of a tool developed based on the ICU framework.

7.1.3 Method

TAM is used to explain a specific behaviour toward a specific target within a specific context (Davis 1993). The task for tools based on the ICU framework in information seeking. The target is the tool created using the framework. ISeeYou is an example of such a tool. The context is the user's use of the World Wide Web. The first task in applying TAM is to clearly identify these entities.

Identify task (information seeking). People who are searching for information on the World Wide Web are engaged in information-seeking behaviours. These behaviours can be categorized as browsing and searching. The exact nature of the browsing and seeking depends on the user and their information need. In situations where the users tasks are variable and situation dependent, it is appropriate to allow the user to perform whatever task they perform normally (Nielsen 1994). This evaluation will ask users to perform whatever information-seeking tasks they would ordinarily perform.

Identify context (on the World Wide Web). This evaluation is only concerned with the users information seeking on the World Wide Web, using the browser with the tool installed. No effort is made to assist or track information-seeking behaviours performed in other systems, such as help files or non-web based library catalogues. Similarly, no effort is made to track information-seeking behaviours performed in other browsers or with other devices like mobile phones.

Identify users (users of the World Wide Web). TAM has been shown to be generalisable across groups of users. This is clearly true when results from groups of students are generalized to groups of professional or expert users. There have been widely varying attempts to identify the properties of web user that constitute an expert, and no clear quantification of expert is available for this work (Martzoukou 2004). However, it would be acceptable to state that the students with web expertise are a user group for whom results would be generalisable to other web experts, according to studies in to the generalisability of TAM results.

Having a clear understanding of the tasks to be evaluated, the users who will perform the evaluation, and the context within which the evaluation will be performed, the final step is to identify the method for performing the evaluation. The method used in this evaluation is based on an analysis of a number of TAM evaluations (Laitenberger & Dreyer 1998; Morris & Dillon 1997; Wang et al. 2003; Zettel 2001), and involves the following steps:

1. Select a set of questions to measure the factors.
2. Rewrite the questions to reflect the tool being evaluated, and the tasks and context within which it is being evaluated.
3. Distribute the tool to a group of users.
4. Distribute the survey instrument.
5. Collect survey responses.
6. Analyse responses.

The following section describes how these steps were performed for an evaluation of ISeeYou.

7.2 Evaluation with TAM

TAM is designed to be fairly straight forward to apply as it is intended to be used regularly during a development cycle to provide feedback about the likely success (or otherwise) of a technology. The principles are reasonably simple and are presented in the following section. The next section describes an approach to carrying out a TAM evaluation that is drawn from several of the many TAM case studies that are available (Davis 1989; Laitenberger & Dreyer 1998; Morris & Dillon 1997). Finally, some of the extensions to TAM and their relationship to the work are discussed.

7.2.1 Applying TAM

There is not set of rules associated with TAM about how to carry out an evaluation. The basic idea is that a survey is given to a group of potential users of the system. Their responses to the survey are used to determine the weighting of the PU and PEOU factors. If the weighting for both is over 0.5, then the system is likely to be adopted by users. The stronger the weighting of these factors, the more likely it is the technology will be adopted.

The following steps are derived from case studies on TAM and are intended to make explicit the steps that are typically involved in a TAM evaluation. These steps are expanded on in the following section to show how TAM can be applied to the problem domain and the software.

1. Identify the system. Describe what it does in terms of user tasks,

2. Develop a set of hypotheses about the usefulness and ease of use of the system.

3. Develop a set of survey questions to test the hypotheses. Base these on the chart-master questions used by Davis if a guide is required.

4. Identify the group of people who will be surveyed. Ensure this group has similar levels of experience to ensure the results are generalisable.

5. Have the subjects perform tasks using the system.

6. Distribute the survey and have the subjects fill it out.

7. Collate the results and calculate the weighting for each factor.

These steps have been used to create an evaluation methodology that can be used to evaluate tools based on the ICU framework, such as ISeeYou. The steps used in this methodology are described in the next section.

7.2.2 An Evaluation Methodology

For the purposes of this work, the original TAM (Davis 1989) provides a sufficient indication of user acceptance to support the hypothesis. For this reason, any of the extensions or the new factors that have been proposed are not applied to this evaluation. There is scope in other projects to identify extensions for applying TAM to information-seeking behaviours, but that is outside the scope of this work. Developing TAM to specifically describe the kinds of behaviours this work seeks to support in this work would be a useful step in supporting the development of context-based user interfaces that help users find information on the World Wide Web.

To evaluate the user interfaces a methodology that can be reused for evaluating similar tools in future work is used. This methodology consists of two phases, planning and execution. The planning phase consists of four steps as shown in Table 7-1.

Step 1. Identify the evaluation goal.
Step 2. Identify what is being identified.
Step 3. Identify the target audience.
Step 4. Create a survey.

Table 7-1 Evaluation Planning Phase

The execution phase consists of three steps as shown in table 7-2.

Step 1. Distribute and have people use the tool.
Step 2. Distribute and collect the survey.
Step 3. Collate and analyse responses.

Table 7-2 Evaluation Execution Phase

The application of these steps to evaluating ISeeYou is described in the following section.

7.2.2.1 Planning

The focus of the planning phase is on developing a survey that can be used to evaluate the system that has been implemented. In this case it is the ICU User Interfaces. This section describes the goal of the evaluation and identifies the factors that are being evaluated. It then identifies questions that will allow these factors to be measured, and structure these questions into a survey that can be presented to the subjects.

Step 1. Identify the evaluation goal

The goal of the evaluation is to determine whether the ICU User Interface is perceived to be useful and easy to use. A positive result on these factors means that users will use the tools for the information-seeking behaviours they have been designed to support.

Step 2. Identify what is being evaluated

Two key factors described by TAM are being evaluated, these being perceived ease of use and perceived usefulness. These factors can be determined by a survey based on questions about how the user performed tasks with the tool. These questions are based directly on a set of questions used to evaluate Internet usage (Shih 2004) and indirectly on the original chart-master questions used in validating TAM (Davis 1989)

Perceived Usefulness

Perceived Usefulness	
PU1	Using I See You helps me find information on the web more quickly.
PU2	Using I See You improves my performance when finding information on the web.
PU3	Using I See You can increase my productivity when finding information on the web
PU4	Using I See You can enhance my effectiveness when finding information on the web

Table 7-3 Scale items of the usefulness determinant

Perceived Ease-of-Use	
PEOU1	Learning to use I See You is easy for me.
PEOU2	I can use the I See You in a manner that helps me find information.
PEOU3	My interaction with I See You is clear and understandable.
PEOU4	In general, I find I See You easy to use.

Table 7-4 Scale items of the ease of use determinant

Step 3. Identify the target audience

In order to get a set of generalisable results the survey and tool were released to target groups of 20-30 subjects who have similar levels of experience with the internet. Class groups are ideal here because by their nature they tend to contain people of similar education and technical background. Groups within the Faculty of Engineering were targeted to assist with this research. Another advantage of using class groups is that they are inexpensive and are reasonably easy to access. While an industry-based analysis would be interesting, it is not necessary for the purpose of establishing the hypothesis.

Step 4. Create a survey

The chart-master survey developed by Davis is the original example of how a TAM survey should be organised. This survey is included in Appendix D. The survey has been constructed based on the questions identified as helping to measure PEOU and PU of the ICU User Interface. The survey measures user responses using a seven point Likert-type scale. The scale used for the items is shown in table 8-3.

Extremely Likely	1
Strongly Likely	2
Likely	3
Neutral/Undecided	4
Unlikely	5
Strongly Unlikely	6
Extremely Unlikely	7

Table 7-5 Likert scale items

7.2.3 Execution

In this phase, ISeeYou is made available to the target groups for installation on the machine. In addition, the survey instrument is made available to the target groups. Participants use the tool and complete the survey. The results of the survey are analysed. These steps are outlined in table 7-2, above.

Step 1. Distribute and have people use the tool

In order for users to evaluate the ICU User Interface, it must be installed and usable on the subject's computers. The interface has been built to be installed either off the World Wide Web via download, or from a CD. Both methods will be available to subjects, along with a document describing setup and usage. Support will also be available via email to ensure subjects are able to participate.

Step 2. Distribute and Collect the Survey

When distributing the survey there are three key issues that need to be considered.

1. Distribution Method. There are four common methods for distributing surveys.

- By Phone. The surveyor contacts people either from a call list or at random and asks them a set of survey questions, recording their responses.
- By Mail. The surveyor mails out the survey in a hardcopy form with instructions for completion to a target group of subjects. A paid return address envelope is included for subjects to return the survey when they are done.
- Personal Interview. The interviewer identifies a group of potential subjects and schedules interviews with them. The survey questions are asked face to face and the interviewer records the responses along with other observations that are deemed significant.
- Self-Administered. The surveyor takes the survey to a group meeting of potential subjects and distributes it to the group members. Subjects complete the survey and return it to the surveyor.

The TAM survey has been designed to be self-administered. The survey was distributed after the target group had an opportunity to use the software that has been have developed. The self-administered approach was chosen because it has the lowest costs involved, and there is ready access to groups of potential subjects through the University.

2. Distribution Audience. The main target audience will be class groups in the Department of Computer Systems Engineering at the University of Technology, Sydney. The advantage of using class groups is that the members tend to be of similar backgrounds. Consequently, variances in results tend to arise from variances in responses to the tool rather than from

variance in knowledge background or demographics of the subject group. This means that the resulting analysis is generalisable.

3. Ethics Approval. Studies involving human subjects and the collection of information about them require ethics approval to ensure that the information gathered is appropriate and will be handled in way that protects the privacy of the subject. At UTS the guidelines of the Human Research Ethics committee must be followed for any research involving human subjects. While technically ethics approval is not required for anonymous surveys, working through the approval process and reviewing the methodology with ethics board members was an important step in creating an effective survey. Also, the oversight of the board raised issues around ensuring that a survey is truly anonymous. In particular, it is important to ensure that there was no way of tracking which users responded to the survey. This requirement reinforced the decision to use the Survey Manager provided by ILM. Because the World Wide Web sites for survey manager are run by a group outside of the Faculty of Engineering the identities of users participating cannot be accessed by the Faculty without going through the Institute for Multimedia and Learning.

A self-administered survey that is made available electronically has been chosen to collect responses in this evaluation.

Step 3. Analysis

TAM does not specify a process for analysing results. In order for the evaluation to repeatable and reusable, a simple process has been elicited from the existing literature. A number of TAM evaluations were reviewed (Laitenberger & Dreyer 1998) (Zettel 2001) (Morris & Dillon 1997) with the goal of identifying common steps in analysing the data.

These studies were selected because they had similar goals to ours, focusing on using TAM as a usability evaluation. A process of five steps has been identified that allows the analysis of data that has been obtained from a TAM-based survey.

1. Remove incomplete responses. Any survey responses with empty or illegible response items are removed. This ensures that the results are not affected by undefined values.
2. Remove invalid responses. Invalid survey responses are removed based on qualifying questions. Asking users for demographic information or self-described experience are examples of qualifying questions.
3. Identify Factors. A factor analysis to assess the construct validity for the variables considered in this research is performed.
4. Calculate Reliability. Cronbach's alpha coefficient for the constructs to asses scale reliability is calculated.

Calculate Medians. Having established the factors being measured, and the reliability with which the items measure these factors, the median for the factor based on the responses to all its measuring variables is calculated. These medians reflect the extent to which users agree with the factors, specifically with Perceived Usefulness and Perceived Ease-Of-Use.

7.2.4 Conclusion

In this section, the use of TAM to evaluate ISeeYou has been described. It began by discussing the factors measured by TAM and how the relate to the work. It then described a survey instrument based on TAM that can be used to evaluate ISeeYou, and how the results of this survey are to be analysed. Finally, it discussed the operational issues in carrying out the evaluation of ISeeYou and how the results from such an evaluation can be analysed. In the next section the results of the evaluation and an analysis of their significance is presented.

7.3 Results

The following sections describe the results obtained from the survey and the outcomes of applying the analysis process to these results.

7.3.1 Data Collected

The raw data can be downloaded from the Survey Manager website as an excel spreadsheet. Table 7-6 shows the data downloaded, with the question labels modified to save space. In the download, the entire question is shown. In the following analysis, the questions have been replaced with their corresponding mnemonics as used in 8.2.3. This has been done purely to save room.

For the evaluation volunteers from three masters level courses run by the faculty of engineering at the University of Technology, Sydney were recruited. In Autumn Semester of 2006 the ISeeYou tool and the survey were made available to 55 students in the Web Technologies subject. From this group of students six responses were received. To be statistically significant it is necessary to have at least six responses for each of the two factors that are being analysed, so it was decided to recruit more participants the following semester. In Spring 2006 the same tool and survey was made available to 55 students in Web Technologies and 15 students in MITE. It was decided to leave the survey open until the semester break, a period of 7 weeks, to allow students time to respond. By the end of the survey period, a total of 19 responses to the survey had been received. These responses are shown in Table 7-6.

ISeeYou Evaluation

Date	Time	Q1	PU1	PU2	PU3	PU4	PEOU1	PEOU2	PEOU3	PEOU4
30-May-06	19:20	1	3	3	2	2	3	4	4	4
5-Jun-06	15:15	1	2	4	3	2	1	1	2	2
5-Jun-06	21:35	1	3	3	4	2	3	2	3	3
6-Jun-06	14:07	1	2	3	3	4	3	4	5	3
12-Jun-06	11:34	1	6	6	6	4	2	4	2	1
12-Jun-06	20:53	1	2	2	2	2	3	3	4	4
27-Aug-06	20:08	1	3	3	2	2	2	1	3	2
31-Aug-06	0:14	1	3	4	4	4	5	4	3	4
31-Aug-06	1:30	1	3	3	2	3	3	2	2	2
31-Aug-06	15:40	1	2	2	2	2	2	2	2	2
2-Sep-06	11:59	1	3	4	3	3	2	3	5	3
3-Sep-06	19:32	1	3	4	4	4	3	2	2	3
4-Sep-06	16:51	2	3	2	3	4	5	3	5	4
6-Sep-06	16:16	1	3	3	3	3	2	2	3	3
13-Sep-06	16:05	1	3	2	3	3	3	3	3	2
14-Sep-06	16:16	1	3	3	3	3	2	2	3	3
16-Sep-06	10:53		5	4	4	4	5	5	5	4
16-Sep-06	17:40	1	2	3	2	3	2	2	2	3
16-Sep-06	21:11	1	1	2	2	1	2	3	2	2

Table 7-6 Survey Responses

7.3.2 Analysis Tools

The Survey Manager application makes the results of the survey available for download either for Microsoft Word as a Rich Text Format (RTF) document, or as an Excel spreadsheet. For this work, the choice was made to download the results as an Excel spreadsheet as the data requires manipulation before it is presented in a document. Having the data in excel makes it possible to remove invalid responses. Excel also has built in support for calculating medians of sets of data.

To perform factor analysis and reliability calculations it is necessary to perform calculations that are not built in to Excel. The approach used in this work was to use the functionality provided by a statistical analysis package, SPSS (SPSS). SPSS was chosen because it is readily available on machines at UTS.

7.3.3 Analysis

The analysis of the results involves five steps These steps are shown in table 7-7, and the application of each step to analysing the results of the ISeeYou evaluation are described below.

Step 1. Remove incomplete responses.
Step 2. Remove invalid responses.
Step 3. Identify constructs.
Step 4. Calculate reliability.
Step 5. Calculate medians.

Table 7-7 Analysing evaluation results

Step 1. Remove Incomplete Responses

The first step is to remove all incomplete survey responses. The incomplete rows could have been removed by Survey Manager but the incomplete information was downloaded to see if people were completing the entire survey. Response 17 is removed from the result set using this filter. The row to be removed is highlighted in Table 7-8.

Response	Q1	PU1	PU2	PU3	PU4	PEOU1	PEOU2	PEOU3	PEOU4
1	1	3	3	2	2	3	4	4	4
2	1	2	4	3	2	1	1	2	2
3	1	3	3	4	2	3	2	3	3
4	1	2	3	3	4	3	4	5	3
5	1	6	6	6	4	2	4	2	1
6	1	2	2	2	2	3	3	4	4
7	1	3	3	2	2	2	1	3	2
8	1	3	4	4	4	5	4	3	4
9	1	3	3	2	3	3	2	2	2
10	1	2	2	2	2	2	2	2	2
11	1	3	4	3	3	2	3	5	3
12	1	3	4	4	4	3	2	2	3
13	2	3	2	3	4	5	3	5	4
14	1	3	3	3	3	2	2	3	3
15	1	3	2	3	3	3	3	3	2
16	1	3	3	3	3	2	2	3	3
17		5	4	4	4	5	5	5	4
18	1	2	3	2	3	2	2	2	3
19	1	1	2	2	1	2	3	2	2

Table 7-8 Remove Incomplete Responses

Step 2. Remove Invalid Responses

A response is determined to be invalid if the answer to Q1 has a value of 2. This corresponds to the respondent answering no to the statement "I was able to successfully install and use ISeeYou.". Failure to install or use the tool could occur because of factors outside the control of this evaluation, such as a lack of administrator privileges on the machine, the user running a version of IE that is not compatible with ISeeYou, or a problem with their java configuration. It is reasonable to expect that users who cannot install or use the tool cannot form reliable opinions about the usefulness or the ease of use of the tool, so these responses are discarded. Table 7-9 shows the responses that were removed by applying this filter.

Response	Q1	PU1	PU2	PU3	PU4	PEOU1	PEOU2	PEOU3	PEOU4
1	1	3	3	2	2	3	4	4	4
2	1	2	4	3	2	1	1	2	2
3	1	3	3	4	2	3	2	3	3
4	1	2	3	3	4	3	4	5	3
5	1	6	6	6	4	2	4	2	1
6	1	2	2	2	2	3	3	4	4
7	1	3	3	2	2	2	1	3	2
8	1	3	4	4	4	5	4	3	4
9	1	3	3	2	3	3	2	2	2
10	1	2	2	2	2	2	2	2	2
11	1	3	4	3	3	2	3	5	3
12	1	3	4	4	4	3	2	2	3
13	2	3	2	3	4	5	3	5	4
14	1	3	3	3	3	2	2	3	3
15	1	3	2	3	3	3	3	3	2
16	1	3	3	3	3	2	2	3	3
18	1	2	3	2	3	2	2	2	3
19	1	1	2	2	1	2	3	2	2

Table 7-9 Remove Invalid Responses

Step 3. Identify Constructs.

Factor analysis allows us to identify underlying constructs from a set of items. A Principal Component Analysis is used on the set of survey responses. This analysis identifies underlying constructs as components by reducing closely correlated items in to a single factor. A factor analysis is performed on the set of filtered data obtained after step 2 is complete, shown in Table 7-10. Note that the response column and the column for Q1 have been removed, as there are not measurement items.

PU1	PU2	PU3	PU4	PEOU1	PEOU2	PEOU3	PEOU4
3	3	2	2	3	4	4	4
2	4	3	2	1	1	2	2
3	3	4	2	3	2	3	3
2	3	3	4	3	4	5	3
6	6	6	4	2	4	2	1
2	2	2	2	3	3	4	4
3	3	2	2	2	1	3	2
3	4	4	4	5	4	3	4
3	3	2	3	3	2	2	2
2	2	2	2	2	2	2	2
3	4	3	3	2	3	5	3
3	4	4	4	3	2	2	3
3	3	3	3	2	2	3	3
3	2	3	3	3	3	3	2
3	3	3	3	2	2	3	3
2	3	2	3	2	2	2	3
1	2	2	1	2	3	2	2

Table 7-10 Complete and Valid Responses

For the survey, it is expected there will be two components, Perceived Usefulness (PU) and Perceived Ease-Of-Use (PEOU). The results of performing a factor analysis using the Principal Component Extraction method in SPSS are shown in Table 7-11.

Item	Component	
	PU	PEOU
PU1	0.856	-0.245
PU2	0.855	-0.287
PU3	0.905	-0.183
PU4	0.803	0.183
PEOU1	0.316	0.738
PEOU2	0.497	0.624
PEOU3	0.035	0.733
PEOU4	-0.112	0.848

Table 7-11 Principal Component Analysis

The factor analysis identified two components. The first component consists of PU1, PU2, PU3 and PU4. The second component consists of PEOU1, PEOU2, PEOU3 and PEOU4. Factor 1 corresponds to Perceived Usefulness (PU) and factor 2 corresponds to Perceived Ease of Use

(PEOU). This correspondence indicates that the items used do measure the constructs they were designed to measure.

Step 4. Calculate reliability.

Cronbach's alpha is a coefficient that measures how well a set of items measure a single underlying construct. If the coefficient for a set of items is greater than 0.7 they can be said to be reliable measures of the underlying construct (Nunally 1978). From the factor analysis, it can be seen that there are two constructs, which have been identified as PU, corresponding to Perceived Usefulness, and PEOU, corresponding to Perceived Ease of Use. Cronbach's alpha for each of these constructs are shown in Table 7-12

	Cronbach's Alpha
PU	0.896
PEOU	0.755

Table 7-12 Cronbach's Alpha for Constructs

Cronbach's alpha is greater than 0.7 for both components. This tells us that the items that make up each component reliably measure the component. This means that the items can be used to deduce the results for each component.

Step 5. Calculate medians.

The PU and PEOU of I See You can now be calculated. Factor analysis has been used to identify the variables that measure each component. It has also been shown that these variables reliably measure these components. For each component, the mean of all the items that measure the component is calculated. The results of this calculation are shown in Table 7-13.

	Median
PU	3
PEOU	3

Table 7-13 PU and PEOU for ISeeYou

The median for PU is 3, which corresponds to likely. The median for PEOU is 3, which corresponds to likely.

7.3.4 Results

The median Perceived Usefulness of I See You is **3**. From this, it can be concluded that it is **likely** that I See You is perceived to be useful.

The median Perceived Ease of Use of I See You is **3**. From this, it can be concluded that it is **likely** that I See You is perceived to be easy to use.

From these measurements, it can be determined that I See You is perceived to be useful and easy to use. In answer to research question 15 then, the answer is yes. I See You is perceived to be *easy to use* and *useful*.

7.4 Conclusion

Evaluation is performed at the end of a software development cycle to determine if the software developed has met the goals that drove its design. In Scenario-Based Design this is referred to as summative evaluation.

