INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

IMPROVED INTERNET INFORMATION RETRIEVAL THROUGH THE USE OF USER MODELS, FILTERING AGENTS AND A KNOWLEDGE-BASED SYSTEM

by

Sima C. Newell

September 1997

Department of Electrical Engineering

McGill University

Montréal, Québec, Canada

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF ENGINEERING

Abstract

Over the past few years, the amount of electronic information available from the Internet has increased dramatically. Unfortunately, the search tools currently available for retrieving and filtering such information remain inadequate in balancing relevance and comprehensiveness. This work proposes a new Internet search paradigm. Simple user models are first combined with search specifications (the "user needs"), to form an Enhanced User Need (EUN). Uniform Resource Agents are then constructed to filter information, based on the EUN parameters. Finally, a knowledge-based system is developed to suggest those agents that are best for a researcher with a given background and search requirements. Two experiments show that users can obtain better search results from a set of HTML documents by invoking the agents recommended by the knowledge-based system, than by conducting a traditional keyword-based search. This work thus demonstrates that the use of user models, filtering agents and a knowledge-based system improves search results and may be extended to improve Internet information retrieval.

Résumé

Depuis plusieurs années, la quantité d'information électronique disponible sur l'Internet a augmenté dramatiquement. Malheureusement, les outils de recherche disponibles pour retrouver et filtrer ces informations restent insuffisants pour trouver des résultats qui sont à la fois pertinents et compréhensifs. Cette dissertation suggère une nouvelle méthode pour accomplir des recherches sur l'Internet. En premier, de simples profils d'usagers sont ajoutés à la spécification du sujet des recherches, pour former un Besoin d'Usager Augmenté (*Enhanced User Need, EUN*). Ensuite, des *Uniform Resource Agents* sont construits pour filtrer les informations en utilisant les paramètres des Besoins Augmentés. Enfin, un système expert est developé pour suggérer les meilleurs agents pour un usager quelconque, qui tient son propre formation et ses besoins de recherches particuliers. Deux expériences, chaque utilisant un groupe fixe de documents en HTML, illustrent que des usagers peuvent obtenir de meilleurs résultats en utilisant les agents recommendés par le system expert, au lieu de faire leurs recherches par la méthode traditionelle d'utiliser des mots clés. Ce travail démontre alors qu'ensemble, des profils d'usagers, des agents qui filtrent, ainsi qu'un systéme expert, ameliorent les résultats des recherches, et que ces éléments pourraient également ameliorer la retrouve de l'information sur l'Internet.

Acknowledgements

I would like to express my gratitude to all the generous people I have had the pleasure of working with while I was researching and writing this thesis. I am amazed at how fortunate I have been to encounter so many people who have put aside their own overwhelming schedules to help me through various parts of this work — simply because I asked. To *all* of them, thank you.

There are two people to whom I am most indebted — for their technical insight, their unfailing support, and for each having read more than one draft version of this thesis! First, my thanks go to my supervisor at McGill, Professor David Lowther. He provided insight, guidance, understanding, and more than a little patience. One could not hope for a better thesis supervisor! Second, I would like to thank Leslie Daigle of Bunyip Information Systems, for her precious time, her guidance and her support. Leslie provided a wealth of technical and organizational insight; she has been a huge inspiration to me.

There are many others who have helped along the way.

I must express my gratitude to Bunyip Information Systems Inc., for its financial support and the use of its resources, including a giant whiteboard in an office with a door! I would also like to extend my thanks to all the people I met at Bunyip, who have been so helpful: Peter Deutsch, for taking me seriously when I approached him at RISQ'95 and for introducing me to Leslie; Philippe Boucher and Jeff Allen for insightful brainstorming; Charles Snow, Colin Bradley, Sandro Mazzucato, William Heelan and Bibi Ali for the help with Archie and the crash course in LaTeX; Jonathan Levine for lending me his thesis as an example; and David Holmes and Ray Daoud, for keeping my workstation at Bunyip alive and well.

From McGill, I would like to thank Debbie Morzajew, Nigel Ayoung-Chee, Patrick Chan, and Mingway Huang for keeping the Engineering Microcomputing Facilities networks up and running. I would also like to thank Camille Mseikeh who provided me with the LaTeX style files I needed to format this work.

I would also like to thank Dr. Alfred Kobsa, Editor, and all the reviewers from "User Modeling and User-Adapted Interaction: An International Journal" for their helpful comments on the paper I submitted. Their feedback spilled into this thesis and helped make it more focused, insightful, and meaningful to the research community.

Finally, I would like to express my deep gratitude to my companion, Steven Uggowitzer, for his endless support and patience with me during the course of my work.

This thesis is dedicated to Aunt Anita.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past few years, the amount of electronic information available through the Internet has increased dramatically. The topic areas and target audiences have also expanded well beyond the original set of scientists and academic researchers. Unfortunately, the search tools currently available for retrieving and filtering information from the Internet remain inadequate in balancing relevance and comprehensiveness [30].

Part of the problem is the way in which a search query is specified. While users' search requirements may include information about the topic areas, the reasons for the search, and the type of results they desire, all this information must be condensed into a few words for input into most search engines. Another aspect of the problem lies in the types of search engines available, which are massive, brute-force data indexers. They perform either direct matching of keywords, or conceptual matching of phrase elements based on statistical measures. Unlike a human librarian, these engines have no memory, and cannot use information about the success of previous searches to evaluate or improve current results.

Because people are able to specify their information needs in much finer detail than what has traditionally been used as a search key, and machines with intelligent software can manipulate data in human-like ways, it would seem that an intelligent software system would be of use to keep track of what searches are conducted and why. Such an "electronic librarian" could then return data that is of real and specific interest to those who seek it. In order to develop such a system, it is first necessary to examine the existing problem in detail, and to understand the various issues involved in solving it. Then it becomes possible to develop a search paradigm designed to address the current problems and, in so doing, to improve Internet search mechanisms.

1.1 Overview of the Problem Space

The current Internet search paradigm is shown in Figure 1.1. A typical search for a given topic is conducted as follows. First, users must identify the resources they wish to use; these form the space in which the search will be performed. For example, a user seeking information regarding microwave communications might decide to search the World Wide Web (WWW), and to join a microwave-topics mailing list. In this case the Web and the mailing list form the search space for retrieving information regarding microwave communications.

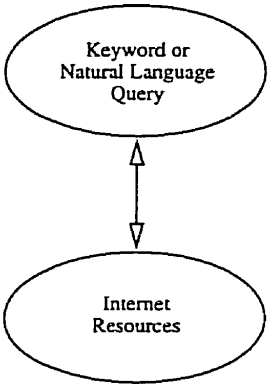


Figure 1.1: The Current Internet Search Paradigm

Users must then determine which type of search mechanism should be used for each resource type. A plethora of search engines, such as Excite, Infoseek and AltaVista, are available for use on the WWW. For searching mailing lists, users must set up software filters to sort their electronic mail (e-mail) automatically, or they must read each piece of e-mail individually (or dig through the mail archives of individual lists), and pick out the information of interest themselves.

Finally, users must specify their needs in the manner required by each search mechanism. For example, in the search for microwave communications, someone using a typical WWW-based search engine might specify the keywords "microwave and communications" as the search term.

Several problems exist with the current search paradigm. The structure of the underlying data makes searching difficult, because the information is organized by

data-type and access-protocol rather than by topic or content. The choice of search mechanisms is left to the user, and there are usually many search mechanisms associated with a single data-type. In addition, the user must create the appropriate search specification for each type of search mechanism chosen. For Web indexers, this may be simple natural language specifications or keywords linked with logical operators. There is never the opportunity to specify why the search is being conducted, or the types of results expected (such as product listings, course outlines, or job opportunities).

1.1.1 Problems with Underlying Data

The structure and content of the underlying data itself is the first source of problems in conducting searches. On the Internet, every person who sends e-mail, posts to a news group or creates a World Wide Web homepage is an electronic publisher. While this system is one of the strengths of the Internet, it does not lend itself to traditional search methods. First of all, the information is referenced by data-type, not content. This means that searching for a given topic across several mailing lists, news groups, gopher sites, World Wide Web sites, and ftp archives is non-trivial. Usually, however, it is the topic that is of interest to a researcher, not the underlying data-type or access-protocol. A few of the different types of Internet resources, distinguished by data-type and access-protocol, are given in Table 1.1. It is up to the users to access the different data-types separately (such as the WWW and the mailing list in the previous example), even if they are searching for a single topic.

Resource	Data-Type	Access Protocol
WWW	HTML	HTTP
gopher	ASCII Text	gopher
UseNet News	ASCII Text	NNTP
Mailing list	ASCII Text	SMTP
File Archives	Any	FTP
White Pages	ASCII Text,binary	Whois++, LDAP

Table 1.1: Internet Resources and Corresponding Data-Types and Access Protocols

Furthermore, while there is a huge amount of information available from all these electronic publishers, there is no reasonable measure of its validity. In the traditional system of publications, researchers know which reputable journals in their fields are likely to publish articles that are interesting, and that have also been subject to a peer review. An Internet search may provide results of all grades and levels pertaining to a given topic. The results are usually plentiful, but many — if not most — are irrelevant.

1.1.2 A Variety of Search Mechanisms

While a variety of search mechanisms exist for conducting Internet searches, none perform extraordinarily well in balancing relevance and comprehensiveness [30]. There is no measure of reliability — different search engines will return different results for the same search specification. Some return few results, while others offer the user a deluge of Uniform Resource Locators (URLs) from which to choose.

A quick test with the Excite search engine (http://www.excite.com/) exemplifies the problem. Consider, for instance, the topic defined by "microwave AND communications". A test search using Excite returned the top ten hits of a series of 19640 documents about "microwave AND communications". These hits were ranked with a relevance of 81% or less (based on unspecified criteria), and a pointer to the the next set of ten documents was provided. Unfortunately, sifting through nearly 20000 documents, ten at a time, is somewhat unwieldy. Moreover, the basis of the relevance rankings is unclear. (A similar query, "+ microwave + communications", using InfoSeek (http://www.infoseek.com/) provided 86425 results.) Search engines such as these perform an absolutely necessary task — that of indexing the vast numbers of documents available through the WWW quickly and reliably. However, for an end-user conducting an Internet search, the results can be overwhelming in their abundance. Some method is needed, therefore, to find the subset of those results that are most relevant to a particular user at a particular time.

1.1.3 The Search Specifications

All Internet search tools attempt to retrieve a set of documents that match an input search request. Some search engines take boolean statements as input, while others allow users to enter natural language descriptions of the information they are seeking. The engines then return documents that have matched the keywords given, or that have matched concepts associated with them based on statistical measures. No search engine lets users specify *why* they are conducting the search. None permits them to provide information regarding their background as it pertains to the search. None gives them a means to specify what data type needs to be indexed and accessed. None allows them to describe the type of result they expect — such as research papers, product specifications or course descriptions.

It would seem that all this information should be used to return results that are of greater relevance to the user. Simply specifying a topic by the phrase "microwave and communications" provides a poor description of a user need such as "I'm an electrical engineer searching for information on microwave communications. I've been working on synthetic aperture radar systems for five years, and I would like to find any on-line journals that are available." This is the type of description one would expect a user to provide a librarian. Therefore, to provide improved results in searching for information on the Internet, it is important to take into account elements of the user's background in addition to the specification of the search topic.

1.2 Characteristics of the Solution

To solve the problems with the current search paradigm, all three issues must be addressed. The solution must provide the ability to search by topic across different data types. The search specifications should include data such as user background information in addition to the search topic. The success may then be measured in the improvement in the quantity and quality of the hits returned, with the goal of improving the balance of relevance and comprehensiveness in the set of results.

While these form the minimum criteria for improving searches on the Internet, the new paradigm could be further refined if users with similar backgrounds were

able to share search processes and results. Suppose, for example, that John is an electrical engineer with a doctoral degree who has been working for ten years in satellite design. Jill has a master's degree in electrical engineering and has also been working several years in satellite design. Information regarding "microwave and communications" that is of interest to John might very well be of interest to Jill as well. Once John has performed a search that has satisfied his requirements, it would be interesting for Jill to be able to perform the same search. That is, if John and Jill have similar enough backgrounds, the results tailored to John's interests regarding "microwave and communications" would probably be of greater interest to Jill than general information on the topic.

Unfortunately, the structure of the Internet does not lend itself to implementing this type of solution, since existing engines do not allow information other than the search topic to be used as search criteria — nor would it be appropriate for large data indexers to store user profiles. Some other method must therefore be devised to make use of this type of information.

Another difficulty is that even if there were a means to specify aspects of a user's background in a search query, there is currently little meta-data associated with Internet information (such as HTML documents) that characterizes it — aside from the data-type, access protocol, and the occasional piece of information entered for a site's own use. There is certainly no explicit information separate from any given document that describes its target audience.

Any new search paradigm thus requires three fundamental changes in the current methodology. First, a user model must be provided as part of the search specification. Second, a mechanism must be created to retrieve existing Internet-based data using the information in the user model and the search specification. Finally, the search activity corresponding to the needs of a given user with a particular background seeking information regarding a specific topic should be encapsulated and described so that other users with similar needs can perform the same search at a future time. This work describes and evaluates a paradigm that implements these three criteria, using user models, intelligent agents, and a case-based reasoning system, with the goal of providing improved Internet information retrieval.

