Approaches to Locating Expertise Using Corporate Knowledge

Richard Crowder,* Gareth Hughes and Wendy Hall
IAM Group, University of Southampton, Southampton, UK

ABSTRACT **In many organizations people need to locate colleagues with knowledge and information to resolve a problem. Computer-based systems that assist users with finding such expertise are increasingly important to industrial organizations. In this paper we discuss the development of Expertise Finders suitable for use within the engineering design environment, as illustrated through the use of a scenario. A key feature of this work is that the Expertise Finder returns both recommended contacts and supporting documentation. The Expertise Finder bases its results on information held within the organization, e.g. on-line publications repositories, human resource records, and not on individually compiled Curriculum Vitaes or other forms of user-maintained records. The recommendations are presented to the user with due regard to the social context, and are supported by the documents used to make the recommendation. Copyright © 2003 John Wiley & Sons, Ltd.**

INTRODUCTION

In the course of most activities, people face problems that they cannot solve alone. Their natural response is to study past experiences and re-use previously acquired knowledge, either from their own experiences or from resources within their organization. Goa *et al*. (1998) estimated that 90% of industrial design activity is based on variant design, whereas in a redesign activity 70% of the information is re-used from previous solutions (Khadilkar and Stauffer, 1996). For many problems, access to documentation through hypermedia or similar systems may give adequate solutions (Crowder *et al*., 1998). However, for many problems people need to have specific information above that given by documents alone to resolve the issue. In this paper, the term *expertise* assumes the embodiment of knowledge and skills within individuals. This definition distinguishes expertise from an expert. An individual may have different levels of expertise about different topics. Expertise can be topical or procedural and is arranged and valued within the organization. In some cases, expertise can be captured from a person and used to populate a database. This works very well when the problem is restricted to a very specific domain, e.g. robot maintenance (Auriol *et al*., 1999). However, for many problems the required expertise can only be accessed through a social network.

To solve a specific problem people want to find other people with the required expertise quickly. In many organizations, key personnel (managers, senior employees, information concierges; McDonald and Ackerman, 2000) will facilitate the contacts. Recommender systems are one approach to automate this process, by augmenting and assisting the natural expertise-locating behaviour within an organization. A

* Correspondence to: R. Crowder, IAM Group, Department of Electronics and Computer Science, University of Southampton, Southampton, UK.
E-mail: rmc@ecs.soton.ac.uk
Contract/grant sponsor: UK Engineering and Physical Sciences Research Council; Contract/grant number: GR/M83582.

recommender system that suggests people who have some expertise with a problem holds the promise to provide, in a small way, a service similar to these key personnel. Expertise recommender systems can also reduce the load on people in these roles and provide alternative recommendations when these people are unavailable.

In the recommendations provided by the Expertise Finder, trust is important; this can be achieved by showing why people were not recommended or why a document was not considered so important. A document might seem relevant based on a full text search but it is actually 20 years old; this is an important factor in some situations, but not in others. The provision of evidence for its decisions in the form of a list of documents and other data is considered a key Expertise Finder output. This approach contrasts with a number of reported systems where Web-based information is used to provide the recommendation (Bollacker, *et al.*, 1998; Becerra-Fernandez, 2000; Chandrasekaran and Joshi, 2001). Answer Garden 2 (Ackerman and McDonald, 1996) has an explicit expertise-location engine and provides computer-mediated communications mechanisms to find others with a range of expertise, though the mechanisms a not very elaborate. A different approach was taken by McDonnald and Ackerman (1998), who used software developed by employees to identify their expertise in various aspects of software development.

ENGINEERING DESIGN ENVIRONMENT

Our work with Expertise Finders has been targeted towards use within the engineering design environment. The design environment is currently undergoing rapid changes with social and technical drivers. In general, the design environment is highly distributed in nature and is characterized by a large number of information sources, which, together with the designers, forms a complex sociotechnical system. The paper by Wallace *et al.* (2001), discussed an outline for the future vision of the engineering design environment, and concluded that a range of knowledge management tools would be required to support their vision. One of the objectives of our work has been to define a future engineering design environment, with particular emphasis on the social and technical systems that will support designers in their day-to-day activities (Crowder *et al.*, 2003).

In order to determine the future requirements of the engineering design environment, we gathered a considerable amount of information through a range of techniques, including interviews with designers and their managers, allocation of function exercises (Clegg *et al.*, 2000), and the analysis of the current design practice. The results were developed into a detailed scenario which was evolved through discussions with the current design community. The scenario adopts a sociotechnical perspective in dealing with the capture, sharing and re-use of knowledge within the design context (Crowder *et al.*, 2003). As such, the ideas and practices outlined throughout the scenario promote a complementary social and technical approach to managing knowledge in the future engineering design process.

The Scenario

The scenario was not intended to be the detailed script for future work, but rather a resource for discussing the design process, for planning the research activities to realize the vision and for clarifying the interests and concerns that motivate it. In the scenario it is assumed that the technical elements of the future design environment have been embodied in an application termed KTfD (Knowledge Tools for Designers). The proposed architecture, Figure 1, shows that all the information required by the designer can be accessed through the KTfD desktop. It is recognized that populating the databases and the associated links is a key issue, but outside the scope of this paper. All objects within the KTfD database are version controlled and the system is configured so that all documents are stored locally, ensuring that they are available even if the original source is irretrievable. As shown in Figure 1, KTfD is able to access information from anywhere in the design office, including, if required, through the local wireless network. The flexibility gives a degree of pervasive computing to the user, as it permits active reconfiguration as

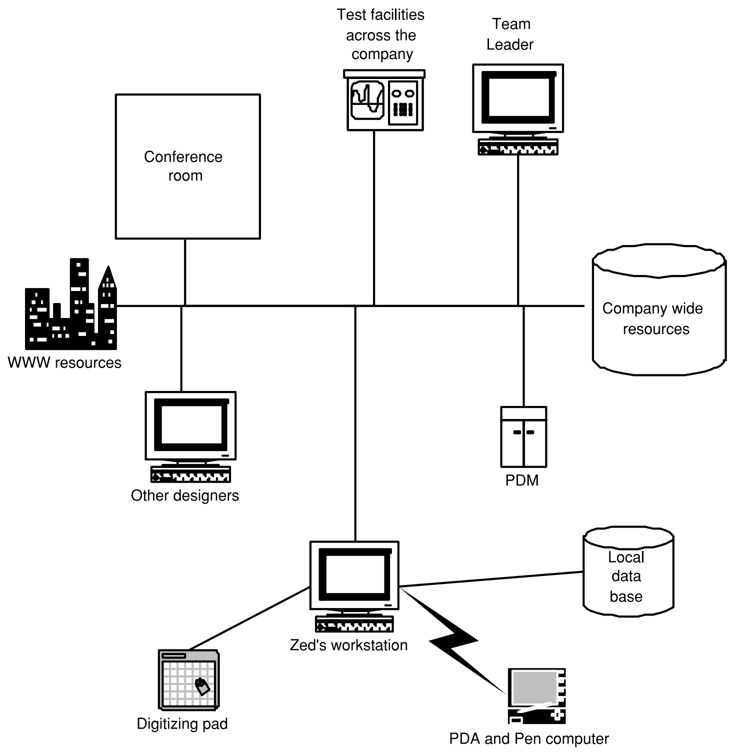


Figure 1 The KTfD concept

a function of location. It is the widespread integration of corporate systems through the KTfD environment that is a cornerstone to the operation of an Expertise Finder. It is worth noting that the KTfD is not only for knowledge management, it also has access to the full range of office and data analysis tools. While the full scenario covers all aspects of the designers' activities during the design process and the subsequent interaction with the knowledge cycle, this paper will only consider those activities that relate directly to Expertise Finders.

