Sharing Searches: Developing Open Support for Collaborative Searching

James Walkerdine & Tom Rodden

Computing Department, Lancaster University, Lancaster, UK

{walkerdi, tam}@comp.lancs.ac.uk

Abstract: This paper describes a system to encourage and manage collaborative searching, with a particular focus on query management. The model proposed in this paper attempts to raise the importance of the search, treating it as an information resource it its own right. This in turn allows the development of cooperative mechanisms that can be used to manipulate and manage the searches in a manner similar to that which exists for other information resources. This paper discusses the general approach to querying before moving on to describe the developed query management model and supporting system. Finally the on going evaluations of the system are discussed.

Keywords: Cooperative information management, querying, heterogeneous information repositories

1 Introduction

This paper presents a system to help support the activities of groups of people who regularly need to search information repositories. The objective is to provide a set of mechanisms that allow a number of users to collectively support each other in the discovery of information gained from, what are increasingly becoming, heterogeneous and highly active information repositories. Examples of these repositories include Digital Libraries, Corporate Databases or the Web itself.

Although search engines exist that provide simplified interfaces to the web (e.g. www.yahoo.com), users still find developing search terms difficult (Fitzpatrick, 1997). Many repositories still require users to construct complex queries. We wish to support searching when it is undertaken by a community of users. We do this by promoting the idea of independent 'search entities' and providing a user community with facilities that allow them to share these search entities.

Our approach is somewhat in contrast to existing considerations of resource discovery, which focus on associating information management with the discovered resources. By focusing on supporting the cooperative development and use of searches we allow a community of users to share their collective experiences in discovering information.

In this paper we describe the model and supporting system that has been developed. We concentrate on cooperative query management, which we define to encompass the construction and manipulation of queries. We present the developed system and demonstrate its use to provide access to a range of information repositories. Finally we describe an on going evaluation of the implemented system.

2 Active Information Sharing

The development of systems to support the sharing of information between a community of users has long been a core concern among CSCW researchers (Bannon, 1997). It is possible to divide these into two main categories.

- *Structuring mechanisms* that let users structure collections of information resources. Categorisation models (Dourish, 1999; Simone, 1999) and data warehousing (Chaudhuri, 1997), are two examples.

- *Recommendation techniques* that communicate to users which information could interest them (Resnick, 1994).

Although beneficial, these solutions tend to focus on the information resource. Inherent within both these approaches is an assumption that information resources are *stable*. However, the rapid growth of on-line information has seen information repositories become increasingly active. This impacts on the stability of information resources and makes management through the information resource more difficult. An information resource that was valid yesterday might not be today, web links die, information changes, and all of this has an obvious effect on both the potential accessibility and utility of information

We wish to develop management facilities that can be used in active information settings where repositories are routinely updated and where the set of repositories used are often changed. In order to provide this support we focus on supporting the cooperative development and use of searches.

We have already seen the development of search management systems such as Copernic (Copernic, 2000), and Sherlock (Sherlock, 2000). The growing importance of searching is also reflected in the development of systems such as DLITE (Cousins, 1997) and Presto (Dourish, 1999) where search expressions are used to dynamically structure information gained from a heterogeneous collection of repositories. However, these systems have tended to focus on searching as an individual activity.

2.1 Sharing Searches

Ethnographic studies of users looking for information (Twidale, 1996) suggest that users make considerable use of common searches as a means of understanding on-line catalogues. Library users would routinely support each other in amending and refining searches and would often exchange searches they have previously found useful.

Searches were used in a number of ways by users of the library and provided a significant resource for a number of cooperative activities. The list below highlights a few examples of the cooperative activities supported by sharing searches within a Library context.