Chapter 2

Related Work

The problem of searching the Internet provides a new space for exploring an old problem: that of information retrieval. Information retrieval began as a science dealing with finding a comprehensive set of relevant printed documents based on a given set of specifications. Because of the explosion of electronic publishing (in various media, including the Internet), the problem of information retrieval and information filtering in electronic media has become very significant.

Several different areas of research have been applied to the generic problem of electronic information retrieval: user modeling technology has applications in this area, as do software agents, as well as case-based reasoning systems. User models provide a mechanism by which search results can be personalized. They effectively enhance the query by providing background information about the person conducting the search. Software agents are modular programmes that act on behalf of the user. An agent can filter results based on the preferences of a user contained in his or her model — they essentially act as personal electronic librarians. In addition, an agent that was successful for one user could be used for someone else with a similar background and query. If the agents are viewed as cases, then cased-based reasoning techniques may be used to find the best one. (In our comparison, this amounts to finding a good librarian.)

These three areas are important to explore, because they provide the technology needed to make the required changes to the current Internet search paradigm. The work done in the field of user modeling provides ways of incorporating a user model into a search specification. Software agents allow search activities to be encapsulated and conducted on behalf of a user, and case-based reasoning techniques can then be used to find the best agent for another user with similar needs. By examining the

work in these fields, it becomes possible to apply these various technologies to the problem of information retrieval on the Internet.

2.1 User Models in Information Retrieval and Information Filtering

In order to describe the current applications of user models in information retrieval, it is first important to note the similarities that exist in the fields of information filtering and information retrieval. Belkin and Croft [2] indicate that information retrieval research is very relevant to the domain of information filtering. The duality of these domains is important because much of the work in user modeling has been for the purpose of information filtering rather than information retrieval.

In his survey of the field of text-based information filtering, Oard [21] expands on Belkin and Croft's concept and summarizes the differences between information filtering and retrieval as follows: in the case of information retrieval, the information sources are typically "Stable & Unstructured", whereas the user's information need is "Dynamic & Specific". In the case of information filtering, on the other hand, the information source is usually "Dynamic & Unstructured", and the information need is "Stable & Specific". For the problem of finding useful information on the Internet, we are faced with a dynamic and unstructured information source as well as a dynamic (and specific) information need. Thus, both information retrieval and information filtering systems are important to the problem of information discovery on the Internet.

2.1.1 Applications of User Models in Information Filtering Systems

User modeling techniques have been applied to improve the relevance of the results of information retrieval and filtering systems. Foltz and Dumais [12] proposed using keyword-based filters, to be constructed and employed by each user, to personalize information delivery without constructing explicit user models. While the concept of *personalized* information delivery is very important, its implementation can be greatly

enhanced by using explicit, structured user models. This method permits the re-use of the same model at a later date, or on different sets of information.

For example, Raskutti et al. [25] describe the use of explicit user models in information filtering for video-on-demand (VOD) systems. The profile they use consists of attribute-value pairs which describe the preferences of the user. The index of the underlying data (in this case video data) uses the same set of attributes, thus enabling the data to be filtered according to the user's profile. Such a system has the advantage of modeling the data in a way that makes filtering easier and more fruitful for the user.

For the VOD system, the hierarchy of attributes is flat — that is, the (default) value of one attribute cannot be inferred from the value associated with another. In a more general case, inferences about the value associated with one attribute may also be made based on stereotypes associated with other attributes [27]. For example, it may be assumed that a user with an advanced degree (in any domain) is also familiar with research methods. Thus, if required, it is possible to build more complex user model systems which could be applied to the problem of information filtering and retrieval [6].

The type of system used by Raskutti et al. for VOD can be applied in a somewhat modified form to information retrieval on the Internet. While it is not possible to index Internet information based on the attributes contained in a user model, it is possible to filter Internet information (indexed by other means) based on a model consisting of attribute-value pairs. The encapsulation of such filters in software agents will be discussed in Section 2.2.

2.1.2 Collaborative Filtering Systems

Given that improved information delivery for any single user can be achieved by including a user model in the information filtering or retrieval process, further improvement can be obtained if users with similar profiles can share their results. This is the premise of *collaborative filtering* systems.

Goldberg et al. [13] first discussed the application of collaborative filtering in an electronic mail (e-mail) system called Tapestry. Their premise was that users can

help one another by contributing their feedback on messages, and thus improve upon information filtering systems targeted at a single user.

The collaborative filtering movement has since taken off in the development of recommender systems for the World Wide Web (WWW) [26]. PHOAKS (People Helping One Another Know Stuff) [29] uses recommendations of Uniform Resource Locators (URLs) mined from Usenet messages, but does not describe the user providing the recommendation. On the other hand, Kautz et al.'s ReferralWeb [16] takes into account *who* is providing a recommendation by way of his or her placement in a social network. ReferralWeb thus enhances the notion of collaborative filtering by making use of user profiles to model the originator of a recommendation. Fab [1], developed by Balbanovic and Shoham, is very appealing, in that it consists of a *hybrid* content-based recommendation system (which recommends items similar to those previously preferred by a given user) and a collaborative recommendation system (which recommends items preferred by other users with similar tastes). Rucker and Polanco's SiteSeer [28] uses bookmark files as profiles of users' interests, and thus as a basis for recommending URLs to users with similar profiles. GroupLens [18] (by Konstan et al.) uses a collaborative filtering approach to predicting articles of interest in Usenet news. One commercial system, Firefly(TM) [15] [11] uses user profiles extensively for Automated Collaborative Filtering(TM) on Internet sites.

As the term "collaborative filtering" implies, all of these systems are information *filtering* rather than information retrieval systems — they do not prompt the user for a search topic. Instead, they recommend resources deemed of interest on any topic, but for a specific type of data (e-mail messages, WWW pages or Usenet news articles). However, the idea of collaborative filtering is appealing since users with similar preferences are able to share results and find each others' favourite Internet resources. Similar technology, if applied to information retrieval, could be used to build a system that encapsulates one user's search activities pertaining to a specific topic, using software agents, so that another user with a similar background might perform the same activity at another time.

2.2 Software Agents in Information Retrieval

Software agents are programmes that act on a user's behalf — essentially as personal assistants [19]. Much work has been done in the area of using software agents as tools in electronic information retrieval, because they encapsulate activities, and thus modularize elements of the search.

For instance, agents such as Etzioni and Weld's Softbot [10] may be used to abstract the goal (in our case, information retrieval) from the method, actions or underlying protocols that are required to accomplish it. This is an important technology for information retrieval, since a user's particular information need is no longer limited by the specifications of the protocols used to fulfill it. (It is also the basis of Uniform Resource Agent technology[5], which is discussed further in Chapter 4.) In a larger system, such agents may be coordinated in a "distributed problem solving" approach, such as the "collaborative information gathering" (CIG) method proposed by Oates et al. [22]. In cases in which there are sets of heterogeneous information sources and complex queries (such as for information retrieval on the Internet), Oates suggests using a set of cooperating, semi-autonomous agents that would form a search organization to respond to a complex query.

Cooperative agents ("userbots" and "corpusbots") are also at the core of Vorhees' InfoScout [31] system. Each userbot comprises a user profile, statistical data regarding the user's categorization of information, scripts for automating recurring tasks, and a set of references to other userbots of interest. The userbots communicate queries to the corpusbots (which represent a particular data collection). The advantage of such a system is that it not only abstracts the data retrieval and manipulation from the query (as did Etzioni and Weld's Softbot), but it also allows users with similar interests to share query results. However, it does not provide an automated recommendation of different userbots to a user — the user must know a priori about other userbots of interest. Whereas Fab [1] provides a method of collaborative information filtering, InfoScout provides both collaborative information retrieval and filtering. This is an effective system, and is similar in paradigm to the Internet information retrieval solution proposed in this work.

2.3 Case-Based Reasoning in Information Retrieval

The component that is lacking from Vorhees' InfoScout system is a method of automating the recommendation of userbots to other users with similar needs. If each userbot is viewed as a case, then the field of case-based reasoning (CBR) lends itself naturally to solving this problem.

Research has been conducted by Daniels and Rissland [7] in applying CBR to information retrieval. They propose a hybrid case-based reasoning and information retrieval system geared toward finding relevant cases in the legal domain. Their premise is that CBR systems are designed to retrieve highly relevant cases, but are limited by the availability of cases. Conversely, in the domain of full-text information retrieval, there is no lack of cases, but it is difficult to find those that are relevant. They found that their hybrid CBR-information retrieval system "significantly outscores straight information retrieval alone". Thus, applying CBR methods to the problem of Internet information retrieval would seem like a very promising approach.

Hammond et al. [14] address the problem of knowledge navigation and information filtering using "Find-Me Agents", which act in a preset domain. One type of Find-Me agent, the "Car Navigator" was constructed to help car buyers choose cars that best meet their needs. The CBR system constructs a user model based on the users' selections (e.g the type of car sought), and then presents cases that may be of most interest. The user model is refined as the users narrow their selections, by specifying elements such as price range and fuel economy. Hammond et al. have paid special attention to developing a user interface that is exceptional at shielding users from the underlying searches conducted. The drawback to their system is that the structure of the user models is domain dependent — that is, the semantics of the domain dictate the elements used to find the best cases. This limits the possibility of adapting the system to other domains, and creating a more generalized information retrieval system, as would be required for the Internet.

2.4 Summary

The current problem of the explosion of information available electronically is widely recognized by researchers in many domains. Work in the areas of user models, intelligent agents, collaborative filtering, and case-based reasoning systems is being applied to traditional information retrieval problems. Each paradigm has its advantages and limitations, but each offers the promise of achieving partial solutions. A hybrid system that makes use of all these elements — perhaps something similar to Vorhees' InfoScout system — could be constructed to meet the requirements for a search paradigm that will overcome the current difficulties in Internet information retrieval.

Chapter 3

Proposed Solution

In order to provide improved information retrieval on the Internet, a search paradigm was designed that uses information filtering techniques, user models, and a case-based reasoning system to overcome the problems inherent in the current search systems. The system architecture devised allows an enhanced search specification that contains user background information in addition to the topic description. It also encapsulates sets of Internet activities in order to perform searches based on common content rather than on a single data-type. This encapsulation is achieved through the addition of software agents that use the enhanced search specification to define their search criteria. Information that describes the agents is introduced in the model so that users may find and use the agents of others with similar backgrounds. The following sections provide an overview of the proposed system architecture, and describe the implementation goals of the experimental work conducted in evaluating a prototype system.

3.1 System Architecture

The proposed system architecture is shown in Figure 3.1.

With this model, a user doing a search is faced with two options:

1) The user may construct a software agent that filters search results from several traditional search mechanisms according to his/her background and topic specification. For example, John might create an agent that would filter the hits returned from the search specification "microwave and communications and radar" for research papers pertaining to satellite design. If John were to use such an agent to perform his search, he should obtain results that are better suited to his needs

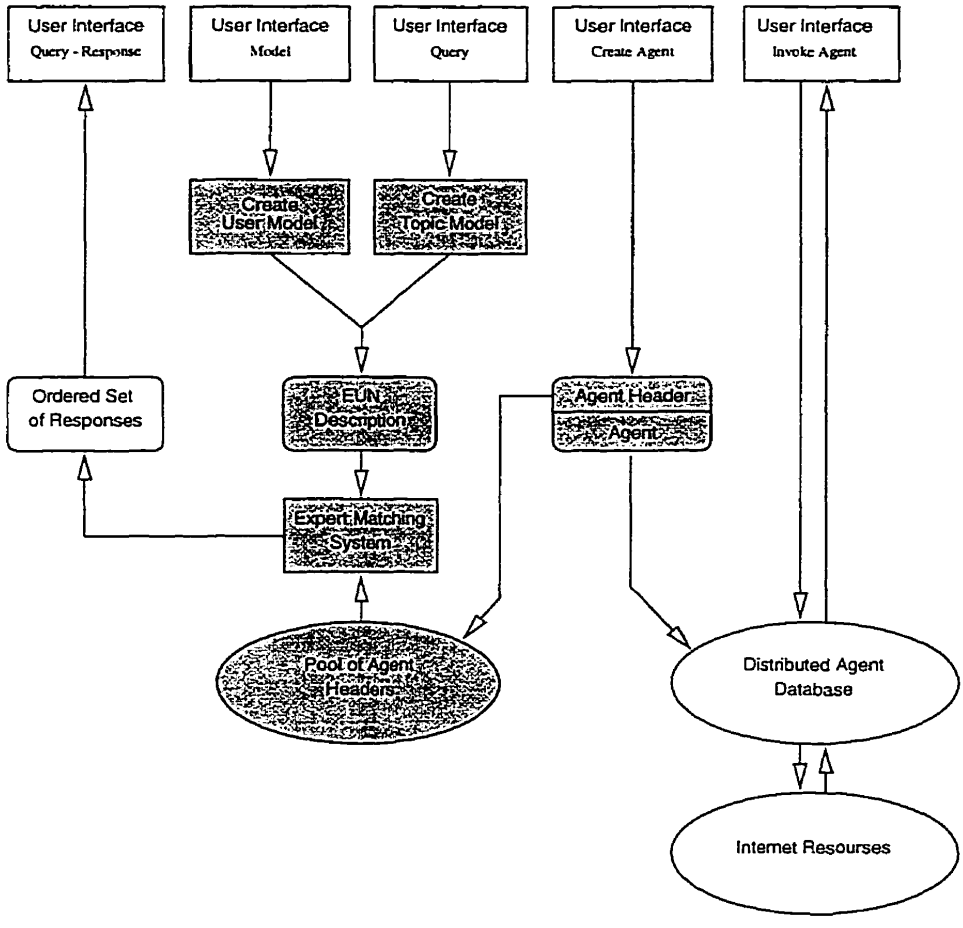


Figure 3.1: Architecture of the Proposed System

than if he were to do a simple keyword search for "microwave and communications and radar", because they have been filtered appropriately.

or

2) The user may search a pool of existing agents to find one already created by another user with a similar background to do a similar search. Suppose Jill has a similar background to John: he has a Master's degree in Electrical Engineering and has been working in satellite design for five years. In this case, the results returned by John's agent would probably be of interest to Jill.