In the scenario developed, a design engineer has been tasked to resolve a problem relating to the bonding of components in a gas turbine. Using the KTfD desktop, the Team Leader initiates the activity by retrieving the debonding problem report from the Product Data Management (PDM) system and defining it as a new KTfD prime issue to be resolved, by opening a new design folder. As part of the definition process a new job number is issued, together with links to the background information.

Obtaining Previous Work and Background Information

From the KTfD workspace the designer is able to locate final reports relating to previous projects across the company, using a conventional search engine and the PDM system. On opening the reports and browsing the documents, the designer is able to identify the options considered and the reasons for the choices made. The designer does not immediately recognize any of the people involved, but sees that there is a short audio-visual item provided by the original designer of discussion with an adhesives expert. Selecting the expert present in the discussion, KTfD opens an information window that informs the designer that the original designer is now a manager in an adjacent area; the designer makes contact with the manager and arranges a meeting. One of the documents reveals that there was a debonding problem on a previous product. Browsing the documents reveals a primary cause, and the designer decides to contact the engineer identified against that item through KTfD, but first reviews the presentations given at various design review meetings.

The people who are contacted by the designer may be considered experts, and they are located by past information in the design files, and information provided by previous users. In many cases the information provided may be incomplete, or selective, depending on the editing history of the documents.

Searching for Colleagues with Prior Knowledge

During a review of the original design reports it quickly becomes apparent that this problem has been looked at before. The designer enters a query into the Expertise Finder through the KTfD desktop that searches for colleagues that have prior knowledge of using adhesives to bond metals within a high-temperature environment. The results reveal a number of possible contacts across the organization.

As envisaged, the Expertise Finder will allow the user to find people primarily, not documents, i.e. answering the *'who knows about . . .'* questions. The system will use design output, stored documents and personnel information to generate a technical profile of an individual. The solution will exploit linking within the information space, e.g. between documents, workflow information within PDM, and an individual's job profile within the human resources databases. The general requirement for such a system is that the architecture should be modular and flexible and allow queries from multiple viewpoints to be resolved. Our proposed systems will be discussed in the section 'Expertise Finding: Problem Definition and Context'.

Other Design Activities

It rapidly becomes clear that the problem involves the choice of adhesive and the loads to which the bond is subjected. To clarify a number of points, the designer arranges for tests to be undertaken on a number of samples. The results are downloaded directly to the workstation, allowing their rapid analysis. In the design process, a large number of elements are drawn together; hence, KTfD provides facilities to prioritize tasks. As the work progresses, the results of discussions, contact and queries are added to the record of the design process; these could include test results for a material laboratory, requests to manufacture test components and e-mails to external suppliers for support and quotations. As this information is processed and stored, KTfD will add any required metadata, allowing the information to be used subsequently by the designer, and during its retrieval by any search queries, including those from the Expertise Finder.

At the regular review meeting, the team is able to review the current situation regarding the resolution of the problem with senior staff. It is clear that a number of key sub-issues have yet to be resolved; as the meeting progresses it is clear that a slight change of emphasis is required. As the decisions are made, the designer edits the design rationale, by rejecting a number of issues, and defining a number of other issues that require more information. At the conclusion of the review meeting all the work-package modifications and queries are distributed immediately to the team. In addition, the system will identify any issue or activity that has slipped with reference to the initial time plan. The use of KTfD in the context of a meeting brings a number of significant advantages, namely all design documentation can be presented electronically at the meeting and annotated immediately as required. Feedback to the design team is instantaneous, as the meeting notes are produced on the fly and are fully cross-referenced to the design item. Experienced designers routinely attend design reviews; this information can be used by the Expertise Finder algorithm to rank people's experience in a particular subject. Part of their role is to coach less-experienced designers in good approaches to design problems. This face-to-face approach enables experience that is hard to document, or which has not been documented, to be transferred.

The final report describes the problem being considered and the details of the solution chosen, with the analysis of the problem, and the alternatives explored as linked rationale graphs. Exporting a snapshot of the active documents in the KTfD database generates the final version of the final report to be submitted for approval. The designer(s) creates the final report by linking in appropriate pieces of text and graphics contained in it to a wider design rationale graph. This consists of a network of issues, proposed answers, arguments and quantitative selection criteria, with attached e-mails, faxes, models, test result files, etc. The on-line final report and design rationale are easily browsed, with the ability to follow links in either direction between the two documents, and again can be used to locate expertise.

Feedback on the Scenario's View of Expertise Finding

In our discussions with practising designers and managers, the overall impressions of the scenario were very positive about the concepts and proposed implementation of the Expertise Finder. It was clear that there were a number of reservations regarding its implementation across a major multinational company, in particular with regard to the accuracy and freshness of the electronic data. Although this is a critical issue, it is more a reflection of the company's Knowledge Management policy than on the limitation of an Expertise Finder. It should be recognized that, within the KTfD concept, automated capture of information will resolve this problem for current, but not for legacy, designs. In any case, if the information and knowledge is 'hoarded' by an individual there is no way it can be used by an Expertise Finder, or by any other information tool.

In reviews, current designers were particularly supportive of the notion of supplying both experts and supporting documents. This ensures that the user can make informed, justified decisions about who to contact.

EXPERTISE FINDING: PROBLEM DEFINITION AND CONTEXT

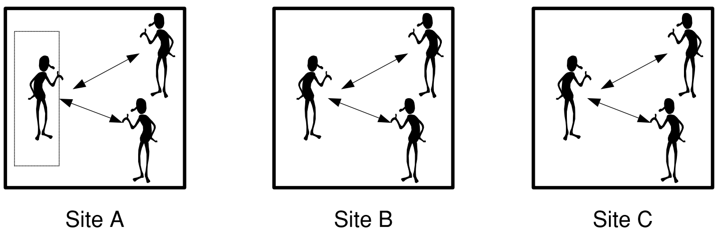
When attempting to find an answer to a problem people will tend to use the social network around them. It is natural to first ask people nearby if they know the answer or if they can recommend someone else who may know the answer. Thus, a chain of connections are made utilizing the experienced members of an organization. As people are now being moved around organizations at a faster rate and organizations are becoming increasingly distributed, this model is starting to fail. There may be no social connection between spatially separated groups even though they work on similar problems. Our approach to Expertise Finders attempts to alleviate this by using the company's own resources to recommend people to contact. It does not replace the social network, rather it attempts to speed up the connection-making process.