- *Asking for help*
  The search was routinely used when asking for help – "*I'm trying to find about this (description of users goal). I've tried with this search (search the user has created and used) and got only 4 hits. I'm sure there should be more. Any suggestions?"*
- *Learning by Example*
  Users would often teach others how to create queries to drive their searches. This included the use of complex queries– "*Here's one I created earlier*".
- *Collaborative searches*
  Users often worked together to find information and would share a terminal to collectively search for information and needed to explain their actions to others – "*Ok this is what I have found so far, and these are the searches I have used.*"
- *Reusing searches to discover new information*
  Users would often use a search at a later date and comparing the new results with the old. Any new information would help the user focus on changes.

Despite the obvious advantages of sharing searches there is little existing support for cooperative searching (Sandusky (1998) has carried out related work and the issue has also been discussed by Karamuftuoglu (1998)). In fact, systems to aid information searching (such as digital libraries) do not consider collaborative interactions. Previous studies have concluded:

> *"unless steps are taken to preserve the social aspects of information searching then much of the (presently unrecognised and under-valued) collaboration amongst library users and staff may be lost."(Twidale, 1996) page 4*

In this paper we propose to address the current imbalance by providing similar support to the searches (categorisation, recommendation, sharing) that currently exists for the information resources. The idea is not merely to shift the focus away from the resources, but instead to promote the importance and utility of sharing and managing searches.

3 Querying

Querying is a common search technique for locating information. Traditionally, a query has been associated with an information repository where a query is processed to return a collection of results. The development of meta search facilities such as the Stanford Information Bus (Paepcke, 1996), products such as Copernic (Copernic, 2000), and Sherlock (Sherlock, 2000), and the formation of standards such as LDAP (LDAP, 1997) has reduced the coupling between queries and information repositories. The development of heterogeneous query facilities allows us to consider how a community of users may cooperatively manage queries that may be applied across a heterogeneous collection of active information repositories.

One advantage of focusing on the cooperative management of queries is that this provides a means of constructing stable points of access to dynamic information. Essentially, the searcher's aim is reflected within the query and the dynamic nature of the information repositories does not directly impact on that expressed purpose

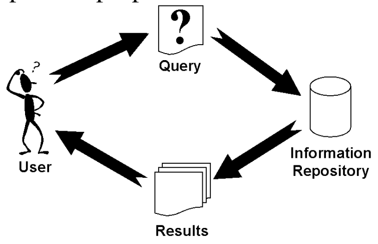


Figure 1: The Querying Process

We consider the querying process in terms of the user, the query, the information repository and the results. This process is illustrated in Figure 1.

- The *user* provides the input to the querying process by creating a query. The user receives, as output, the results.
- A *query*, is a request for specific information; an expression that is constructed from a set of query terms and operators. Query expressions vary in complexity from very simple web queries of the form *'dog + cat'* to more complex queries using query languages such as SQL "*SELECT * FROM animals WHERE type='cat' OR type='dog'* ". The success of a query is often dependent on its composition and query construction should be considered an important factor in any query management model. However, users do not wish to invest time in learning the details of query languages (Fitzpatrick, 1997).
- The *information repository* is a physical data source that the user is interrogating. Numerous types of repositories exist and a number of researchers are actively developing systems to allow queries to be passed to a heterogeneous collection of sources (Paepcke, 1996; ISO 1995).
- The *results* are the information resources produced as a consequence of processing a query.

Throughout the querying process it is not necessarily the case that a single user interacts with the query and results. As we have already indicated results are often shared amongst users and techniques have emerged to manage results in this context. We wish to allow users to be able to cooperatively use queries and many of the techniques developed in our approach build upon the experiences of previous strategies to managing information resources.

4 Cooperative Query Management

Our model for Query Management is designed to support a cooperative management environment. An important notion in the model is that of a *Generic Query*. We define a Generic Query as being a more abstract query, one that has no connections with an information repository. Such a query essentially consists of attribute name: value pairs (e.g. 'Person Name: James'). It is these Generic Queries that are managed by the system and that are converted into repository-specific queries only when they are to be sent to the repository. Three main components are central to the developed system realising the model.

Query Controller. The essential roles of the Query Controller are to administer the query objects that exist within the system, to convert generic queries into a type suitable for the connected information repositories, and to launch the subsequent queries and collect the results.