By adding the intermediary of a software agent to the traditional search system, it is possible for the user to access many resources of different data-types through a single search request. The agent can then filter the search results from the different

resources based on the user's background. This system permits other users to perform the set of Internet search and filtering activities encapsulated by the agent, thus making use of the experience of the agent's creator.

3.1.1 Overview of the System Components

The top level of the system shown in Figure 3.1 is the User Interface. The users have the option to (from left to right):

- view the results of their queries
- create their models by inputing background information
- perform search queries
- create their own agents
- invoke the agents returned as results

Note that the construction of a perfect user interface was not the goal of this work, and only the necessary components of it were implemented in a very rudimentary fashion.

The middle section of the diagram (the shaded boxes in the diagram) constitutes the focus of this work. The function of each shaded box is described below, but the specific implementation details for the experiments performed will be discussed in Chapters 4 and 5 . The components may be scaled in complexity depending on the desired functionality of the final system.

Create User Model

This module creates the system's model of the user. Such elements as education level and area, career area and job function [6] would be typical elements of an explicit user model. Information such as the statistical data on the user's categorization of information used in InfoScout [31] could also be used as a system-generated user model.

Create Topic Model

This element creates the system's model of the search topic. This model could include a keyword or natural language description of the topic, an evaluation of the user's knowledge level in the area (novice, intermediate or expert, for instance), and the depth and breadth of his/her knowledge of the topic[6].

EUN Description

The Enhanced User Need (EUN) is an enhanced search specification that combines the information in the User and Topic Models. It is hypothesized that elements of both models are relevant to information retrieval requirements. The EUNs associated with the background and search requirements of an agent's creator are included within the Agent Header. New EUNs may then be matched against the agents' descriptors in order to find those agents that perform searches similar to those requested by other users.

Agent Header and Agent and Pool of Agent Headers

The agents encapsulate a set of Internet search activities associated with an EUN of the creator, similar to the Softbots proposed by Etzioni and Weld [10] and Vorhees' userbots [31]. Their headers form meta-data that describe the agent's functionality and the creator's EUN. The headers in the pool will be used to match new EUNs to existing agents.

Expert Matching System and Order Set of Responses

The case-based reasoning system is used to match the EUNs and agent headers. It should be noted that this is the primary functionality missing from the InfoScout [31] system — a method for automatically finding other users' agents. The case-based reasoning system's size and complexity will depend in part on the completeness of the set of descriptors used in the EUNs. (With more background descriptors, for instance, more rules would be needed in the system.) It could also comprise a variety of useful sub-systems such as a thesaurus or semantic mesh. This would allow different terms with similar meanings (such as "voltage" and "electric potential") to be compared. The case-based reasoning

system should output an ordered set of responses to be directed back to the user.

The last two elements of Figure 3.1, are the Distributed Agent Database and the Internet Resources being acted upon. A distributed database of agents would be necessary in an environment such as the Internet. For example, a Whois++[9] server at each site hosting an agent database could provide access to the agent headers. The structure of such a database is beyond the scope of this work, but is mentioned here for the sake of completeness.

3.1.2 Purpose and Goals

The underlying hypothesis is that if the user models, search specifications and filtering agents are properly constructed, a user finding the agent of someone with a similar background will obtain better search results than by conducting a search using traditional methods.

The stated hypothesis raises a number of questions: how close in background must Jill and John be? Should certain aspects of the users' models be matched more closely than others? How is it possible to quantify "better" search results? This work attempts to answer some of these questions through a constrained experimental implementation of the system in Figure 3.1. The methodology involved determining some of the elements of users' backgrounds that influence their search requirements, developing simple filtering agents based on the search criteria, and creating a small case-based reasoning system to allow users to find the agents of those with similar search requirements. The goal is then to evaluate the results of the interaction of these components in providing more effective search results.

3.2 Summary of methodology

To overcome the problems associated with current Internet search systems, a new information retrieval paradigm has been proposed that may enable Internet researchers to obtain more useful and accurate search results. The proposed system allows users to create software agents that encapsulate different Internet activities based on a

common search topic rather than on a common data type or access protocol. The agents filter the search results based on elements of the users' backgrounds that affect the type of results they desire to obtain. The combination of search area and user model descriptors is referred to as an Enhanced User Need, and forms a part of the meta-information used to specify the functioning of each agent.

Because meta-information exists to describe the agents' contents and behaviour, it is possible for other users to find and use the agents whose meta-data most closely match their own EUNs. A case-based reasoning system is used to recommend existing agents that are best suited to fulfill incoming user requests with existing agents. By using a suitable distance function, the results from the expert system can then be compared to the results of a straight keyword-based search, and to the results of the user's own agent. By experimenting with these components, it is then possible to evaluate the validity of the proposed paradigm.

Chapter 4

Methods

Given the architecture of the proposed system, it is necessary to evaluate its validity — that is, to determine whether such a model is able to offer consistently better search results than the existing Internet search paradigm. While research has been conducted in each area individually (user modeling, intelligent agents, case-based reasoning (CBR) systems and Internet information retrieval), the purpose of this work is to evaluate a system that combines all these technologies. As a starting point, Figure 4.1 compares the conventional Internet search paradigm with the proposed model. The shaded boxes correspond to the elements of the current search paradigm (from Figure 1.1. The other elements illustrate the components added in the new paradigm: EUNs, agents, a CBR system and a group of agent creators.

In order to evaluate the effect that the addition of *all* these components has on the search results, an implementation of the new paradigm must be developed. Obviously, choices regarding the structure of the EUNs, the functionality of the agents and the construction of the CBR system will affect the final results, and it is the performance of a particular implementation rather than that of the architecture that will ultimately be evaluated. The hope is that the performance of the chosen implementation will *validate* the original architecture and pave the way for further research in developing an improved method for Internet information retrieval based on the proposed paradigm.

Given the goal of validating the new model, the methodology chosen comprised three steps. First, the components added in the new search model were constructed. Then, a method was devised to compare traditional results with those obtained by the new system. Finally, experiments were conducted to compare the results of specific implementations of the system to those of the traditional methods. This section

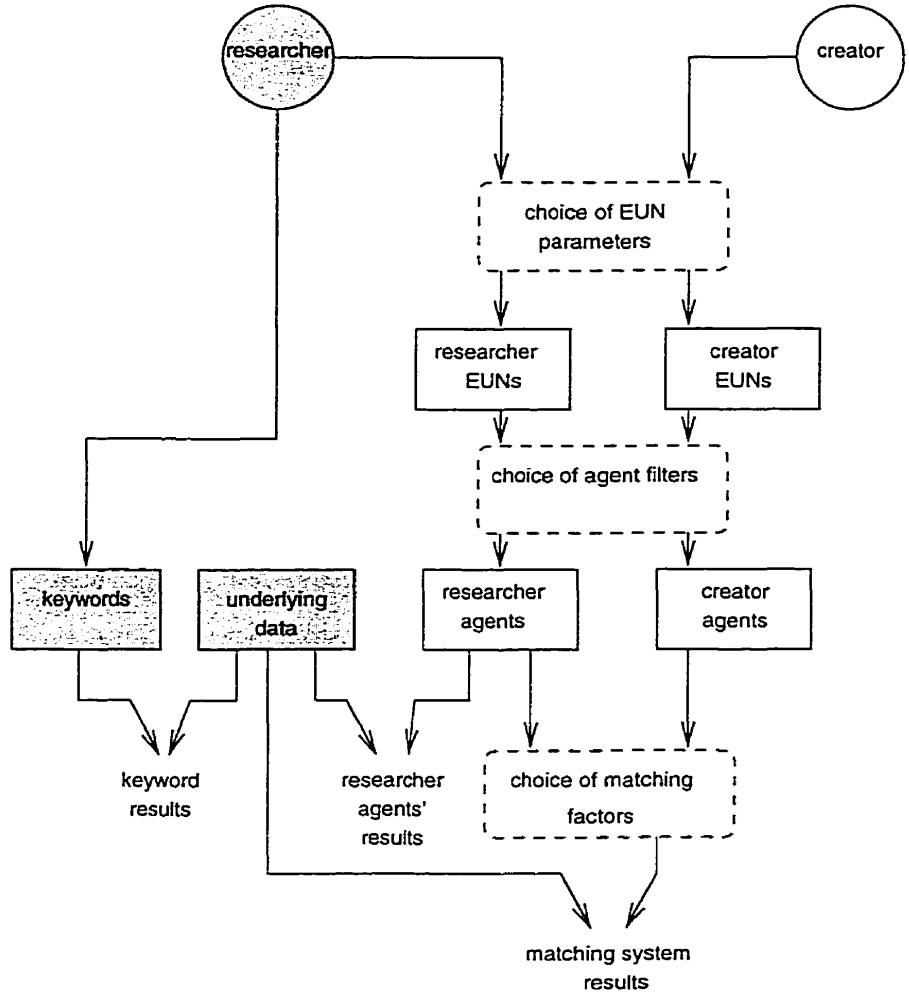


Figure 4.1: Comparing Search Paradigms

describes the design of the EUNs, agents and the CBR system, as well as the method by which search results were compared to determine whether the implementation of the proposed architecture lead to improved search results.

4.1 Component design

The components of the model shown in Figure 4.1 were designed with the goal of validating the proposed paradigm using as simple an implementation as possible. Thus, the complexity of each component was minimized since only a validated paradigm would justify improving each component to produce a better overall implementation.

The design of the user-need specifications, the filtering agents and the CBR system is detailed below.

4.1.1 User-Need Specifications

The first important component of the experimental system is the specification of the user's needs. This was implemented as a set of Enhanced User Needs (EUNs) — a combination of information about the search requirements and the users' backgrounds. All data in the EUNs was structured as lists of attribute-value pairs, similar to those in Raskutti's[25] video-on-demand system. Symbolic values are advantageous in that they may be input easily to a rule-based system's knowledge base (for symbolic matching), and they also form a simple basis on which agents may filter search results. In addition, using attribute-value sets permits data about the search topic to be separated from that regarding the user's background, allowing the effects of each to be evaluated individually.

Structure of the Enhanced User Need Specification

The Enhanced User Needs were created by combining a topic model (search specification) and a model of the user's background. An example EUN would be:

```
(EUN example-1
  (TOPIC        (SPECIFIC_AREA       radar)
                (KNOWLEDGE_LEVEL     expert))
  (BACKGROUND   (DEGREE_AREA         electrical engineering)
                (DEGREE_LEVEL        bachelor))
                (WORK_AREA           electronics)))
```

where the attribute names are in capitals (e.g. "SPECIFIC_AREA"), and the symbolic values are in lower case (e.g. "radar").

Several assumptions are inherent in this model. First, the user model is based on the traditional written user model, the curriculum vitae, and thus contains information about both academic and work experience. Although initial work was done in building a much more complex user model based on stereotypes [27] from a CV [6]

(which included information about several previous jobs and all degrees obtained, as well as information about hobbies and interests), the experiments in this work used only a small number of parameters. While previous job experience and academic experience may affect a user's view of a specific area, and hence his/her rating of a set of search results, it was much simpler to use a small number of parameters identifying the user's most recent experience as an initial test model. From this test, it may ultimately be possible to generalize the results in order to understand the influence of *classes* of parameters instead of that of individual descriptors. However, a simple test EUN, like the one above, provides a good starting point for validating the approach under consideration.

In addition, there is no semantic knowledge of the EUN values built into the system. That is, there is no means to know how closely related "electronics", "electrical", "engineering" and "radar" actually are in the previous example. Semantic knowledge, such as that provided in libraries such as LinguistX from Xerox Parc [8] would easily permit searches based on related words, and could help retrieve documents that are contextually relevant. However, for our purposes (with the goal of simplicity), the assumption was made that the specific area of the search is closely related to the user's background, and that a search for some combination of the terms provided will return meaningful results.

4.1.2 Agent Structures

Given a set of EUNs, the requirement for the agents was to filter Internet search results based on the user needs. Uniform Resource Agent (URA) technology [5] provided the means to encapsulate the types of search and filter mechanisms required. Two sets of six agents were created for each experiment using simple filtering mechanisms based on the EUNs, and meta-information was created that described each agent. This meta-information was then used as a basis for the comparisons performed by the CBR system.

Uniform Resource Agents

Uniform Resource Agents [5] provide a structured means to encapsulate different Internet activities, such as repetitive searching, as well as the parsing and filtering of results. The agents were developed within the Silk [4] environment.

There is a subtle, but important, difference between creating a search agent that encapsulates the necessary filtering knowledge that is based on a given EUN, and adding a generic user-model-based filtering mechanism to an existing search engine. By creating a filtering agent, the choice of filter is in the hands of the user, not the search service provider. For instance, a user may choose to place the search and filtering mechanisms into separate URAs, hence allowing different types of searches and filters to be mixed and matched. Also, using agents allows users with similar EUNs to find and use agents already written by others with similar needs — since all the filtering knowledge is contained within the agent itself. It is for these reasons that URA technology was deemed appropriate for this purpose.