The work reported in this paper presents details of two Expertise Finders. The key problem that is being addressed is summarized in Figure 2(a): How does a person located in Site A locate the best expertise to solve a specific problem? The person's local network will, in all probability, only extend to within the site; therefore, expertise in other sites cannot accessed. It should be remembered that sites can share common problems, but not necessarily be easily accessible to each other. For example, within the academic community, a question on robotics could easily draw on expertise from either a Department of Electrical Engineering or a Department of Cognitive Physiology. Although the sites may not form a cohesive social network, they do share common sets of resources, including e-mail, phone books, publication and report repositories, Figure 2(b). In our approach to Expertise Finder systems, these information repositories are used to identify the required expert, Figure 2(c).

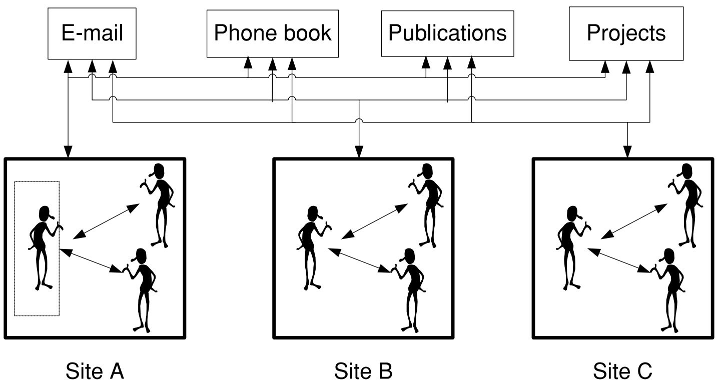
How do we Identify an Expert?

The identification of an expert can be considered to be a function of a number of social and technical factors. For example, in an academic organization an expert will be the person who has the most publications, largest number of grants, and extensive experience, either with the current or similar organization. In addition, they will tend to hold senior posts.

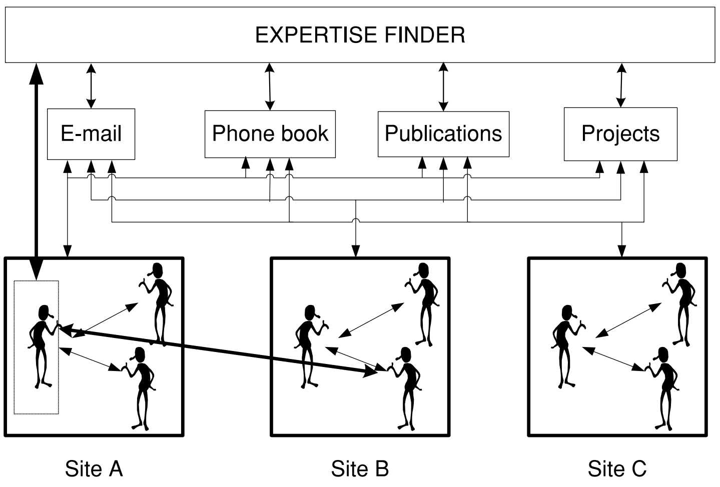
However, when a person wishes to contact an expert there are additional social factors that need to be taken into account. For example, within the academic environment, without social factors the single expert will be swamped with queries for everyone ranging from undergraduates to Vice-Chancellors. In practice, the appropriate person



(a) Local social network independent of each other



(b) Local groups are known to organisation wide electronic resources



(c) Expertise Finder finds people-to-people links regardless of location

Figure 2 Overview of the Expertise Finder

depends on the query and the user's requirements. Typically, the peer-to-peer approach is considered best in the first instance; however, the person requiring the expertise needs to be free to make a valued judgement about whom to approach. It is for this reason that we make available all the sources used for the recommendation available for review.

As discussed by McDonald and Ackerman (2000), the details matter in successful expertise location. The heuristics used to select the expert are bound to the organizational environment. Systems that augment expertise locating must be capable of handling large number of details that depend on the specific context and problem.

In practice, an Expertise Finder system is designed to mimic the reality of an organization in terms of its social structures and information infrastructure. We have developed two approaches to Expertise Finders: one is based on agent technology discussed in the next section, and the other is based on Java servlets technology (discussed in the section after next).

IMPLEMENTATION OF AN AGENT-BASED EXPERTISE FINDER

The implementation of the agent-based Expertise Finder consists of a number of Distributed Information Management (DIM) Agents operating within the Southampton Framework for Agent Research (SoFAR) (Moreau *et al.*, 2000). SoFAR was developed at the University of Southampton as an agent framework designed to address the problems of distributed information management. On each occasion that the Expertise Finder system is deployed the sources of data available to be used and their structures will be different. There will be commonalities due to the use of standards, such as being able to access a database using standard query language or the use of protocols such as LDAP. There will still be subtle differences that require the customization of the system. Therefore, it is apparent that the high-level steps that any system should take to identify an expert will be unique on each occasion.

In order to communicate with each other, agents use a shared understanding of a domain called an ontology. Ontologies are a conceptualization of a domain into a form which can be understood both by humans and computers. A well-known definition is *an ontology is an explicit specification of a conceptualisation* (Gruber, 1993). Ontologies provide a mechanism to allow communication and interaction about a real-world domain. They remove ambiguity from language through careful design. Pragmatically, they allow us to concentrate on high-level concepts rather than spend time on the implementation details, such as communications and data representation. It follows, therefore, that the design of the ontology is crucial to implementing an Expertise Finder, and careful work was required to understand and map the real-world situation correctly into the ontological vocabulary. Further technical details of how ontologies are implemented and used in the SoFAR framework can be found in Moreau *et al.* (2000). The ontologies used in design of the agent-based Expertise Finder were designed previously within the IAM Group at Southampton but extended for this application. They represent the activities and people in our research group. A detailed explanation of their design and implementation can be found in Weal *et al.* (2001).

Figure 3 shows the system architecture of the implementation of the agent-based Expertise Finder. The Expertise Finder system consists of a main agent, the *Expertise Finder Agent*, which uses a set of simpler *source agents* in some algorithm to determine a list of people and documents to recommend to a user. The Expertise Finder Agent builds an answer as XML before transforming that to HTML for delivery to the user via the Web server agent. The use of XML allows the Expertise Finder Agent to be reused in other systems and its results transformed as required. In the diagram we show all of the agents we have at our disposal, but here we concentrate on the core interactions between those outlined in solid lines.

The source agents (the *Academic Publications* and *Directory Services* agents) are designed to represent sources of information and data within the organization. These can range from the simple (an agent that understands the data stored in the internal phone book) to more complex knowledge (such as an agent interface to a publications database). In Figure 3 we include examples of some of the ontological predicates that the agents support. For instance, the Directory Services Agent can answer queries about the location of people or return all of the people with a certain phone number.