There are many techniques available for evaluating software. TAM has been chosen because it is a measure of how likely it is that people will use a technology. Users will adopt a technology if they find it useful and easy to use. The technology to which TAM is applied is the user interface software that has been developed for supporting information-seeking behaviours. The adoption of a technology can be predicted by measuring its PEOU and PU. These are subjective factors measured using the Technology Acceptance Model.

A subjective evaluation was chosen because it can be applied easily across the range of information-seeking behaviours. There is no single empirical measure that applies to all the kinds of information seeking behaviours that are being supported. Subjective measures of usefulness and ease-of use can be applied to a variety of tasks.

The evaluation has been constructed based on the Technology Acceptance Model, an evaluation technique that has been repeatedly shown to be effective in determining whether users will adopt a new technology for a given task. A general-purpose evaluation such as TAM has been used rather than a IR evaluation because ISeeYou is not an IR system. It is a system that contextualises information retrieved in IR activities such as browsing and searching. It has more in common with Personal Information (PIM) systems than with IR systems, and there is no effective evaluation for PIM systems (Kelly 2006).

Using TAM to determine the PEOU and PU of the user interfaces allows us to determine the validity of the hypothesis. It has already been shown through theoretical modelling that it is possible to create an ontology of context. It has also been shown that it is possible to create user interfaces that use this ontology of context to support information-seeking behaviours.

The final claim of the hypothesis is that these user interfaces are perceived to be useful and easy-to-use for these behaviours. The methodology that has been developed in this chapter allows the measurements of these factors. The next chapter reports the results from applying this evaluation to the tools that have been developed and discusses how these relate to the validity of the hypothesis.

8 Conclusions and Further Work

This chapter presents the key findings of the work. It begins by discussing the findings regarding the validity of the hypothesis. Next, it presents the knowledge outcomes and associated technical outcomes of this research. Finally, it discusses research questions and potential applications emerging from this work.

8.1 Hypothesis Analysis

At the beginning of this work, the following hypothesis was proposed in response to the problems of information overload, disorientation and cognitive overload facing users of the World Wide Web.

Hypothesis

It is possible to create an ontology of context that can be used to create tools that users perceive to be useful and easy to use when performing information-seeking behaviours on the World Wide Web.

The research presented in this work focuses on establishing the validity of this hypothesis. To this end, an action research-based method that investigates the hypothesis by developing and working through a set of research questions has been applied. Based on the outcomes of the investigations presented in the previous chapter, this work concludes that there is evidence to support the validity of the hypothesis. This conclusion is based on the outcomes of the research questions, as discussed throughout this work. The research action approach has involved questions that develop concepts and models that inform the research process, and questions that contribute new knowledge that is used to assess the validity of the hypothesis. The questions that contribute new knowledge are discussed below, followed by an analysis of how the knowledge derived from the questions relates to the validity of the hypothesis.

Is ontology an appropriate way to model context? Ontology provides a way of structuring information in a way that captures semantic relationships between the information elements. This is appropriate for modelling context because it supports the expression of relationships between contextual elements. Context is a collection of information entities and relationships between them. A knowledge engineering methodology can be used to create a model of this information. This model can then be formally defined this model using the Ontology Web Language (OWL).

Contextual information represented using an ontology can be used to support common approaches to contextualisation. Two mechanisms that are widely used are trails, such as history or favourites, and summaries. This work creates trail-based interfaces that display dynamic trails of the users behaviour based on information in the model of context. These trails act as signposts that help mitigate the effects of user disorientation. It also creates summaries that highlight links and data based on semantic knowledge of the document structure and on link ranking heuristics. Using the ontology in this way shows that it can be used to support different approaches to the contextualisation of information.

How can an ontology of context be integrated into existing web infrastructure? The ontology can be integrated with existing web infrastructure by wrapping it in a web service. The web service allows any developer to obtain the API that allows interaction with the service and to use or add to the users model of context. Using a web service to manage the interaction of components with the ontology of context means that a service-oriented architecture can be used to describe a system that contextualises information on the World Wide Web. Such an architecture can be abstracted in to an application framework. This framework describes the interaction of software components that support contextualisation. This work has described the ICU Framework, which is a SOA framework that uses an ontology to manage contextual information. This approach allows any application that can invoke web services to manage and retrieve contextual information.

Are the tools useful and easy-to-use?

PU and PEOU are factors measured by the Technology Acceptance Model (TAM). They have been shown to be reliable indicators of the likelihood that a technology such as a software tool will be used for a given task. A TAM evaluation was performed that involved having people use ISeeYou while browsing and searching the World Wide Web and then fill out a survey. The set of TAM survey questions was created based on previous TAM research. The results of the survey indicate that users perceived ISeeYou to be useful and easy-to-use.

The outcomes of these research questions provide evidence that the hypothesis is valid.

8.2 Research Outcomes

The focus of this work has been the investigation of the hypothesis. As a result of this investigation, there are several outcomes. These outcomes can be grouped in to two categories. The first category is new knowledge. These are assertions of fact that can, after this investigation, be made with supporting evidence to back them up. The second category is a collection of technical and analytical knowledge that informs this work but has not been rigourously investigated to determine its validity.

8.2.1 New Knowledge

After investigating the hypothesis through the research question, the following are now known:

1. *Ontology is an appropriate approach to representing contextual information.* This work is based on the idea that existing approaches to contextualisation suffer from a lack of interoperability and an application specific approach to managing contextual information. The investigation in chapter 4 shows that an ontology can be used to manage contextual information. Chapter 4 also showed that ontologies have characteristics such as formality and reusability that make them appropriate for representing contextual information. The ability to share and reuse ontologies across applications and even between domains opens up new approaches to contextualisation.
2. *It is possible to realise an ontological representation of context that enables the sharing of contextual information between applications.* While it is possible to share contextual information between applications in existing system, support for this kind of sharing is not explicitly offered. Rather, it must be achieved by third party applications that have the rules for data formats built in to them. The approach developed in this work explicitly supports interoperability between applications that contextualise information by creating a software framework that uses web services to publish an API for managing contextual information. This approach allows contextual information to be aggregated in one location and structured in a way that allows applications to access the entire set of a user's contextual information, rather than just the elements created by that particular application. The discussion in chapter 6 shows how such a framework can be constructed. The framework and implementation presented in chapter 6 take idea from conjecture to a concrete demonstration of the utility of such an approach.
3. *The representation can be implemented in tools that are*

perceived by users as useful when undertaking tasks involving contextualisation. The representation can be used to implement tools that are perceived by users as useful when undertaking tasks involving contextualisation. A key contribution of this work is demonstrating that a new approach to managing contextual information can be applied to creating tools that contextualise information. The ability to create tools that are useful using the framework based on ontology is taken from being an interesting possibility to a demonstrable fact by this work. This result encourages further work to be performed into using this approach to managing contextual information for contextualising information across different applications. It is important to note here that it is not the tools that are the research outcome. Rather, it is the ability to create useful tools using an ontology of context that is significant.

4. *It is possible to use ontology to model and share contextual information about a user.* The implementation of ISeeYou demonstrates the utility of ontology for managing contextual information. The new work in the thesis is a demonstration of how a personal information management engine can be constructed using an ontology, and how this engine can be used to support existing approaches to contextualisation in a way that supports cross-application sharing of contextual information. Existing approaches use custom data stores that only support the task for which they were developed. The approach developed in this work shows that ontology can be used to model a users context. The fact that the ontology is not tied to any specific implementation of an application means that the contextual information can be reused across different applications.

8.2.2 Technical and Conceptual Knowledge

This research has produced the following technical and conceptual contributions:

- A concept of context for web-based systems is developed.
- An analysis of existing approaches to using context to augment information seeking.
- An understanding of the role of context in helping users find information.
- A model showing how context can be applied to information seeking.
- A reusable software framework for developing context-based tools.
- A demonstration of the construction of context-based tools using the software framework.
- A methodology for evaluating context-based tools. A set of tools that help users find information on the World Wide Web.

The technical and analytical contributions noted here will contribute to further investigations of research questions arising from this work. Combining these outcomes with the knowledge contributions of this thesis leads us to a number of future research and development directions.

8.3 Further Research and Development

The outcomes of this work encourage the pursuit of further research in to the development of ontology-based models of context that augment information-seeking behaviours on the World Wide Web. This work builds on the existing research in to information seeking, personalisation and contextualisation to develop a framework that supports the development of software tools that help users find information on the World Wide Web. The outcomes of this work suggest further research possibilities in several directions. First, there are areas of this research that would benefit from a deeper exploration. Second, there are areas of this research that would benefit from broader integration with existing research. Finally, there are opportunities for developing tools that can be distributed to help users find information on the World Wide Web. These areas can be called deeper research, wider research, and applied research. Each of these is discussed in turn.

8.3.1 Research Questions

This work has focussed on a specific research question that is expressed in the hypothesis. Having answered this question, there are now further questions arising that are worth pursuing. Of particular interest for the researchers involved in this project are three general questions.

To what extent can contextual information be shared between applications? The investigation in this work has shown that contextual information can be made available using a SOA-based framework. Using a web service to publish the API's means that any application that can invoke web services can manage contextual information. This shows the mechanism for sharing contextual information between applications. What is still unknown is whether it is meaningful to share contextual information between applications. In the example of information-seeking behaviours used in this work, an assumption has been made that doing so is meaningful, since different browsers store similar kinds of contextual information such as browser history and favourites. A first step in exploring this issue would be to create concrete examples of cross-browser contextualisation to determine the validity of this assumption. The goal of this investigation would be to identify elements of context that are application specific and elements that are generic, and to assess what contextual information should be managed to support contextualisation of different applications.

To what extent can contextual information be shared across domains? Contextualisation is, in theory, not limited to a single domain of applications. This research has investigated the domain of information-seeking behaviours. Within this domain, there is scope for further investigation as noted in the previous questions discussion. Beyond this, however, is the question of managing contextual information and applying contextualisation across different application domains. Such an investigation would look at case studies such as sharing contextual information between a web browser and a word processor, or between a spreadsheet and an email tool. One goal of such investigations would be to identify the extent to which contextual elements can be reused across domains, and also the extent to which ontologies themselves could be reused across domains. Another goal would be to identify whether different domains have unique requirements that alter the way ontologies are used or the way the framework is structured and/or applied.

What is the right ontology to use? The ontology of context developed in this work is acknowledged to be nothing more than a proof-of-concept based on the researchers knowledge of the domain. A significant research question emerging from this work is how can such ontologies be created in a rigourous way that captures all the necessary contextual information to support contextualisation within a knowledge domain. An associated question is how is the set of necessary contextual information identified? It may be that a domain requires a number of ontologies within it that capture different views of context. Investigating the issues around the creation of ontologies is an ongoing research area emerging from this research.

The work presented in this thesis has established that ontology can be an effective tool in supporting contextualisation. This is only a beginning and this result serves as the starting point for a range of further investigation rather than being an end in itself. The value of the hypothesis is that it possible to move forward from asking 'can ontology be used' and 'how can an ontology be used' to asking questions like 'what ontology should be used' and 'what new things can be done with ontology'. The outcomes from this work provide a platform on which to base further research.

8.3.2 Applications

Existing approaches to World Wide Web browsers focuses on large stand-alone applications like Internet Explorer and Firefox. The user interaction with the World Wide Web is handled by dynamic scripts embedded in the content being viewed. Given the variety of user contexts and the wide variety of tasks that user perform on the World Wide Web, the idea of task-specific and context-specific information interfaces, rather than a single interface trying to be all things to all people, holds promise for a new front in developing tools for managing information on the World Wide Web.

There is a range of tools for authoring and publishing information. Yet for viewing information, users are generally restricted to the choice of two web browsers. The availability of plug-ins for these browsers is, as seen in this work, an avenue for providing users with tools that augment their information behaviour. This work would benefit from being browser independent, and being available as tools the users can access for managing information as appropriate.

The use of a knowledge-based approach to modelling context could readily be applied to the development of the Database of Intentions described by John Battelle (Battelle 2005). By building knowledge of the user's link clicks and queries, the Context Store contains knowledge about the user's intentions. These intentions are not only useful for providing navigation and search aids, but also as a source of information for targeting marketing information.

A further step here would be to share the Context Store between more than one user, thus building a Database of Intentions for a community. This idea of groups of users is included in the ontology but no use of the ontologies ability to manage groups is made. The service-based, distributed architecture of the ICU framework allows this kind of functionality to be implemented. One possibility would be to have community stores in addition to personal stores. The community knowledge would then be useful for providing navigation aids and marketing to a user based on the interests that have been expressed by the group.

8.4 Conclusion

This work has investigated how ontology can be used to manage contextual information in support of contextualisation. The discussion has used contextualising information on the World Wide Web during information-seeking behaviours to make the ideas being developed concrete, but this is by no means the only domain to which this approach can be applied. A concept of context for such systems has been developed, and a method for modelling context based on this concept has been developed. A software framework has been developed that allows these concepts and this model to be used in the development of tools that help users overcome the problems of information overload and disorientation when seeking information on the World Wide Web. Finally a technique for evaluation such tools has been described, which was used to show that a group of users do find tools developed based on this approach useful for finding information on the World Wide Web. The results of this investigation encourage further research in to the concepts, models, framework and evaluation developed in this work.

Appendix A. Ontology of Context

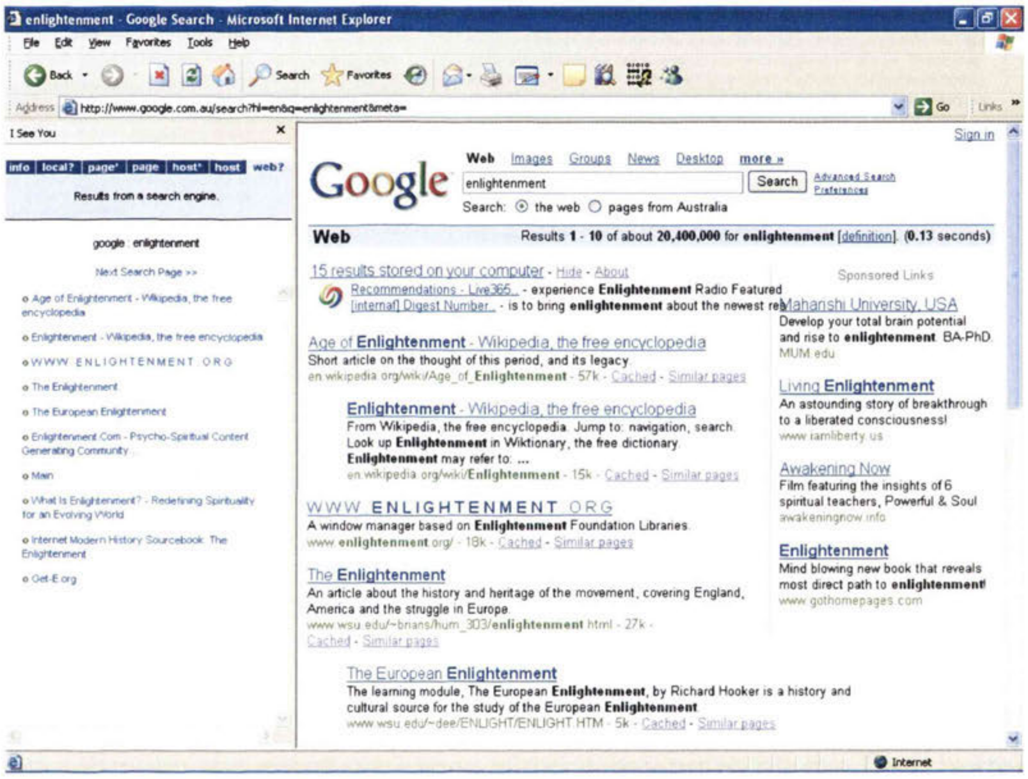
The following is a simple view of the OWL description of the ontology of context used to validate the classes and attributes identified as being part of the model.