Filtering Methodology

The URAs were constructed to retrieve data based on the search topic and then to filter them based on elements of the creator's background. The simplest choice of filter was to conduct a series of searches, refining each incrementally, and then to return the results in order — from the most specific to the least specific search.

For example, suppose the sample EUN given in Section 4.1.1 described the needs of an agent's creator. The agent in this case would perform five separate sub-string searches for the following:

(a) `radar`
(b) `radar and electrical`
(c) `radar and electronic`
(d) `radar and electronic and electrical`
(e) `radar and electronic and electrical`
`and not (introductory or novice)`

and would then return hits in reverse order (from (e) to (a)), and eliminate multiple

instances of any hit. To search for a different topic, the user would have to construct a separate agent — or find a suitable one constructed by someone else.

This filtering system is simple, but functional. A better agent might allow for relevance feedback from the user, and thus adapt its filtering based on the user's ranking of the returned documents. However, the chosen filtering method provides a direct relationship between the elements of the EUN and the search performed, thus singling out the effects of each EUN parameter.

Meta-Information

An important problem with existing Internet search systems is the lack of information available to describe the contents of electronic documents such as web pages or posts to UseNet news. The only way to determine the content of a document is to parse it and conduct keyword-based estimations of its contents. To avoid a similar problem in evaluating the agents' behaviour, their headers comprise sets of meta-information that describe the agents, but that may be stored separately from the agents themselves. Ideally the meta-information would contain data regarding both the EUN of the agent's creator and the functioning of the agent itself.

To reduce the complexity of the CBR system, the meta-information for each agent was limited to the information in its associated EUN. Since all the agents performed the same type of filtering, descriptors of the filtering mechanisms were omitted. It would be possible, however, for different agents to perform different types of searching and filtering based on the same EUN. In such cases, descriptors of an agent's functionality would form a necessary component of its meta-information. In any case, given the meta-information, it becomes possible to match existing agents automatically to the EUNs corresponding to the search requirements of different users.

4.1.3 Matching New Queries with Existing Agents

With the structures of the EUNs and agents defined, the system permits users to specify elements of their backgrounds as they pertain to a search, and to construct agents that use these elements in filtering search results. Suppose, however, that another Internet researcher has a similar background and search requirements to one

of the agent-creators. The system is still lacking the structure needed to match a researcher's query with the meta-information describing existing agents. For this reason, a scaled-down CBR system — effectively a simple knowledge-based system was written to compare researchers' EUNs with those described in the meta-information of existing agents. A complete CBR system should adapt previous cases (in our case, agents) to solve new problems (here, new queries)[17]. However, such a complex system was not required for the purposes of validating the proposed paradigm. Thus, the KBS implemented found the closest agent for a given query, but did not modify the agent in any way to better fulfill the request.

To build the knowledge-based system, rules were devised which govern the matching between the sets of topic attributes (search area and knowledge level) and the user profile attributes (degree area, degree level and work area) of the researchers and agent-creators. The purpose of the system was to evaluate how good the match was between a researcher's EUN and that of each agent already written. Each rule modified a factor that provided a measure of how close the match was, based on a specific set of heuristics.

System Design

The system was designed to rank researchers' EUNs with those of the agents' creators by modifying a parameter (the match factor) describing how good the match was for any EUN-agent pair. Note that the match factor does not describe the validity or applicability of the rule, but rather the cumulative effects of differences in values associated with the EUN attributes. The match factor (m_f) associated with such a pair had an initial value $m_{f0} = 1.0$. This value was then modified with each rule fired, by a multiplicative factor f_j:

$$m_f = \prod_j m_{f0} \cdot f_j$$

where f_j is the multiplicative factor associated with the j^{th} rule. These multiplicative factors are categorized by attribute-type, as shown in Table 4.1.

As seen from the first column, if a perfect match was found between the researcher and the agent-creator for any of the five attributes, the factor f was set to

	Match	No-match
TOPIC_AREA	$f = 1.0$	$f = 0.0$
KNOWLEDGE_LEVEL	$f = 1.0$	$0.0 < f < 1.0$
WORK_AREA	$f = 1.0$	$0.0 < f < 1.0$
DEGREE_AREA	$f = 1.0$	$0.0 < f < 1.0$
DEGREE_LEVEL	$f = 1.0$	$0.0 < f < 1.0$

Table 4.1: Categorization of Match Factors

1.0, and the match factor for that EUN-agent pair was not modified. On the other hand, if there was no match for any given attribute, the multiplicative factor was set to 0.0 indicating no match was found. Cases in which the multiplicative factor would be zero would include those in which the subject area differed (as above), and, in an enhanced system, those in which semantic knowledge indicated that a potential match was invalid. In the simple system implemented, only differences in TOPIC_AREA resulted in a multiplicative factor (and thus, a match factor) of 0.0.

For the other four parameters, if the values did not match, the factor was chosen between 0.0 and 1.0. For example, if the researcher had an intermediate level of knowledge in the topic area, and the agent was created by an expert, then the match should be better than if an expert researcher found an agents written by a novice. For each corresponding rule, the factor would be set accordingly:

```
If the (creator-knowledge-level = expert)
   and (researcher-knowledge-level=intermediate)
 then f=0.9
```

```
If the (creator-knowledge-level = novice)
   and (researcher-knowledge-level=expert)
 then f=0.65
```

Thus, the first scenario would result in a better match than the second. The specific choices of all the f_j are discussed further in Chapter 5.

Knowledge-Based System Output

Once the inferencing was complete (there were no rules left on the agenda), the final value of m_f for each EUN-agent pair allowed the system to rank the agents in terms of how well they were matched against the given search request. The system could then suggest which agent(s) might be of greatest interest to the researcher. The effectiveness of the inferencing obviously depends on the consistency of the knowledge base and the validity of the chosen match factors. Because agents are ranked based on how well the researcher's and agent-creators' backgrounds matched, the hypothesis is that the top ranking agents will produce better results than a simple keyword search for the researcher's topic-area. The remaining problem, therefore, is to be able to compare the results of the suggested agents to those of the corresponding keyword search in order to verify this hypothesis.

4.2 Comparing the results

Given two different sets of search results for the same topic, the task is to determine which set is better — that is, which set is closer to the researcher's ideal set of results. However, if the researcher's ideal set of results were known a priori, the search problem would be solved by definition! Therefore, in order to verify the hypothesis, it is necessary to conduct experiments in which the ideal set may be identified. Once the ideal set has been established, a method must be devised to compute the distance between it and each set of results. We may then determine if the results given by the suggested agent are (or are not) closer to the ideal set than those of the keyword search, and thus verify the stated hypothesis.

4.2.1 Identifying the Ideal Set

Consider a researcher looking for information on "microwave AND communications AND radar". This researcher may obtain three different sets of results:

S1: from a keyword search for "microwave AND communications AND radar";

S2: by writing an agent which searches for "microwave AND communications AND

radar" and then orders the results based on the researcher's background and preferences;

S3: by writing an agent which searches for "microwave AND communications AND radar" and orders the results based on the preferences of someone with a similar background (a collaborative retrieval approach).

While the ideal set of results is unknown, it may be assumed that of the three sets of results, the second set is the best one for the researcher. If s/he wrote an agent that did no filtering, S2 and S1 would be identical; by adding filters based on his/her background, S2 will be better for the researcher than S1. Since the filters are for the researcher's preferences (and not those of someone else), S2 will also be better than S3. Ideally, the agent in (2) would be perfectly tailored to the researcher's needs, and would thus offer a good approximation of the ideal set. The only way to ascertain this would be through experimental user testing. However, we may assume that S2 is the best known approximation.

Therefore, by writing an agent that conducts a search based on a researcher's EUN, we may assume that it returns a reasonable approximation of the ideal set for that researcher. We may then compute and compare the distances $D(S1, S2)$ and $D(S3, S2)$ and verify which set of results (S1 or S3) is closest to the set S2, which approximates the ideal set.

If we use this method to compare the results of the agents found by the knowledge-based system to those written by the researchers, a measure may be obtained of how well the knowledge-based system performed. Because of the way in which the filtering was conducted by the agents, they all operated on the same set of hits — only the *ordering* of the results differed between agents operating on the same topic. The following section describes the measure chosen to evaluate the distance between two sets of results.

4.2.2 Choice of a Distance Function

If each set of hits in a file is considered to be a point in the space consisting of all possible orderings of hits for the SPECIFIC_AREA, the distance between two sets of hits (two points in the space) should have the following characteristics:

1. if the hits are in the same order, the distance is zero;

2. if the hits are in different orders, the distance will be greater than zero;

3. if the hits at the top of the list are out of place, the distance will be greater than if the hits at the bottom of the list are out of order. This means that in a set of 100 hits for "radar" , it is worse for elements of the top 10 hits to be out of place than for those of the last 10.

From these requirements, the following was used as the distance function $D(x, y)$ for two sets of hits, x and y, in the space:

$$D(x,y) = \sum_{i=0}^{N} Z_i(x,y) \ , \ where \ Z_i = \{ \mid w_i \cdot i - w_j \cdot j \mid \mid x_i = y_j \}$$

where i is the index of the element in the set x, j is the index of the corresponding element in the set y, and w_i is the weight associated with the position i in the set. N is the total number of hits (for the SPECIFIC_AREA). The algorithm to calculate $D(x, y)$ is as follows:

1. while there are still elements left, get x_i, the i^{th} element of x

2. search y for the element $y_j = x_i$

3. calculate $|(w_i)i - (w_j)j|$, where w is the weight associated with the index of an element, and $w_1 > w_2 > ... > w_N$

4. increment i

5. go to 1

$D(x, y)$ may be shown to be a valid distance function using the criteria in [24]. The function

$$D(x,y) = \sum_{i=0}^{N} Z_i(x,y) \ , \ where \ Z_i = \{ \mid w_i \cdot i - w_j \cdot j \mid \mid x_i = y_j \},$$

where i is the index of the element in the set x, j is the index of the corresponding element in the set y, and the w_i are the weights associated with the position i in the

set, is said to be a distance function iff:

1.

$$x \neq y \Rightarrow D(x,y) > 0 \; and \; D(x,y) = D(y,x)$$

2.

$$x = y \Rightarrow D(x,y) = 0$$

3.

$$D(x,y) \leq D(x,z) + D(z,y)$$

Property 1 is true since $|w_i i - w_j j| > 0$ by definition of the absolute value.

Property 2 is true since if $x = y$, $i = j$ for $x_i = y_j$ in all cases. Therefore, $|w_i i - w_j j| = 0 \; \forall \; i, j$ and $D(x,y) = 0$.

Property 3 is shown to be true since:

$$D(x,z) + D(z,y) = \sum_{i=0}^{N} \{ \mid w_i \cdot i - w_k \cdot k \mid \mid x_i = z_k \} + \sum_{k=0}^{N} \{ \mid w_k \cdot k - w_j \cdot j \mid \mid z_k = y_j \}$$

Since all values of x_i are unique, we get:

$$D(x,z) + D(z,y) = \sum_{i=0}^{N} \{ \mid w_i \cdot i - w_k \cdot k \mid + \mid w_k \cdot k - w_j \cdot j \mid \mid x_i = z_k = y_j \}$$

Since $|A + B| \leq |A| + |B|$,

$$D(x,z) + D(z,y) \geq \sum_{i=0}^{N} \{ \mid w_i \cdot i - w_k \cdot k + w_k \cdot k - w_j \cdot j \mid \mid x_i = z_k = y_j \}$$

Giving

$$D(x,z) + D(z,y) \geq D(x,y)$$

So, all three properties hold, and $D(x,y)$ is a valid distance function.

Therefore, the chosen $D(x,y)$ is a valid distance function for all w_i, and we may

choose the w_i such that a greater weight is placed on elements at the top of the list than on those at the bottom. The following function was used to determine the w_i:

$$w_i = \frac{(N-i)^2}{N^2}$$

With this weighting function and the given distance function, it is possible to compare the results of the suggested agents and the keyword search to the results obtained from the researcher's agent, and hence to determine how well the proposed paradigm performs.

4.3 Summary of the Methods

Given the goal of validating the proposed paradigm, a specific implementation has been developed with the following characteristics. Instead of a traditional query consisting of a keyword or natural language phrase describing the topic, an Enhanced User Need with five parameters is used. The EUN describes the subject area, the user's proficiency (knowledge level) in that area, as well as the user's degree area, degree level and work area. The EUNs are then used as bases for filtering systems encapsulated in Uniform Resource Agents. Meta-data is provided to describe the URAs, so that a knowledge-based system can retrieve those of interest to other users. Finally, a distance function is developed in order to compare the results obtained using a traditional keyword search to those retrieved using the implementation of the proposed paradigm. The specific experiments and results obtained using these methods are described in the following chapter.

Chapter 5

Experiments and Results

Two experiments were conducted using the methodology described in the previous chapter. Each experiment was run on a subset of documents in a particular domain ("microwave and communications" in the first experiment, and "english and literature" in the second). Sets of EUNs were chosen, and filtering agents constructed. The results from those agents recommended by the knowledge-based system to each individual of a set of researchers were then compared to those obtained in a random order from an Archie keyword search. Each experiment and its results are described in the following sections.