The agent-based Expertise Finder application was based on a previous agent application, the Dynamic CV (Weal *et al.*, 2001). This application used the notion of query recipes to construct an

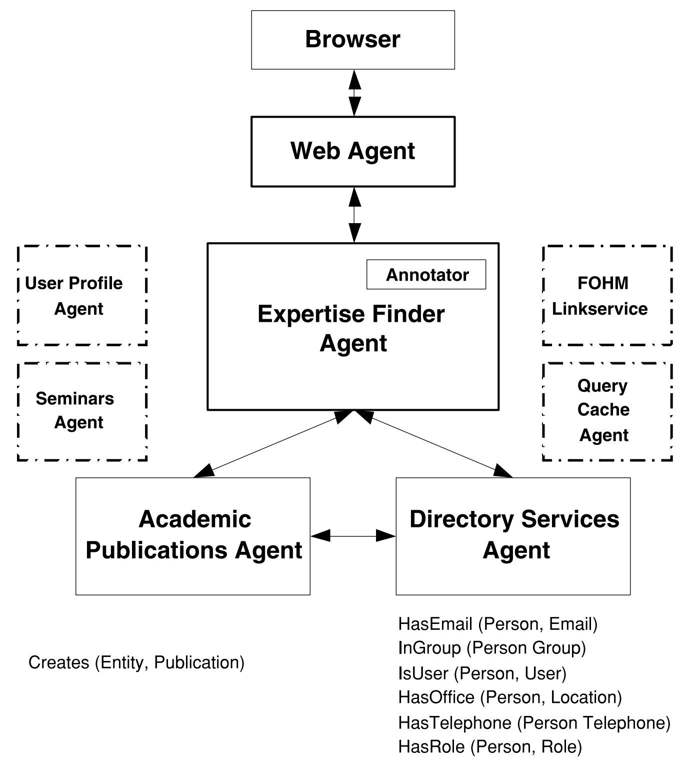


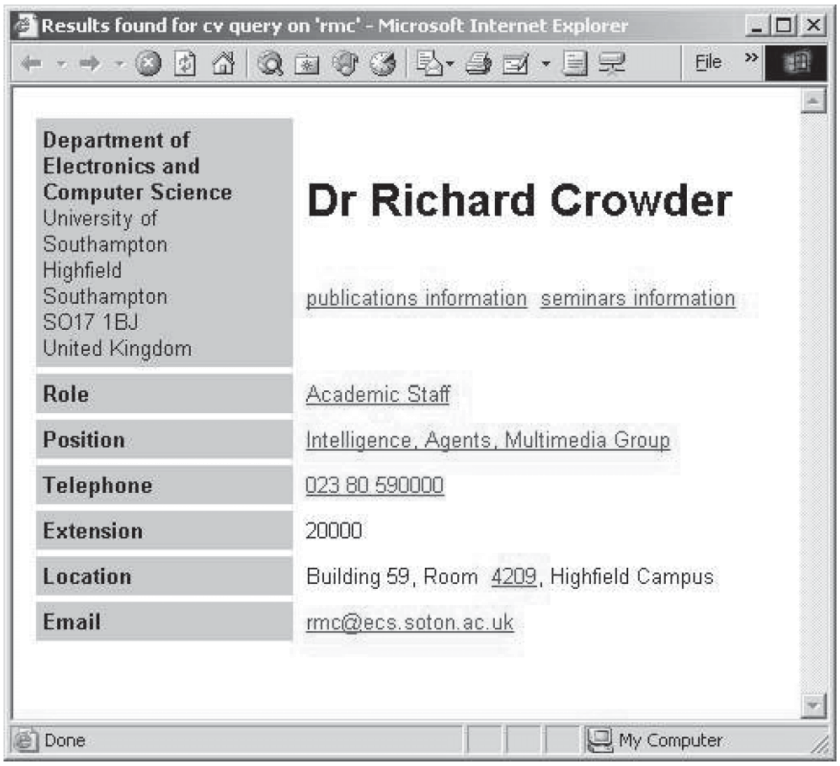
Figure 3 Expertise Finder architecture. Typical predicates used are given below the respective agents

on-line Curriculum Vitae dynamically. For instance, for the CV query, a general information page about a person, it would find and use agents to obtain telephone number, office location, and e-mail address. The answers were combined into a Web page in which links to new queries were automatically added, and thus a user could navigate around the information space. Figure 4 shows the result of a CV query.

The key weakness of the Dynamic CV application was that the main agent would gather information from source agents following the instructions of a query template. It would extract the data and place it onto the Web page with no understanding of the results. The Expertise Finder Agent is a total redesign of this, with the express intention not only of supporting the types of query performed by Dynamic CV but also if performing complex interactions with Source Agents in order to build towards a final answer. In the Expertise Finder the Source Agents have been radically improved and the services they provide have been expanded considerably.

Implementation

The current version of the agent-based Expertise Finder has been used to locate people using the scientific publication repository within the authors' Department. The goal is to aid people to find experts on a topic amongst the people in the department. A user enters a query on a research subject into a Web search page. This query is given to the Expertise Finder Agent by the Web Agent. The Expertise Finder Agent first asks the Publications Agent to find publications using the search terms. The Publications Agent takes the query terms from the predicate and uses them to form an SQL query. The query is run on the department publications database. The publication database lists authors by a list of full names and a corresponding parallel list of full e-mail


Results found for cv query on 'rmc' - Microsoft Internet Explorer
File
Done
My Computer

Department of Electronics and Computer Science University of Southampton Highfield Southampton SO17 1BJ United Kingdom	**Dr Richard Crowder** publications information seminars information
Role	Academic Staff
Position	Intelligence, Agents, Multimedia Group
Telephone	023 80 590000
Extension	20000
Location	Building 59, Room 4209, Highfield Campus
Email	rmc@ecs.soton.ac.uk

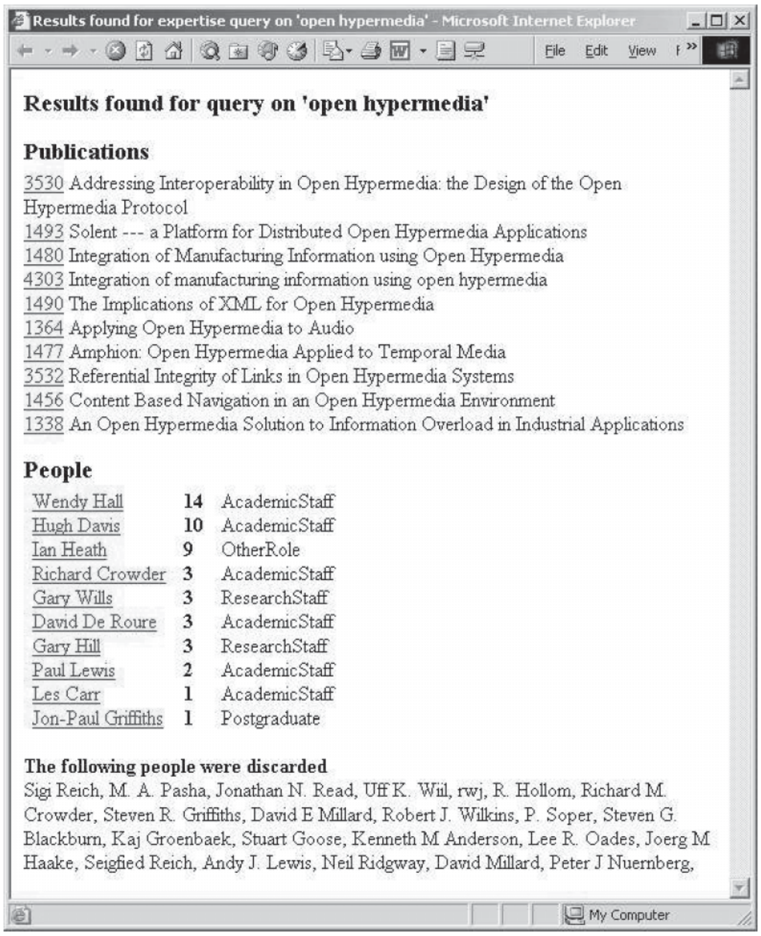
Figure 4 The Dynamic CV agent system found agents to fill in query templates. There was no attempt to understand or use the information that is returned