Query Management Tools. The Query Management Tools provide support for essential cooperative query management. Essentially, they provide the interface between the user and the generic queries; they deal with a query's creation and its subsequent manipulation. In order for the query management system to be easily expandable, the tools have been designed to be as independent as possible. Example tools include, a query browser, a query creator and an email query facility.

Central Server. The primary purpose of the Central Server is to provide a place where users can make any queries they have created, publicly available. When a user saves a query to the Central Server details of the repositories the user is connected to are also sent along with the query. This means that it is possible (subject to security restrictions) that when another user uses the query not only are the repositories he or she is attached to interrogated, but also the ones the original owner of the query is attached to. In addition the server also stores the details of all the users who have registered with the system. This information can then be used to build up a directory of users who use the system. Any users who have specified that they want to be ex-directory are not included in this directory.

A Local Query Management System runs at each client and carries out the processing of queries. The client also deals with all communication with respective information repositories. A network connection supports communication between the client and Central Server. This is illustrated in Figure 2.

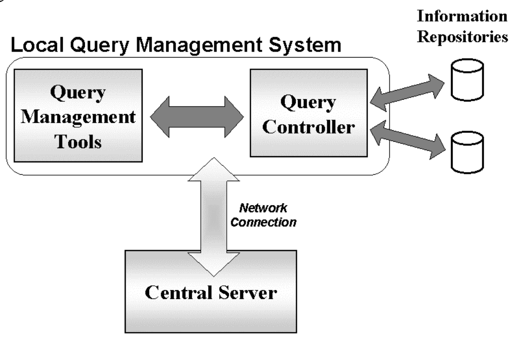


Figure 2: The General Architecture Overview

Our implementation has been carried out in Java and Swing, as these tools support the creation of comprehensive user interfaces. Communication between the Local Query Management tools and the Central Server is supported by a Remote Method Invocation (RMI) connection. Information about the queries at both the local and central level are extracted and stored within a database, and because of this it is possible to have efficient mapping between queries.

The initial version of the system focuses on web search engines (such as Yahoo and Altavista) and information repositories that possess Z39.50 gateways (ISO, 1995). As the majority of our work is to do with managing queries, we are using existing technology (i.e. Z39.50) to deal with the query mapping between different repositories.

Queries are stored automatically and so when a user wants to create a query, they can utilise past queries, either by re-launching them or extracting terms from them.

5 Providing Cooperative Support

As well as making use of some of the techniques produced from results management based research, our current implementation supports a number of cooperative features.

Query Recommendations

When a query has been stored onto the Central Server it is possible for it to receive user recommendations. Query recommendations are split into two parts. The first part, is a simple voting system with a range of 1-5 stars. The second part allows a user to write a written general comment about the query. Any user who has been granted read access to the query can provide recommendations.

When a user browses through the stored queries the average rating is shown. It is also possible to view a breakdown of all the recommendations that have been made and to which user they relate. As recommendations and users are linked then a possible future extension would be to display all recommendations provided by a specific user. This would be useful if a user finds another user with similar tastes.

Query Versions

A query's owner can specify if a query supports versioning. Versioning allow users to create and possess their own customised rendition of a query. Multiple users can make alterations to the original query without affecting it. Versions are stored inside the original query, however they are just like standard queries in that they each possess an owner and can also possess their own recommendations. At any time it is possible to convert a query to support versioning and vice versa.

Annotation of Queries

Our system supports the annotation of queries by allowing users to attach text comments to a query. Each comment is stored with details about its creator and a log is built up of all the comments that have been made. Any user who has access to the query can then view all the annotations that have been attached to the query.

Exchanging Queries

Although users can find queries by either browsing those that are stored locally or those on the Central Server, there are likely to be situations when a user wants a colleague to use or possess a specific query. Our implementation supports the exchange of queries by allowing queries to be emailed.