5.1 Description of Experiment 1

The first experiment consisted of implementing some of the components of the proposed system and evaluating their performance. To this end, a few additional constraints were placed on the system. First, a test set of 500 HTML documents was created with which to experiment. The fixed test set ensured that the data did not change over the course of the experiment, and restricted indexing and search methods to www-based resources. Eleven EUNs were then developed, and the twelve corresponding agents were written. Six of the EUNs were placed in the *creator* group, corresponding to users that create their own agents. The other five belonged to the *researchers* – those that were seeking information. The goal of the experiment was to determine whether it is better for a researcher to find a pre-existing agent recommended by the system (i.e one of the creators' agents), or to conduct an Archie keyword search, which returns results in a random order. Preliminary work by the author in this area is described in [20].

Using the researchers' EUNs, the knowledge-based system then ranked the creators' agents (by matching their meta-data to the researchers' EUNs), producing an ordered list — from the best matched creator's agent to the worst one — for each researcher. Finally, the results returned by the highest ranking creators' agents were compared to those of the researchers' agents and to those from straight keyword searches, in order to determine which method (using the recommended agent or performing a keyword search) produced better results for the researcher.

5.1.1 Choice of the Test Set

The 500 documents from the test set were chosen arbitrarily as the top 500 hits from Excite on the search request "microwave and communications". A Tcl [23] script was written to parse the output from Excite and to obtain the URLs of the top 500 documents. Another script was then used to download each document to a local site, so that the set would remain unchanged over the duration of the experiment. Sites that could not be accessed resulted in thirty-six (36) zero-length files; these files represented search hits that would have been of no relevance to the user.

The set of 500 HTML documents was then indexed using the Archie [3] Web indexing software. The zero length files did not contribute to the index, and thus would not ever be returned as hits from Archie. Archie was chosen because its current version includes a parameter that, when set, forces the output to be in attribute-value format, in plain ASCII text. This feature makes it very easy for software agents to parse the output of an Archie search. In other cases, agents must parse HTML output — which can change frequently, as the look and feel of a page is altered (even if the content is not).

5.1.2 Choice of Creators and their Agents

A set of EUNs describing the six creators was constructed with the parameter values given in Table 5.1. Note that the EUNs of Users B-F each differ from that of User A by one parameter.

The creators' URAs were constructed to search the test set of 500 documents, and then to filter the results based on each of the six EUNs. The filtering of Agents

	A	B	C	D	E	F
S_AREA	radar	digital	radar	radar	radar	radar
K_LEVEL	expert	expert	novice	expert	expert	expert
D_AREA	electrical	electrical	electrical	mechanical	electrical	electrical
D_LEVEL	bachelor	bachelor	bachelor	bachelor	master	bachelor
W_AREA	electronic	electronic	electronic	electronic	electronic	antenna

Table 5.1: EUN Parameters of the Creators in Experiment 1

A-F was conducted as follows:

1. they performed Archie sub-string searches for the following:

   (a) SPECIFIC_AREA

   (b) SPECIFIC_AREA and DEGREE_AREA

   (c) SPECIFIC_AREA and WORK_AREA

   (d) SPECIFIC_AREA and DEGREE_AREA and WORK_AREA

   (e) SPECIFIC_AREA and DEGREE_AREA and WORK_AREA and term_0 and term_1

   where term_0 and term_1 are given by:

```
term_0 =     (introductory or novice) if  (KNOWLEDGE_LEVEL =  novice)
term_0 = not(introductory or novice) if  (KNOWLEDGE_LEVEL <> novice)
term_1 =     (research)                iff ((DEGREE_LEVEL=master) or
                                            (DEGREE_LEVEL=doctor))
```

2. they returned hits in reverse order (from (e) to (a)), and eliminated multiple instances of any hit.

As a reference, the Tcl source code for Agent A is included in the Appendix A. Given the test set of 500 hits, the six URAs were run, producing six sets of results: one set for each of the creators' EUNs.

5.1.3 Choice of Researchers and their EUNs

The search needs of the researchers, labeled as Users ALPHA-EPSILON, were described by the EUNs in Table 5.2. Each researcher's EUN differed from that of a creator by a single parameter.

	ALPHA	BETA	GAMMA	DELTA	EPSILON
S_AREA	digital	radar	radar	radar	radar
K_LEVEL	expert	intermediate	expert	expert	expert
D_AREA	electrical	electrical	computer	electrical	electrical
D_LEVEL	bachelor	bachelor	bachelor	doctor	bachelor
W_AREA	electronic	electronic	electronic	electronic	transmitter

Table 5.2: EUN Parameters of the Researchers in Experiment 1

These EUNs were then used as input to the knowledge-based system, which suggested the best match with a creator's EUN in each case.

5.1.4 Matching System Rule Base

The knowledge-based system was written in CLIPS v6.0. It consisted of approximately 35 rules (including initialization rules). The rules were grouped according to the attribute on which the matching was performed. The heuristics to modify the matching factor m_f (see Section 4.1.3) were chosen based on how relevant the results would be to the researcher in each case. The factors are shown in Table 5.3.

For example, if rule 28 described the match between an "intermediate" researcher and a "novice" creator, it would take the following form:

```
((researcher_knowledge_level=intermediate)
and (creator_knowledge_level=novice))
=>
f_28=0.8
```

The overall matching factor m_f for any creator-researcher pair is

		researcher		
creator	K_LEVEL	novice	intermediate	expert
	novice	1.0	0.8	0.65
	intermediate	0.85	1.0	0.8
	expert	0.6	0.9	1.0
	D_LEVEL	bachelor	master	doctor
	bachelor	1.0	0.8	0.7
	master	0.85	1.0	0.9
	doctor	0.75	0.95	1.0
		match	no_match	
	S_AREA	1.0	0.0	
	D_AREA	1.0	0.8	
	W_AREA	1.0	0.8	

Table 5.3: Match Factors for Experiment 1

$$m_f = \prod_j m_{f0} \cdot f_j,$$

where f_j is the factor in Table 5.3. In the above example, $f_j = f_{28} = 0.8$. The CLIPS source code is given in Appendix B.

Given the matching factors, it may be seen that despite differences in certain areas, reasonably good matches can still be found. For example, if the researcher's knowledge level is "intermediate" and the creator's is "expert" the match of 0.9 (90%) indicates a very good match. Nonetheless, the error accumulates if many differences exist, and the quality of the match deteriorates accordingly. Note that these heuristics were chosen as a good starting point. To know exactly what quantitative effect the difference between a master and a doctoral degree would have on the desired search results would probably be a topic of psychology research, and is beyond the scope of this work.

The output of the knowledge-based system was simply the final match factor (m_f) associated with the matches between the EUNs of each creator-researcher pair.

5.1.5 Results from the Matching System

The output from the knowledge-based system is given in in Table 5.4. Note that ALPHA was given a 0% recommendation because the SPECIFIC_AREA differed from that of each of the creators.

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	0.00	0.9	0.8	0.7	0.8
B	0.00	0.0	0.0	0.0	0.0
C	0.00	0.8	0.52	0.46	0.52
D	0.00	0.72	0.8	0.56	0.64
E	0.00	0.77	0.8	0.9	0.68
F	0.00	0.72	0.68	0.56	0.8

Table 5.4: Knowledge-Based System Output for Experiment 1

The results are shown in ranked order in Table 5.5,

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	-	1	1	2	1
B	-	-	-	-	-
C	-	2	4	4	4
D	-	4	1	3	3
E	-	3	2	1	2
F	-	4	3	3	1

Table 5.5: Knowledge-Based System Output in Ranked Order for Experiment 1

where

- 1 indicates the highest recommendation
- 4 indicates the lowest recommendation that is greater than 0%
- - indicates a 0% recommendation

For example, for researcher BETA, the knowledge-based system estimated that the results found from agent A would be most relevant, whereas those from agent D

or F would be least relevant. The next problem was to evaluate how accurate these recommendations were. In the case of BETA, for instance, it is necessary to know how close the results produced by agents A, D and F actually were to BETA's ideal set of results.

5.1.6 Actual Distance

The distances between the results of the creators and researchers agents are given in Table 5.6. The entries for "RADAR" correspond to the results of an Archie keyword search for "radar" only (these are assumed to be in random order). An entry of -1.0 indicates that the SPECIFIC-AREA parameter has different values for the creator and researcher, and the distance between the results sets was not calculated.

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	-1.00	0.00	19.36	5.99	47.04
B	-1.00	-1.00	-1.00	-1.00	-1.00
C	-1.00	0.00	19.36	5.99	47.04
D	-1.00	32.87	29.07	29.66	39.04
E	-1.00	5.99	18.46	0.00	45.46
F	-1.00	53.88	52.03	51.40	56.95
RADAR	-1.00	56.17	57.20	58.04	40.09

Table 5.6: Distances Between Results for Experiment 1

The results are shown in ranked order in Table 5.7,
where

- 1 indicates the smallest distance
- 5 indicates the highest distance
- - indicates an unknown distance (different numbers of hits)

From this table, it may be seen that agent A did indeed produce results that were closest to BETA's ideal set, whereas those of agent F were the furthest from it (aside from the results of the keyword search). The next step is therefore to conduct similar comparisons with the results obtained from the other researchers.

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	-	1	2	2	4
B	-	-	-	-	-
C	-	1	2	2	4
D	-	3	3	3	1
E	-	2	1	1	3
F	-	4	4	4	5
RADAR	-	5	5	5	2

Table 5.7: Distances Between Results in Ranked Order for Experiment 1

5.1.7 Analysis

In order to evaluate the effectiveness of the proposed system, the results from tables 5.5 and 5.7 must be compared. Table 5.8 compares the rankings of agents A to F for each researcher, by the knowledge-based system, to those of the distances of each creator's agent and the researchers' ideal sets. The last row ranks the results of a simple keyword search for "radar."

	ALPHA		BETA		GAMMA		DELTA		EPSILON	
	ES	DIST	ES	DIST	ES	DIST	ES	DIST	ES	DIST
A	-	-	1	1	1	2	2	2	1	4
B	-	-	-	-	-	-	-	-	-	-
C	-	-	2	1	4	2	4	2	4	4
D	-	-	4	3	1	3	3	3	3	1
E	-	-	3	2	2	1	1	1	2	3
F	-	-	4	4	3	4	3	4	1	5
RADAR	-	-		5		5		5		2

Table 5.8: Comparison Between Ranked Results from Knowledge-Based System and Distances for Experiment 1

The first observation is that in three out of four cases, the knowledge-based system recommended an agent that would yield better results than a simple keyword search. Although it must be noted that in these three cases, *any* agent would have returned better results than the keyword search, in the cases of BETA and DELTA, the knowledge-based system correctly predicted the creator's agent that *best* matched the researcher's needs. In case of EPSILON, any agent other than D would have returned results that were worse than those of a simple keyword search. This phe-

nomenon could be attributed to the content of the underlying data, to the behaviour of the knowledge-based system, or to the filtering mechanism of the agents.

The second observation is that the distances between agents A and C and the ideal set are identical for each one of the researchers. Since the EUNs of agents A and C differ from each other in the KNOWLEDGE_LEVEL, this parameter is either unnecessary, or the filtering on "introductory AND novice" is ineffective. A different filter would need to be evaluated in order to draw further conclusions from this observation.

From this experiment, it would seem that the EUN description of users and their search needs can be effectively used by a knowledge-based system to recommend the agents of users with similar requirements. In order to verify these preliminary results, a second experiment was conducted on a larger data set of HTML documents pertaining to a different subject area.

5.2 Description of Experiment 2

The second experiment differed from the first in the underlying dataset, and consequently in the search terms used in the EUNs. In experiment 2, the test set consisted of 995 documents pertaining to "english and literature". For this experiment, eleven more EUNs were developed, and the twelve corresponding agents were written. As with experiment 1, six of the EUNs were placed in the *creator* group, and the rest were were placed in the *researcher* group. The same knowledge-based system ranked the creators' agents to produce a list for each researcher, ordered from the best-matched creator's agent to the worst one. As for experiment 1, the results returned by the highest ranking creators' agents were compared to those of the researchers' agents and to straight keyword searches, in order to determine which method (using the recommended agent or performing a keyword search) produced better results for the researcher. By performing the same type of experiment on the larger and topically different dataset, it is then possible to evaluate whether the results of the first experiment were due solely to the underlying data, or whether it is possible for the knowledge-based system to make recommendations that will provide better search results for the researchers independent of the topic area of the search.

5.2.1 Choice of the Test Set

The scripts from experiment 1 were used to parse the output from Excite and to obtain the URLs of the top 995 documents (on "english and literature"), and to download each document to a local site. Sites that could not be accessed resulted in ninety-three zero-length files (of no interest to the user). The 995 HTML documents were then indexed using Archie [3]. As for experiment 1, the zero length files did not contribute to the index.

5.2.2 Choice of Creators and their Agents

The EUN parameters given in Table 5.9 describe the six creators in experiment 2. Once again, the EUNs of Users B-F each differ from that of User A by one parameter.

	A	B	C	D	E	F
S_AREA	shakespeare	chaucer	shakespeare	shakespeare	shakespeare	shakespeare
K_LEVEL	expert	expert	novice	expert	expert	expert
D_AREA	history	history	history	english	history	history
D_LEVEL	bachelor	bachelor	bachelor	bachelor	master	bachelor
W_AREA	renaissance	renaissance	renaissance	renaissance	renaissance	linguistic

Table 5.9: EUN Parameters of the Creators in Experiment 2

These EUNs were then used as the basis for the filtering by the creators' URAs. The filtering of Agents A-F was conducted using the same method as in experiment 1 (see Section 5.1.2).

5.2.3 Choice of Researchers and their EUNs

The following EUNs describe the search needs of the researchers (Table 5.10). Each differed from that of a creator by one parameter.