addresses. Hence, some understanding of this and some data translation must be performed. The Publications Agent uses the Directory Services Agent to help identify authors. It then uses the results of the query to build new Creates Predicates and return them to the Expertise Finder Agent. The Expertise Finder Agent will maintain a record of their details, saving duplication of queries, and begin to count the number of times the person appears in the returned publications. The Expertise Finder Agent will also maintain a list of people not identified, Figure 5.

The final results page is made up of the returned publications, the list of authors found with a count of their occurrences and their status within the department. The list of unknown authors is also returned to allow users to decide for themselves the usefulness of such information. In the context of this application this list consists of people who have left the department or external collaborators, and is less useful to the user.

JAVA SERVLETS-BASED EXPERTISE FINDER

The difficulties we faced in improving the original Expertise Finder system, particularly with regard to scaling, led to a new design emerging. The system was implemented using Java servlets and is interfaced via a simple search-form Web page. As in the agent-based approach, the user enters a query and the system will return a ranked list of documents plus a ranked list of people to contact. On entering a technical query the system's goal is to find documents and people pertinent to that query according to some predefined


Results found for expertise query on 'open hypermedia' - Microsoft Internet Explorer
File Edit View
My Computer

Results found for query on 'open hypermedia'

Publications

3530 Addressing Interoperability in Open Hypermedia: the Design of the Open Hypermedia Protocol
1493 Solent --- a Platform for Distributed Open Hypermedia Applications
1480 Integration of Manufacturing Information using Open Hypermedia
4303 Integration of manufacturing information using open hypermedia
1490 The Implications of XML for Open Hypermedia
1364 Applying Open Hypermedia to Audio
1477 Amphion: Open Hypermedia Applied to Temporal Media
3532 Referential Integrity of Links in Open Hypermedia Systems
1456 Content Based Navigation in an Open Hypermedia Environment
1338 An Open Hypermedia Solution to Information Overload in Industrial Applications

People

Wendy Hall	**14**	AcademicStaff
Hugh Davis	**10**	AcademicStaff
Ian Heath	**9**	OtherRole
Richard Crowder	**3**	AcademicStaff
Gary Wills	**3**	ResearchStaff
David De Roure	**3**	AcademicStaff
Gary Hill	**3**	ResearchStaff
Paul Lewis	**2**	AcademicStaff
Les Carr	**1**	AcademicStaff
Jon-Paul Griffiths	**1**	Postgraduate

The following people were discarded

Sigi Reich, M. A. Pasha, Jonathan N. Read, Uff K. Wiil, rwj, R. Hollom, Richard M. Crowder, Steven R. Griffiths, David E Millard, Robert J. Wilkins, P. Soper, Steven G. Blackburn, Kaj Groenbaek, Stuart Goose, Kenneth M Anderson, Lee R. Oades, Joerg M Haake, Seigfied Reich, Andy J. Lewis, Neil Ridgway, David Millard, Peter J Nuernberg,

Figure 5 The results of the prototype Expertise Finder, the first 10 publications used to rank the experts are given as an unranked list

strategy. Figure 6 shows the system architecture of our current implementation of the Java servlets-based Expertise Finder.

When the system is invoked by a user's submission a predefined *Expertise finding strategy* will be activated. This is a piece of code that decides which *components* to use and how to manipulate the results they return. The strategy will employ various other components in a sequence. The first types of component are termed *generators*. These perform the search and will produce lists of documents or people. The second set of components are termed *scorers*. These will take the existing results and give them new scores based on a predefined algorithm. The *strategy* keeps all of the scores that each *component* produces. The system then computes a final score for each document or person. This is done using a ratings system for each of the scores. The ratings can come from the strategy itself or from an individual's preferred ratings.

Figures 7 and 8 show the final results of a query; in the current implementation these are shown on the same Web page. The Web page shows a ranked list of documents followed by a ranked list of people. Each is given a score that is represented by a horizontal bar graph. The score is the result of a series of queries to find and score