When an email is sent the query is included as an attachment. When the email arrives at the recipient their query management system recognises the attachment. Information about the query is then displayed to the user.

A User View of Query Categories

Users can use categories to manage the stored queries. The system provides a standard set of categories in which queries can be placed, or users can create new ones themselves. When a query is saved on to the central server the categories are publicly available. Each user can establish their own customisable views of these categories. This allows users to develop personalised views. Figure 3 illustrates the use of user views supported by the system.

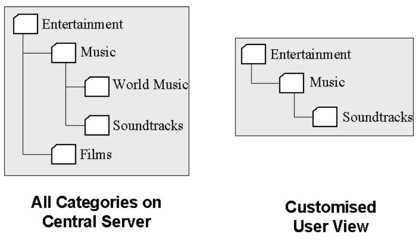


Figure 3: User view of the categories

The user has significant control over their view; a category can be added, deleted or have its name changed. Only when a user adds a category are the categories at the Central Server actually affected, for the other two operations only the view is changed.

Awareness Mechanisms and Query History

As the system is designed to support cooperative activities there will be times when the actions of others will affect users. In order to inform users of such actions the system incorporates a set of awareness mechanisms that reflect any changes in its state. Currently the system supports a query usage history and mechanisms to inform the user whenever a new query or category has been added to the central server.

A usage history is built up for each query. Whenever a query is then downloaded from the server a new entry is added to its log. The information held within this history list is viewable any time a user browses through the queries.

6 The System in Use

This section illustrates the features described above by presenting an example of the Query Management System in use. John is a user who has the Local Query Management system installed on his computer. A separate Central Server resides elsewhere.

6.1 Constructing a query

John wants to create a query to find information about Virtual Reality systems and in particular, Virtual Worlds. He clicks on the 'Create New Query' icon and is presented with a window similar to the left one shown in Figure 4. This window represents John's work area for constructing the query. The window can have keywords or 'phrases' added, deleted, moved around or negated (the user wants to find information that does *not* contain that keyword) on it. The vertical position of keywords within the query window represents their priority.

John has an active interest in Virtual Reality and has made similar searches in the past. Because of this he selects the 'Use Past Queries' option, which opens a window that will show past queries similar to the query he is constructing. John proceeds to add the keywords Virtual and Worlds to his query. As he does so, the Past Queries window is updated to show similar queries. At this point John notices that one of his previous queries contains terms that he believes are suited to the query he is trying to create. He clicks on the 'Add Terms' button and the terms from this past query are added to his query (Figure 4).

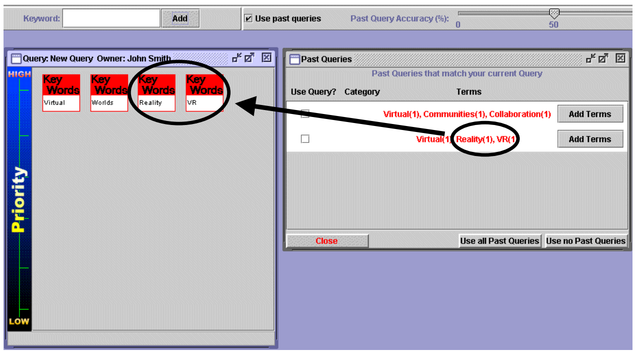


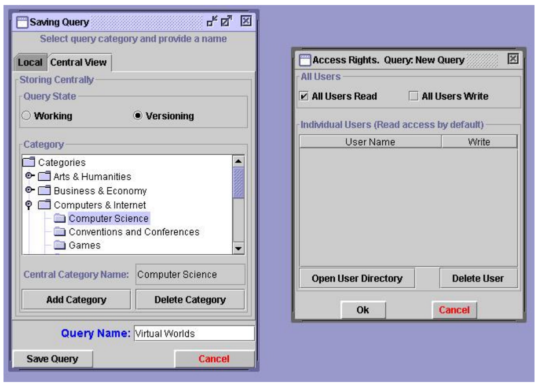
Figure 4: Creating a query and adding terms from a past query to the current query

6.2 Saving the query

John decides that he should make his new query publicly accessible and decides to save it onto the central server.