The knowledge-based system took the researchers' EUNs as input, and returned the best match with a creator's EUN for each one.

	ALPHA	BETA	GAMMA	DELTA	EPSILON
S_AREA	curriculum	shakespeare	shakespeare	shakespeare	shakespeare
K_LEVEL	expert	intermediate	expert	expert	expert
D_AREA	history	history	psychology	history	history
D_LEVEL	bachelor	bachelor	bachelor	doctor	bachelor
W_AREA	renaissance	renaissance	renaissance	renaissance	academic

Table 5.10: EUN Parameters of the Researchers in Experiment 2

5.2.4 Matching System Rule Base

The rule base for the knowledge-based system in experiment 2 was identical to that used in experiment 1. (See section 5.1.4.)

5.2.5 Results from the Matching System

The output from the knowledge-based system for experiment 2 is given in Table 5.11:

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	0.00	0.9	0.8	0.7	0.8
B	0.00	0.0	0.0	0.0	0.0
C	0.00	0.8	0.52	0.46	0.52
D	0.00	0.72	0.8	0.56	0.64
E	0.00	0.77	0.8	0.9	0.68
F	0.00	0.72	0.68	0.56	0.8

Table 5.11: Knowledge-Based System Output for Experiment 2

The results are shown in ranked order in Table 5.12,
where

- 1 indicates the highest recommendation
- 4 indicates the lowest recommendation that is greater than 0%
- - indicates a 0% recommendation

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	-	1	1	2	1
B	-	-	-	-	-
C	-	2	4	4	4
D	-	4	1	3	3
E	-	3	2	1	2
F	-	4	3	3	1

Table 5.12: Knowledge-Based System Output in Ranked Order for Experiment 2

These results are identical to those of experiment 1 for two reasons. First, for any pair of EUNs, the attributes with differing values are the same in both experiments. For instance, the difference between EUN BETA and EUN A lies in the value associated with the KNOWLEDGE_LEVEL attribute in both experiments. In addition, the same knowledge-based system is used in the two experiments, and it evaluates the match based solely on whether the EUN parameter values are the same (and the match factor is 1) or different (and the match factor is less than 1) for each pair of EUNs. Because the system has no semantic knowledge, it can not make any inferences about how close an "intermediate" is to an "expert" in the domain of "english and literature", versus the domain of "microwave and communications". It will therefore return the same results (using the same match factor) in both cases.

5.2.6 Actual Distance

Table 5.13 shows the distances between the results of the creators and researchers agents. The entries for "SHAKE" are assumed to be in random order, as they correspond to the results of an Archie keyword search for "shakespeare". If the SPECIFIC_AREA parameter has different values for the creator and researcher, an entry of -1.0 is used to indicate that the results contained different numbers of hits.
The results are shown in ranked order in Table 5.14,
where

- 1 indicates the smallest distance

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	-1.00	0.00	1013.86	302.30	935.25
B	-1.00	-1.00	-1.00	-1.00	-1.00
C	-1.00	0.00	1013.86	302.30	935.25
D	-1.00	1551.70	1540.82	1512.77	1132.57
E	-1.00	302.30	945.69	0.00	918.24
F	-1.00	1067.13	445.14	984.97	824.52
SHAKE	-1.00	1456.91	1157.70	1469.05	1326.53

Table 5.13: Distances Between Results for Experiment 2

	ALPHA	BETA	GAMMA	DELTA	EPSILON
A	-	1	3	2	3
B	-	-	-	-	-
C	-	1	3	2	3
D	-	5	5	5	4
E	-	2	2	1	2
F	-	3	1	3	1
SHAKE	-	4	4	4	5

Table 5.14: Distances Between Results in Ranked Order for Experiment 2

- 5 indicates the highest distance
- - indicates an unknown distance (different numbers of hits)

From this table, it may be seen that as in experiment 1, agent A produced results that were closest to BETA's ideal set, whereas those of agents D and F were the furthest from it (aside from the results of the keyword search). It appears that, in this case, the system's behaviour is independent of the underlying data set. Similar comparisons with the results from the other researchers are made in the following section in order to generalize this conclusion.

5.2.7 Analysis

As for the first experiment, the results from 5.12 and 5.14 must be compared. These results are given in Table 5.15, which shows the rankings of agents A to F for each researcher, by the knowledge-based system, and the ranked distances of each creator's agent and the researchers' ideal sets. The last row ranks the results of a simple keyword search for "shakespeare".

	ALPHA		BETA		GAMMA		DELTA		EPSILON	
	ES	DIST	ES	DIST	ES	DIST	ES	DIST	ES	DIST
A	-	-	1	1	1	3	2	2	1	3
B	-	-	-	-	-	-	-	-	-	-
C	-	-	2	1	4	3	4	2	4	3
D	-	-	4	5	1	5	3	5	3	4
E	-	-	3	2	2	2	1	1	2	2
F	-	-	4	3	3	1	3	3	1	1
RADAR	-	-		4		4		4		5

Table 5.15: Comparison Between Ranked Results from Knowledge-Based System and Distances for Experiment 2

The primary observation is that in three out of four cases (GAMMA being the exception), the knowledge-based system recommended an agent whose results were better suited to the researcher's need that were the results of a simple keyword search for "shakespeare". It should be noted that these results are comparable to those of experiment 1, and indicate that better search results may be obtained using the proposed methodology. However, a few more observations should be made from the data in Table 5.15.

First, in the case of GAMMA, the knowledge-based system gave two highest recommendations, one of which was worse than the keyword search. The EUNs of each of the two agents (A and D) recommended by the knowledge-based system for GAMMA differs from GAMMA's EUN only in the degree area parameter. (GAMMA's DEGREE_AREA was psychology, whereas creator A's DEGREE_AREA was history, and D's was english.) Because the same parameter differed, the knowledge-based system treated agents A and D as equally relevant. However, the results indicate that this was not the case. The anomaly is due either to the particular data set in this domain, or to the system's lack of semantic knowledge in the domain of "english and literature". To narrow the cause, it would be necessary to conduct the same

experiment in the same domain, but with a different data-set — that is, with 995 *different* documents on "english and literature".

Second, it should be noted that, as with experiment 1, the distances between the results of the researchers' agents and those of agents A and C are identical. This problem is therefore independent of the underlying data and may be attributed to term_0 of the filter (which filters on the words "introductory and novice"). The knowledge-based system, however, takes the KNOWLEDGE_LEVEL parameter into consideration in its recommendation, and better results might be obtained if a better filter were constructed to take this parameter into account when searching. It is also possible that the KNOWLEDGE_LEVEL parameter has no bearing on a user's search requirements. Different filters would have to be tested in order to understand its effect.

Third, the results from agent D are seen to be furthest from those of agents BETA, GAMMA and DELTA — further even than the randomly ordered results of a keyword search for "shakespeare". Because this phenomenon did not occur in the first experiment, it may be attributed to the domain. By using the same agents on different data in the "english and literature" domain, it would be possible to determine if the underlying data itself (i.e. the contents of these particular 995 documents) is the cause, or whether it is a lack of semantic knowledge and inferencing that is the reason for this radical difference in results based on a change in the degree area. This problem is significant since the proposed search method would be more robust if it was entirely domain independent.

Finally, there is a need for more differentiation in rankings from the knowledge-based system. In the case of GAMMA, for instance, agents A and D are both recommended, even though they produce very different results. Improved match factors, a greater number of EUN parameters on which matches are conducted, and semantic knowledge of the domain could reduce this effect.

Chapter 6

Summary and Conclusions

While there are currently many difficulties in conducting an effective search on the Internet, this work demonstrates that it is possible to develop a search paradigm that addresses the problems and improves the relevance of the results obtained. In effect, we have proposed an alternative search methodology and demonstrated its utility.

In summary, we saw that three main factors lie at the root of the existing search difficulties. First, there is no meta-data available that describes the underlying documents being searched: their textual content is the only information describing them. Second, the data is categorized by data type and access protocol rather than by content. Finally, the search mechanisms that are available (currently primarily WWW-based) provide only a restricted query description — limited to a boolean or natural language description of the topic itself. No information is obtained regarding *who* is conducting the search and *why* the information is sought. These factors imply that users can only search a single type of data at a time, using queries that are poor representations of their information needs, and which are then matched against an index of the documents derived only from raw textual content. (Even if the query included data describing the users or their reasons for the search, this information could not be matched against similar descriptions for the documents, because such meta-data does not exist.)

To address each of these fundamental problems, changes are required to the current search methodology. The proposed search paradigm addresses each problem as follows. The query has been expanded to include a user model as part of the search specification. Search activities — which may span various data-types and access protocols — are conducted based on the expanded query, and are encapsulated into software agents. Meta-data describing the agents is included in the paradigm,

allowing successful agents to be retrieved and re-used at a later date by other users with similar information needs.

The goal of the work has been to validate the paradigm by building an implementation using 5 parameters in the expanded query (the Enhanced User Need), Uniform Resource Agents to encapsulate the search activity, and a knowledge-based system to match the agents to users with similar EUNs. Two experiments were conducted in separate domains ("microwave and communications" and "english and literature"). In three out of four cases in each experiment, the knowledge-based system successfully suggested an agent that would return better results than those in random order obtained from an Archie keyword search. These results certainly indicate that the paradigm is valid, and that future work using this methodology could be used to improve the results from Internet searches. Additional experimental work using the same implementation has been suggested, including using the same domain with different underlying data (e.g. a different set of documents in the "microwave and communications" domain). Further steps may also be taken to improve the implementation, and thus to construct a useful Internet-wide search system.

Not only does the introduction of the additional components improve search results within the proposed framework, but it provides new avenues to explore, in terms of finding ways to match people and their needs to underlying information resources. The ultimate goal of this research is to improve Internet-wide information retrieval. With the proposed paradigm validated, two main areas may be targeted for future work. The first revolves around improving the agents' access to resources. Several changes would address this issue:

- The agents must be constructed to retrieve and filter information obtained from the entire pool of Internet resources, rather than restricting them to act on a limited set of documents.

- Agents should be implemented to make use of the full capabilities of URA technology, both in searching across data-types (the experiments in this work were limited to HTML documents), and in implementing a variety of filtering methods. (This is non-trivial given that HTML is oriented towards human understanding of the information, rather than intelligent machine interpretation.)

The second area of future work is to explore ways to improve the matches of people and existing information filtering agents. These would include:

- adding semantic knowledge of the domain in the knowledge-based system's knowledge base;
- expanding the EUNs to include a more complete user description [6];
- enhancing the agents' descriptions to include not only the EUNs of their creators, but also some description of their specific filtering methods;
- building an excellent WWW-based user interface — both for creating agents, and for conducting searches. (The work of Hammond [14] could be of help in this area);
- replacing the simple knowledge-based system with a full case-based reasoning system, permitting the closest agents (the cases) to be modified based on current needs.

Fortunately, the proposed architecture is flexible and modular enough that the different elements may be individually enhanced and expanded, without affecting the structure or functionality of the other components.

Moreover, because the architecture consolidates elements from a variety of fields – user modeling, intelligent agents, collaborative filtering, and case-base reasoning, it is possible to take advantage of the useful aspects of all these technologies. The experimental work presented validates this approach, and provides a demonstrated method of achieving improved results in Internet information retrieval.