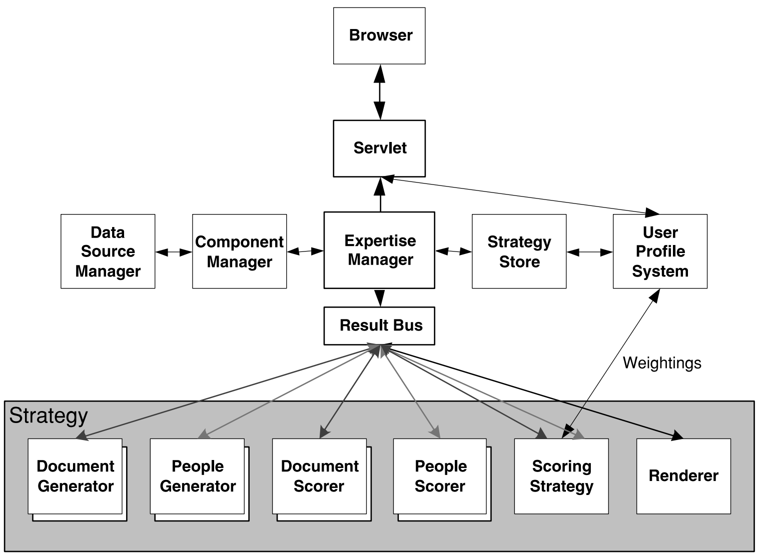


Figure 6 Java servlets-based Expertise Finder

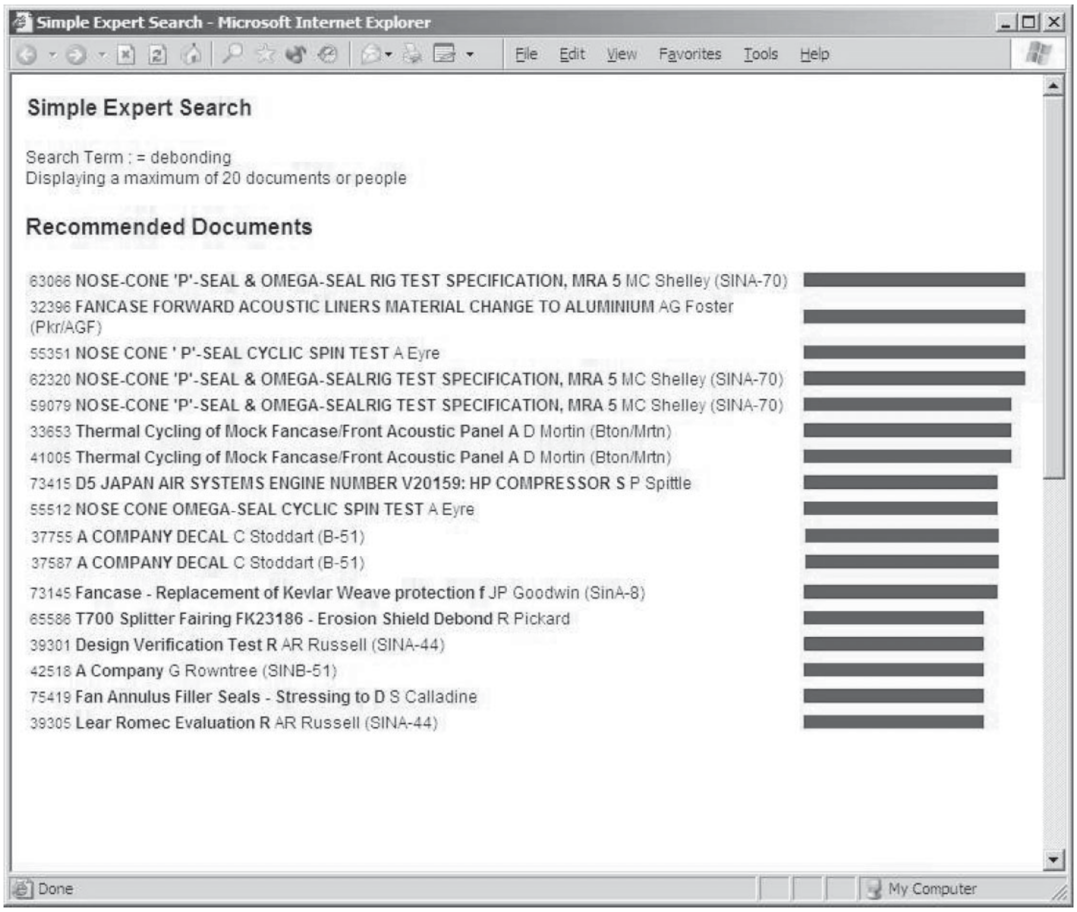
documents and people from various sources. The final score is an accumulation of all of these scores according to an interchangeable strategy that can be influenced by the user. Therefore, the final score is relative and not normalized across strategies or users.

The servlet component is responsible for generating the finished pages and providing the necessary responses to the Web browser. It passes the query to the Expertise Manager. This acts as the central hub and is responsible for starting the various components and running the requisite search *strategy*.

In the architecture the *generators* produce new results, either documents or people. They typically do this by interfacing with existing systems within an organization, such as databases or document repositories. *Generators* will produce lists of results from a query and may provide a score for each result. For instance, a *generator* based on a search engine might return a list of documents and the relevancy score given by the search engine. In this system being discussed in this paper all scores are integers between zero and ten. The final scores can be any value; this is in keeping with the nature of the system, that the final scores are only relative and not normalized. *Scorers* take results produced by *generators* and give a new score for each result depending on some other algorithm. For instance, the same list of documents found by the full text search engine could be given a score based on their age. New documents might be given a higher score than older documents. This design implies an implementation relationship between *scorers* and *generators* and that each component is not totally independent.

The Expertise Manager runs a particular strategy to solve the query. The *strategy* manages the execution of the components in a certain order and manages the manipulation of the data they return. For instance, a *scorer* might need certain information about certain documents or can only function on a subset of results. Again, the *strategy* component will be highly specific to the application and the organization within which it is run. The Expertise Finder Manager chooses which *strategy* to run at query time.

For every result found, the system keeps all the scores for that particular entry as well recording the origin of that score; hence, for each document or person located a *result* object is created. In the API, a *result* object is comprised of a *source* object plus an array of *score* objects. The *source* object refers to the document or person and the *score*


Simple Expert Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

Simple Expert Search

Search Term : = debonding
Displaying a maximum of 20 documents or people

Recommended Documents

63066 NOSE-CONE 'P'-SEAL & OMEGA-SEAL RIG TEST SPECIFICATION, MRA 5 MC Shelley (SINA-70)
32396 FANCASE FORWARD ACOUSTIC LINERS MATERIAL CHANGE TO ALUMINIUM AG Foster (Pkr/AGF)
55351 NOSE CONE ' P'-SEAL CYCLIC SPIN TEST A Eyre
62320 NOSE-CONE 'P'-SEAL & OMEGA-SEALRIG TEST SPECIFICATION, MRA 5 MC Shelley (SINA-70)
59079 NOSE-CONE 'P'-SEAL & OMEGA-SEALRIG TEST SPECIFICATION, MRA 5 MC Shelley (SINA-70)
33653 Thermal Cycling of Mock Fancase/Front Acoustic Panel A D Mortin (Bton/Mrtn)
41005 Thermal Cycling of Mock Fancase/Front Acoustic Panel A D Mortin (Bton/Mrtn)
73415 D5 JAPAN AIR SYSTEMS ENGINE NUMBER V20159: HP COMPRESSOR S P Spittle
55512 NOSE CONE OMEGA-SEAL CYCLIC SPIN TEST A Eyre
37755 A COMPANY DECAL C Stoddart (B-51)
37587 A COMPANY DECAL C Stoddart (B-51)
73145 Fancase - Replacement of Kevlar Weave protection f JP Goodwin (SinA-8)
65586 T700 Splitter Fairing FK23186 - Erosion Shield Debond R Pickard
39301 Design Verification Test R AR Russell (SINA-44)
42518 A Company G Rowntree (SINB-51)
75419 Fan Annulus Filler Seals - Stressing to D S Calladine
39305 Lear Romec Evaluation R AR Russell (SINA-44)