When a query is saved it maybe placed into one or more categories. If the query is to be saved on the Central Server access rights and versions also come into play. By setting the access rights it is possible to specify which users are granted access permission to the query. The owner of the query can decide whether a query can be read and/or altered by all users, or whether individual users can be granted these access rights (or a mixture of both, i.e. all users have read access, but only a few have write access). The owner of the query can also specify whether the query supports versioning or not. If versioning is enabled then every time an alteration to the query is made and saved, a new query version is generated.

John opens up the Save Query window, enters a name for the query and selects the 'Computer Science' category. He does not mind if other users want to alter his query, as long as the original is not changed, so he decides to make it a versioning query. He opens up the access rights window and ticks the 'All Users Read' and 'All Users Write' options and saves the query. Figure 5 shows the query being saved and the access rights set.


Saving Query

Select query category and provide a name

Local | Central View

Storing Centrally

Query State: Working [✓ Versioning]

Category: Categories, Arts & Humanities, Business & Economy, Computers & Internet, Computer Science, Conventions and Conferences, Games

Central Category Name: Computer Science

Add Category | Delete Category

Query Name: Virtual Worlds

Save Query | Cancel

Access Rights. Query: New Query

All Users: [✓ All Users Read] All Users Write

Individual Users (Read access by default)

User Name	Write

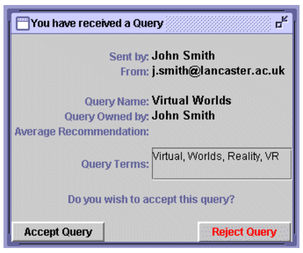
Open User Directory | Delete User

Ok | Cancel

Figure 5: Setting the access rights and saving the query on the Central Server

6.3 Sending a query to another user

John knows that a couple of colleagues are also interested in Virtual Worlds and decides to send his query to them by pressing the “Email Query” button. When John’s colleagues receive his email the Query Managers on their machines recognise an incoming query and a window pops up to inform them that a new query has arrived (Figure 6).


You have received a Query

Sent by: John Smith

From: j.smith@lancaster.ac.uk

Query Name: Virtual Worlds

Query Owned by: John Smith

Average Recommendation:

Query Terms: Virtual, Worlds, Reality, VR

Do you wish to accept this query?

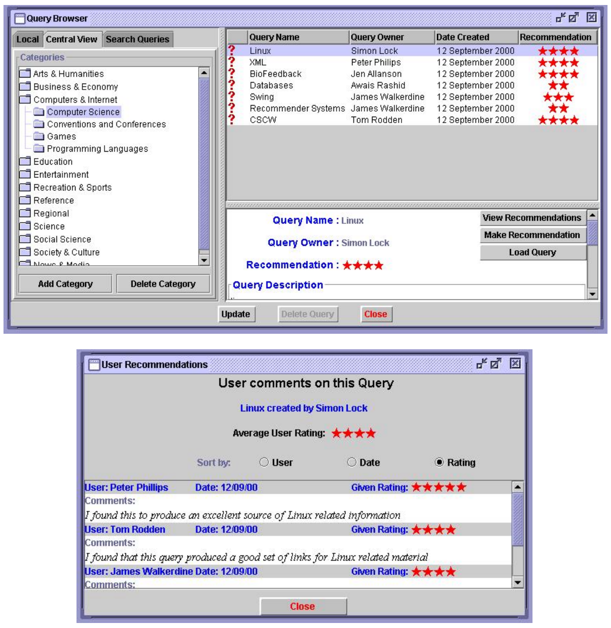
Accept Query | Reject Query

Figure 6: Receiving the query

The users can see who the sender is, who owns the query, what recommendations that query possesses, and the terms it contains. It is then up to them to decided whether or not they wish to accept it.