Appendix A

Sample URA: Tcl Code for Agent A, Experiment 1

```
# ------------------------------------------------------------------------
#
#                              URA initialization
#
# ------------------------------------------------------------------------

#
# Initialize the URA, its search specs and searchable services.
#

# URA Creator info
# URA Name:   Agent A
#     Topic:  Specific Area:  radar
#       Knowledge Level:expert
#     Background:  Degree:      electrical, b.eng
# Work: electronics
#

# URA init.

uraInit {
  {name {Agent A out}}
  {author {Sima Newell}}
  {version 0.0a}
  {description "This URA finds items about radar for an novice EE in
   electronics"}
  {help "This is help on the search script."}
}
```

```
# Search spec. init.

foreach item {
  {
    {name            {Specific_Area}}
    {field           Specific_Area}
    {description     {Specific Area for this Search}}
    {prompt          {Specific Area}}
    {help            {Specific Area indicates the specific subject
                      matter of this search.}}
    {type            STRING}
    {subtype         {}}
    {allowed         .*}
    {numvals         1}
    {required        1}
    {response        {radar}}
    {respset         1}
  }
  {
    {name            {Knowledge_Level}}
    {field           Knowledge_Level}
    {description     {Knowledge Level for this Search}}
    {prompt          {Knowledge Level}}
    {help            {Knowledge Level indicates the level of expertise
                      in the Specific Area specified.}}
    {type            STRING}
    {subtype         {}}
    {allowed         .*}
    {numvals         1}
    {required        1}
    {response        {expert}}
    {respset         1}
  }
  {
    {name            {Degree_Name}}
    {field           Degree_Name}
    {description     {Degree_Name for this Search}}
```

```
  {prompt          {Degree Name}}
  {help            {Degree Name indicates the name of the degree held
                    by the search creator or target user of this URA}}
  {type            STRING}
  {subtype         {}}
  {allowed         .*}
  {numvals         1}
  {required        1}
  {response        {"bachelor engineering"}}
  {respset         1}
}
{
  {name            {Degree_Area}}
  {field           Degree_Area}
  {description     {Degree_Area of search creator}}
  {prompt          {Degree Area}}
  {help            {Degree Area indicates the area of study of the most
                    recent degree of the search creator or target
                    user of this URA}}
  {type            STRING}
  {subtype         {}}
  {allowed         .*}
  {numvals         1}
  {required        1}
  {response        {electrical}}
  {respset         1}
}
{
  {name            {Work_Area}}
  {field           Work_Area}
  {description     {Work_Area for this Search}}
  {prompt          {Work Area}}
  {help            {Work Area indicates the career area of the search
                    creator or target user of this URA}}
  {type            STRING}
  {subtype         {}}
  {allowed         .*}
  {numvals         1}
```

```
    {required        1}
    {response        {electronic}}
    {respset         1}
  }
} {
  uraSearchSpecInit $item
}

uraAnnotationInit {
  {help        {Enter comments to store with an instance}}
  {numvals     1}
  {subtype     {}}
  {response    {}}
  {name        Comments}
  {required    0}
  {class       ANNOTATION}
  {type        TEXT}
  {description {General comments or reminders about this URA.}}
  {respset     1}
  {prompt      "Comments or Reminders\nAbout this Search"}
  {field       {}}
  {allowed     .*}
}

uraResultInit {
  {name {Resource Handles}}
  {contents {}}
}

# Searchable services init.

# [More to come.]
foreach item {
  {
    {name     archie}
    {protocol http-post}
    {url      http://java.bunyip.com:8010/~sima/cgi-bin/archie.cgi}
  }
```

```
} {
  uraServicesInit $item
}

#    {url      http://vulnavia.bunyip.com:8888/~sima/cgi-bin/archie.cgi}

# ------------------------------------------------------------------------
#
#                              Mapping procedures
#
# ------------------------------------------------------------------------

rename uraHTTPPostSearch ""

proc uraHTTPPostSearch {url nmvals} {
  global uraConnectionTimeout

  set path ""
  if {[scan $url "http://%\[^/]/%\[^\n]" hostport path] < 1} {
    error [format "malformed URL: %s" $url]
  }

  if {[scan $hostport "%\[^:]:%d" host port] != 2} {
    set host $hostport ; set port 80
  }

  set nv {}
  foreach elt $nmvals {
    lappend nv [format "%s=%s" [uraHTTPQuote [lindex $elt 0]]
                               [uraHTTPQuote [lindex $elt 1]]]
  }
  set body [join $nv "&"]

  uraDebugPuts stderr [format "uraHTTPPostSearch: message body is '%s'." $body]

  uraOpenTCP -timeout $uraConnectionTimeout $host $port ifp ofp

  puts $ofp [format "POST /%s HTTP/1.0\r" $path]
```

```
  puts $ofp "Accept: */*\r"
  puts $ofp [format "Content-Length: %d\r" [string length $body]]
  puts $ofp [format "Content-Type: application/x-www-form-urlencoded\r"]
  puts $ofp "\r"
  puts $ofp $body ; close $ofp

  set in [gets $ifp]

  if {[string compare $in ""] == 0} {
    set resp ""
  } elseif [regexp -- "\[^ \t]+\[ \t]+200" $in] {
    while {[gets $ifp junk] > 1} {}
    set resp [read $ifp]
  } else {
    append resp $in [read $ifp]
  }

  close $ifp

  return $resp
}


#
# There is one procedure, for each searchable service, to map the search
# spec responses to a form suitable for inclusion into a search URL (or
# whatever form the particular query procedure accepts).
#

proc mapResponsesToArchie {} {
  set r {}

  # get values for each topic and background parameter
  foreach spec [uraListOfSetSpecs] {
    switch -- $spec {
      Specific_Area {
set spec_area [lindex [uraGetSpecResponse Specific_Area] 0]
      }
```

```
      Knowledge_Level {
set know_level [lindex [uraGetSpecResponse Knowledge_Level] 0]
      }
      Degree_Name {
set deg_name [lindex [uraGetSpecResponse Degree_Name] 0]
puts stderr [lindex $deg_name 0]
      }
      Degree_Area {
set deg_area [lindex [uraGetSpecResponse Degree_Area] 0]
      }
      Work_Area {
set work_area [lindex [uraGetSpecResponse Work_Area] 0]
      }
   }
 }

 # query0..4 represent different queries that increase in specificity
 # query4 represents the optimal query for this user and this topic
 set query0 "$spec_area"
 set query1 "$spec_area and $deg_area"
 set query2 "$spec_area and $work_area"
 if {[lindex $deg_name 0]=="bachelor"} {
  set query3 "$spec_area and $deg_area and $work_area"
 } else {
set query3 "$spec_area and $deg_area and $work_area and research"
 }
 set query4 ""
 if {$know_level=="expert"} {
set query4 "$query3 and not introductory and not novice"
 } elseif {$know_level=="novice"} {
set query4 "$query3 and introductory or $query3 and novice"
 }

 # this sets up the query as Archie needs to see it.
 # r is the standard set of Archie parameters
 # r0..4 append the query string to r and thus form a complete
 #       Archie boolean search specification
 set r {
```

```
            {url http://java.bunyip.com:8010/~sima/cgi-bin/archie.cgi}
            {gifurl http://java.bunyip.com:8010/~sima/results.gif}
            {oflag 1}
            {case Insensitive}
            {database {Web Index}}
            {type {Sub String}}
            {maxhits 200}
            {format {Keywords Only}}
          }
  set r0 [linsert $r [llength $r] [list query $query0]]
  set r1 [linsert $r [llength $r] [list query $query1]]
  set r2 [linsert $r [llength $r] [list query $query2]]
  set r3 [linsert $r [llength $r] [list query $query3]]

  # query4 only exists if the user is an expert in the topic.
  # it causes introductory level references to be ignored.
  if {$query4 != ""} {
   set r4 [linsert $r [llength $r] [list query $query4]]
      set response [list $r0 $r1 $r2 $r3 $r4]
  } else {
   set response [list $r0 $r1 $r2 $r3]
  }

  uraDebugPuts stderr [format "response is %s" $response]

  if {[string compare $response ""] == 0} {
     return ""
  }

  return $response

}



# ---------------------------------
proc uraArchieCanonicalize {textRes} {
  set result_value {}
```

```
# ditch the header lines returned by Archie

  regsub {.*START_RESULT=0} $textRes "START_RESULT=0" textRes
  set list [split $textRes "\n"]
  set length [llength $list]


# get each line returned by Archie one at a time
  for {set i 0} {$i < $length} {incr i} {
  set index [lindex $list $i]

#   get the next url and append it to the results
  if {[regexp {^URL=.*} $index]} {
regsub {^URL=} $index {} url
lappend r [list URL $url]
  }

#   get the title associated with the url
#   and append it to the results
#   also apend the TYPE
  if {[regexp {^N?0?_?TITLE=.*} $index]} {
  regsub {^N?0?_?TITLE=} $index {} headln
lappend r [list HEADLINE $headln]
lappend r [list TYPE "text/plain"]
  }

#  at the end of the lines associated with this result
#  append the URL, HEADLINE and TYPE attribute-values
#  to the final results list and resent the results
#  variable (r)

 if {[regexp {^END_RESULT=.*} $index]} {
   lappend result_value $r
set r {}
 }
  }
  return $result_value
```

```
}



proc uraRun {} {
  global errorInfo
  global uraDebug

  set errorInfo ""

  set uraDebug 1

  foreach serv [uraListOfServices] {
    set u [uraGetServiceURL $serv]

    switch -- $serv {
      archie {
        if [catch {

#   create a list of queries
          set query_list [mapResponsesToArchie]

#   initialize the list of urls
  set url_list {}

#   post the search and canonicalize the results
#   for each query in the list
#   start with the most constrained query and
#   end with the most general query
  for {set i [expr [llength $query_list]-1]} {$i >= 0} {incr i -1} {
          set result [uraHTTPPostSearch $u [lindex $query_list $i]]
set url_list [concat $url_list [uraArchieCanonicalize $result]]
uraDebugPuts stderr [format "Got a total of %s Hits!" [llength $url_list]]
  }

#   initialize the final list
  set ordered_list {}
  set u 0
```

```
#   ditch multiple instances of the same URL/TITLE/TYPE set
  for {set i 0} {$i < [llength $url_list]} {incr i +1} {
if {[lsearch -exact $ordered_list [lindex $url_list $i]] == -1} {
lappend ordered_list [lindex $url_list $i]
# write ordered_list to a file
set outfile [open out-a.txt a]
puts $outfile [lindex [lindex $ordered_list $u] 0]
close $outfile
incr u +1
}
  }
  uraDebugPuts stderr [format "Ordered List length...%s" [llength $ordered_list]]

#   display the list
  puts $ordered_list
        }] {puts stderr $errorInfo}
      }

      default {
        # can't handle other searches, yet.
      }
    }
  }
}
```

Appendix B

CLIPS Source Code for Knowledge-Based System

```
; -- Author: Sima Newell
; -- Version: 0.99a
; -- Date: Nov. 20, 1996

; associated files:
; agents.clp:  contains current URA meta-information

;------------------ EUN AGENT and MATCH Deftempates ----------------------
; Define EUN and AGENT templates
; AGENT descriptions represent URA meta-information
; EUN (Enhanced User Need) descriptions represent a combination of search
; description and user model information
;-------------------------------------------------------------------------


(deftemplate EUN "EUN description"
(slot  name     (type SYMBOL) (default ?DERIVE))
(multislot  search       (type SYMBOL) (default ?DERIVE))
(slot  knowledge-level (type SYMBOL) (default ?DERIVE))
(multislot  degree-area  (type SYMBOL) (default ?DERIVE))
(slot  degree-level    (type SYMBOL) (default ?DERIVE))
(slot  work-area  (type SYMBOL) (default ?DERIVE)))

(deftemplate AGENT "AGENT description"
(slot  name     (type SYMBOL) (default ?DERIVE))
(multislot  search       (type SYMBOL) (default ?DERIVE))
(slot  knowledge-level (type SYMBOL) (default ?DERIVE))
(multislot  degree-area  (type SYMBOL) (default ?DERIVE))
```

```
(slot  degree-level    (type SYMBOL) (default ?DERIVE))
(slot  work-area  (type SYMBOL) (default ?DERIVE)))

(deftemplate MATCH "associated with AGENT to keep track of matches"
(slot  ura-name     (type SYMBOL) (default ?DERIVE))
(slot  eun-name     (type SYMBOL) (default ?DERIVE))
(slot  search       (type SYMBOL) (default no))
(slot  knowledge-level (type SYMBOL) (default no))
(slot  degree-area  (type SYMBOL) (default no))
(slot  degree-level    (type SYMBOL) (default no))
(slot  work-area  (type SYMBOL) (default no))
(slot factor (type NUMBER) (default 1.0)))
; "no" means that the slot in the corresponding agent has not yet been
; compared to the EUN slot value.


;---------------------INIT-MATCH---------------------------------
; initialize the match facts between a given agent and user
;----------------------------------------------------------------

(defrule init-match

(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))
```

```
=>
(assert (MATCH (ura-name ?ura-name) (eun-name ?name))))


;---------------------SEARCH-AREA-MATCH-x-y----------------------
; Match EUN search with URA meta-data
; 'x' is search in the EUN of the user doing the search
; 'y' is search of each agent
;----------------------------------------------------------------

(defrule search-match-not-good

(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name) ;match for above agent
(eun-name ?name)    ;and above user
(search  no)    ; search not matched
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))
```

```
(test (neq $?ura-search $?search)) ;are EUN and agent searches the
   ; same?
=>
(modify ?match (search diff) (factor (* ?match-factor 0.0))))



(defrule search-match-good

(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?search)  ;are EUN and agent searches the
   ; same?
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name) ;match for above agent
(eun-name ?name)    ;and above user
(search  no)    ; search not matched
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))
```

```
=>
(modify ?match (search same) (factor (* ?match-factor 1.0))))



;---------------------KNOW-LEVEL-MATCH-x-y----------------------
; Match EUN knowledge level with URA meta-data
; 'x' is knowledge level in the EUN of the user doing the search
; 'y' is knowledge level of each agent
;---------------------------------------------------------------

(defrule know-level-match-novice-novice

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  novice)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  novice)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
```

```
(factor  ?match-factor))

=>
(modify ?match (knowledge-level same)))


(defrule know-level-match-novice-inter

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  novice)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  intermediate)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; user may learn, but may have trouble understanding
(modify ?match (knowledge-level diff) (factor (* ?match-factor 0.85))))
```

```
(defrule know-level-match-novice-expert

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  novice)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  expert)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; user wouldn't understand
(modify ?match (knowledge-level diff) (factor (* ?match-factor 0.6))))

(defrule know-level-match-inter-novice

(EUN
(name  ?eun-name)
```

```
(search  $?search)
(knowledge-level  intermediate)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  novice)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; close, but user might be bored
(modify ?match (knowledge-level diff) (factor (* ?match-factor 0.8))))

(defrule know-level-match-inter-inter

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  intermediate)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))
```

```
(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  intermediate)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
(modify ?match (knowledge-level same)))

(defrule know-level-match-inter-expert

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  intermediate)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  expert)
(degree-area  $?ura-degree-area)
```

```
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; user stands to learn -- factor=0.9
(modify ?match (knowledge-level diff) (factor (* ?match-factor 0.9))))

(defrule know-level-match-expert-novice

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  expert)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  novice)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
```

```
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; close, but user might be bored
(modify ?match (knowledge-level diff) (factor (* ?match-factor 0.65))))

(defrule know-level-match-expert-inter

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  expert)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  intermediate)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
```

```
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; close, but user might be bored
(modify ?match (knowledge-level diff) (factor (* ?match-factor 0.8))))


(defrule know-level-match-expert-expert

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  expert)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  expert)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  no)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))
```

```
=>
(modify ?match (knowledge-level same)))

;---------------------DEGREE-AREA-MATCH-x-y----------------------
; Match EUN degree-area with URA meta-data
; 'x' is degree-area in the EUN of the user doing the search
; 'y' is degree-area of each agent
;----------------------------------------------------------

(defrule degree-area-match-not-good
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  no)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

(test (neq $?degree-area $?ura-degree-area))
```

```
=>
(modify ?match (degree-area diff) (factor (* ?match-factor 0.8))))

(defrule degree-area-match-good
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  no)
(degree-level  ?match-degree-level)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
(modify ?match (degree-area same)))

;---------------------DEGREE-LEVEL-MATCH-x-y----------------------
; Match EUN degree-area with URA meta-data
; 'x' is degree-area in the EUN of the user doing the search
```

```
; 'y' is degree-area of each agent
;-------------------------------------------------------

(defrule degree-level-match-bach-bach
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  bachelor)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  bachelor)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
(modify ?match (degree-level same)))

(defrule degree-level-match-bach-mast
(EUN
(name  ?name)
(search  $?search)
```

```
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  bachelor)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  master)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; good but may be too theoretical for user
(modify ?match (degree-level diff) (factor (* ?match-factor 0.85))))


(defrule degree-level-match-bach-doct
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  bachelor)
(work-area  ?work-area))
```

```
(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  doctor)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; good but may be much too theoretical for user
(modify ?match (degree-level diff) (factor (* ?match-factor 0.75))))



(defrule degree-level-match-mast-bach
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  master)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
```

```
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  bachelor)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; good but may not be research-oriented enough
(modify ?match (degree-level diff) (factor (* ?match-factor 0.8))))



(defrule degree-level-match-mast-mast
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  master)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  master)
(work-area  ?ura-work-area))
```

```
?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; good but may be too theoretical for user
(modify ?match (degree-level same)))


(defrule degree-level-match-mast-doct
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  master)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  doctor)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
```

```
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; very good but may be a little too theoretical for user
(modify ?match (degree-level diff) (factor (* ?match-factor 0.95))))


(defrule degree-level-match-doct-bach
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  doctor)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  bachelor)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
```

```
(factor  ?match-factor))

=>
; good but may be too simple
(modify ?match (degree-level diff) (factor (* ?match-factor 0.7))))



(defrule degree-level-match-doct-mast
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  doctor)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  master)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; good but may be a little too simple
```

```
(modify ?match (degree-level diff) (factor (* ?match-factor 0.9))))



(defrule degree-level-match-doct-doct
(EUN
(name  ?name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  doctor)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  doctor)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name ?name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  no)
(work-area  ?match-work-area)
(factor  ?match-factor))

=>
; good but may be a little too simple
(modify ?match (degree-level same)))
```

```
;--------------------WORK-AREA-MATCH-x-y-----------------
; Match EUN work-area with URA meta-data
; 'x' is work-area in the EUN of the user doing the search
; 'y' is work-area of each agent
;--------------------------------------------------------

(defrule work-area-match-not-good

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?ura-work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  no)
(factor  ?match-factor))

(test (neq ?ura-work-area ?work-area))

=>
```

```
(modify ?match (work-area diff) (factor (* ?match-factor 0.8))))

(defrule work-area-match-good

(EUN
(name  ?eun-name)
(search  $?search)
(knowledge-level  ?knowledge-level)
(degree-area  $?degree-area)
(degree-level  ?degree-level)
(work-area  ?work-area))

(AGENT
(name  ?ura-name)
(search  $?ura-search)
(knowledge-level  ?ura-knowledge-level)
(degree-area  $?ura-degree-area)
(degree-level  ?ura-degree-level)
(work-area  ?work-area))

?match <-
(MATCH
(ura-name  ?ura-name)
(eun-name  ?eun-name)
(search  ?match-search)
(knowledge-level  ?match-knowledge-level)
(degree-area  ?match-degree-area)
(degree-level  ?match-degree-level)
(work-area  no)
(factor  ?match-factor))

=>
(modify ?match (work-area same)))
```