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns="http://www.eng.uts.edu.au/web/context#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.eng.uts.edu.au/web/context">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Source"/>
  <owl:Class rdf:ID="Event"/>
  <owl:Class rdf:ID="Entity"/>
  <owl:Class rdf:ID="Account">
    <rdfs:subClassOf rdf:resource="#Source"/>
  </owl:Class>
  <owl:Class rdf:ID="UndirectedBrowse">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Browse"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="DirectedSearch">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Search"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Browse">
    <rdfs:subClassOf rdf:resource="#Event"/>
  </owl:Class>
  <owl:Class rdf:ID="DirectedBrowse">
    <rdfs:subClassOf rdf:resource="#Browse"/>
  </owl:Class>
  <owl:Class rdf:ID="BodyWord">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Element"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MetaTag">
    <rdfs:subClassOf rdf:resource="#Element"/>
  </owl:Class>
  <owl:Class rdf:about="#Search">
    <rdfs:subClassOf rdf:resource="#Event"/>
  </owl:Class>
  <owl:Class rdf:ID="UndirectedSearch">
    <rdfs:subClassOf rdf:resource="#Search"/>
  </owl:Class>
  <owl:Class rdf:ID="Group">
    <rdfs:subClassOf rdf:resource="#Source"/>
  </owl:Class>
  <owl:Class rdf:ID="WebPage">
    <rdfs:subClassOf rdf:resource="#Entity"/>
  </owl:Class>
  <owl:Class rdf:ID="Keyword">
    <rdfs:subClassOf rdf:resource="#MetaTag"/>
  </owl:Class>
  <owl:Class rdf:ID="Query">
    <rdfs:subClassOf rdf:resource="#Entity"/>
  </owl:Class>
  <owl:Class rdf:ID="User">
    <rdfs:subClassOf rdf:resource="#Source"/>
  </owl:Class>
  <owl:Class rdf:ID="Title">
    <rdfs:subClassOf rdf:resource="#Element"/>
  </owl:Class>
  <owl:Class rdf:ID="Host">
    <rdfs:subClassOf rdf:resource="#Element"/>
  </owl:Class>
  <owl:Class rdf:ID="ResultSet">
    <rdfs:subClassOf rdf:resource="#Entity"/>
  </owl:Class>
  <owl:ObjectProperty rdf:ID="WebPageProperties">
    <rdfs:range>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Element"/>
          <owl:Class rdf:about="#Event"/>
          <owl:Class rdf:about="#Source"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:range>
  </owl:ObjectProperty>
  <owl:DatatypeProperty rdf:ID="valueText">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="timeStamp">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="Text">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="visitCount">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="nameText">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="word">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="queryEngine">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="queryString">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="URL">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.2, Build 355)  http://protege.stanford.edu -->
```

Appendix B. ISeeYou Screenshots

The following screenshots show the ISeeYou useApp running in Internet Explorer 6.

Google Search (1)



enlightenment - Google Search - Microsoft Internet Explorer

Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=

I See You

info | local? | page' | page | host' | host | web?

Results from a search engine.

google : enlightenment

Next Search Page >>

- Age of Enlightenment - Wikipedia, the free encyclopedia
- Enlightenment - Wikipedia, the free encyclopedia
- WWW ENLIGHTENMENT ORG
- The Enlightenment
- The European Enlightenment
- Enlightenment Com - Psycho-Spiritual Content Generating Community
- Main
- What Is Enlightenment? - Redefining Spirituality for an Evolving World
- Internet Modern History Sourcebook: The Enlightenment
- Get-E org

Web Images Groups News Desktop more »

enlightenment [Search] Advanced Search Preferences

Search: (•) the web ( ) pages from Australia

Web Results **1** - **10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

15 results stored on your computer - Hide - About

Recommendations - Live365... - experience **Enlightenment** Radio Featured

[internal] Digest Number... - is to bring **enlightenment** about the newest re

Age of **Enlightenment** - Wikipedia, the free encyclopedia
Short article on the thought of this period, and its legacy.
en.wikipedia.org/wiki/Age_of_**Enlightenment** - 57k - Cached - Similar pages

Enlightenment - Wikipedia, the free encyclopedia
From Wikipedia, the free encyclopedia. Jump to: navigation, search. Look up **Enlightenment** in Wiktionary, the free dictionary. **Enlightenment** may refer to: ...
en.wikipedia.org/wiki/**Enlightenment** - 15k - Cached - Similar pages

WWW ENLIGHTENMENT ORG
A window manager based on **Enlightenment** Foundation Libraries.
www.**enlightenment**.org/ - 18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k - Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Sponsored Links

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD.
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

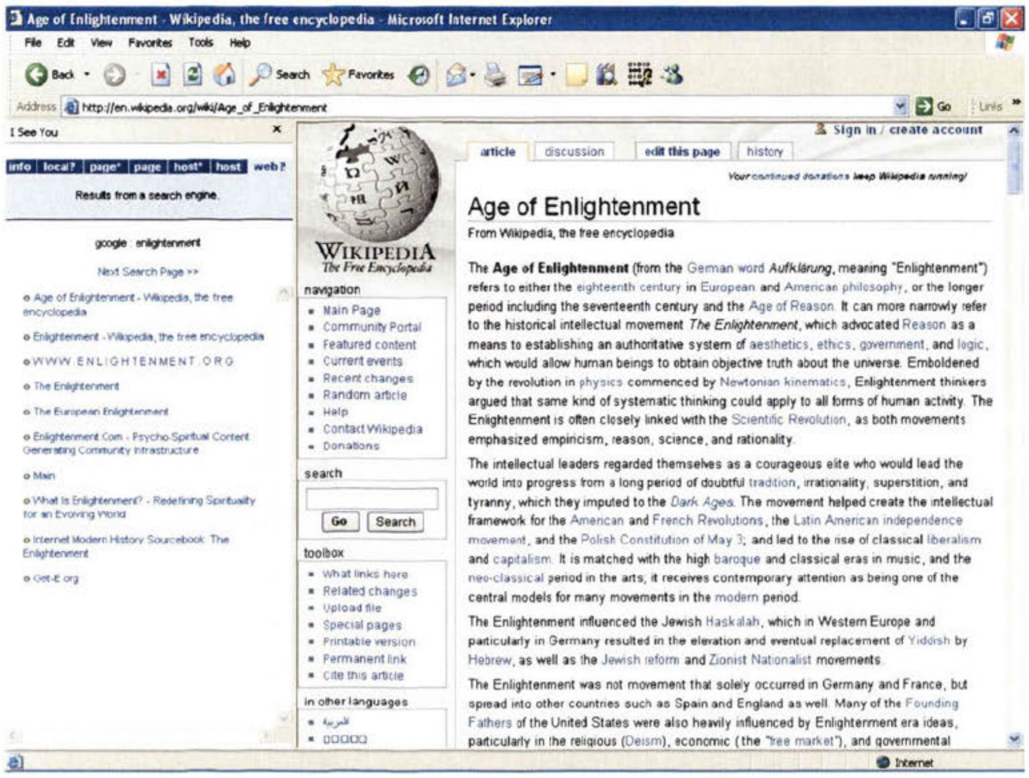
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-1 Google Search (1)

When a user of ISeeYou performs a search using the Google search engine, a summary of the result links is shown in the ISeeYou Band.

Google Search (2)


Age of Enlightenment - Wikipedia, the free encyclopedia - Microsoft Internet Explorer

Address http://en.wikipedia.org/wiki/Age_of_Enlightenment

I See You

info | local? | page* | page | host* | host | web?

Results from a search engine.

google : enlightenment

Next Search Page >>

- Age of Enlightenment - Wikipedia, the free encyclopedia
- Enlightenment - Wikipedia, the free encyclopedia
- WWW.ENLIGHTENMENT.ORG
- The Enlightenment
- The European Enlightenment
- Enlightenment.Com - Psycho-Spiritual Content Generating Community Infrastructure
- Main
- What Is Enlightenment? - Redefining Spirituality for an Evolving World
- Internet Modern History Sourcebook: The Enlightenment
- Get-E.org

Your continued donations keep Wikipedia running!

Age of Enlightenment

From Wikipedia, the free encyclopedia

The **Age of Enlightenment** (from the German word *Aufklärung*, meaning "Enlightenment") refers to either the eighteenth century in European and American philosophy, or the longer period including the seventeenth century and the Age of Reason. It can more narrowly refer to the historical intellectual movement *The Enlightenment*, which advocated Reason as a means to establishing an authoritative system of aesthetics, ethics, government, and logic, which would allow human beings to obtain objective truth about the universe. Emboldened by the revolution in physics commenced by Newtonian kinematics, Enlightenment thinkers argued that same kind of systematic thinking could apply to all forms of human activity. The Enlightenment is often closely linked with the Scientific Revolution, as both movements emphasized empiricism, reason, science, and rationality.

The intellectual leaders regarded themselves as a courageous elite who would lead the world into progress from a long period of doubtful tradition, irrationality, superstition, and tyranny, which they imputed to the *Dark Ages*. The movement helped create the intellectual framework for the American and French Revolutions, the Latin American independence movement, and the Polish Constitution of May 3; and led to the rise of classical liberalism and capitalism. It is matched with the high baroque and classical eras in music, and the neo-classical period in the arts; it receives contemporary attention as being one of the central models for many movements in the modern period.

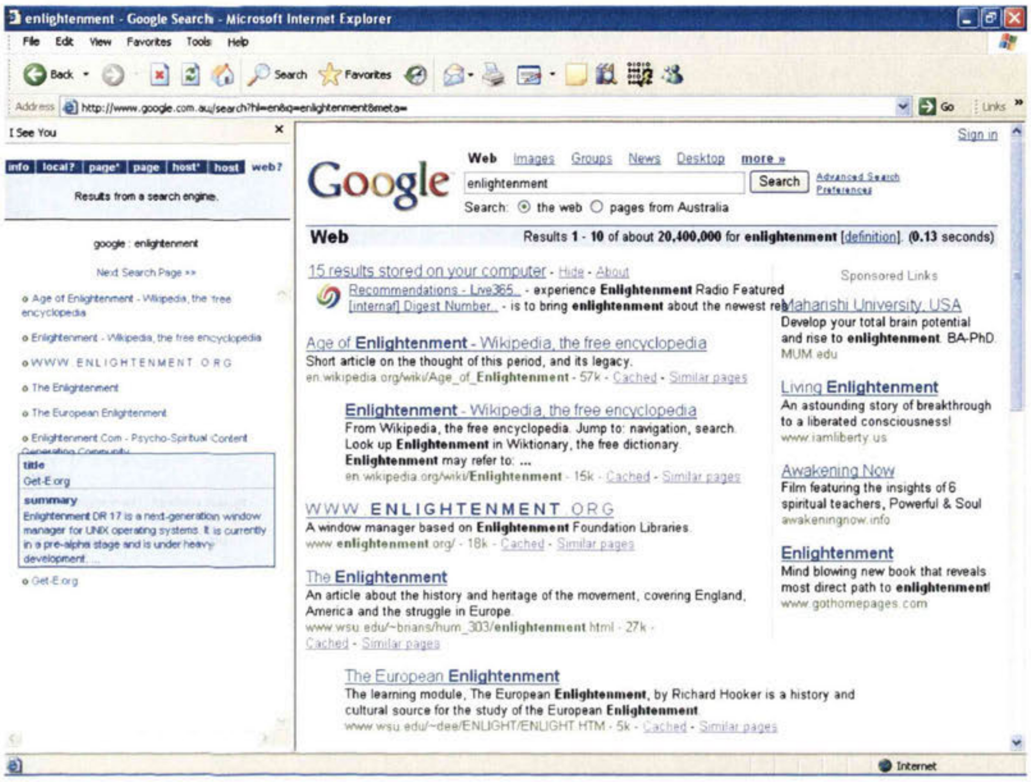
The Enlightenment influenced the Jewish Haskalah, which in Western Europe and particularly in Germany resulted in the elevation and eventual replacement of Yiddish by Hebrew, as well as the Jewish reform and Zionist Nationalist movements.

The Enlightenment was not movement that solely occurred in Germany and France, but spread into other countries such as Spain and England as well. Many of the Founding Fathers of the United States were also heavily influenced by Enlighterment era ideas, particularly in the religious (Deism), economic (the "free market"), and governmental

Figure B-2 Google Search (2)

The resultset is still visible when the user clicks on a link. The user can scan through the resultset pages without having to hit the back button all the time.

Google Search (3)


enlightenment - Google Search - Microsoft Internet Explorer

Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=

I See You

info | local? | page* | page | host* | host | web?

Results from a search engine.

google : enlightenment

Next Search Page »»

- Age of Enlightenment - Wikipedia, the free encyclopedia
- Enlightenment - Wikipedia, the free encyclopedia
- WWW ENLIGHTENMENT ORG
- The Enlightenment
- The European Enlightenment
- Enlightenment Com - Psycho-Spiritual Content Generating Community

title
Get-E org

summary
Enlightenment DR 17 is a next-generation window manager for UNIX operating systems. It is currently in a pre-alpha stage and is under heavy development. ...

- Get-E org

Web Images Groups News Desktop more »

enlightenment [Search] Advanced Search Preferences

Search: (•) the web ( ) pages from Australia

Web Results **1** - **10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

15 results stored on your computer - Hide - About
Recommendations - Live365... - experience **Enlightenment** Radio Featured
[internal] Digest Number... - is to bring **enlightenment** about the newest re

Age of **Enlightenment** - Wikipedia, the free encyclopedia
Short article on the thought of this period, and its legacy.
en.wikipedia.org/wiki/Age_of_**Enlightenment** - 57k - Cached - Similar pages

Enlightenment - Wikipedia, the free encyclopedia
From Wikipedia, the free encyclopedia. Jump to: navigation, search. Look up **Enlightenment** in Wiktionary, the free dictionary. **Enlightenment** may refer to: ...
en.wikipedia.org/wiki/**Enlightenment** - 15k - Cached - Similar pages

WWW **ENLIGHTENMENT** ORG
A window manager based on **Enlightenment** Foundation Libraries.
www.**enlightenment**.org/ - 18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k - Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Sponsored Links

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD.
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

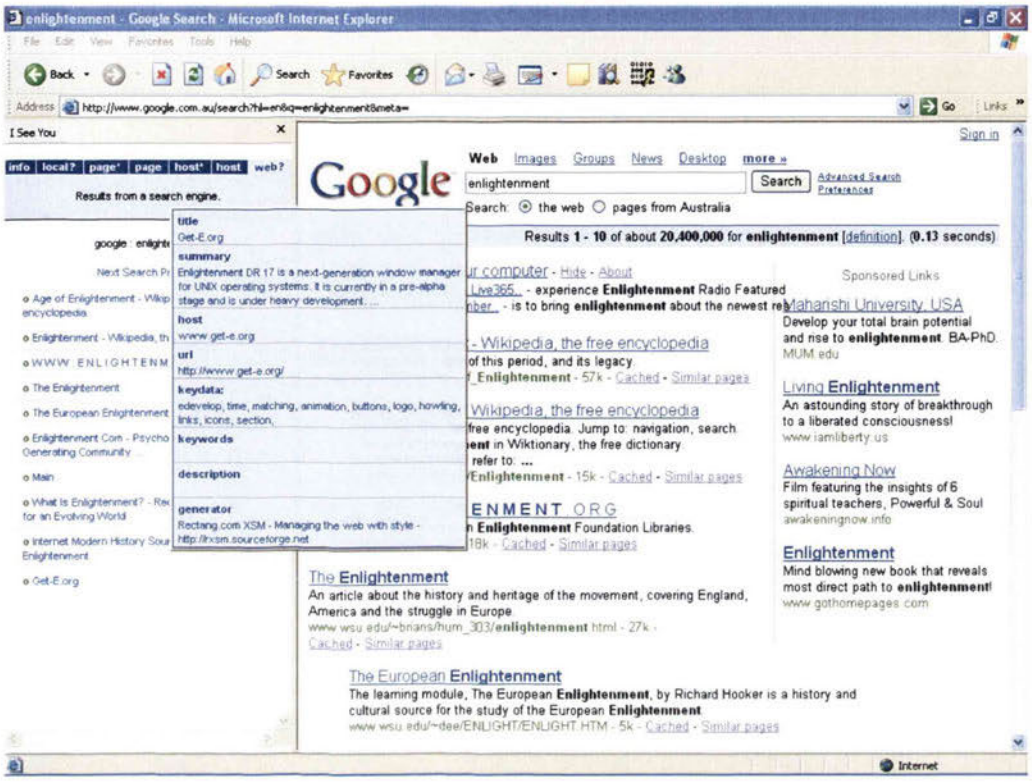
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-3 Google Search(3)

When the mouse cursor moves over a link in a resultset displayed in the ISeeYou Band, a summary of the pages details is shown in a small popup.

Google Search (4)


enlightenment - Google Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=
Go Links
I See You
info local? page' page host' host web?
Web Images Groups News Desktop more »
Search
Advanced Search
Preferences
Sign in
Sponsored Links
Internet

Results from a search engine.

google : enlight…

Next Search Pr…

- Age of Enlightenment - Wikip… encyclopedia
- Enlightenment - Wikipedia, th…
- WWW ENLIGHTENM…
- The Enlightenment
- The European Enlightenment
- Enlightenment Com - Psycho… Generating Community
- Main
- What Is Enlightenment? - Re… for an Evolving World
- Internet Modern History Sour… Enlightenment
- Get-E.org

title
Get-E.org
summary
Enlightenment DR 17 is a next-generation window manager for UNIX operating systems. It is currently in a pre-alpha stage and is under heavy development. ...
host
www.get-e.org
url
http://www.get-e.org/
keydata:
edevelop, time, matching, animation, buttons, logo, howling, links, icons, section,
keywords

description

generator
Rectang.com XSM - Managing the web with style - http://rxsm.sourceforge.net

enlightenment

Search: ⦿ the web ○ pages from Australia

Results **1 - 10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

…r computer - Hide - About
…Live365. - experience **Enlightenment** Radio Featured
…nber. - is to bring **enlightenment** about the newest re…

…- Wikipedia, the free encyclopedia
…of this period, and its legacy.
…f_**Enlightenment** - 57k - Cached - Similar pages

…Wikipedia, the free encyclopedia
…free encyclopedia. Jump to: navigation, search
…**ent** in Wiktionary, the free dictionary.
… refer to: ...
…/**Enlightenment** - 15k - Cached - Similar pages

…ENMENT.ORG
…n **Enlightenment** Foundation Libraries.
…18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k -
Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD.
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

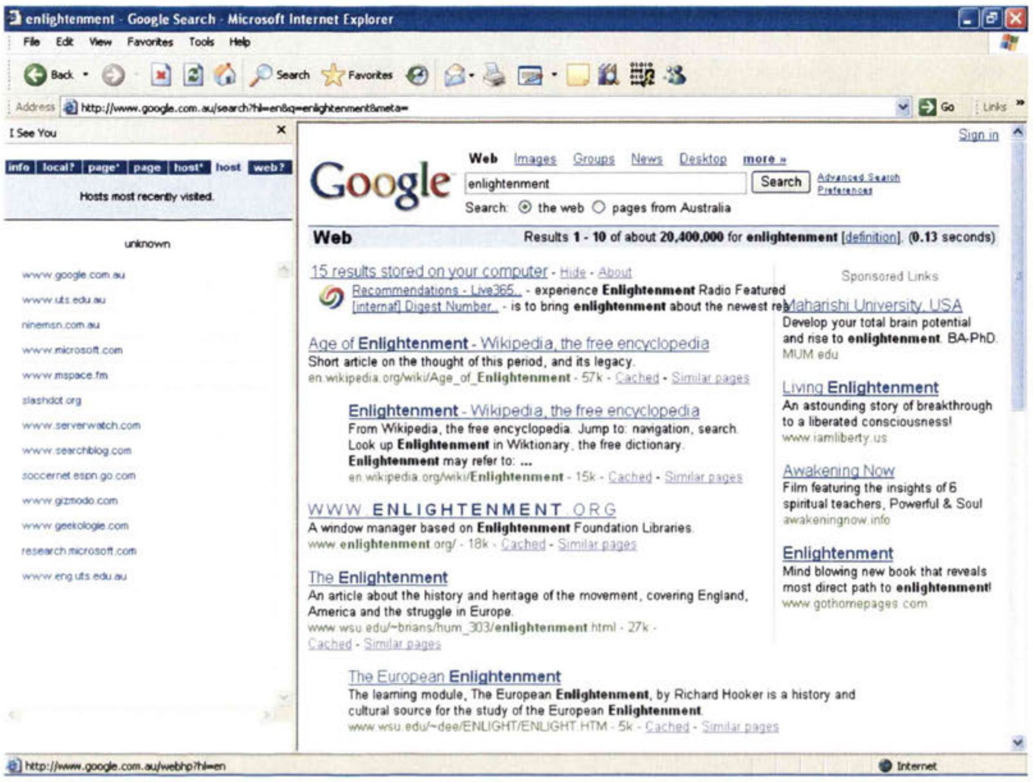
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-4 Google Search (4)

When the cursor is left over a link in the result set displayed in the ISeeYou Band for more than 5 seconds, a more detailed summary of the link based on a look-ahead retrieval of the page to which it points is shown.

Most Recent Hosts


enlightenment - Google Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=
Go Links
I See You
info local? page* page host* host web?
Sign in
Web Images Groups News Desktop more »
Search
Advanced Search
Preferences
http://www.google.com.au/webhp?hl=en
Internet

Hosts most recently visited.

unknown

www.google.com.au
www.uts.edu.au
ninemsn.com.au
www.microsoft.com
www.mspace.fm
slashdot.org
www.serverwatch.com
www.searchblog.com
soccernet.espn.go.com
www.gizmodo.com
www.geekologie.com
research.microsoft.com
www.eng.uts.edu.au

enlightenment

Search: ⦿ the web ○ pages from Australia

Web Results **1** - **10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

15 results stored on your computer - Hide - About

Recommendations - Live365... - experience **Enlightenment** Radio Featured

[internal] Digest Number... - is to bring **enlightenment** about the newest re

Age of **Enlightenment** - Wikipedia, the free encyclopedia
Short article on the thought of this period, and its legacy.
en.wikipedia.org/wiki/Age_of_**Enlightenment** - 57k - Cached - Similar pages

Enlightenment - Wikipedia, the free encyclopedia
From Wikipedia, the free encyclopedia. Jump to: navigation, search.
Look up **Enlightenment** in Wiktionary, the free dictionary.
Enlightenment may refer to: ...
en.wikipedia.org/wiki/**Enlightenment** - 15k - Cached - Similar pages

WWW.**ENLIGHTENMENT**.ORG
A window manager based on **Enlightenment** Foundation Libraries.
www.**enlightenment**.org/ - 18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k - Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Sponsored Links

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD.
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

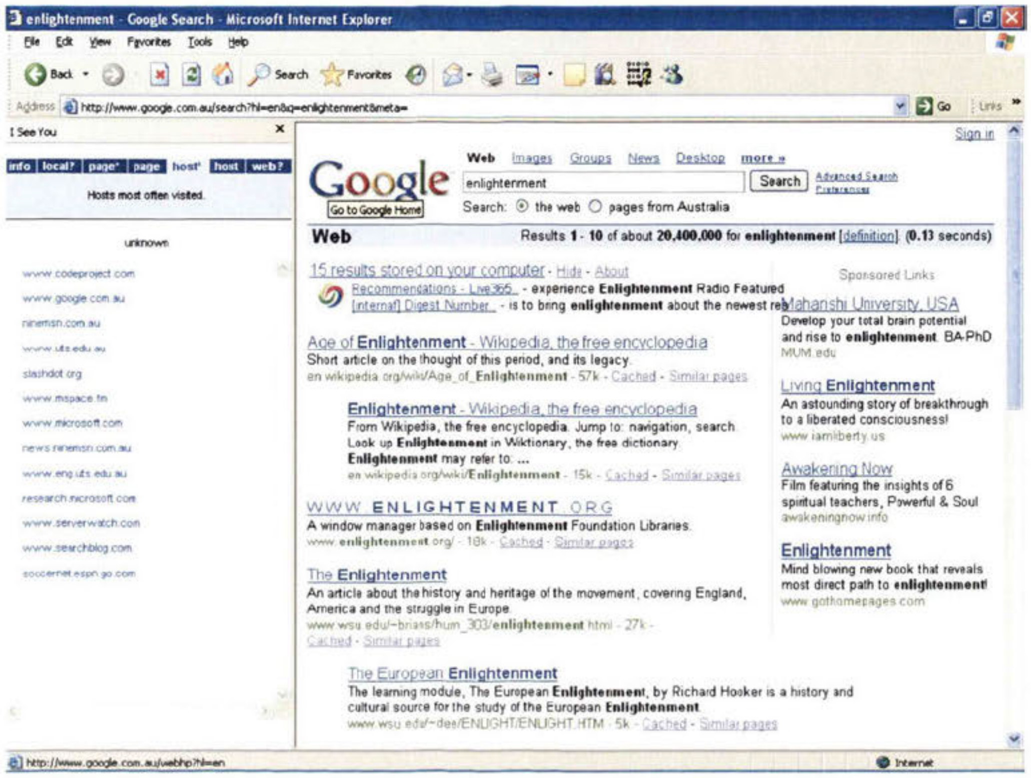
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-5 Most Recent Hosts

The host most recently visited view shows the list of hosts that have pages that have been visited recently. The list is constructed by showing unique host names from the most recently visited pages list. The length of the list is capped at 13. The list displays the domain name of the host. Clicking on the domain name causes the default page for that host to load.

Most Visited Hosts


enlightenment - Google Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=
Go Links
I See You
info local? page* page host* host web?
Sign in
Web Images Groups News Desktop more »
Search
Advanced Search
Preferences
Go to Google Home
http://www.google.com.au/webhp?hl=en
Internet

Hosts most often visited.

unknown

www.codeproject.com
www.google.com.au
ninemsn.com.au
www.uts.edu.au
slashdot.org
www.mspace.fm
www.microsoft.com
news.ninemsn.com.au
www.eng.uts.edu.au
research.microsoft.com
www.serverwatch.com
www.searchblog.com
soccernet.espn.go.com

enlightenment

Search: ⦿ the web ○ pages from Australia

Web Results **1** - **10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

15 results stored on your computer - Hide - About
Recommendations - Live365 - experience **Enlightenment** Radio Featured
[internal] Digest Number - is to bring **enlightenment** about the newest re

Age of **Enlightenment** - Wikipedia, the free encyclopedia
Short article on the thought of this period, and its legacy.
en.wikipedia.org/wiki/Age_of_**Enlightenment** - 57k - Cached - Similar pages

Enlightenment - Wikipedia, the free encyclopedia
From Wikipedia, the free encyclopedia. Jump to: navigation, search
Look up **Enlightenment** in Wiktionary, the free dictionary
Enlightenment may refer to: ...
en.wikipedia.org/wiki/**Enlightenment** - 15k - Cached - Similar pages

WWW.ENLIGHTENMENT.ORG
A window manager based on **Enlightenment** Foundation Libraries.
www.**enlightenment**.org/ - 18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k - Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Sponsored Links

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

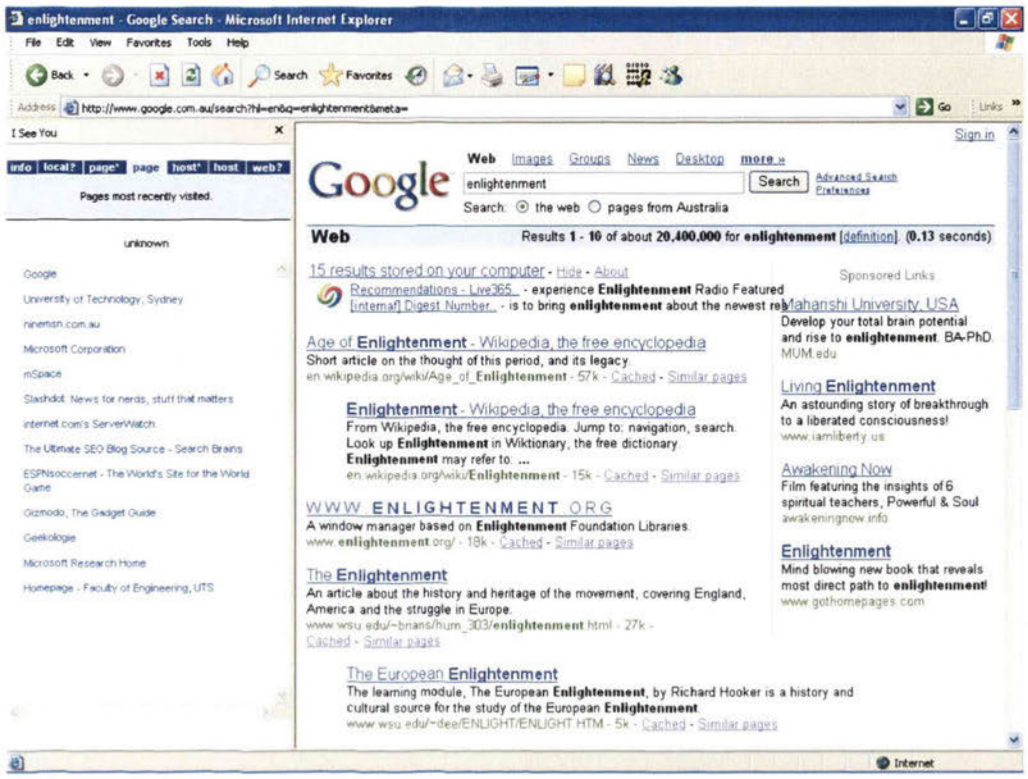
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-6 Most Visited Hosts

The hosts most often visited view shows the hosts that have the most pages visited by the user. The list is constructed by counting pages visited by the user grouped by the hostname part of their URL. The list is ordered descending from most visited down, and is capped at 13 entries. The list displays the domain name of the host. Clicking on the domain name causes the default page for that host to load.

Most Recent Pages


enlightenment - Google Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=
Go Links
I See You
info local? page* page host* host web?
Sign in
Web Images Groups News Desktop more »
Google
Search
Advanced Search
Preferences
Internet

Pages most recently visited.

unknown

Google
University of Technology, Sydney
ninemsn.com.au
Microsoft Corporation
mSpace
Slashdot: News for nerds, stuff that matters
internet.com's ServerWatch
The Ultimate SEO Blog Source - Search Brains
ESPNsoccernet - The World's Site for the World Game
Gizmodo, The Gadget Guide
Geekologie
Microsoft Research Home
Homepage - Faculty of Engineering, UTS

enlightenment

Search: ⦿ the web ○ pages from Australia

Web Results **1** - **10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

15 results stored on your computer - Hide - About
Recommendations - Live365 - experience **Enlightenment** Radio Featured
[internal] Digest Number.. - is to bring **enlightenment** about the newest re

Age of **Enlightenment** - Wikipedia, the free encyclopedia
Short article on the thought of this period, and its **legacy**.
en.wikipedia.org/wiki/Age_of_**Enlightenment** - 57k - Cached - Similar pages

Enlightenment - Wikipedia, the free encyclopedia
From Wikipedia, the free encyclopedia. Jump to: navigation, search. Look up **Enlightenment** in Wiktionary, the free dictionary. **Enlightenment** may refer to: ...
en.wikipedia.org/wiki/**Enlightenment** - 15k - Cached - Similar pages

WWW.**ENLIGHTENMENT**.ORG
A window manager based on **Enlightenment** Foundation Libraries.
www.**enlightenment**.org/ - 18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k - Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Sponsored Links

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD.
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

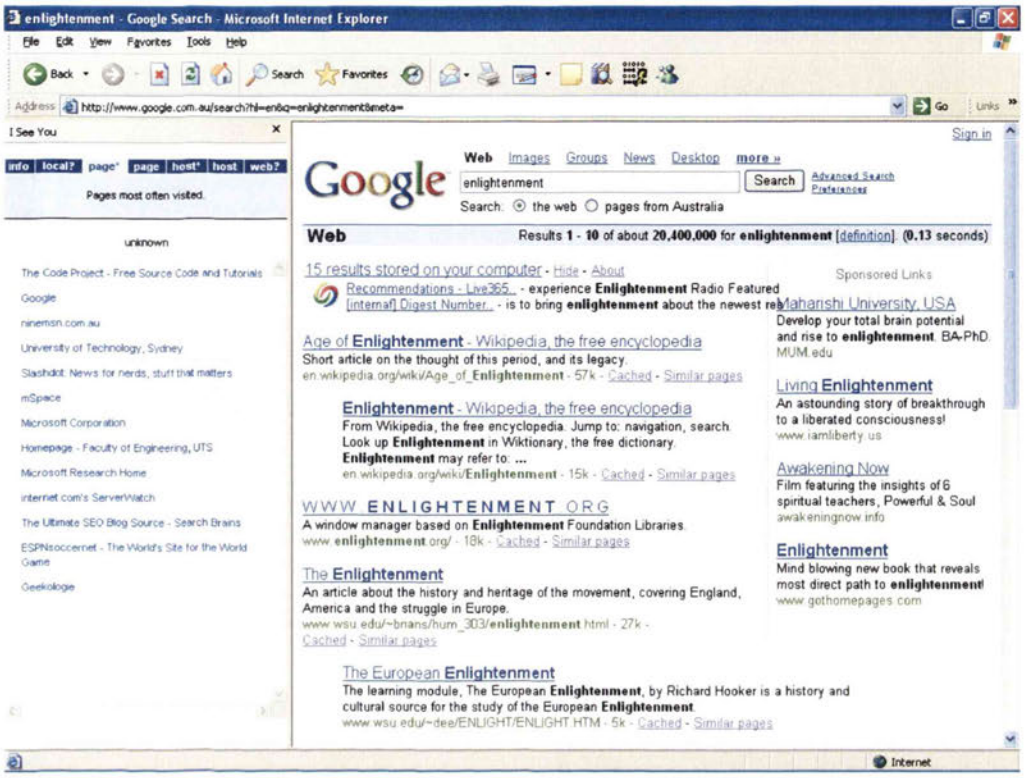
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-7 Most Recent Pages

The pages most often visited view shows the titles of the pages most often visited by the user. Clicking on the title causes the main window to navigate to the page. The list is ordered descending from most recent, and is capped at 13 entries.

Most Visited Pages


enlightenment - Google Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.google.com.au/search?hl=en&q=enlightenment&meta=
Go
Links
I See You
info local? page* page host* host web?
Web Images Groups News Desktop more »
Search
Advanced Search
Preferences
Sign in
Internet

Pages most often visited.

unknown

The Code Project - Free Source Code and Tutorials

Google

ninemsn.com.au

University of Technology, Sydney

Slashdot: News for nerds, stuff that matters

mSpace

Microsoft Corporation

Homepage - Faculty of Engineering, UTS

Microsoft Research Home

internet.com's ServerWatch

The Ultimate SEO Blog Source - Search Brains

ESPNsoccernet - The World's Site for the World Game

Geekologie

enlightenment

Search: ⦿ the web ○ pages from Australia

Web Results **1** - **10** of about **20,400,000** for **enlightenment** [definition]. (**0.13** seconds)

15 results stored on your computer - Hide - About

Recommendations - Live365... - experience **Enlightenment** Radio Featured

[internal] Digest Number... - is to bring **enlightenment** about the newest re

Age of **Enlightenment** - Wikipedia, the free encyclopedia
Short article on the thought of this period, and its legacy.
en.wikipedia.org/wiki/Age_of_**Enlightenment** - 57k - Cached - Similar pages

Enlightenment - Wikipedia, the free encyclopedia
From Wikipedia, the free encyclopedia. Jump to: navigation, search. Look up **Enlightenment** in Wiktionary, the free dictionary. **Enlightenment** may refer to: ...
en.wikipedia.org/wiki/**Enlightenment** - 15k - Cached - Similar pages

WWW.**ENLIGHTENMENT**.ORG
A window manager based on **Enlightenment** Foundation Libraries.
www.**enlightenment**.org/ - 18k - Cached - Similar pages

The **Enlightenment**
An article about the history and heritage of the movement, covering England, America and the struggle in Europe.
www.wsu.edu/~brians/hum_303/**enlightenment**.html - 27k - Cached - Similar pages

The European **Enlightenment**
The learning module, The European **Enlightenment**, by Richard Hooker is a history and cultural source for the study of the European **Enlightenment**.
www.wsu.edu/~dee/ENLIGHT/ENLIGHT.HTM - 5k - Cached - Similar pages

Sponsored Links

Maharishi University, USA
Develop your total brain potential and rise to **enlightenment**. BA-PhD.
MUM.edu

Living **Enlightenment**
An astounding story of breakthrough to a liberated consciousness!
www.iamliberty.us

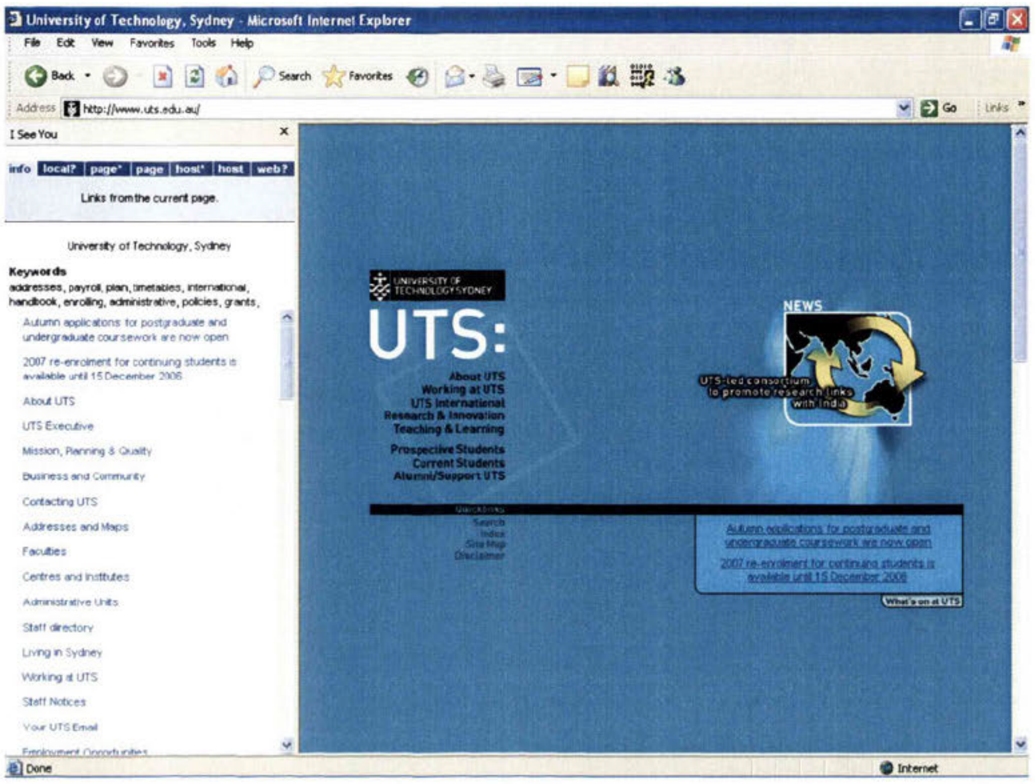
Awakening Now
Film featuring the insights of 6 spiritual teachers, Powerful & Soul
awakeningnow.info

Enlightenment
Mind blowing new book that reveals most direct path to **enlightenment**!
www.gothomepages.com

Figure B-8 Most Visited Pages

The pages most recently visited view shows the titles of the pages most recently visited by the user. Clicking on the title causes the main window to navigate to the page. The list is ordered descending from most recent, and is capped at 13 entries.

Page Information


University of Technology, Sydney - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.uts.edu.au/
Go Links
I See You
info local? page* page host* host web?
UTS:
NEWS
Done
Internet

Links from the current page.

University of Technology, Sydney

Keywords
addresses, payroll, plan, timetables, international, handbook, enrolling, administrative, policies, grants,

Autumn applications for postgraduate and undergraduate coursework are now open

2007 re-enrolment for continuing students is available until 15 December 2006

About UTS

UTS Executive

Mission, Planning & Quality

Business and Community

Contacting UTS

Addresses and Maps

Faculties

Centres and Institutes

Administrative Units

Staff directory

Living in Sydney

Working at UTS

Staff Notices

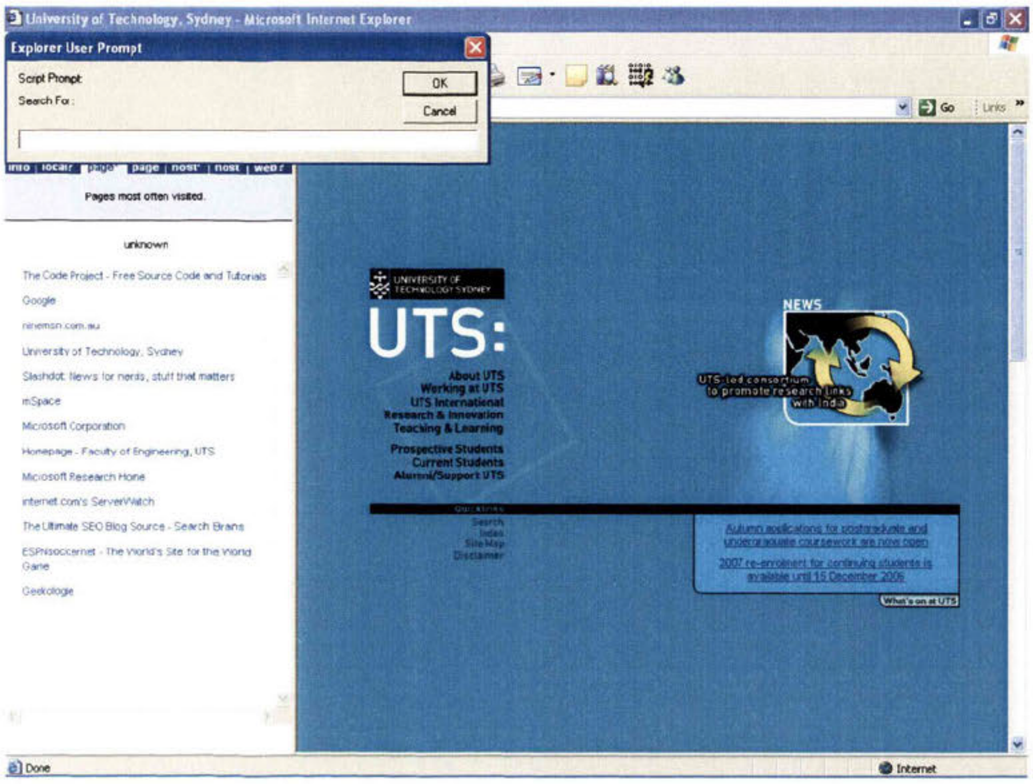
Your UTS Email

Employment Opportunities

Figure B-9 Page Information

The info option shows the keywords associated with a page, followed by all the links contained in the current page. The links are is ordered using a heuristic that analyses how often the user clicks links containing words in the anchor text for the link in the page.

Local Search (1)


University of Technology, Sydney - Microsoft Internet Explorer
Explorer User Prompt
OK
Cancel
info | local? | page? | page | host? | host | web?
Go
Links
Done
Internet

Script Prompt
Search For :

Pages most often visited.

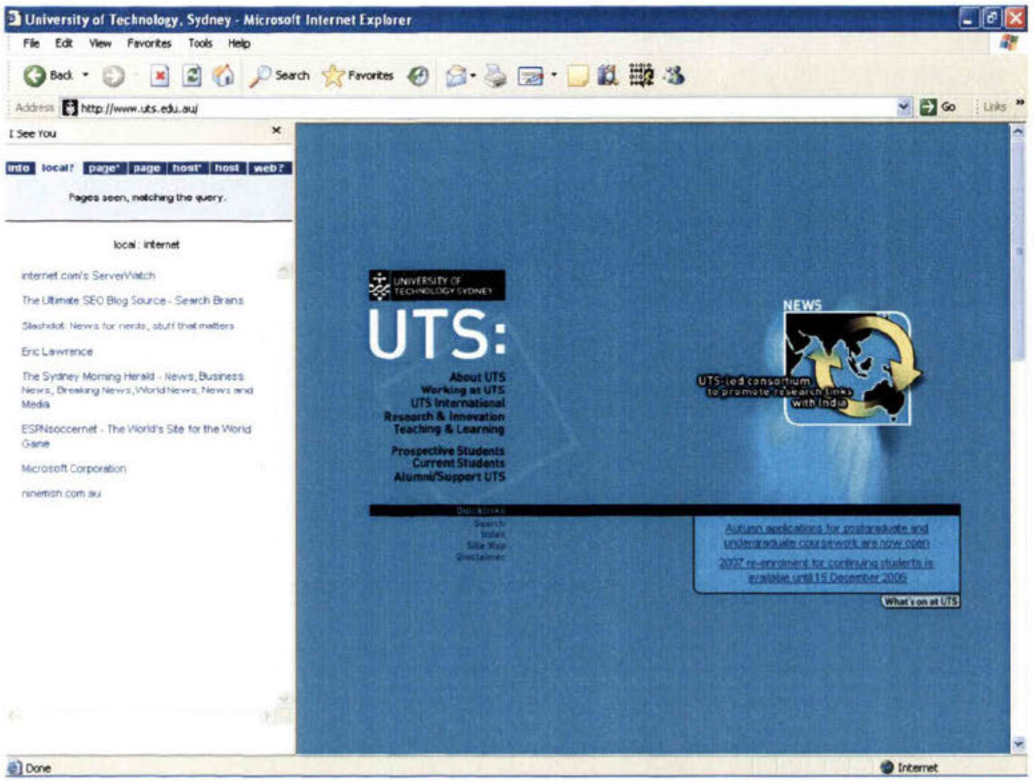
unknown

The Code Project - Free Source Code and Tutorials
Google
ninemsn.com.au
University of Technology, Sydney
Slashdot: News for nerds, stuff that matters
mSpace
Microsoft Corporation
Homepage - Faculty of Engineering, UTS
Microsoft Research Home
internet.com's ServerWatch
The Ultimate SEO Blog Source - Search Brains
ESPNsoccernet - The World's Site for the World Game
Geekologie

Figure B-10 Local Search (1)

The local search prompts the user to enter a search string. The user types one or more keywords and clicks OK.

Local Search (2)


University of Technology, Sydney - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address http://www.uts.edu.au/
Go
Links
I See You
info local? page' page host' host web?
UTS:
About UTS
Working at UTS
UTS International
Research & Innovation
Teaching & Learning
Prospective Students
Current Students
Alumni/Support UTS
NEWS
UTS-led consortium to promote research links with India
What's on at UTS
Done
Internet

Pages seen, matching the query.

local : internet

internet.com's ServerWatch

The Ultimate SEO Blog Source - Search Brains

Slashdot: News for nerds, stuff that matters

Eric Lawrence

The Sydney Morning Herald - News, Business News, Breaking News, World News, News and Media

ESPNsoccernet - The World's Site for the World Game

Microsoft Corporation

ninemsn.com.au

Figure B-11 Local Search (2)

A result set containing all the pages the user has visited that contain the words in the search string is displayed. The result set is ordered using a heuristic that analyses how often the user clicks links containing words in the anchor text for the link in the result set.

Appendix C. IContextStoreService Interface

This appendix contains the WSDL description of the interface published by the Message Service of the Collection component of ISeeYou. This interface is used by agents and interface that identify and use context in ISeeYou.