6.4 Browsing the Queries

Having created, saved and emailed his Virtual World query, John decides that he could do with finding some information on the Linux operating system. This is the first time he has performed such a search in this area and he is not entirely sure what he is looking for, therefore he decides to browse the queries that have been provided by others. He opens up the Query Browser and starts to browse through all the categories. In the category ‘Computing & Internet -> Computer Science’ he finds a Linux query created by Simon Lock. Looking in the preview pane, he examines the query’s terms and also notices that it has an average recommendation rating of four stars. Though John is happy with the query he is not entirely sure and takes a closer look at the recommendations to find what comments individual people have made. Figure 7 shows the query browser and the recommendations that have been made.


Query Browser

Local | Central View | Search Queries

Categories: Arts & Humanities, Business & Economy, Computers & Internet, Computer Science, Conventions and Conferences, Games, Programming Languages, Education, Entertainment, Recreation & Sports, Reference, Regional, Science, Social Science, Society & Culture, News & Media

Query Name	Query Owner	Date Created	Recommendation
Linux	Simon Lock	12 September 2000	★★★★
XML	Peter Philips	12 September 2000	★★★★
BioFeedback	Jen Allanson	12 September 2000	★★★★
Databases	Awais Rashid	12 September 2000	★★
Swing	James Walkerdine	12 September 2000	★★★
Recommender Systems	James Walkerdine	12 September 2000	★★
CSCW	Tom Rodden	12 September 2000	★★★★

Query Name : Linux

Query Owner : Simon Lock

Recommendation : ★★★★

Query Description

View Recommendations | Make Recommendation | Load Query

Add Category | Delete Category

Update | Delete Query | Close

User Recommendations

User comments on this Query

Linux created by Simon Lock

Average User Rating: ★★★★

Sort by: User Date [✓ Rating]

User: Peter Phillips Date: 12/09/00 Given Rating: ★★★★★

Comments:

I found this to produce an excellent source of Linux related information

User: Tom Rodden Date: 12/09/00 Given Rating: ★★★★

Comments:

I found that this query produced a good set of links for Linux related material

User: James Walkerdine Date: 12/09/00 Given Rating: ★★★★

Comments:

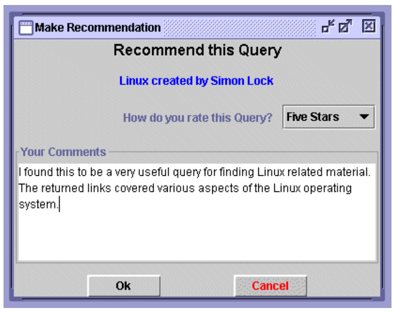
Close

Figure 7: Browsing for a query and viewing its recommendations

Satisfied by the positive feedback John loads the query and launches it.

6.5 Recommending a Query

The Linux query John used proved to be very successful and resulted in him quickly finding the information he required. John therefore decides to add his own recommendation. He opens up the recommendation window, provides a comment and gives the query a rating of five stars. Figure 8 shows the recommendation being made.


Make Recommendation

Recommend this Query

Linux created by Simon Lock

How do you rate this Query? Five Stars

Your Comments

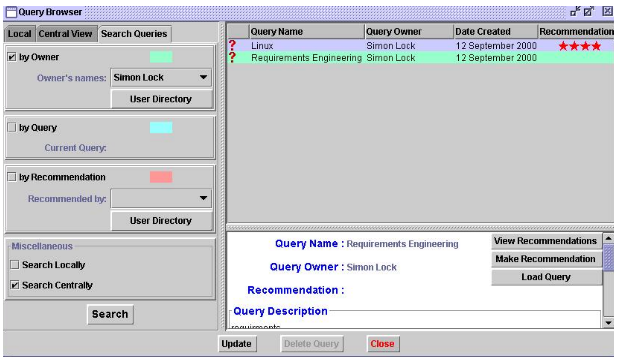
I found this to be a very useful query for finding Linux related material. The returned links covered various aspects of the Linux operating system.