Figure 7 List of recommended documents; score is given by a bar graphic

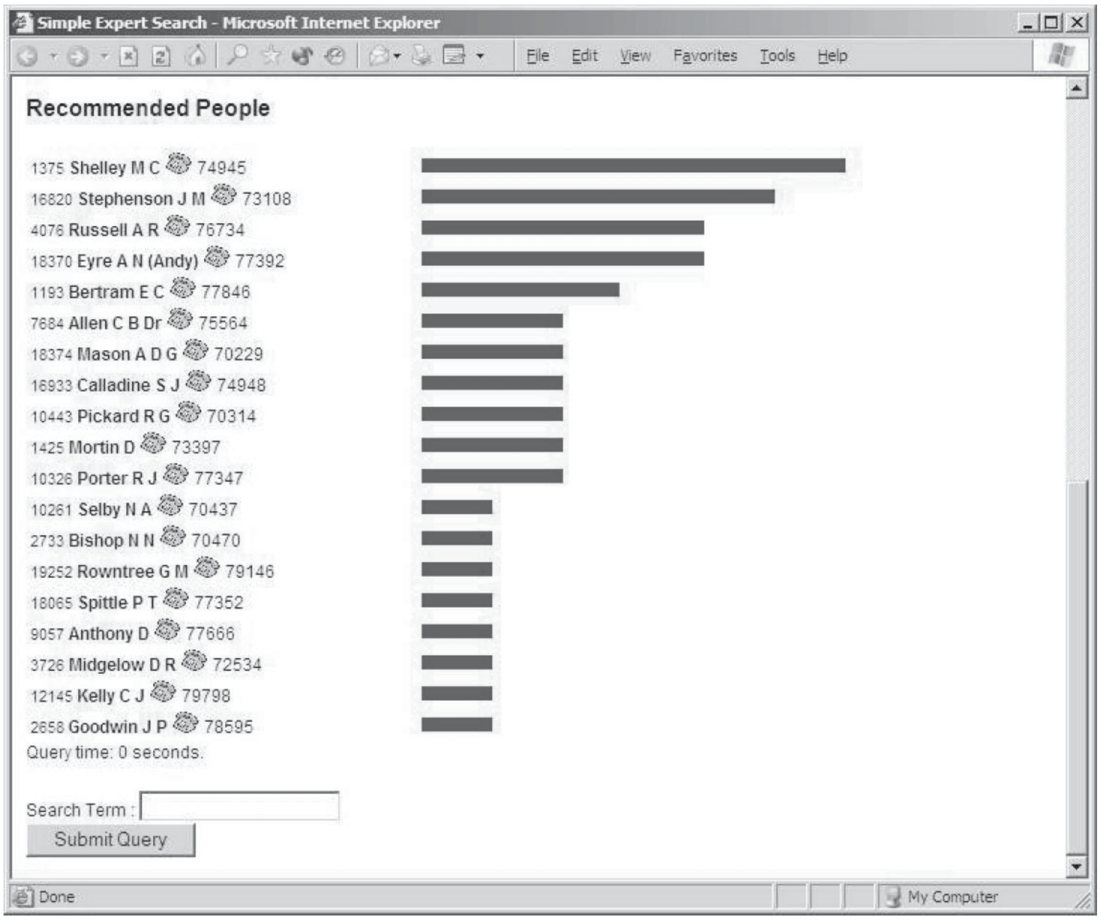
objects are the scores given by each *component* in the current *strategy*.

The final list of *results* is given to the *ScoreCompiler*, which is responsible for producing the final score. To produce this final score, a list of ratings or multiplication factors is required. The ratings is a list of values used as multiplication factors for each type of score. The *ScoreCompiler* performs the simple calculation of multiplying each score by its factor and then summing the results. It adds this to each *result* as a new *score* entry with the origin name 'Final'. The *strategy* provides the ratings values but can also obtain or alter them by using the preferences system and hence allow for individual user intervention in the final result.

The preferences system is a general purpose way of storing preferences for each user. A simple API maintains a unique ID for each user, managed using cookies, and allows the store of any key-value pair in a database against this ID. Hence any part of the system can store and retrieve arbitrary data for an individual user. This can be used to override default values for the ratings given by the *strategy*.

The finished *results* are ranked and then used to produce the final display. This service is provided by a renderer component. The current component produces HTML, but other renderers could provide other types of data, such as XML.

The data source manager controls access to underlying database connections in order to


Simple Expert Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

Recommended People

1375 Shelley M C 74945
16820 Stephenson J M 73108
4076 Russell A R 76734
18370 Eyre A N (Andy) 77392
1193 Bertram E C 77846
7684 Allen C B Dr 75564
18374 Mason A D G 70229
16933 Calladine S J 74948
10443 Pickard R G 70314
1425 Mortin D 73397
10326 Porter R J 77347
10261 Selby N A 70437
2733 Bishop N N 70470
19252 Rowntree G M 79146
18065 Spittle P T 77352
9057 Anthony D 77666
3726 Midgelow D R 72534
12145 Kelly C J 79798
2658 Goodwin J P 78595
Query time: 0 seconds.

Search Term :

Submit Query

Figure 8 List of people with contact information; score is given by a bar graphic

help cope with large-scale deployments of the system.

Implementation

The current system is built around data supplied by a large manufacturing company. The major source comes in the form of databases of internal publications. These databases list details on internal technical reports produced at individual company sites. We also have a source of phone numbers and locations for all the employees of the company; this database has over 20 000 entries.

The system we have developed comprises two document generators and a number of scorers. Each technical report database has been placed into MySQL and a generator written to perform full text searches of these resources. This has required careful design, as there are a total of 300 000 entries in the two databases. Unless care is taken, then the time to query such resources can be considerable and the number of results returned can easily overwhelm the rest of the system. We currently use the full text search capabilities of MySQL and use the relevancy scores it produces.

We use the phone book database as a people generator. Again, the raw data we received was placed into a MySQL table. For each document found by the *document generators* an attempt is made to match the author against the phone book. This entails understanding the correct fields that hold the author data, that the formatting of the fields do not match and that the author field of the document databases are free text and highly inconsistent. This is a prime example of where the *strategy* code is required to perform raw manipulation of the data in order to perform this matching process.

If the author is matched and the department they have given in the report database matches that in their phone book entry then the people generator deems to have a strong match for that person, because they have not moved department since the report was written, and returns a high score. If the person is matched but not in the same department then the score is less. If there is no match then no score is returned. This crude algorithm can be improved, but data sources are required to aid in tracing and matching people. As such resources become available, this generator could be improved or replaced without altering the rest of the system. At such a time the scoring algorithm may need rebalancing to take account of the improvement of the accuracy of this component.

We have developed a number of demonstrator scoring components. One examines the date of the documents and gives them a score based on their age. Recent documents are given higher scores. The databases contain data for approximately 20 years of work, so this is an important way to re-rank the data. Again, there have been issues with attempting to work with free text fields and trying to understand the format of the document dates.

The new Expertise Finder framework has been designed with the specific aim of allowing search strategies and ranking algorithms to be interchanged. This allows the system to be heavily customized to the particular social system found within an organization. This is achieved by breaking down the process into steps performed by interchangeable components.

During the building of this new system it was realized that there cannot be general purpose ontologies through which all components communicate. There needs to be highly application-specific code in the system to deal with the low-level interchange and manipulation of data from the various sources in the system. The strategy code is the way of dealing with this problem. The code in this component is totally specific to a certain application of the framework and manipulates raw data from sources. Therefore, it is reliant on certain sources being available to work.

It is unavoidable that the strategy code is interlinked with the available components found at a particular site. It is not possible to design a single system that can be installed at a site with no additional work. Components need to be written to match available sources and the strategy needs to be written to match the social situation.

Instead of data exchange via ontologies the new system has a simple way to represent data. The data produced by one generator or *scorer* may or may not be useful or recognizable to another. Therefore, the data are represented simply by a unique ID and the source of that ID. If another component wishes to make use of that data then it needs to obtain it from the original component. This produces yet more interdependencies in the system design, but clear APIs to each component help here. For instance, a scoring component might rank documents based on their date. Therefore, the generator that supplied those documents needs to be able to supply dates of documents via its API.

By dealing with the absolute minimum of data the system can handle relatively large numbers of results. In the previous system, large amounts of data were generated as ontologies were instantiated and exchanged. A great deal of this data was often redundant to the final requirements. The decision to provide minimum results but provide full provenance is a trade off between results that are interchangeable against results that allow for speed and low overheads. The default usage of the system is to provide only a limited number of results, for instance the top 20 documents and people; therefore, it is much faster just to extract the required data for those 40 results at the end of the run.