```
<?xml version="1.0" encoding="UTF-8"?>

<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:ns1="http://dom.w3c.org"
xmlns:soap11="http://schemas.xmlsoap.org/soap/envelope/" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope" xmlns:soapenc11="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soapenc12="http://www.w3.org/2003/05/soap-encoding" xmlns:tns="http://localhost"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://localhost">

  <wsdl:types>
   <xsd:schema targetNamespace="http://localhost" elementFormDefault="qualified"
attributeFormDefault="qualified">
    <xsd:element name="getCtxInfo">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
    <xsd:element name="getCtxInfoResponse">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
    <xsd:element name="getMsg">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
    <xsd:element name="getMsgResponse">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="out" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
    <xsd:element name="getSearchItemById">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
       <xsd:element name="in1" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
```

```
<xsd:element name="getSearchItemByIdResponse">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="getStoreSearch">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="getStoreSearchResponse">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="processEntity">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="processEntityResponse">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="checkSearchStatusById">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   <xsd:element name="in1" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="checkSearchStatusByIdResponse">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="getPageInfo">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="getPageInfoResponse">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="addDocumentContext">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="addDocumentContextResponse">
```

```
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="addFavourite">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="addFavouriteResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="checkSearchEntity">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="checkSearchEntityResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getHistory">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getHistoryResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getStatus">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getStatusResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="checkEntity">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="checkEntityResponse">
  <xsd:complexType>
   <xsd:sequence>
```

```
    <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getEntityStatus">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getEntityStatusResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="showFreqWords">
  <xsd:complexType />
 </xsd:element>
 <xsd:element name="showFreqWordsResponse">
  <xsd:complexType />
 </xsd:element>
 <xsd:element name="initPrevayler">
  <xsd:complexType />
 </xsd:element>
 <xsd:element name="initPrevaylerResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="xsd:boolean" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="addClick">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="addClickResponse">
  <xsd:complexType />
 </xsd:element>
 <xsd:element name="getSearchInfo">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="getSearchInfoResponse">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="showPageMap">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="showPageMapResponse">
  <xsd:complexType />
 </xsd:element>
 <xsd:element name="testMe">
  <xsd:complexType />
 </xsd:element>
 <xsd:element name="testMeResponse">
```

```
    <xsd:complexType />
   </xsd:element>
   <xsd:element name="addPage">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="addPageResponse">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="getUUID">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="in0" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="getUUIDResponse">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="out" type="ns1:ArrayOfElement" nillable="true" minOccurs="1" maxOccurs="1" />
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="getStyleSheet">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="in0" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="getStyleSheetResponse">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="out" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1" />
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:schema>
  <xsd:schema targetNamespace="http://dom.w3c.org" elementFormDefault="qualified"
attributeFormDefault="qualified">
   <xsd:complexType name="ArrayOfElement">
    <xsd:sequence>
     <xsd:element name="Element" type="ns1:Element" nillable="true" minOccurs="0"
maxOccurs="unbounded" />
    </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="Element">
    <xsd:sequence>
     <xsd:element name="schemaTypeInfo" type="ns1:TypeInfo" minOccurs="0" nillable="true" />
     <xsd:element name="tagName" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="attributes" type="ns1:NamedNodeMap" minOccurs="0" nillable="true" />
     <xsd:element name="baseURI" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="childNodes" type="ns1:NodeList" minOccurs="0" nillable="true" />
     <xsd:element name="firstChild" type="ns1:Node" minOccurs="0" nillable="true" />
     <xsd:element name="lastChild" type="ns1:Node" minOccurs="0" nillable="true" />
     <xsd:element name="localName" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="namespaceURI" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="nextSibling" type="ns1:Node" minOccurs="0" nillable="true" />
     <xsd:element name="nodeName" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="nodeType" type="xsd:short" minOccurs="0" />
     <xsd:element name="nodeValue" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="ownerDocument" type="xsd:anyType" minOccurs="0" />
     <xsd:element name="parentNode" type="ns1:Node" minOccurs="0" nillable="true" />
     <xsd:element name="prefix" type="xsd:string" minOccurs="0" nillable="true" />
     <xsd:element name="previousSibling" type="ns1:Node" minOccurs="0" nillable="true" />
```