Ok | Cancel

Figure 8: Recommending a Query

6.6 Searching the queries

Now that he has made his recommendation, John is curious to know what other queries Simon Lock has created. Returning to the Browser, he uses the query search facilities to sets up a search for all the stored queries created by Simon Lock. The search is performed and all the results displayed on the right. John can now browse through these queries and if desired launch them, modify them, email them to a friend, etc. Figure 9 shows how a query search is made and the returned results.



	Query Name	Query Owner	Date Created	Recommendation
?	Linux	Simon Lock	12 September 2000	★★★★
?	Requirements Engineering	Simon Lock	12 September 2000	

[✓] by Owner — Owner's names: Simon Lock
[ ] by Query — Current Query:
[ ] by Recommendation — Recommended by:
Miscellaneous: [ ] Search Locally [✓] Search Centrally

Query Name : Requirements Engineering
Query Owner : Simon Lock
Recommendation :
Query Description

Figure 9: Searching the queries

This scenario only highlights a few of the features supplied by the system. Others include, the creation of queries using natural language processing (users provide a question which the system then converts to a query) and the 'bookmarking' of queries in a web browser.

7 Evaluation

In order to assess our system we returned to the library studies that motivated the original work. Twelve Computer Science MSc students participated in the evaluation. As part of their normal coursework, groups of students were given the task of collating information related to research in the area of CSCW. The coursework represented a real task that students needed to complete for assessment. They were asked to make use of the Query Management System. Some basic queries were placed on the Central Server, as examples and starting points and a number of introductory sessions were provided.

Feedback about the system was obtained via workshops and questionnaires. Weekly workshops were arranged in which the students could highlight bugs or difficulties they were having. At the end of the evaluation period, the participants were asked to fill in a questionnaire and also to take part in a debriefing session, where as a group, they could provide verbal feedback.

Overall the participants made use of the system and exchanged queries across the community of users. Feedback was positive about the general nature of the system and offered supportive comments about its use *"useful if someone doesn't really know what they are looking for"* and some users even suggested future benefits *"a huge commercial potential. Information retrieval is such a problem"*

The feedback sessions also highlighted some specific design issues about the prototype and the approach.

Knowing other users

The participants felt that the system worked best when they were able to 'understand' another users query.

"… I need to be able to understand the query"
"I don't know from what viewpoint that query has been formed, the results may not be relevant to my approach"

This understanding would not only come from the content and structure of the query, but also from the motivation and viewpoint of the user who constructed it. The participants believed that the sharing of searches would be most beneficial in environments where users knew each other and had a common goal. However, they worried about its use as a general search facility outside this context.

Sharing the Queries

The use of a central server to store the queries received a mixed response. One user commented positively on the fact that the server would evolve over time. Other users were concerned that the collection of saved queries would achieve such a size that to successfully navigate it became a task in its own right. Users were unsure as to the extent that the use of categories would reduce this problem.

The overhead of use

A large number of the participants found the system cumbersome to use and suggested this was mainly due to its standalone nature. They were concerned about startup *"longer to load than a normal browser"* and suggested closer integration with browsers *"if it was more accessible, then yes, definitely would use such a system"*

The participants suggested that the system could be embedded into parts of the operating system, such as in a Web Browser to reduce the overhead of using the system.

Loss of amended queries and versions

During the workshops a number of users pointed out that amended queries could be lost. For example, a user would fetch the query *"information + sharing"* from the server, edit it to say *"information + sharing + virtual + reality"* and then use this query. Users tended not to resave extended queries and these were lost when the system was exited. Users made little use of the explicit versioning model developed in the system suggesting that this was heavyweight for their needs. This perhaps suggests the need for an automatic saving mechanism, or at least, prompting the user when they quit the system. It also suggests the need to consider a more implicit model of versions.