DISCUSSION

This paper reports the development of two Expertise Finders, the initial version based on agents and the subsequent version using a more simplistic architecture based around the need to adapt the search strategy heavily to the organization. The current servlet-based Expertise Finder produces results of limited value to designers due to problems associated with the data used. Two particular problems are worth noting. First, in identical databases supplied by two company sites, the fields were similar but not exactly the same. In addition, neither the publication nor the phonebook data could be guaranteed to be correctly maintained, nor can the entries in any fields be guaranteed to be 'correct'. For instance, people could edit their own entries in the phone book to give themselves any title or enter any name for their particular work group. Although this information was stored correctly elsewhere in other sources, it was not available for this work. These problems did demonstrate the dangers of individuals configuring information without regard to the overall knowledge management requirements of the organization.

Even with these limitations our work did demonstrate the infrastructure in use and successfully allowed the *strategy* to utilize various document *generators* and *scorers* before computing a score. We are currently working on producing more meaningful *scoring* components.

The notion of trust in the results has been a key point made by our industrial partners at every step of the design process. For instance, in our first Expertise Finder implementation the agents tried to match author names in a publications database against the live staff database. Those names that failed to match were returned to the user along with the names of those which did match. This crude example showed users exactly what was happening within the system and taught us a great number of implementation lessons.

The current system attempts to solve this problem with its transparency. Developers and users can see what scores are being given to each result and can see how these scores are brought together. All results are kept, along with their origin, for precisely this reason. It allows users to evaluate for themselves the whole process and to go into more detail on what one particular component has produced. This allows for possibilities such as saving the results of a query for caching and recalculating.

The process of finalizing the details of a particular strategy will happen in two phases. There will need to be an initial requirements capture phase in which the users, or specific users with experience, are asked to explain the sources and methods that they would use to find an expert. From this, a strategy component can be built that reflects a best attempt to turn this experience into algorithms and scoring schemes. We will then require an evaluation loop to refine this strategy.

One tool to help this process would be to be able to visualize all of the score sets for each result. For instance, a Web page output could list all of the scores for each result as well as listing the ratings given to each contributing score. This will allow users to look deeper into the results, spot anomalies and have a much greater trust in the final results.

As expected, our experiences have clearly shown that to produce an Expertise Finder to support designers within a manufacturing organization is a non-trivial undertaking. The main problem is the sheer scale and complexity of the organization we have been working with. Initially, it took a considerable amount of time before we felt we understood our partner and their actual processes. The second is the age and quality of the data we have been able to access. In this particular organization the knowledge is kept for many years, in many forms. Each area of the company uses and produces different types of information, so no single solution will be appropriate organization-wide, a major factor in the evolution of our design.

In conclusion, our work has shown that it is possible to develop Expertise Finders that rely on information currently available to organizations. With our approach, companies do not need to undertake a knowledge audit across the organization to 'frontload' the system. In practice, the viability of our approach is dependent on the ability to access all the information available in electronic format. With many organizations this can be rather spasmodic, as the IT infrastructure has evolved over time with a range of conflicting priorities. Even with these caveats we believe that the introduction of this approach is feasible, though the challenges may be more of a social than technical nature.

ACKNOWLEDGEMENTS

This work was funded in part by the University Technology Partnership for Design, which is a collaboration between Rolls-Royce, BAE SYSTEMS and the Universities of Cambridge, Sheffield and Southampton. The UK Engineering and Physical Sciences Research Council funded the work under grant number GR/M83582. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of the EPSRC or any other member of the UTP.

References

Ackerman M, McDonald D. 1996. Answer garden 2: merging organisational memory with collaberative help. In *Proceedings of ACM 1998 Conference on Computer Supported Cooperative Work, CSCW 96*. ACM: New York; 97–105.

Auriol E, Crowder RM, McKendrick RJ, Rowe R, Knudsen T. 1999. Integrating case-based reasoning and hypermedia documentation: an application for the diagnosis of a welding robot at Odense steel shipyard. In *Proceeding of the International Conference on Case-Based Reasoning (ICCBR'99)*.

Becerra-Fernandez I. 2000. The role of artificial intelligence technologies in the implementation of people-finder knowledge management systems. In *Proceedings of the 2000 American Association for Artificial Intelligence Spring Workshop*, AAAI, March.

Bollacker K, Lawrence S, Giles C. 1998. Citeseer: an autonomous Web agent for automatic retrieval and identification of interesting publications. In *Proceedings of the Second International Conference on Autonomous Agents*, Sycara K, Wooldridge M (eds). ACM: 116–123.

Chandrasekaran P, Joshi A. 2001. An expertise recommender using Web mining. In *Proceedings of AAAI 14th Annual International Florida Artificial Intelligence Research Symposium*.

Clegg C, Gray M, Waterson P. 2000. The charge of the byte brigade and a socio-technical response. *International Journal of Human–Computer Studies* **52**(2): 235–251.

Crowder R, Wills G, Heath I, Hall W. 1998. Hypermedia information management: a new paradigm. In *3rd International Conference on Managing Innovation in Manufacture*, University of Nottingham; 329–334.

Crowder R, Bracewell R, Clegg C, Hall W, Hughes G, Kerr M, Moss M, Knott D, Wallace K, Waterson P. 2003. A future vision for the engineering design environment: a future sociotechnical scenario. In *Proceedings of ICED 03*, Stockholm.

Goa Y, Zeid I, Bardez T. 1998. Charcteristics of an effective design plan to support re-use in case-based mechanical design. *Knowledge-based Systems* **10**: 337–350.

Gruber TR. 1993. Toward principles for the design of ontologies used for knowledge sharing. Technical report KSL-93-04, Knowledge Systems Laboratory, Stanford University.

Khadilkar D, Stauffer L. 1996. An experimental evaluation of design information reuse during conceptual design. *Journal of Engineering Design* **7**(4): 331–339.

McDonald D, Ackerman M. 1998. Just talk to me: a field study of expertise location. In *Proceedings of ACM 1998 Conference on Computer Supported Cooperative Work, CSCW 98*. ACM: New York; 315–324.

McDonald D, Ackerman M. 2000. Expertise recommender: a flexible recommendation system and architecture. In *ACM 2000 Conference on Computer Supported Cooperative Work*. ACM: New York; 231–240.

Moreau L, Gibbins N, DeRoure D, El-Beltagy S, Hall W, Hughes G, Joyce D, Kim S, Michaelides D, Millard D, Reich S, Tansley R, Weal M. 2000. SoFAR with DIM agents: an agent framework for distributed information management. In *Proceedings of PAAM00*, Manchester, UK.

Wallace K, Clegg C, Keane A. 2001. Visions for engineering design: a multidisciplinary perspective. In *Proceedings of ICED 01*, Glasgow.

Weal M, Hughes G, Millard D, Moreau L. 2001. Open hypermedia as a navigational interface to ontological information. In *Proceedings of Hypertext'01*, Davis H, Douglas Y, Durand D (eds). ACM: 227–236.