```
      <xsd:element name="textContent" type="xsd:string" minOccurs="0" nillable="true" />
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="NamedNodeMap">
    <xsd:sequence>
      <xsd:element name="length" type="xsd:int" minOccurs="0" />
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Node">
    <xsd:sequence>
      <xsd:element name="attributes" type="ns1:NamedNodeMap" minOccurs="0" nillable="true" />
      <xsd:element name="baseURI" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="childNodes" type="ns1:NodeList" minOccurs="0" nillable="true" />
      <xsd:element name="firstChild" type="ns1:Node" minOccurs="0" nillable="true" />
      <xsd:element name="lastChild" type="ns1:Node" minOccurs="0" nillable="true" />
      <xsd:element name="localName" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="namespaceURI" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="nextSibling" type="ns1:Node" minOccurs="0" nillable="true" />
      <xsd:element name="nodeName" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="nodeType" type="xsd:short" minOccurs="0" />
      <xsd:element name="nodeValue" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="ownerDocument" type="xsd:anyType" minOccurs="0" />
      <xsd:element name="parentNode" type="ns1:Node" minOccurs="0" nillable="true" />
      <xsd:element name="prefix" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="previousSibling" type="ns1:Node" minOccurs="0" nillable="true" />
      <xsd:element name="textContent" type="xsd:string" minOccurs="0" nillable="true" />
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="NodeList">
    <xsd:sequence>
      <xsd:element name="length" type="xsd:int" minOccurs="0" />
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="TypeInfo">
    <xsd:sequence>
      <xsd:element name="typeName" type="xsd:string" minOccurs="0" nillable="true" />
      <xsd:element name="typeNamespace" type="xsd:string" minOccurs="0" nillable="true" />
    </xsd:sequence>
  </xsd:complexType>
 </xsd:schema>
</wsdl:types>
<wsdl:message name="getStatusResponse">
 <wsdl:part element="tns:getStatusResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="addPageResponse">
 <wsdl:part element="tns:addPageResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="checkEntityRequest">
 <wsdl:part element="tns:checkEntity" name="parameters" />
</wsdl:message>
<wsdl:message name="showPageMapResponse">
 <wsdl:part element="tns:showPageMapResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="showFreqWordsResponse">
 <wsdl:part element="tns:showFreqWordsResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="testMeRequest">
 <wsdl:part element="tns:testMe" name="parameters" />
</wsdl:message>
<wsdl:message name="addFavouriteResponse">
 <wsdl:part element="tns:addFavouriteResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getStyleSheetResponse">
 <wsdl:part element="tns:getStyleSheetResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getStatusRequest">
 <wsdl:part element="tns:getStatus" name="parameters" />
</wsdl:message>
<wsdl:message name="showPageMapRequest">
 <wsdl:part element="tns:showPageMap" name="parameters" />
</wsdl:message>
<wsdl:message name="addFavouriteRequest">
```

```
 <wsdl:part element="tns:addFavourite" name="parameters" />
</wsdl:message>
<wsdl:message name="getStoreSearchRequest">
 <wsdl:part element="tns:getStoreSearch" name="parameters" />
</wsdl:message>
<wsdl:message name="addDocumentContextResponse">
 <wsdl:part element="tns:addDocumentContextResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="checkEntityResponse">
 <wsdl:part element="tns:checkEntityResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="addClickResponse">
 <wsdl:part element="tns:addClickResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getCtxInfoResponse">
 <wsdl:part element="tns:getCtxInfoResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getUUIDResponse">
 <wsdl:part element="tns:getUUIDResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="checkSearchEntityResponse">
 <wsdl:part element="tns:checkSearchEntityResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getEntityStatusRequest">
 <wsdl:part element="tns:getEntityStatus" name="parameters" />
</wsdl:message>
<wsdl:message name="processEntityResponse">
 <wsdl:part element="tns:processEntityResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="addDocumentContextRequest">
 <wsdl:part element="tns:addDocumentContext" name="parameters" />
</wsdl:message>
<wsdl:message name="addClickRequest">
 <wsdl:part element="tns:addClick" name="parameters" />
</wsdl:message>
<wsdl:message name="getCtxInfoRequest">
 <wsdl:part element="tns:getCtxInfo" name="parameters" />
</wsdl:message>
<wsdl:message name="getHistoryRequest">
 <wsdl:part element="tns:getHistory" name="parameters" />
</wsdl:message>
<wsdl:message name="getSearchItemByIdRequest">
 <wsdl:part element="tns:getSearchItemById" name="parameters" />
</wsdl:message>
<wsdl:message name="addPageRequest">
 <wsdl:part element="tns:addPage" name="parameters" />
</wsdl:message>
<wsdl:message name="getMsgRequest">
 <wsdl:part element="tns:getMsg" name="parameters" />
</wsdl:message>
<wsdl:message name="getHistoryResponse">
 <wsdl:part element="tns:getHistoryResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="checkSearchEntityRequest">
 <wsdl:part element="tns:checkSearchEntity" name="parameters" />
</wsdl:message>
<wsdl:message name="getStyleSheetRequest">
 <wsdl:part element="tns:getStyleSheet" name="parameters" />
</wsdl:message>
<wsdl:message name="showFreqWordsRequest">
 <wsdl:part element="tns:showFreqWords" name="parameters" />
</wsdl:message>
<wsdl:message name="getPageInfoRequest">
 <wsdl:part element="tns:getPageInfo" name="parameters" />
</wsdl:message>
<wsdl:message name="getUUIDRequest">
 <wsdl:part element="tns:getUUID" name="parameters" />
</wsdl:message>
<wsdl:message name="initPrevaylerResponse">
 <wsdl:part element="tns:initPrevaylerResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getSearchInfoResponse">
```