As a whole the evaluation has provided further sustenance to the notion of supporting the sharing of searches between users. The lessons learnt from this study will be used to inform the future development of the system with closer integration with the browser and a lighter weight auto save facility an immediate priority. We intend to perform further evaluations on the system, possibly using a more quantitative approach similar to the one described by Baeza-Yates (1997).

8 Conclusions

In this paper we have argued for the need to extend existing considerations of cooperative information management. In particular we have argued for the need to consider queries as information resources that can be managed in their right. We are able to do this because of the developments in managing heterogeneous information repositories that have allowed query expressions to become increasingly independent of the information repositories they are normally associated with.

A focus on the cooperative management of queries allows us to manage pools of information that are held across a heterogeneous and active collection of repositories where we can make few assumptions about the nature of the information held within these repositories.

A developed model that manages the representation and dispatching of queries supports our focus on the query as an entity to be cooperatively managed in its own right. In this paper we have presented an overall description of the model and presented the initial implementation developed to support cooperative query management.

Finally we have discussed the evaluation the developed tools have been put through and highlighted some of the initial findings.

References

Baeza-Yates, R., Pino, J.A., A first step to formally evaluate collaborative work. In *ACM GROUP'97*, Phoenix, USA, 56-60.

Bannon, L., Bodker, S., Constructing Common Information Spaces. In *Proceedings of ECSCW'97*, Lancaster, UK. Dordrecht: Kluwer.

Chaudhuri, S., Dayal, U., An Overview of Data Warehousing and OLAP Technology. *SIGMOD Record* 26(1), 1997, ACM press, 65-74.

Copernic 2000. Available via URL http://www.copernic.com

Cousins, S. B., Paepcke, A., Winograd, T., Bier, E. A., Pier, K., The Digital Library Integrated Task Environment (DLITE). *In Proceedings of DL 97*, Philadelphia, USA, 1997, ACM press, 142-151.

Dourish, P., Edwards, K., LaMarca, A., Salisbury, M., Presto: An Experimental Architecture for Fluid Interactive Document Spaces, *ACM Trans. Comput. - Human Interaction*. 6, 2. June 1999. 133-161.

Dourish, P., Lamping, J., Rodden, T., Building Bridges: Customisation and Mutual Intelligibility in Shared Category Management. *In Proceedings of SIGGROUP 99*, Phoenix, AZ, USA, 1999, ACM Press, 11-20.

Fitzpatrick, L., Dent, M., Automatic Feedback Using Past Queries: Social Searching?, In *Proc. of SIGIR'97*, Philadelphia PA, USA, 1997, ACM press, 306-313.

ISO 23950, Information Retrieval (Z39.50): Application Service Definition and Protocol Specification, 1995

Karamuftuoglu, M., Collaborative Information Retrieval: Toward a Social Informatics View of IR Interaction. In *J. of the ASIS*, 1998, 49(12), 1070-1080.

Lightweight Directory Access Protocol. Specified in RFC-1777, 1997. Available from Michigan University via URL http://www.umich.edu/~dirsvcs/ldap

Paepcke, A., Cousins, S., Garcia-Molina, H., Hassan, S., Ketchpel, S., Roscheisen, M., Winograd, T., Using distributed objects for digital library interoperability. *IEEE Computer Magazine*, 29 (5), May 1996, 61-68.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Reidl, J., GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of CSCW'94*, NC. 1994, ACM press, 175-186.

Sherlock. Available from Apple Computers via URL http://www.apple.com/sherlock

Simone, C., Mark, G., Giubbilei, D., Interoperability as a Means of Articulation Work. *In Proc. of WACC'99*, ACM Press, San Francisco. Febuary 1999, 39-48

Sandusky, R.J., Powell, K.R., Feng, A.C., Design for Collaboration in Networked Information Retrieval. *In Proceedings of ASIS* Midyear '98, Orlando, Florida, May 16-20, 1998

Twidale, M.B., Nichols, D.M., Collaborative browsing and visualisation of the search process. *Aslib Proceedings*, 1996, 48(7-8), pages 177-182