References

[1] M. Balbanovic and Y. Shoham. Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66-72, 1997.

[2] N.J. Belkin and W.B. Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29-38, 1992.

[3] Bunyip. *Archie Home Page.* Bunyip Information Systems, Inc., Montréal, Québec, 1996. URL: http://www.bunyip.com/products/archie.

[4] L. Daigle and P. Deutsch. Agents for Internet information clients. In *Proceedings of the CIKM'95 Workshop on Intelligent Information Agents*, 1995. URL: http://www.cs.umbc.edu/~cikm/iia/submitted/viewing/daigle.ps.

[5] L. Daigle, P. Deutsch, B. Heelan, C. Alpaugh, and M. Maclachlan. Uniform resource agents (uras). Internet Architecture Board RFC: 2016, October 1996. URL: http://ds.internic.net/rfc/rfc2016.txt.

[6] L. Daigle and S. Newell. Intelligent agent structures for the Internet. *Canadian AI Magazine*, (40), 1996.

[7] J. J. Daniels and E.L. Rissland. A case-based approach to intelligent information retreival. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 238-245, 1995.

[8] E.X. DeJesus. The searchable kingdom. *Byte*, 22(8):92NA1-92NA12, 1997.

[9] P. Deutsch, R. Schoultz, P. Faltstrom, and C. Weider. Uniform resource agents (uras). Internet Architecture Board RFC: 1835, October 1995. URL: http://ds.internic.net/rfc/rfc1835.txt.

[10] O. Etzioni and D. Weld. A softbot-based interface to the Internet. *Communications of the ACM*, 37(7):72-76, 1994.

[11] Inc. Firefly Network. Building intelligent relationships: The firefly tools. Internet URL, 1997. URL: http://www.firefly.net/products/FireflyTools.html.

[12] P.W. Foltz and S.T Dumais. Personalized information delivery: An analysis of information filtering methods. *Communications of the ACM*, 35(12):51-60, 1992.

[13] D. Goldber, D. Nichols, B.M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61-70, 1992.

[14] K.J. Hammond, R. Burke, and K. Schmitt. A case-based approach to knowledge navigation. In D.B. Leake, editor, *Case-Based Reasoning Experiences, Lessons & Future Directions*, pages 125-136. AAAI Press The MIT Press, Cambridge, Massachusetts, 1996.

[15] P.C. Judge. Why firefly has mad ave. buzzing (sic). *Business Week*, (October 7):100-104, 1996.

[16] H. Kautz, B. Selman, and M. Shah. Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63-65, 1997.

[17] J.L. Kolodner and D.B. Leake. A tutorial introduction to case-based reasoning. In D.B. Leake, editor, *Case-Based Reasoning Experiences, Lessons & Future Directions*, pages 31-66. AAAI Press The MIT Press, Cambridge, Massachusetts, 1996.

[18] J.A Konstan, B.N. Miller, D. Maltz, J.L. Herlocker, L.R. Gordon, and J. Riedl. Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77-87, 1997.

[19] P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):31-40, 1994.

[20] S.C. Newell. User models and filtering agents for improved Internet information retrieval. *User Modeling and User-Adapted Interaction*, 1997. in press.

[21] D.W. Oard. The state of the art in text filtering. *User Modeling and User-Adapted Interaction*, 1997.

[22] T. Oates, Prasad M.V.N., and V.R. Lesser. Cooperative information gathering: A distributed problem solving approach. Technical Report 94-66, Department of Computer Science, University of Massachusetts, Amherst, MA 01003, USA, 11 1994. URL: ftp://ftp.cs.umass.edu/pub/lesser/oates-94-66.v2.ps.

[23] J. K. Ousterhout. *Tcl and the Tk Toolkit.* Addison-Wesley Publishing Company, Don Mills, Ontario, 1994.

[24] M.J.D. Powell. *Approximation Theory and Methods.* Cambridge University Press, New York, 1994.

[25] B. Raskutti, A. Beitz, and B. Ward. A feature-based approach to recommending selections based on past preferences. *User Modeling and User-Adapted Interaction*, 1997.

[26] P. Resnick and H.R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56-58, 1997.

[27] E. Rich. Stereotypes and user modeling. In A. Kobsa and W. Wahlster, editors, *User Models in Dialog Systems*. Springer-Verlag, New York, 1989.

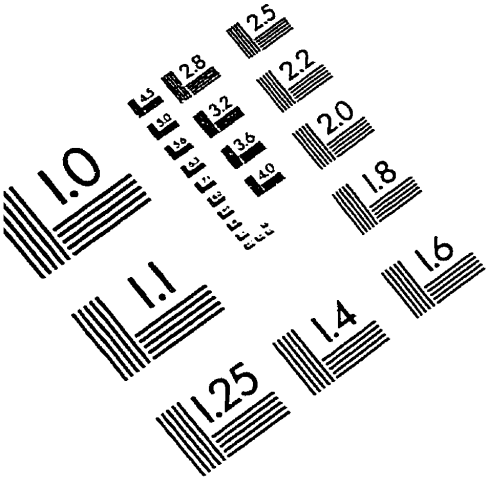
[28] J. Rucker and M.J. Polanco. Personalized navigation for the web. *Communications of the ACM*, 40(3):73-75, 1997.

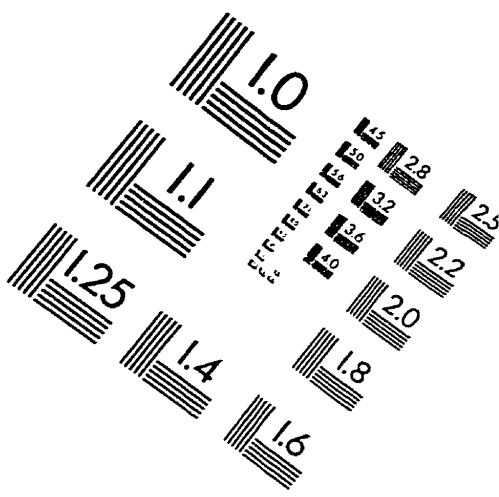
[29] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter. Phoaks: A system for sharing recommendations. *Communications of the ACM*, 40(3):59-62, 1997.

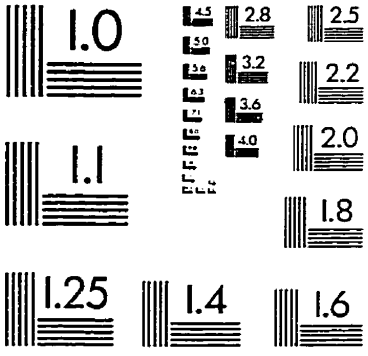
[30] G. Venditto. Search engine showdown. *Internet World*, 7(5):78–86, 1996.

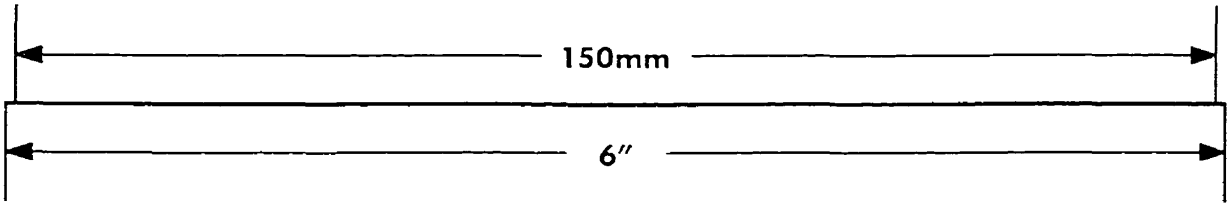
[31] E.M. Vorhees. Agent collaboration as a resource discovery technique. In *Proceedings of the CIKM'94 Workshop on Intelligent Information Agents*, 1994. URL: http://www.cs.umbc.edu/cikm/1994/iia/papers/vorhees.ps.

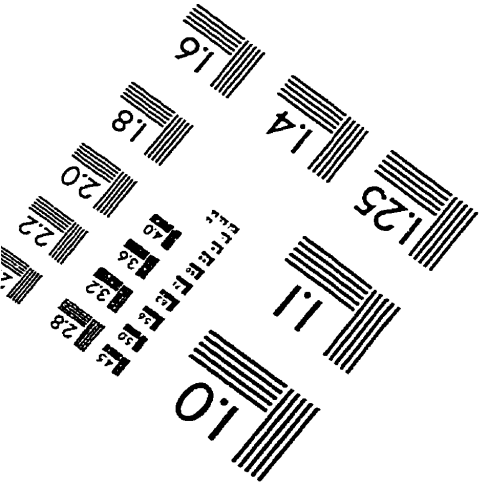
IMAGE EVALUATION
TEST TARGET (QA-3)

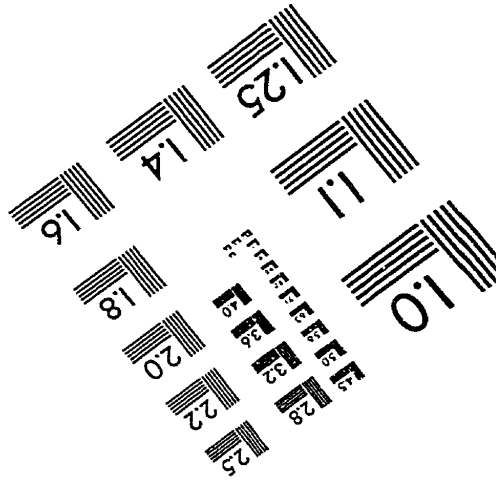












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989