```
  <wsdl:part element="tns:getSearchInfoResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="getSearchInfoRequest">
  <wsdl:part element="tns:getSearchInfo" name="parameters" />
 </wsdl:message>
 <wsdl:message name="testMeResponse">
  <wsdl:part element="tns:testMeResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="getEntityStatusResponse">
  <wsdl:part element="tns:getEntityStatusResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="getMsgResponse">
  <wsdl:part element="tns:getMsgResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="checkSearchStatusByIdRequest">
  <wsdl:part element="tns:checkSearchStatusById" name="parameters" />
 </wsdl:message>
 <wsdl:message name="processEntityRequest">
  <wsdl:part element="tns:processEntity" name="parameters" />
 </wsdl:message>
 <wsdl:message name="getPageInfoResponse">
  <wsdl:part element="tns:getPageInfoResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="checkSearchStatusByIdResponse">
  <wsdl:part element="tns:checkSearchStatusByIdResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="getStoreSearchResponse">
  <wsdl:part element="tns:getStoreSearchResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="getSearchItemByIdResponse">
  <wsdl:part element="tns:getSearchItemByIdResponse" name="parameters" />
 </wsdl:message>
 <wsdl:message name="initPrevaylerRequest">
  <wsdl:part element="tns:initPrevayler" name="parameters" />
 </wsdl:message>
 <wsdl:portType name="CtxStoreService">
  <wsdl:operation name="getCtxInfo">
   <wsdl:input message="tns:getCtxInfoRequest" name="getCtxInfoRequest" />
   <wsdl:output message="tns:getCtxInfoResponse" name="getCtxInfoResponse" />
  </wsdl:operation>
  <wsdl:operation name="getMsg">
   <wsdl:input message="tns:getMsgRequest" name="getMsgRequest" />
   <wsdl:output message="tns:getMsgResponse" name="getMsgResponse" />
  </wsdl:operation>
  <wsdl:operation name="getSearchItemById">
   <wsdl:input message="tns:getSearchItemByIdRequest" name="getSearchItemByIdRequest" />
   <wsdl:output message="tns:getSearchItemByIdResponse" name="getSearchItemByIdResponse" />
  </wsdl:operation>
  <wsdl:operation name="getStoreSearch">
   <wsdl:input message="tns:getStoreSearchRequest" name="getStoreSearchRequest" />
   <wsdl:output message="tns:getStoreSearchResponse" name="getStoreSearchResponse" />
  </wsdl:operation>
  <wsdl:operation name="processEntity">
   <wsdl:input message="tns:processEntityRequest" name="processEntityRequest" />
   <wsdl:output message="tns:processEntityResponse" name="processEntityResponse" />
  </wsdl:operation>
  <wsdl:operation name="checkSearchStatusById">
   <wsdl:input message="tns:checkSearchStatusByIdRequest" name="checkSearchStatusByIdRequest" />
   <wsdl:output message="tns:checkSearchStatusByIdResponse" name="checkSearchStatusByIdResponse"
/>
  </wsdl:operation>
  <wsdl:operation name="getPageInfo">
   <wsdl:input message="tns:getPageInfoRequest" name="getPageInfoRequest" />
   <wsdl:output message="tns:getPageInfoResponse" name="getPageInfoResponse" />
  </wsdl:operation>
  <wsdl:operation name="addDocumentContext">
   <wsdl:input message="tns:addDocumentContextRequest" name="addDocumentContextRequest" />
   <wsdl:output message="tns:addDocumentContextResponse" name="addDocumentContextResponse" />
  </wsdl:operation>
  <wsdl:operation name="addFavourite">
   <wsdl:input message="tns:addFavouriteRequest" name="addFavouriteRequest" />
   <wsdl:output message="tns:addFavouriteResponse" name="addFavouriteResponse" />
  </wsdl:operation>
  <wsdl:operation name="checkSearchEntity">
    <wsdl:input message="tns:checkSearchEntityRequest" name="checkSearchEntityRequest" />
    <wsdl:output message="tns:checkSearchEntityResponse" name="checkSearchEntityResponse" />
  </wsdl:operation>
  <wsdl:operation name="getHistory">
    <wsdl:input message="tns:getHistoryRequest" name="getHistoryRequest" />
    <wsdl:output message="tns:getHistoryResponse" name="getHistoryResponse" />
  </wsdl:operation>
  <wsdl:operation name="getStatus">
    <wsdl:input message="tns:getStatusRequest" name="getStatusRequest" />
    <wsdl:output message="tns:getStatusResponse" name="getStatusResponse" />
  </wsdl:operation>
  <wsdl:operation name="checkEntity">
    <wsdl:input message="tns:checkEntityRequest" name="checkEntityRequest" />
    <wsdl:output message="tns:checkEntityResponse" name="checkEntityResponse" />
  </wsdl:operation>
  <wsdl:operation name="getEntityStatus">
    <wsdl:input message="tns:getEntityStatusRequest" name="getEntityStatusRequest" />
    <wsdl:output message="tns:getEntityStatusResponse" name="getEntityStatusResponse" />
  </wsdl:operation>
  <wsdl:operation name="showFreqWords">
    <wsdl:input message="tns:showFreqWordsRequest" name="showFreqWordsRequest" />
    <wsdl:output message="tns:showFreqWordsResponse" name="showFreqWordsResponse" />
  </wsdl:operation>
  <wsdl:operation name="initPrevayler">
    <wsdl:input message="tns:initPrevaylerRequest" name="initPrevaylerRequest" />
    <wsdl:output message="tns:initPrevaylerResponse" name="initPrevaylerResponse" />
  </wsdl:operation>
  <wsdl:operation name="addClick">
    <wsdl:input message="tns:addClickRequest" name="addClickRequest" />
    <wsdl:output message="tns:addClickResponse" name="addClickResponse" />
  </wsdl:operation>
  <wsdl:operation name="getSearchInfo">
    <wsdl:input message="tns:getSearchInfoRequest" name="getSearchInfoRequest" />
    <wsdl:output message="tns:getSearchInfoResponse" name="getSearchInfoResponse" />
  </wsdl:operation>
  <wsdl:operation name="showPageMap">
    <wsdl:input message="tns:showPageMapRequest" name="showPageMapRequest" />
    <wsdl:output message="tns:showPageMapResponse" name="showPageMapResponse" />
  </wsdl:operation>
  <wsdl:operation name="testMe">
    <wsdl:input message="tns:testMeRequest" name="testMeRequest" />
    <wsdl:output message="tns:testMeResponse" name="testMeResponse" />
  </wsdl:operation>
  <wsdl:operation name="addPage">
    <wsdl:input message="tns:addPageRequest" name="addPageRequest" />
    <wsdl:output message="tns:addPageResponse" name="addPageResponse" />
  </wsdl:operation>
  <wsdl:operation name="getUUID">
    <wsdl:input message="tns:getUUIDRequest" name="getUUIDRequest" />
    <wsdl:output message="tns:getUUIDResponse" name="getUUIDResponse" />
  </wsdl:operation>
  <wsdl:operation name="getStyleSheet">
    <wsdl:input message="tns:getStyleSheetRequest" name="getStyleSheetRequest" />
    <wsdl:output message="tns:getStyleSheetResponse" name="getStyleSheetResponse" />
</wsdl:operation>
 </wsdl:portType>  <wsdl:binding name="CtxStoreServiceImplHttpBinding" type="tns:CtxStoreService">
  <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="getCtxInfo">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="getCtxInfoRequest">
      <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getCtxInfoResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getMsg">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="getMsgRequest">
      <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getMsgResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getSearchItemById">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getSearchItemByIdRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getSearchItemByIdResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getStoreSearch">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getStoreSearchRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getStoreSearchResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="processEntity">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="processEntityRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="processEntityResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="checkSearchStatusById">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="checkSearchStatusByIdRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="checkSearchStatusByIdResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getPageInfo">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getPageInfoRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getPageInfoResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="addDocumentContext">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="addDocumentContextRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="addDocumentContextResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="addFavourite">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="addFavouriteRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="addFavouriteResponse">
   <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="checkSearchEntity">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="checkSearchEntityRequest">
   <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="checkSearchEntityResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getHistory">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getHistoryRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getHistoryResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getStatus">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getStatusRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getStatusResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="checkEntity">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="checkEntityRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="checkEntityResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getEntityStatus">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getEntityStatusRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="getEntityStatusResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="showFreqWords">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="showFreqWordsRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="showFreqWordsResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="initPrevayler">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="initPrevaylerRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="initPrevaylerResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="addClick">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="addClickRequest">
    <wsdlsoap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="addClickResponse">
    <wsdlsoap:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getSearchInfo">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="getSearchInfoRequest">
    <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getSearchInfoResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="showPageMap">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="showPageMapRequest">
      <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="showPageMapResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="testMe">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="testMeRequest">
      <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="testMeResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="addPage">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="addPageRequest">
      <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="addPageResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getUUID">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="getUUIDRequest">
      <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getUUIDResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getStyleSheet">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="getStyleSheetRequest">
      <wsdlsoap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getStyleSheetResponse">
      <wsdlsoap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="CtxStoreServiceImpl">
  <wsdl:port binding="tns:CtxStoreServiceImplHttpBinding" name="CtxStoreServiceImplHttpPort">
    <wsdlsoap:address location="http://localhost:5150/CtxStoreServiceImpl" />
  </wsdl:port>
</wsdl:service></wsdl:definitions>
```

Appendix D. Contextualisation Scenarios

This appendix contains the scenario descriptions and action sequences, developed using Scenario-Based Design in chapter 5. These scenarios are instances of the information-seeking behaviours on the World Wide Web. The scenarios shown in table D-1 below are discussed in the following sections.

Contextualisation Scenarios
Search Context Store
Search Result Set
Browse Context Store
Browse Result Set
View Hotlist.
View Dynamic Favourites.

Table D-1 Contextualisation Scenarios

D.1 Search Context Store

Description

The user searches a context store to find pages relating to a search term.

Information Need

The user remembers reading a page about a Ducati motorbike the previous week, but cannot remember the location of the page. Performing a Google search for Ducati returns a large resultset, but does not help locate the desired page. Refining the query to include reference to the location of the user still returns a large result set that is more qualified, but the user still cannot locate the page they wish to view.

Main Flow

1. The user selects the Search Context Store function.

2. The search pane displays a text box to accept a query string, and a submit button to execute the search.

3. The user enters the query string *ducati*.

4. The user selects the submit button.

5. The search pane submits the query to the context store.

6. The context store locates all pages the user has visited that contains the query string.

7. The context store builds a list of pages and contextual information and returns it to the search pane.

8. The search pane displays the list.

9. The user uses the contextual information to locate the required page.

10. The user browses to the required page.

11. The use case ends.

D.2 Search Result Set

Description

The user searches a result set from a search engine to find a page of interest.

Information Need

The user has been browsing a result set for a while, and after going through 10 pages realizes that one of the pages earlier in the result set was actually most useful. Remembering a keyword off the title of the interesting page, the user enters that term and the search result set if filtered to only show links to pages with that term in the title. This saves the user having to click back through all the result set pages looking for the desired page.

The user searches a result set from a search engine to find a page of interest.

Pre-condition

The user is viewing a page from the result set of a search.

Main Flow

1. The user selects the Search Result Set function.
2. The search pane displays a text box and a submit button.
3. The user enters their query term and selects the submit button.
4. The search pane submits the query terms to the context store.
5. The context store retrieves all pages from the result set list whose contextual information matches the submitted query term.
6. The context store passes a list of pages relevant to the query terms to the search pane.

7. The search pane displays the list of relevant pages along with their contextual information.
8. The user uses the mouse over highlighting to browse the result set until the desired page is found.
9. The use case ends.

D.3 Browse Context Store

Description

A user browses a context store to find pages that relate to an information need.

Information Need

The user has read through a number of articles relating to steam engines this past Sunday. Starting out from an interest piece in the paper, the user then followed a link to a club homepage; then from the club homepage followed links through to pages on building steam engines. The user wishes to follow the same trail to locate some contact details that were listed on some of these pages. The user does not remember enough about the details to formulate a query, but knows the relevant details were somewhere on one of those pages.

Main Flow

1. The user selects the Browse Context Store function.
2. The search pane requests a list of top-level categories from the context store (default is by date).
3. The search pane displays a tree view of pages organized by date.
4. The user selects the date for Sunday.
5. The search pane retrieves the contextual information for all pages visited Sunday.
6. The tree view opens showing all pages visited on Sunday in chronological order.
7. The user identifies the article heading and selects it from the list.
8. LOOP
9. The user reads the page.

10.If page satisfies information need, END LOOP

11.The user selects the next function on the search pane.

12.If no more pages, END LOOP.

The use case ends.

D.4 Browse Result Set

Description

The user is using a search engine to locate information specified by a query.

Information Need

The user has entered a set of query terms describing an information need. The term entered is Outback Steakhouse. The user is currently in Melbourne Australia, but plans to visit Seattle, WA soon and has been reading up about Seattle places of interest on the World Wide Web. The user wants to find the location of Outback steakhouses in Seattle, WA.

The user is using a search engine to locate information specified by a query.

Main Flow

1. The user starts the scenario by entering a search term at Google and submitting it.
2. LOOP UNTIL information located or result set exhausted.
3. The search pane detects the page loaded is a Google result set and sends the page links in the resultset page to the context store.
4. The context store retrieves the pages at the URL's submitted by the search pane and stores their contextual information. If the query terms have been used recently, the pages are added to that list. Otherwise, a new list is created.
5. The context store adds relevancy rankings based on a "recent terms of interest" heuristic.
6. The context store notifies the search pane its look ahead is complete.

7. The search pane retrieves the pages that are part of the result set for the current search.
8. The search pane displays the list of pages from the result set, highlighting pages that exceed a specified relevancy threshold.
9. The user reviews the contextual information for each page by hovering the mouse pointer over it.

The user selects next on the search page.

D.5 View Hotlist.

Description

The user uses a list of frequently visited pages to quickly navigate to a page.

Information Need

The user wishes to visit a page they regularly visit, such as a newspaper home page. Instead of using a favourites bookmark or a shortcut, the user selects the item from a dynamically generated the list. This removes the need for the user to maintain favourite lists or to update shortcuts. The hotlist is generated based on pages recently visited, and thus allows new preferences to emerge over time.

Main Flow

1. The user selects the View Hotlist function.
2. The search pane requests a list of frequently visited pages from the context store.
3. The context store assembles a list and returns it to the search store.
4. The search pane displays the list of pages.
5. The user selects the desired page.
6. The use case ends.

D.6 View Dynamic Favourites.

Description

The user views a list of pages that the context store identifies as being relevant to the user based on various heuristics.

Information Need

Users of tools such as Internet Explorer or Mozilla often wind up with lists of bookmarks that are too long and too out of date to be useful. Pages are only bookmarked if the user thinks to do it. Sometimes they would like to be able to locate a page that they wish they had bookmarked.

Main Flow

1. The user selects the View Dynamic Favourites function.
2. The search pane sends a request to the context store for a list of favourites.
3. The context store prepares a list of pages in the store ordered by relevancy to emergent interest keywords and by frequency of visit heuristics.
4. The context store sends the list of pages to the search pane.
5. The search pane displays the list of favourites.
6. The use case ends.

Appendix E. Web Services Overview

Web services use the World Wide Web to perform application-to-application integration (Kreger 2003). Advances in web technology have seen it move from a two tier client-server system delivering static pages to a three-tier system delivering dynamic content out of databases. Web services support a n-tier architecture for web-based applications by allowing software services to be invoked remotely over the World Wide Web by other software components (Ferris & Farrell 2003). This communication between components is supported by having the components publish interfaces. Other components can then call in to the component using these interfaces. These services make information available in XML so it is machine readable, and independent of any presentation semantics. The consuming application can use the information however it chooses. The consuming application can be scripts, end-user application, or other service components.

E.1 Understanding Web Services

A web service is any service that is available over the Internet, uses a standardized XML messaging system, and is not tied to any one operating system or programming language (Curbera et al. 2002).

A web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML-based messages conveyed by Internet protocols. (Booth et al. 2004). Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks.

A service is an application that exposes its functionality through an application-programming interface (API). In other words, a service is a resource that is designed to be consumed by software rather than by humans.

A service is a piece of software that does work for other software. In most circumstances, a service runs on a server, waiting for an application to call it and ask it to do some work. In many cases services don't provide any type of human interface, and the only way to access the service is through its API.

A service can perform system functions or business application functions. For example, a file service can create, find, save, or delete a file. A stock quote service can retrieve the current ask and bid prices of an equity.

Web Services have two key characteristics. Web Services are discoverable. Web Services can be listed in directory services such as UDDI. Web Services are self-describing. A web-service is described using WSDL. Any software that wishes to act as a client of a web-service can identify the interface provided by the service by requesting the WSDL that describes the interface.

A Web service is a Web resource. You access a Web service using platform-independent and language-neutral Web protocols, such as HTTP. These Web protocols ensure easy integration of heterogeneous environments.

A Web service provides an interface—a Web API—that can be called from another program. This application-to-application programming interface can be invoked from any type of application. The Web API provides access to the application logic that implements the service.

E.1.1 Using Web Services

Web Services are defined by a set of interfaces (Booth et al. 2004).

The interfaces are specified using WDSL. A web service is deployed using an application server. The application server accepts incoming HTTP requests and replies with HTTP responses. The requests and responses are encoded as SOAP messages. SOAP is an XML DTD that describes how services can communicate with one another. A SOAP message consists of an envelope that contains a header and a body. The header describes how the message is structure. The body contains information about procedures to be invoked and any objects being transported from the sender to the receiver.

Publish. The web service implementation is deployed to an application server. The WDSL description of its interface is made available via a URI, and the services location is added to a registry service such as a UDDI server.

Find. Agents locate the service instance via the directory registry and determine which interfaces are available to interact with. They set up initial connections and authenticate if necessary.

Interact. The agent begins to exchange SOAP messages with the service. The agent invokes methods of the service by sending SOAP requests containing the name of the method and a list of parameters. The service sends back a SOAP response containing the results of the operation.

E.1.2 Web 2.0

The World Wide Web as architected by Tim Berners-Lee focused on serving pages in response to requests from client applications. This architecture led to web applications being built as client-server applications, where the client sends requests to a server and receives a response. The emergence of web services has seen a major evolution in the way web applications are architected (McIlraith & Zeng 2001). Clients are more complex and can consume data from multiple web services. Web 2.0 is an approach to developing software solutions using the World Wide Web as the application platform.

Web 2.0 is not a specific technology or protocol in the sense that HTTP or HTML are definite protocols. Rather, Web 2.0 is an approach to software development.

The distributed and collaborative nature of web 2.0 applications means that it supports the activities of communities. Examples of Web 2.0 applications include blogging, forums, peer to peer file sharing, and distributed file downloads via bit torrent.

Web 2.0 applications are characterised by their focus on people and communities. They are based on the idea that knowledge isn't a static entity to be served up in response to a request, but rather that knowledge is the result of people interacting with information. They track the interaction as much as they track the static information. The defining example of this is the use of tags in the del.icio.us application for tracking favourite web pages. Rather than trying to categorise pages into a rigid taxonomy that has been defined by the application developers, the taxonomy of tags develops as users update the system. This emergent taxonomy allows structures that reflect the interest of the users to emerge, rather than relying on a structure developed by the application developer.

Appendix F. Implementation Technical Notes

This appendix discusses technical issues relating to the design and implementation of the ICU framework and the ISeeYou tool. These discussions capture technical and design decisions that are significant to the reproducibility of the work. The issues discussed here are not research outcomes but were significant in supporting the work that lead to the research outcomes achieved by this work. The technologies presented here are grouped in to Client Technologies, Server Technologies, and Deployment technologies.

F.1 Client Technologies

The following technologies were used to develop the Identify and Use components.

F.1.1 Browser

In choosing the web browser to use for the implementation, the most commonly-used browsers were considered. The choice was based on how widespread the browser is at the time of the decision (early 2006), and the level of support it has for extensibility through plug-ins. The browsers with the biggest shares of the market are IE with around 85% and Firefox with around 12%. Whilst other browsers have interesting features they do not have the same level of usage as the leading two browsers (Firefox). To make the tools available to as many people as possible the choice was made to focus the tool building efforts on IE – effectively the tool most often used for carrying out information seeking behaviours on the World Wide Web. At the same time, the tools have been designed with the extensibility mechanisms of Firefox in mind so that porting them in future is a real possibility. By integrating the agents in to IE it is shown that information-seeking behaviours can be augmented by tools that integrate with existing tools.

F.1.2 COM Overview

The Internet Explorer application relies heavily on a technology called COM. The Component Object Model (COM) (Box 1997) is a specification describing how objects can communicate with one another in a distributed system. COM also describes how the lifecycle of objects can be managed and how objects can be defined. The COM specification was developed to be platform independent. In practice COM is only fully available on the Microsoft Windows operating system.

Interfaces define the set of methods that a COM object makes available to COM client applications. Interfaces are defined using the Interface Definition Language (IDL). The set of COM objects and their interfaces available on a machine can be explored using the OLEVIEW tool from Microsoft.

Connection Points provide a way for a COM client to receive the events of another object. The client registers through a Connection Point to receive notifications belonging to a Dispatch Interface. A Dispatch Interface defines the set of events that a COM Object can send to clients.

F.1.3 Browser Helper Objects

A Browser Helper Object (BHO) is a COM object that is loaded by Internet Explorer whenever it starts up (Espisito 1999). When the BHO is loaded, it is given a reference to its hosting container. This reference is used to connect to the Connection Interface of the browser that allows components to receive events that occur within the web browser. The reference is also used to connect to the web browser instance itself.

A Browser Helper Object is able to implement custom handling of events that happen in the browser. For example, when the user clicks on a hyperlink in a page, an OnClick event is generated. A Browser Helper Object is able to detect when this event occurs. An event handler can then be executed that extends or modify's the existing behaviour of the browser. The events the web browser generates are described by the DWebBrowserEvents dispatch interface.

BHO's are able to access the Document Object Model (DOM) (Hors et al. 2000) of the page currently being displayed in the browser through the IWebBrowser2 interface. This means that the BHO is able to read and write content, as well as examine and modify the structure of the page.

F.1.4 Explorer Bands

An explorer band is a child window of the browser that can be used to display information to and receive input from the user (Microsoft 2001). The band is managed separately from the web browser instance being used to display web pages to the user. This makes it an ideal candidate for displaying information about a page in a way that does not affect the page being viewed.

Explorer bands can be displayed vertically, horizontally, or on the desktop. Vertical Explorer Bands are shown in a vertical strip to the left of the main display area. Horizontal Bands are shown in a horizontal strip below the main display area. These areas are totally under the control of the band developer. The bands give the developer a window area that they can control programmatically. A dialog or a custom control can be used to display the user interface in these areas. It is also possible to place another WebBrowser control in this area. This control is independent of the control being used to render content in the main display area.

Explorer Bands provide a mechanism for a developer to integrate their own user interface in to Internet Explorer. This is an appropriate technique for implementing a Use agent.

F.1.5 Extensibility Technology

The primary purpose of a browser is to allow users to view content which is available on the World Wide web. Given the diversity of information types and users found on the internet, it is beneficial to extend browsers to handle information formats that are maintained by third party software developers. It is also beneficial to provide users with customised methods of navigation or new ways of viewing information. These extensions are achieved in different ways in different browsers.

Netscape plug-ins. Plug-ins are software components developed using the Netscape Plugin API (NPAPI). They are used by all major browser except for recent versions of Internet Explorer. They provide a way for native code to be executed inside the browser to allow for handling of new media types. They must be downloaded and installed manually by the user.

ActiveX controls. An ActiveX control is a COM (Box 1997) object that can be instantiated and accessed by a web page. ActiveX is a technology developed by Microsoft and provides a way of exposing to a web page the functionality of applications or libraries already installed on users' machines. For example an ActiveX control can be used to display Word content on a web page. Originally ActiveX controls could be downloaded and installed by the browser seamlessly if they weren't available on a users' machines. Due to security concerns about allowing controls to silently install executable code on a user's machine, this ability has been restricted and now typically requires some kind of user consent before installation occurs.

An explorer band is a region within the browser frame that is used to provide a user interface. They can be implemented as a window using binary code or as registered HTML content. As binary code the band implements a set of interfaces that allow it to be loaded and hosted by IE, and to interact with the user.

Explorer Bands: *E*xplorer Bands in Internet Explorer can be implemented using HTML. By registering an HTML page in appropriate registry locations, IE will load the page into an explorer band. The developer does not need to write any extra code. A similar approach can be used to modify context menus and to add buttons to the main toolbar (Microsoft 2001).

XUL: (pronounced 'zool') This is an XML-based user interface language. It provides a way for user interface elements which it refers to as widgets to be described using XML. The XML defines a hierarchy of available components that the XUL engine knows how to render (Oeschger et al. 2002).

There are two main benefits to this approach. First, XUL allows the browser to be extended without requiring the engine to execute binary code. Allowing third party binary code to run in an application can be a security risk (Garfinkel & Spafford 1997). Second, XUL allows cross platform applications to be developed. Because the widget is described in a non-platform specific way using XML, the widget can be displayed on any platform.

As a platform for developing user interface elements, XUL meets all the requirements of the ICU framework. In this regard, XUL is an important technology to consider for future development. However, XUL is not suitable for this implementation because it is not supported by Internet Explorer.

Greasemonkie: An Internet Explorer extension that supports scripted plug-ins using an XML-based user interface language similar to XUL (Daishar 2005). It was not considered for this implementation because it is not a standard component of Internet Explorer and its availability on target browsers is not certain. It also requires .NET components to be installed and has a large redistributable footprint, which meant that bundling it with the ISeeYou distribution was not an option.

AJAX: An approach to developing web-based applications (Crane 2005). It is based on the use of JavaScript, DHTML, DOM, HTML and XML to create web pages that provide the user with an interactive experience. AJAX uses the XMLHttpRequest object to retrieve data in XML format and uses JavaScript and DHTML to update the page with the retrieved data. This allows the content displayed on a page to be updated without requiring the page to be reloaded. AJAX applications can be embedded in scripted or binary plug-ins. AJAX is a relatively new approach to developing software and it was not chosen because at the time development started on ISeeYou there was a lack of documentation and tool support for developing AJAX based application. It would be appropriate to use AJAX in future development of ISeeYou client agents.

In the Identify and Use agent components of ISeeYou an ActiveX control is implemented to allow the agent components to be loaded and managed by Internet Explorer. At the same time, consideration has been given to developing the components in a way that would them allow them to be ported to other browsers using technologies like AJAX and XUL. For this reason, the use of XML has been relied upon heavily for communication between components, and using JavaScript to implement dynamic functionality.

Browser Re-use: One approach to extending browsers is to reuse the rendering engine and build a custom user interface. There are two main UI components of a web browser (figure N). The first is the frame and the second is the rendering engine. The frame – also known as the chrome in Firefox and Mozilla – is the user interface of the application. It includes the menu bar, tool bars, address bar, status bar and the applications window borders.

The second component is the HTML rendering engine. In Internet Explorer the rendering engine is called MSHTML and is also referred to as the WebOC or the WebBrowser control (MSDNb). In Firefox and Mozilla the rendering engine is called Gecko (Oeschger et al. 2002). Internet Explorer can be considered a host for MSHTML and Firefox can be considered a host for Gecko. There are some differences in how these two engines render HTML, particularly in how they handle stylesheets. In Netscape 8 it is possible to switch between these two rendering engines at runtime.

Reusing the Web browser control is an effective way of providing the user with new navigation aids. A good example is Maxthon, which reused MSHTML to provide the user with tabbed browsing that was completely compatible with pages developed for Internet Explorer.

In the Use tools for ISeeYou the webbrowser control is hosted in the explorer band to simplify the rendering of the response information. By using stylesheets and XML the dependency of the UI on platform specific display features is removed and relies instead on using HTML.

F.2 Server Technology

The following issues were significant in the development of the Collect layer of the ICU framework and the ISeeYou tool.

The Context Collection is the Server layer of the ICU client-server architecture. As defined in the ICU framework it consists of two components. The Message Service component is responsible for listening for requests from client agents and building appropriate responses. The Context Store component is responsible for managing the contextual information.

The Message Service Component is a web service implemented as a Windows service application using Xfire (codehaus). The Context Store Component is a data layer that structures the contextual information described by the ontology in to data structures that can be rapidly accessed and also can be persisted so that the context of the user is available across login sessions.

The following sections discuss technologies that can be applied to implementing the ICU frameworks server layer are discussed. The first section discusses the Message Service and appropriate technologies for its implementation. The next session discusses the Store Service and relevant technologies. Finally, the discussion considers how a server component can be managed by service technologies that allow the server layer implementation to be deployed for use by client agents.

F.2.1 Message Service Component

The ICU framework that was developed in chapter 6 specifies SOAP as the transport mechanism for communication between the client agents and the server layer. XFire is an open source java-based implementation of a SOAP server and SOAP client (codehaus). It is designed to be fast and lightweight. Some of its key features that were of interest in this research include the ability to run as a standalone service and the ability to specify the interface for the webservice using java annotations.

The web service annotations are based on JSR-181 (JCP) and are applied to the Java interface. They are used to generate the appropriate web service definition from the source code. This is useful for us as it means there is only a single point where the interface definition must be defined. Because the WSDL is generated dynamically, it is always up to date with the current interface description. This is very useful during development of a service as it removes the possibility of errors occurring due to the implementation and WSDL being inconsistent.

The ability to run XFire as a standalone service means that it does not require the deployment of an entire servlet infrastructure just to run a SOAP service. This would have been a problem with a container-based SOAP implementation such as Axis 1.1 (Apache). The distributable was large and the performance hit of loading an entire servlet container into memory made implementations based on Axis 1.1 respond poorly. Axis 2.0 does allow standalone deployment, but was still in beta at the time of choosing a SOAP implementation. Additionally, XFire had plenty of documentation and an active community via IRC and mailing lists who were responsive to questions.

The methods exposed by the interfaces fall into three categories; add methods, get methods and check methods. These are described in the following sections.

Add Methods

The add methods cause new information to be added to the context store. These methods take as an input parameter an XML string containing the contextual information to be added. They do not return a result as they are intended to be invoked asynchronously and the calling program is not expected to block waiting for a result.

AddPage - This method is used by an identify agent to add contextual information to the context store about a page being viewed in the browser.

AddClick - This method is used by an identify agent to add contextual information to the context store when a hyperlink is clicked.

AddFavourite – This method is used by an identify agent to add contextual information to the context store when the user creates a favourite in their favourites folder.

AddDocumentContext – This method is used by an identify agent to add contextual information to the context store about a google search result page when it is loaded into the browser.

Get Methods

The get methods return contextual information from the store. Parameters that select the information to be retrieved are passed with the calls to these methods. All these methods take as an input parameter an XML string specifying the contextual information being queried. The return an XML string containing the requested contextual information.

GetCtxInfo – This method is used by use agents to retrieve contextual information about a particular page. The URL of the page and a mode parameter specifying the kind of information to retrieve are passed with the call.

GetStyleSheet – This method is used by use agents to retrieve a stylesheet to apply to contextual information retrieved from the store.

GetSearchItemByID – This method is used by use agents to get the details for an element of a search result. This was implemented so that a set of contextual information can be returned for a page with placeholders for information that is not yet available.

GetEntityStatus - This method is used by use agents to retrieve a status code saying whether contextual information for the entity is available.

GetStoreSearch – This method is used by use agents to retrieve pages from the context store that match a query string passed in with the call.

Check Methods

The check methods are used by use agents to poll whether contextual information for an entity is available. They take as an input parameter an XML string specifying the entity who's status is being queried. They return an XML string specifying the status. A string was used rather than a Boolean to allow multiple states to be specified.

CheckSearchEntity – This method is used by use agents to check whether contextual information for a search result page is available.

CheckEntity – This method is used by search agents to determine whether contextual information for a web page is available.

F.2.2 Context Store Component

The Context Store component of the ICU framework is responsible for managing the storage of and access to contextual information. The ISeeYou framework specifies two requirements for the implementation of a Context Store. First, it must provide persistent (Driscoll et al. 1986) storage of the contextual information, Second, it must provide fast access to the information in the Store when it is queried. In order to meet these requirements the Context Store for ISeeYou has been implemented using using the Prevaylence API (Prevayler 2006). Prevaylence works by storing the data structures containing the contextual information in memory, and creating snapshots and log files on the hard drive to persist the information between sessions.

The store is implemented as a collection of in-memory data structures. The data structures used are members of the Java Collections API. In particular ArrayLists have been used where sequential accesses are important, and HashMaps where lookups of a particular key value are important. A choice was made to implement the data structures in memory because retrieving data already in memory is much faster than retrieving it from a file or even from a database.

Keeping a data structure in memory requires that some kind of persistence mechanism be available so that information is not lost if the process is terminated. The simplest example of a persistence mechanism is writing the data structure out to a file, a process known as serialization. The problem with serialization is that if the process terminates unexpectedly, there may not be an opportunity to write the entire data structure to file and/or the file may be corrupted.

The most common approach to persistence (other than serialization to a file) is to use a database to store records that represent the data structure. The problem with this approach is that database structures suffer from the latency associated with file read and writes to a hard drive (Evans 2000). Storing the information in memory rather than on a hard drive significantly reduces the overhead associated with reading and writing information, and so have much better performance for these operations.

Prevaylence provides a mechanism for retrieving the state of an in-memory data structure after a process has been restarted by requiring all updates to the data structure be performed through a Command. When a command is executed the data being updated and the command itself are written to a command log. Additionally, a snapshot of the data structure can be generated at any time. A typical approach is to generate a snapshot before the process terminates gracefully. By loading the data structure from the last available snapshot and then applying any logged commands since that snapshot, the state of the data structure can be restored even if the process was terminated unexpectedly.

Prevaylence is lightweight and easy to use, which made it ideal for the purposes of this work. The input and output streams of the prevaylence implementation were modified to be base64 encoded so that a casual inspection of the snapshot and log files would not reveal any information about the user of the system. This approach to protecting information is at least as effective as using a typical database, and relatively inexpensive in terms of implementation and performance impact.

F.2.3 Service Technology

A service makes a resource available for consumption by other software components. Section 7.1 has described how the Message Service can be implemented using the XFire SOAP implementation. In order to successfully deploy this Message Service an appropriate service technology must be used.. A service technology manages the lifecycle of a process that implements a service.

A service is an application that provides functionality to other processes. They typically run in the background with little or no user interface. Services can be managed by platform specific infrastructure such as Windows Systems Services or Unix Daemon]. Another approach is to develop a service using language specific features, such as a C++ ATL service (Microsoft), and have this service application launched each time a user logs in. For ISeeYou the choice was made to implement the server layer as a Java Service whose lifecycle corresponds to the login session for each user. The key reason behind this decision was to have the service run as the currently logged in user, with access to the user's environment.

A Java Service is an application written in Java that runs in the background. They usually have no user interface aside from option dialogs to setup configuration and start or stop the service. When implementing ISeeYou a choice needed to be made whether to implement the server layer as a standalone service or as a component of a container.

Java Services can run in containers such as Apache Tomcat (Wiggers et al. 2002) or Jboss (Fleury & Reverbel 2003), or they can run standalone. A container provides system infrastructure such as Http listeners, Servlet engines and Data Layers. They can significantly reduce the amount of code required to be written. The tradeoff for this is that they are large distributions, use a lot of resources and sometimes suffer in terms of performance, because they are coded to handle all possible cases.

Running a Java Service standalone means that the developer must write more code to achieve the functionality they want, but wind up with a smaller distributable and less resource consumption because they only use the components they need. For example, an early implementation of the context store which used Apache Tomcat and the axis 1.1 SOAP service from Apache had a distributable of around 30 MBytes. Implementing the same service as a standalone service resulted in a distributable of around 6 MBytes.

In order to make ISeeYou easily downloadable it was important to minimise the size of the distributable. It was also important keep the resources required to run the server as low as possible to lessen impact on a user's system. For these reasons, the choice was made to implement the ISeeYou service as a standalone java service. This made the implementation a bit more complex because the infrastructure provided by a container was not available. The extra coding effort was considered worthwhile to reduce the size and resource consumption.

The server layer for ISeeYou has been implemented as a Java Service. This means that the service does not need any platform specific code to allow for the lifecycle of the service to be managed. Instead the lifecycle of the service is managed using scripts. This means that the service can be readily deployed to any platform that has a java virtual machine available by simply adding appropriate scripts to the system specific startup points.

This section has discussed how existing web technologies can be used to implement the server layer of the ICU framework. The server layer can be decomposed into two components, the Message Service and the Context Store.

The Message Service is responsible for managing communication with client components so contextual information can be updated and queried. The XFire SOAP implementation is an appropriate technology for implementing this component.

The Context Store is responsible for managing queries and updates to the contextual information so that it is available across sessions. It is also important that the Context Store be efficient with response times so the users browsing experience is not impacted. Prevayler has been presented as an appropriate technology for implementing the Context Store.

Finally, the discussion explored deploying the server layer implementation as a service so that it can be available to manage contextual information with no interaction from the user. The lifecycle of the service is managed using the Start Menu automatic entry startup point of the Windows operating system.

The first two sections have shown how ISeeYou can be implemented using technologies that allow the components to be integrated with existing web infrastructure. The next section describes how these components are constructed to implement the functionality specified by the ISeeYou framework.

F.2.4 ISeeYou Functionality

In order to assess the validity of the hypothesis it is necessary to investigate whether the ICU paradigm can be used to create software tools that users seeking information on the World Wide Web find useful and easy-to use. A tool called ISeeYou has been developed based on the ICU paradigm that can be used to perform an evaluation that will allow us to analyse the validity of the hypothesis. The ISeeYou system uses a Java implementation of the context store and Internet Explorer extensions to identify and use context. ISeeYou implements the functionality specified in the ICU framework, drawing on scenarios of information-seeking behaviours identified in chapter five.

The ICU paradigm is not dependent upon, and does not specify, the use of particular implementation technologies. The Identify and Use agents can be built for different browsers using whatever programming language is appropriate. Likewise, the Context Store and the Message Service can be implemented in any language that supports SOAP and data persistence operations, which includes languages like C++, C#, Perl and Java. The choices of Java and C++ extensions to Internet Explorer were made for several reasons.

Reason 1. Interoperability. It is important to demonstrate that the client and server elements of the architecture could interoperate with one another regardless of the language used for their implementation. Specifically, the interoperability is a result of a clearly defined set of interfaces and not the result of language or platform specific functionality.

Reason 2. Compatibility. Extensions for Internet Explorer were developed because it is the most widely used browser, with over 85% of the browser market. The decision to use C++ and ATL for the development of these extensions was made so that there would be no need for runtime libraries to be distributed. This has become less of an issue recently with version 1.0 of the .NET framework being more widely available, but was a concern when development initially started. Using .NET for future Identify and Use components would be an appropriate design direction to take for future development.

Reason 3. Platform Independence. The context store and the collection services are the more complex components of the system. By implementing them in a language that can be readily re-used on different platforms it is possible to port the ICU tools to new platforms by creating new identify and use agents while using the same store and collection components. Because the Identify and Use components are integrated tightly with the target browsers it is expected they would need to be redeveloped for each new platform. By putting most of the complexity of the system in the collection service and developing that service in a language that can be reused across platforms the ICU paradigm is made available across a wider range of platforms. Java was chosen because it is re-usable across platforms and has effective development tools such as Eclipse and NetBeans freely available. Both of these were used at different points in the development of the ISeeYou system.

The ISeeYou system was built using the ICU Framework developed previously in this work. The architecture for ISeeYou is derived directly from this framework. It was built to be readily usable by as many users as possible so that an evaluation could be performed that would allow us to assess the validity of the hypothesis. The following sections discuss the architecture of the system components and how these components were deployed so they could be evaluated.

F.3 Deployment

ISeeYou is implemented by identify and use agents, and by a collection store. Previous sections have described how these components can be implemented. In order to deploy ISeeYou to a user so they can use it these components must be compiled into distributable binaries. In the following sections, the distributable components are described, focusing on how they are compiled, and how they are installed on a user's machine.

F.3.1 DLL's

Dynamic Link Libraries (DLL's) are libraries of functions that can be loaded by a Windows application. ISeeYou consists of two DLL's that implement the identify agent and the use agent. IdentifyApp.dll is the implementation of an identify agent as a Browser Helper Object. UseApp.dll is the implementation of a use agent as an Explorer Bar.

The DLL's were developed using Visual Studio 2005 Professional Edition. They were implemented in C++ using the ATL library to simplify the implementation of COM interfaces and the invocation of COM services provided by other applications. Visual Studio was used because it is the only IDE that has inbuilt support for the ATL library. Implementing COM interfaces without a supporting library requires significantly more development time. Given that the extension components are only targeted at a browser that runs on Windows, the lack of portability imposed by using VS2005 and ATL was considered not to be an issue.

F.3.2 Service

The collection service is implemented using Java2 Standard Edition 5 (J2SE 5). The Netbeans IDE from Sun was used to develop the classes and generate the distribution package. SOAP support is implemented using the XFire Soap implementation, which is an open source implementation of SOAP. The prevaylence API is also used, with customisations included to base 64 encode the contextual information it stores. The implementation classes are packaged in a runnable jar called storeApp.jar. The lib folder contains all the jars required by Xfire.

F.3.3 Data Location

The data held in the context store will be stored in a common location, specifically the All Users application data folder. A separate prevaylence folder is created for each user. The prevaylence instances were put in a common area so that it would be easy to remove the prevaylence log files on uninstall. The alternative would have been to have a folder in each users default document location. This alternative approach would have meant having to iterate over all the users on uninstall and removing the prevaylence log files. Because the implementation of ISeeYou is being developed for an evaluation, it is expected that ISeeYou will be uninstalled. To make sure uninstall is successful it was simpler to store all the data that will be removed on uninstall in a common folder.

On first run of the store a Prevaylence folder is created and then a folder for each user is created the first time they start the service. This folder contains the snapshot and command log files for the user. On uninstall of ISeeYou the entire Prevaylence folder is removed.

F.3.4 Startup

A shortcut to the batch file that launches the Collection Service is created in the Start Menu -> Programs -> Startup folder. This startup point is shared by all users of the system. This causes the batch file to be launched every time a user logs in.

F.3.5 Installer

The installation of ICU has been automated using the Nullsoft Scriptable Installer System (http://nsis.sourceforge.net/Main_Page). NSIS is an open source tool for developing Windows software installers. It is appropriate for ISeeYou because currently ISeeYou is a Windows specific software system. NSIS was chosen because it is lightweight, efficient, and has plenty of documentation and examples on which to base new installers. Using NSIS made it possible to quickly create an installer that would allow ISeeYou to be easily installed on and uninstalled from Windows systems.

The availability of an uninstaller is important for two reasons. First, it gives the user of ISeeYou the ability to remove the software from their system. This is important because the lack of an uninstall process is one of the criteria that results in software being labelled as spyware. Second, the uninstaller removes the context store from the users system. This is an important part of the evaluation as it allows the users of ISeeYou to ensure that any contextual information stored is removed from their system once they choose to end their participation in the evaluation.

Appendix G. Survey Instrument

The survey instrument for the TAM evaluation of ISeeYou was first constructed as a paper based form (figure G-1). This instrument is based on the instruments used in other TAM evaluations. Once the survey instrument had been developed, it was submitted to the UTS Human Research Ethics Committee (HREC) for approval. Once approval was obtained, a web-based version of the survey was constructed. The web-based survey consists of two pages. The first page contains ethics and consent information (figure G-2). This information must be made available to subjects participating in human-based research at UTS. The second page (figure G-3) contains the questions and fields allowing subjects to specify their responses. Only one radio button can be selected per question, so it is not possible for respondents to specify multiple values.

For each question, please indicate with an X your response.

1. I was able to install and use the ISU system

Yes ___ No ___

2. Using the ISU User Interface helps me find information on the web more quickly.

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

3. Using the ISU User Interface improves my performance when finding information on the web.

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

4. Using the ISU User Interface can increase my productivity when finding information on the Web

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

5. Using the ISU User Interface can enhance my effectiveness when finding information on the Web

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

6. Learning to use the ISU User Interface is easy for me.

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

7. I can use the ISU User Interface in a manner that helps me find information.

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

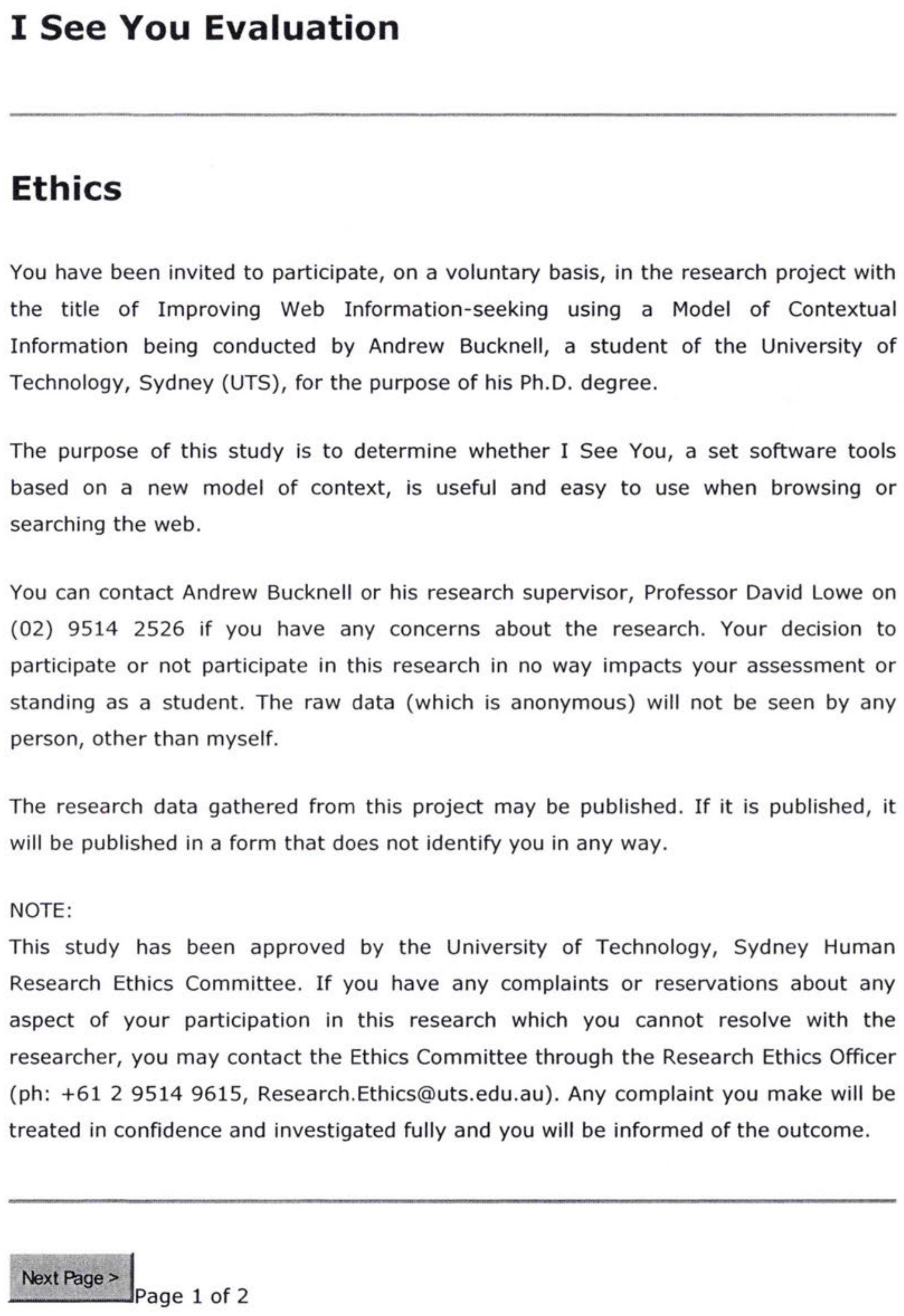
8. My interaction with the ISU User Interface is clear and understandable.

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

9. In general, I find the ISU User Interface easy to use.

Extremely Likely	Quite Likely	Slightly Likely	Neither	Slightly Unlikely	Quite Unlikely	Extremely Unlikely

Figure G-1 Paper-based survey instrument



I See You Evaluation

Ethics

You have been invited to participate, on a voluntary basis, in the research project with the title of Improving Web Information-seeking using a Model of Contextual Information being conducted by Andrew Bucknell, a student of the University of Technology, Sydney (UTS), for the purpose of his Ph.D. degree.

The purpose of this study is to determine whether I See You, a set software tools based on a new model of context, is useful and easy to use when browsing or searching the web.

You can contact Andrew Bucknell or his research supervisor, Professor David Lowe on (02) 9514 2526 if you have any concerns about the research. Your decision to participate or not participate in this research in no way impacts your assessment or standing as a student. The raw data (which is anonymous) will not be seen by any person, other than myself.

The research data gathered from this project may be published. If it is published, it will be published in a form that does not identify you in any way.

NOTE:
This study has been approved by the University of Technology, Sydney Human Research Ethics Committee. If you have any complaints or reservations about any aspect of your participation in this research which you cannot resolve with the researcher, you may contact the Ethics Committee through the Research Ethics Officer (ph: +61 2 9514 9615, Research.Ethics@uts.edu.au). Any complaint you make will be treated in confidence and investigated fully and you will be informed of the outcome.

Page 1 of 2

Figure G-2 Page 1 of Web-based survey instrument

I See You Evaluation

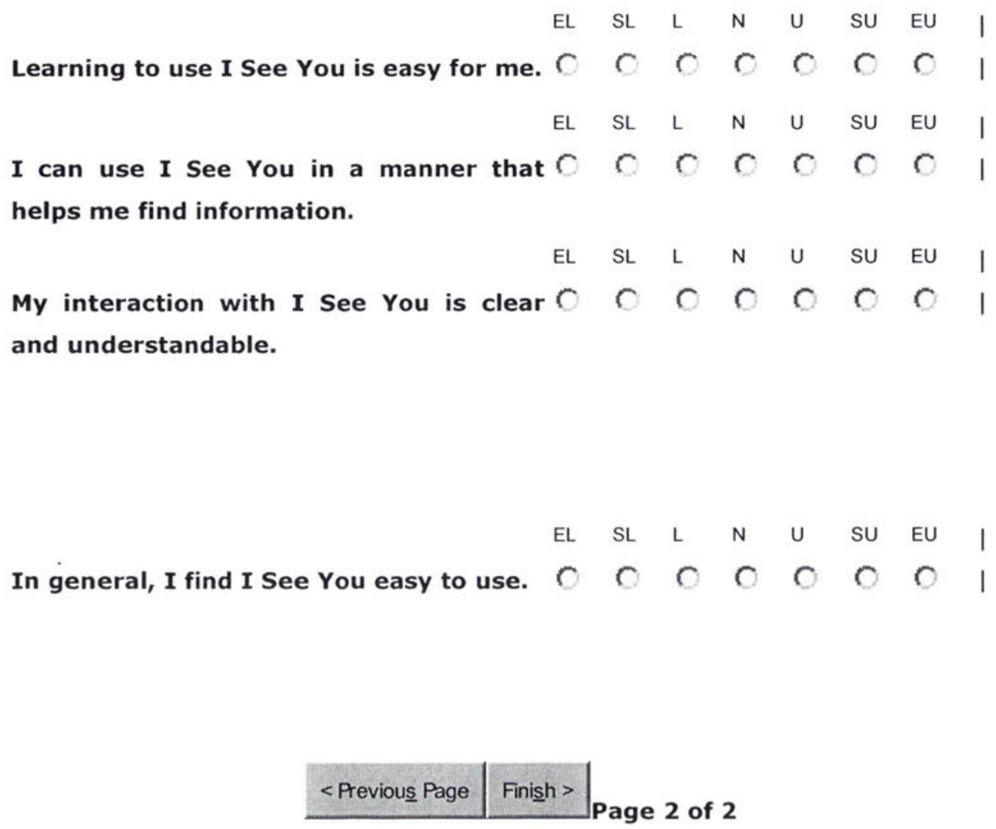
Questions

I was able to install and use I See You

○ Yes

○ No

	Extremely Likely	Strongly Likely	Likely	Undecided/Neutral	Unlikely	Strongly Unlikely	Extremely Unlikely
	EL	SL	L	N	U	SU	EU
Using I See You helps me find information on the World Wide Web more quickly.	○	○	○	○	○	○	○
	EL	SL	L	N	U	SU	EU
Using I See You improves my performance when finding information on the World Wide Web.	○	○	○	○	○	○	○
	EL	SL	L	N	U	SU	EU
Using I See You can increase my productivity when finding information on the World Wide Web.	○	○	○	○	○	○	○
	EL	SL	L	N	U	SU	EU
Using I See You can enhance my effectiveness when finding information on the World Wide Web.	○	○	○	○	○	○	○



	EL	SL	L	N	U	SU	EU
Learning to use I See You is easy for me.	○	○	○	○	○	○	○

	EL	SL	L	N	U	SU	EU
I can use I See You in a manner that helps me find information.	○	○	○	○	○	○	○

	EL	SL	L	N	U	SU	EU
My interaction with I See You is clear and understandable.	○	○	○	○	○	○	○

	EL	SL	L	N	U	SU	EU
In general, I find I See You easy to use.	○	○	○	○	○	○	○

Page 2 of 2

Figure G-3 Page 2 of Web-based survey instrument

Bibliography

Abrams, D., Baecker, R. & Chignell, M. 1998, 'Information archiving with bookmarks: personal Web space construction and organization', *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 41-48.

Akscyn, R., McCracken, D. & Yoder, E. 1987, 'KMS: a distributed hypermedia system for managing knowledge in organizations', *Conference on Hypertext and Hypermedia*, pp. 1-20.

Amitay, E. & Pariš, C. 2000, 'Automatically summarising Web sites: is there a way around it?' *Proceedings of the ninth international conference on Information and knowledge management*, pp. 173-179.

Anderson, K.M. 1997, 'Integrating open hypermedia systems with the World Wide Web', *Proceedings of the eighth ACM conference on Hypertext*, pp. 157-166.

Anderson, K.M., Taylor, R.N. & Whitehead Jr, E.J. 1994, 'Chimera: hypertext for heterogeneous software environments', *Proceedings of the 1994 ACM European conference on Hypermedia technology*, pp. 94-107.

Apache, *Web Services - Axis*, 2006 <http://ws.apache.org/axis/>.

Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A. & Raghavan, S. 2001, 'Searching the Web', *ACM Transactions on Internet Technology*, vol. 1, no. 1, pp. 2-43.

Ashman, H. 1994, 'What is hypermedia?' *ACM SIGWEB Newsletter*, vol. 3, no. 2, pp. 6-8.

Battelle, J. 2005, *The search: how Google and its rivals rewrote the rules of business and transformed our culture*, Portfolio.

Beckett, D.E. 2006, *RDF/XML Syntax Specification (Revised)*, <http://www.w3.org/TR/rdf-syntax-grammar/>.

Belkin, N.J. 1978, 'Information Concepts for Information Science', *Journal of Documentation*, vol. 34, no. 1, pp. 55-85.

Belkin, N.J. 1993, 'Interaction with texts: Information retrieval as information-seeking behavior', *Information Retrieval*, vol. 93, pp. 55–66.

Belkin, N.J. 2000, 'Helping people find what they don't know', *Communications of the ACM*, vol. 43, no. 8, pp. 58-61.

Belotti, R., Decurtins, C., Grossniklaus, M., Norrie, M.C. & Palinginis, A. 2005, 'Interplay of Content and Context', *Journal of Web Engineering*, vol. 4, no. 1, pp. 57–78.

Benjamins, V.R., Contreras, J., Corcho, O. & Gómez-Pérez, A. 2004, 'Six Challenges for the Semantic Web', *AIS SIGSEMIS Bulletin*, vol. 1, no. 1, pp. 24-25.

Bergman, M.K. 2001, 'The deep web: Surfacing hidden value', *Journal of Electronic Publishing*, vol. 7, no. 1, pp. 07-01.

Berners-Lee, T. 1999, *Weaving the Web*, Orion Business Books.

Berners-Lee, T., Luotonen, A., Nielsen, H.F. & Secret, A. 1994, 'The World Wide Web', *Communications of the ACM*, vol. 37, no. 8, pp. 76-82.

Berners-Lee, T., Masinter, L. & McCahill, M. 1994, 'RFC1738: Uniform Resource Locators (URL)', *Internet RFCs*.

Berners-Lee, T.I.M. & Lassila, O.R.A. 2001, 'The Semantic Web', *Scientific American*, vol. 284, no. 5, pp. 28-37.

Bevan, N. 1995, 'Usability is quality of use', *Proc. 6th International Conference on Human Computer Interaction, July*.

Bittner, K., Spence, I. & Jacobson, I. 2002, *Use Case Modeling*, Addison-Wesley Professional.

Boardman, R. & Sasse, M.A. 2004, '" Stuff goes into the computer and doesn't come out": a cross-tool study of personal information management', *Proceedings of the 2004 conference on Human factors in computing systems*, pp. 583-590.

Boegh, J., Depanfilis, S., Kitchenham, B. & Pasquini, A. 1999, 'A method for software quality planning, control, and evaluation', *Software, IEEE*, vol. 16, no. 2, pp. 69-77.

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C. & Orchard, D. 2004, *Web Services Architecture*, viewed 1/11/2006 <http://www.w3.org/TR/ws-arch/>.

Bos, B., Lie, H.W., Lilley, C. & Jacobs, I. 1998, 'Cascading Style Sheets, level 2 CSS2 Specification', *W3C Recommendations are available at http://www. w3. org/TR, May*, vol. 12, p. 80.

Box, D. 1997, *Essential Com*, Addison-Wesley Professional.

Brin, S. & Page, L. 1998, 'The Anatomy of a Large-Scale Hypertextual Web Search Engine', *WWW7 / Computer Networks*, vol. 30, no. 1-7, pp. 107-117.

Broder, A. 2002, 'A taxonomy of web search', *ACM SIGIR Forum*, vol. 36, no. 2, pp. 3-10.

Brookes, B.C. 1977, 'The developing cognitive view in information science', *International Workshop on the Cognitive Viewpoint, CC-77*, pp. 195-203.

Brusilovsky, P. 1996, 'Methods and techniques of adaptive hypermedia', *User Modeling and User-Adapted Interaction*, vol. 6, no. 2, pp. 87-129.

Brusilovsky, P. 2001, 'Adaptive Hypermedia', *User Modeling and User-Adapted Interaction*, vol. 11, no. 1, pp. 87-110.

Buckland, M.K. 1991, 'Information as thing', *Journal of the American Society for Information Science*, vol. 42, no. 5, pp. 351-360.

Budzik, J. & Hammond, K. 1999, 'Watson: Anticipating and contextualizing information needs', *Proceedings of the Sixty-second Annual Meeting of the American Society for Information Science*.

Bunge, M.A. 1977, *Treatise on Basic philosophy. Ontology I: The furniture of the world*, D. Reidel.

Bunnin, N. & Tsui-James, E.P. 2003, *The Blackwell companion to philosophy*, Blackwell.

Bush, V. 1945, *As we May Think. Reprinted from the Atlantic Monthly, Vol. 176, No. 1 (1945)*, 1991). From Memex to Hypertext: Vannevar Bush and the Minds Machine. San Diego: Academic Press.

Card, S.K., Robertson, G.G. & York, W. 1996, 'The WebBook and the Web Forager: an information workspace for the World Wide Web', *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*.

Carlson, C.N. 2003, 'Information Overload, Retrieval Strategies and Internet User Empowerment', *The Good, the Bad and the Irrelevant: The user and the future of information and communication technologies. Conference Proceedings. Haddon, Lua (Eds.), Helsinki*, pp. 169-173.

Carroll, J.M. 1995, *Scenario-based design: envisioning work and technology in system development*, John Wiley & Sons, Inc. New York, NY, USA.

Choo, C.W., Choo, W., Detlor, B. & Turnbull, D. 2000, *Web Work: Information Seeking and Knowledge Work on the World Wide Web*, Kluwer Academic Publishers.

Choo, C.W., Detlor, B. & Turnbull, D. 2000, 'Information Seeking on the Web: An Integrated Model of Browsing and Searching', *First Monday*, vol. 5, no. 2, pp. 67-78.

Clark, J. 1999, 'XSL Transformations (XSLT) Version 1.0, W3C Recommendation, 16 November 1999', *World Wide Web Consortium*.

Clarke, R. 2001, 'Appropriate Research Methods for Electronic Commerce', *International Journal of Electronic Commerce, URL* http://www. *anu. edu. au/people/Roger. Clarke/EC/ResMeth. html Accessed on February 10th*.

codehaus, *XFire*, 2006 <http://xfire.codehaus.org/>.

Conklin, J. 1987, 'Hypertext: a Survey and Introduction', *IEEE Computer*, vol. 20, no. 9, pp. 17-41.

Conklin, J. 1997, 'Hypertext: An Introduction and Survev'.

Dave Crane , Eric Pascarello , Darren James 2005 Ajax in Action, Manning Publications Co., Greenwich, CT

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N. & Weerawarana, S. 2002, 'Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI', *IEEE Internet Computing*, vol. 6, no. 2, pp. 86-93.

Czerwinski, M., Gage, D.W., Gemmell, J., Marshall, C.C., Pérez-Quiñones, M.A., Skeels, M.M. & Catarci, T. 2006, 'Digital memories in an era of ubiquitous computing and abundant storage', *Communications of the ACM*, vol. 49, no. 1, pp. 44-50.

Daishar, T. 2005 Greasemonkey for Internet Explorer - GreasemonkIE, viewed 12/11/2007 <http://www.daishar.com/blog/archives/2005/03/greasemonkey_fo.html>

Davis, F.D. 1989, 'Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology', *MIS Quarterly*, vol. 13, no. 3, pp. 319-340.

Davis, F.D. 1993, 'User acceptance of information technology: system characteristics, user perceptions and behavioral impacts', *International Journal of Man-Machine Studies*, vol. 38, no. 3, pp. 475-487.

Davis, H., Hall, W., Heath, I., Hill, G. & Wilkins, R. 1993, 'Towards an integrated information environment with open hypermedia systems', *Proceedings of the ACM conference on Hypertext*, pp. 181-190.

De Bra, P., Houben, G.J. & Wu, H. 1999, 'AHAM: a Dexter-based reference model for adaptive hypermedia', *Proceedings of the tenth ACM Conference on Hypertext and hypermedia: returning to our diverse roots: returning to our diverse roots*, pp. 147-156.

Dervin, B. 1997, 'Given a context by any other name: methodological tools for taming the unruly beast', *Proceedings of an international conference on Information seeking in context*, pp. 13-38.

Driscoll, J.R., Sarnak, N., Sleator, D.D. & Tarjan, R.E. 1986, 'Making data structures persistent', *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pp. 109-121.

Dumais, S., Cutrell, E., Cadiz, J.J., Jancke, G., Sarin, R. & Robbins, D.C. 2003, 'Stuff I've seen: a system for personal information retrieval and re-use', *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 72-79.

Durfee, E.H., Mullen, T., Park, S., Vidal, J.M. & Weinstein, P. 1998, 'The Dynamics of the UMDL Service Market Society', *Cooperative Information Agents II, LNAI*, pp. 55-78.

Edwards, D.M. & Hardman, L. 1999, 'Lost in hyperspace: cognitive mapping and navigation in a hypertext environment', *Hypertext: theory into practice*, pp. 90-105.

Ellis, D. 1989, 'A behavioral approach to information retrieval system design', *Journal of Documentation*, vol. 45, no. 3, pp. 171-212.

Engelbart, D.C. 1963, 'A conceptual framework for the augmentation of man's intellect', in P. Howerton & Weeks (eds), *Vistas in Information Handling*, Spartan Books, pp. 1-29

Erl, T. 2004, *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*, Prentice Hall PTR Upper Saddle River, NJ, USA.

Espisito, D. 1999, *Browser Helper Objects: The Browser The Way You Want It*, Microsoft Corporation, viewed 1/11/2006 <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebgen/html/bho.asp>.

Evans, H. 2000, *Why Object Serialization is Inappropriate for Providing Persistence in Java*, Technical report, Department of Computing Science, University of Glasgow, Glasgow G12.

Farquhar, A., Fikes, R. & Rice, J. 1997, 'The Ontolingua Server: a tool for collaborative ontology construction', *International Journal of Human Computer Studies*, vol. 46, no. 6, pp. 707-727.

Fayad, M.E. & Johnson, R.E. 1999, *Domain-specific application frameworks: framework experience by industry*, John Wiley & Sons, Inc. New York, NY, USA.

Fayad, M.E., Schmidt, D.C. & Johnson, R.E. 1999, *Building application frameworks: object-oriented foundations of framework design*, John Wiley & Sons, Inc. New York, NY, USA.

Fensel, D., Decker, S., Erdmann, M. & Studer, R. 1998, 'Ontobroker: The very high idea', *Proceedings of the 11th International Flairs Conference (FLAIRS-98)*, vol. 5.

Ferris, C. & Farrell, J. 2003, 'What are Web services?' *Communications of the ACM*, vol. 46, no. 6, p. 31.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H. & Berners-Lee, T. 1999, *Hypertext Transfer Protocol–HTTP/1.1*, RFC 2616, June 1999.

Firefox 2006a, *Firefox Add-ons*, <https://addons.mozilla.org/extensions.php?app=firefox>.

Fishbein, M. & Ajzen, I. 1975, 'Belief, Attitude, Intention, and Behavior: An Introduction to Theory and Research'.

Fleury, M. & Reverbel, F. 2003, 'The JBoss Extensible Server', *ACM/IFIP/USENIX International Middleware Conference*, pp. 344–373.

Fountain, A.M., Hall, W., Heath, I. & Davis, H.C. 1992, 'MICROCOSM: an open model for hypermedia with dynamic linking', *Cambridge Series On Electronic Publishing*, pp. 298-311.

Fox, C. 1992, 'Lexical Analysis and Stoplists, chapter 7', *Frakes and*

Baeza-Yates (Frakes and Baeza-Yates, 1992), pp. 102-130.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. 2002, 'Design patterns: abstraction and reuse of object-oriented design', *Software pioneers: contributions to software engineering*, pp. 701-717.

Garfinkel, S. & Spafford, G. 1997, *Web security & commerce*, O'Reilly Sebastopol.

Gemmell, J., Bell, G., Lueder, R., Drucker, S. & Wong, C. 2002, 'MyLifeBits: fulfilling the Memex vision', *Proceedings of the tenth ACM international conference on Multimedia*, pp. 235-238.

Goldfarb, C.F. & Rubinsky, Y. 1991, *The Sgml Handbook*, Oxford University Press.

Graham, I.S. & Quin, L. 1999, *XML specification guide*, Wiley New York.

Grønbæk, K. & Trigg, R.H. 1993, *Design issues for a Dexter-based hypermedia system*, ACM Press New York, NY, USA.

Gruber, T.R. 1995, 'Toward principles for the design of ontologies used for knowledge sharing?' *International Journal of Human Computer Studies*, vol. 43, no. 5-6, pp. 907-928.

Guarino, N. 1998, *Formal ontology in information systems*, IOS Press.

Guarino, N. & Giaretta, P. 1995, 'Ontologies and Knowledge Bases: Towards a Terminological Clarification', *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*, pp. 25-32.

Guarino, N., Masolo, C., Vetere, G., Council, N.R. & Cnr, P. 1999, 'OntoSeek: content-based access to the Web', *Intelligent Systems and Their Applications, IEEE [see also IEEE Intelligent Systems]*, vol. 14, no. 3, pp. 70-80.

Gulli, A. & Signorini, A. 2005, 'The indexable web is more than 11.5 billion pages', *International World Wide Web Conference*, pp. 902-903.

Halasz, F. & Schwartz, M. 1994, 'The Dexter Hypertext Reference Model', *Communications of the ACM*, vol. 37, no. 2, pp. 30-39.

Halasz, F.G. 2001, 'Reflections on NoteCards: seven issues for the next generation of hypermedia systems', *ACM Journal of Computer Documentation (JCD)*, vol. 25, no. 3, pp. 71-87.

Hardman, L., Bulterman, D.C.A. & van Rossum, G. 1994, 'The Amsterdam hypermedia model: adding time and context to the Dexter model', *Communications of the ACM*, vol. 37, no. 2, pp. 50-62.

Hearst, M.A. 2000, 'User interfaces and visualization', *Modern Information Retrieval*, pp. 257-323.

Heflin, J. 2004, *OWL Web Ontology Language Reference*, 2006 <http://www.w3.org/TR/owl-ref/>.

Heflin, J. & Hendler, J. 2000, 'Searching the Web with SHOE', *AAAI-2000 Workshop on AI for Web Search*.

Heflin, J., Hendler, J. & Luke, S. 1999, 'Applying Ontology to the Web: A Case Study', *International Work-Conference on Artificial and Natural Neural Networks, IWANN*, vol. 99.

Hermans, B. 1998, 'Desperately Seeking: Helping Hands and Human Touch', *First Monday*, vol. 3, no. 11.

Hoffman, D.L. 1999, 'Information Privacy in the Marketspace: Implications for the Commercial Uses of Anonymity on the Web', *The Information Society*, vol. 15, no. 2, pp. 129-139.

Hoffman, D.L., Novak, T.P. & Venkatesh, A. 2004, 'Has the Internet become indispensable?' *Communications of the ACM*, vol. 47, no. 7, pp. 37-42.

Hors, A.L., Hégaret, P.L., Wood, L., Nicol, G., Robie, J., Champion, M. & Byrne, S. 2000, *Document Object Model (DOM) Level 2 Core Specification*, 2006 <http://www.w3.org/TR/DOM-Level-2-Core/>.

Ingwersen, P. 1992, *Information retrieval interaction*, Taylor Graham.

Ingwersen, P. 1996, 'Cognitive Perspectives of Information Retrieval Interaction: Elements of a Cognitive IR Theory', *Journal of Documentation*, vol. 52, no. 1, pp. 3-50.

InternetWorld 2006, *Usage and Population Statistics*, Internet World Stats, viewed 6/12/2006 <http://www.internetworldstats.com/>.

JCP, *Web Services MetaData for the Java Platform*, Java Community Process, <http://www.jcp.org/en/jsr/detail?id=181>.

Kaasten, S. & Greenberg, S. 2001, *Integrating back, history and bookmarks in web browsers*, ACM Press New York, NY, USA.

Kelly, D. 2006, '*Evaluating personal information management behaviors and tools*', *Communications of the ACM*, vol. 49, no. 1, pp. 84-86.

Kernighan, B.W. & Pike, R. 1999, *The Practice of Programming*, Addison-Wesley Professional.

Kimble, C., Hildreth, P. & Grimshaw, D. 1998a, 'The Role of Contextual Clues in the Creation of Information Overload', *Matching Technology with Organisational Needs, Proceedings of the 3rd UKAIS Conference, McGraw Hill, Reading*, pp. 4050-4412.

Kimble, C., Hildreth, P. & Grimshaw, D. 1998b, 'The role of contextual clues in the creation of Information Overload. Matching Technology with Organisational Needs', *Proceedings of 3rd UKAIS Conference, April*, pp. 405-412.

King, W.R. & He, J. 2006, 'A Meta-Analysis of the Technology Acceptance Model', *Information and Management*, vol. 43 (2006), pp. 740-755.

Kirsh, D. 2000, 'A Few Thoughts on Cognitive Overload', *Intellectica*, vol. 1, no. 30, pp. 19–51.

Knublauch, H., Fergerson, R.W., Noy, N.F. & Musen, M.A. 2004, 'The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications', *Third International Semantic Web Conference (ISWC 2004)*.

Kravcík, M. & Gaševic, D. 2006, 'Adaptive hypermedia for the semantic web', *Proceedings of the joint international workshop on Adaptivity, personalization & the semantic web*, pp. 3-10.

Kreger, H. 2003, 'Fulfilling the Web services promise', *Communications of the ACM*, vol. 46, no. 6.

Kuhlthau, C.C. 1991, 'Inside the search process: Information seeking from the user's perspective', *Journal of the American Society for Information Science*, vol. 42, no. 5, pp. 361-371.

Laitenberger, O. & Dreyer, H.M. 1998, 'Evaluating the Usefulness and the Ease of Use of a Web-based Inspection Data Collection Tool', *Proceedings of the Fifth International Software Metrics Symposium*, pp. 122-132.

Lawrence, S. & Giles, C.L. 1998, 'Searching the World Wide Web', *Science*, vol. 280, no. 5360, pp. 98-100.

Legris, P., Ingham, J. & Collerette, P. 2003, 'Why do people use information technology? A critical review of the technology acceptance model', *Information and Management*, vol. 40, no. 3, pp. 191-204.

Losee, R.M. 1997, 'A discipline independent definition of information', *Journal of the American Society for Information Science*, vol. 48, no. 3, pp. 254-269.

Lyman, P. & Varian, H.R. 2006, *How Much Information 2003?*, viewed 3/12/2006 <http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/>.

Ma, Q. & Liu, L. 2004, 'The Technology Acceptance Model: A Meta-Analysis of Empirical Findings', *Journal of Organizational and End User Computing*, vol. 16, no. 1, pp. 59-72.

Machlup, F. 1983, 'Semantic Quirks in Studies of Information', *The Study of Information: Interdisciplinary Messages*, pp. 641-671.

Marchionini, G. 1992, 'Interfaces for end-user information seeking', *Journal of the American Society for Information Science*, vol. 43, no. 2, pp. 156-163.

Marchionini, G. 1997, *Information Seeking in Electronic Environments*, Cambridge University Press.

Martzoukou, K. 2004, 'A review of Web information seeking research: considerations of method and foci of interest', *Information Research*, vol. 10, no. 2, pp. 10-12.

McIlraith, S.A. & Zeng, T.C.H. 2001, 'Semantic Web services', *Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]*, vol. 16, no. 2, pp. 46-53.

Microsoft, *ATL Library Reference*, viewed 1/11/2006

<http://msdn2.microsoft.com/en-us/library/t9adwcde(vs.71).aspx>.

Microsoft 2001, *Creating Custom Explorer Bars, Tool Bands and Desk Bands.*, viewed 1/11/2006 <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/shell/programmersguide/shell_adv/bands.asp>.

Mobasher, B., Cooley, R. & Srivastava, J. 1999, 'Creating Adaptive Web Sites Through Usage-Based Clustering of URLs', *Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange*, pp. 19-25.

Morris, M.G. & Dillon, A. 1997, 'How user perceptions influence software use', *Software, IEEE*, vol. 14, no. 4, pp. 58-65.

MSDNa, *DWebBrowserEvents2 Interface*, 2006 <http://msdn.microsoft.com/workshop/browser/webbrowser/reference/ifaces/dwebbrowserevents2/dwebbrowserevents2.asp>.

MSDNb, *WebBrowser Control Overviews and Tutorials*, 2006 <http://msdn2.microsoft.com/en-us/library/aa752041.aspx>.

MSR, *MSRBot*, Microsoft Research - Web Search and Data Mining, 2006 <http://research.microsoft.com/research/sv/msrbot/>.

Nelson, M.R. 1994, 'We have the information you want, but getting it will cost you!: held hostage by information overload', *Crossroads*, vol. 1, no. 1, pp. 11-15.

Netcraft 2006, *December 2006 Web Server Survey*, viewed 6/12/2006 <http://news.netcraft.com/archives/2006/12/05/december_2006_web_server_survey.html>.

Nielsen, J. 1994, *Usability Engineering*, Morgan Kaufmann.

Nilsson, R.M. & Mayer, R.E. 2002, 'The effects of graphic organizers giving cues to the structure of a hypertext document on users' navigation strategies and performance', *International Journal of Human-Computer Studies*, vol. 57, no. 1, pp. 1-26.

Noy, N.F. & McGuinness, D.L. 2001, 'Ontology Development 101: A Guide to Creating Your First Ontology', *Knowledge Systems Laboratory, March (2001)*.

Nuernberg, P.J. 1998, 'HOSS: an environment to support structural computing', Texas A \& M University.

Oeschger, I., Murphy, E., King, B., Collins, P. & Boswell, D. 2002, *Creating Applications with Mozilla*, O'Reilly.

Perkowitz, M. & Etzioni, O. 2000, 'Adaptive Web sites', *Communications of the ACM*, vol. 43, no. 8, pp. 152-158.

Powers, S. 2003, *Practical RDF*, O'Reilly.

Pratt, W., Unruh, K., Civan, A. & Skeels, M. 2006, 'Personal health information management', *Communications of the ACM*, vol. 49, no. 1, pp. 51-55.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. & Carey, T. 1994, *Human-Computer interaction*, Addison-Wesley Publishing Company.

Pressman, R.S. 2000, *Software engineering: a practitioner's approach: European adaptation*, Ed. 5 edn, McGraw-Hill.

Prevayler 2006, *Prevayler, The Open Source Prevalence Layer*, viewed 1/11/2006 <http://www.prevayler.org/index.pr>.

Qu, Y. 2003, 'A sensemaking-supporting information gathering system', *Conference on Human Factors in Computing Systems*, pp. 906-907.

Raggett, D., Le Hors, A. & Jacobs, I. 1999, 'HTML 4.01 Specification', *W3C Recommendation REC-html401-19991224, World Wide Web Consortium (W3C), Dec.*

Rainie, L. 2005, *Internet: The Mainstreaming of Online Life*, Pew Internet and American Life Project, viewed 6/12/2006 <http://www.pewinternet.org/pdfs/Internet_Status_2005.pdf>.

Rosson, M.B. & Carroll, J.M. 2001, *Usability Engineering: Scenario-Based Development of Human-Computer Interaction*, Morgan Kaufmann.

schaefel, m., Smith, D.A., Owens, A., Russell, A., Harris, C. & Wilson, M. 2005, 'The evolving mSpace platform: leveraging the semantic web on the trail of the memex', *Proceedings of the sixteenth ACM conference on Hypertext and hypermedia*, pp. 174-183.

schraefel, M.C., Karam, M. & Zhao, S. 2003, 'mSpace: interaction design for user-determined, adaptable domain exploration in hypermedia', *Proceedings of the AH2003 Workshop*.

schraefel, M.C., Shadbolt, N.R., Gibbins, N., Harris, S. & Glaser, H. 2004, 'CS AKTive space: representing computer science in the semantic web', *Proceedings of the 13th international conference on World Wide Web*, pp. 384-392.

Schwartz, C. 1998, 'Web search engines', *Journal of the American Society for Information Science*, vol. 49, no. 11, pp. 973-982.

Shadbolt, N., Hall, W. & Berners-Lee, T. 2006, 'The Semantic Web Revisited', *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96-101.

Shannon, C.E. & Weaver, W. 1963, *Mathematical Theory of Communication*, University of Illinois Press.

Shih, H.P. 2004, 'An empirical study on predicting user acceptance of e-shopping on the Web', *Information and Management*, vol. 41, no. 3, pp. 351-368.

Shneiderman, B. 1992, *Designing the user interface: strategies for effective human-computer interaction*, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

Smith, J.B. & Weiss, S.F. 1988, 'Hypertext', *Communications of the ACM*, vol. 31, no. 7, pp. 816-819.

Sommerville, I. 1996, *Software Engineering*, 5th edn, Addison-Wesley Publishers Ltd.

Spink, A. 1999, 'Towards a theoretical framework for information retrieval in an information seeking context', in *Exploring the contexts of information behaviour* , pp. 21-34.

Spink, A. 2003, 'Web search: Emerging patterns', *Library trends*, vol. 52, no. 2, pp. 299-306.

SPSS, *Statistical Package for the Social Sciences*, *SPSS version 6.3*, S Inc.

Teevan, J., Jones, W. & Bederson, B.B. 2006, 'SPECIAL ISSUE: Personal information management, *Communications of the ACM*, vol. 49, no. 1, pp. 40-43.

Theodorakis, M., Analyti, A., Constantopoulos, P. & Spyratos, N. 1999, 'Contextualization as an Abstraction Mechanism for Conceptual Modeling', *Proceedings of the 18th International Conference on Conceptual Modeling, ER*, pp. 475-490.

Turner, J.S. 2002, 'New directions in communications (or which way to the information age?)', *Communications Magazine, IEEE*, vol. 40, no. 5, pp. 50-57.

Utting, K. & Yankelovich, N. 1989, 'Context and Orientation in Hypermedia Networks', *ACM Transactions on Information Systems*, vol. 7, no. 1, pp. 58-84.

Vakkari, P., Savolainen, R. & Dervin, B. 1998, *Information seeking in context*, Taylor Graham Pub [Sl.

van Heijst, G., Schreiber, A.T. & Wielinga, B.J. 1997, 'Using explicit ontologies in KBS development', *International Journal of Human-Computer Studies*, vol. 46, no. 2, pp. 183–292.

Vinoski, S. 2000, 'Introduction to CORBA (tutorial session)', *Proceedings of the 22nd international conference on Software engineering*.

W3C 2000, *Metadata Activity Statement*, 2006 <http://www.w3.org/Metadata/Activity.html>.

Wang, Y.-S., Wang, Y.-M., Lin, H.-H. & Tang, T.-I. 2003, 'Determinants of user acceptance of Internet banking: an empirical study', *International Journal of Service Industry Management*, vol. 14, no. 5, pp. 501-519.

Weibel, S.L. 1995, 'Metadata the foundation of resource description', *Annual review of OCLC research*, pp. 52-56.

Wichansky, A.M. 2000, 'Usability testing in 2000 and beyond', *Ergonomics*, vol. 43, no. 7, pp. 998-1006.

Wiggers, C., Galbraith, B., Chopra, V., Li, S., Bhattacharjee, D., Bakore, A., Irani, R., Bhattacharya, S. & Fowler, C. 2002, *Professional*

Apache Tomcat, Wrox Press Ltd. Birmingham, UK, UK.

Winckler, M.A., Palanque, P. & Freitas, C. 2004, 'Tasks and scenario-based evaluation of information visualization techniques', *Proceedings of the 3rd annual conference on Task models and diagrams*, pp. 165-172.

Wurman, R.S., Leifer, L. & Sume, D. 2001, *Information anxiety 2*, Que Indianapolis, Ind.

Zettel, J. 2001, *Methodological Constraints, Critics, and Technology Acceptance: An Experiment*. ISERN-Report No. ISERN-01-04 Fraunhofer IESE.