Proceedings of the Ninth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. Gaithersburg, MD, 67-74, 2000.

DEMOIR: A Hybrid Architecture for Expertise Modeling and Recommender Systems

Dawit Yimam
GMD – German National Research Center for Information Technology
Institute for Applied Information Technology
D-53755 Sankt Augustin, Germany
dawit.yimam@gmd.de

Alfred Kobsa
Department of Information and Computer Science
University of California, Irvine
Irvine, CA 92697-3425
kobsa@uci.edu

Abstract

Although employees' expertise has long been regarded as an important asset in organizations at least on a par with capital, goods and documented information, it is only recently that automated systems to enhance the visibility and traceability of employees with particular expertise started to receive research attention. In this paper, we introduce a systematic classification of alternative architectures for a core functionality of expert finding systems, namely expertise modeling. We also analyze the advantages and disadvantages of each approach and present our hybrid solution. We discuss how this solution addresses the requirements of (1) distributed and heterogeneous organizational and personal expertise data sources and, (2) centralized access to extracted expertise information.

Keywords: expert finders, expert recommenders, expertise modeling, expertise management, knowledge management, knowledge sharing, software architecture

1. Introduction

Expert finders (aka expert/expertise recommenders) are systems that help in tracing people with needed expertise. Such systems are gaining increasing importance as organizations continue to look for better ways to exploit their internal knowledge capital and foster collaboration among their employees for increased productivity. Basically, this trend is predicated on three fundamental premises:

(1) the tacit knowledge held by individuals (like knowledge of organization-specific processes) is at least as important as the information an organization explicitly documents in the course of its day-to-day business;
(2) the attempt to explicate all tacit knowledge held by individuals is not optimal (if it can ever be done); and
(3) making visible (and quickly traceable) the people holding this tacit knowledge in the organization motivates its optimal utilization through sharing and collaboration.

Traditionally, sources like expert databases (sometimes called "knowledge directories"), personal web pages, etc. have been used for this purpose. Recently, more automated systems that primarily work through mining implicit sources of expertise information from electronically available documents are also being developed. These systems can either stand alone or, more effectively, work as parts of, or in integration with, systems like knowledge management systems, information retrieval systems, CSCW systems, collaborative filtering systems, and electronic expertise marketplaces. A number of prototypes have been reported so far, and commercial applications that incorporate expert finding capability as one of their main features have recently come out or are under development (eg., *Knowledge Server*™ from Autonomy, Inc.[1]; *KnowledgeMail* from Tacit Knowledge Systems[2], *Organik*® from Orbital Software[3]; and *Raven* from Lotus Development Corp[4]).

However, there still remain a significant number of challenges that must be addressed to come up with more general solutions for finding experts, as we elaborated in [14]. Although our research is primarily focused on finding efficient techniques for expertise modeling, we have developed an architecture that we named DEMOIR[5] to serve as a framework for our solutions and algorithms.

[1] http://www.autonomy.com/
[2] http://www.tacit.com/
[3] http://www.orbitalsw.com/
[4] http://www.lotus.com/home.nsf/welcome/km
[5] DEMOIR stands for "Dynamic Expertise Modeling from Organizational Information Resources".

The design of this architecture is informed by our in-depth analysis of the expert finding problem and the domain analysis of expert finding systems [14]. In this paper we discuss and compare the various architectural alternatives (centralized and distributed) that we identified. In the next section, we will describe these architectures and point out their advantages and disadvantages. Then we discuss our hybrid architecture in the subsequent sections.

2. Existing alternative architectures

From our extensive review and domain analysis [14], we have learned that expert finding systems generally comprise the following seven features (or "facets"):

(1) *Expertise evidence source recognition* - which can be straightforwardly based on explicit sources (e.g. assertion by experts themselves or their proxies) or involve mechanisms to recognize implicit sources (e.g., document authorship and occurrence of name in documents).
(2) *Expertise indicator extraction* – mechanisms to extract expertise data from implicit sources.
(3) *Expertise models* – representations of extracted and aggregated expertise descriptors.
(4) *Query mechanisms* – explicit or implicit mechanisms to passively acquire or automatically trace information need or "expertise need" of users.
(5) *Matching operations* – mechanisms to match information and/or expertise need with expertise models.
(6) *Output presentation* – modes and contents of expertise information presentation.
(7) *Adaptation and learning operations* – mechanisms to refine expertise models based on user feedback as well as means to tailor expertise information to users' particular requirements.

There are multiple implementation options for each of the above features and a variety of ways of organizing them. In this paper we focus on what is accomplished by the first three features, to which together we refer as *expertise modeling*. The reader is directed to [14] for a complete discussion of the other features (which focus on mechanisms for exploitation of the expertise models). However, we would like to define what we mean by expertise model here. By *expertise model* we refer to representations of an individual's expertise repertoire that are automatically inferred. We distinguish expertise models from *expert models/profiles* by which we refer to the broad representations of an individual's personal details as well as various relations (social, organizational, etc.) that he or she might have with other individuals.

We distinguish four alternative architectural approaches to expertise modeling, which can be systematically placed in a two-dimensional grid (contingency table) as shown in table 1.

Table 1. Expertise modeling time/location contingency table

	Distributed	**Centralized**
Query-time generated	Dynamic generation by client (at *users'* sites)	Dynamic generation by server
Pre-generated	Personal agents based (at *experts'* sites)	Aggregated expertise model based

The two types of expertise modeling are arrayed along the rows, and the locations where the modeling is done are arrayed along the columns. In the following, we will briefly discuss each of these alternatives individually, pointing out the advantages and disadvantages of each.

2.1 Query-time expertise modeling

In this approach, expertise models are not maintained by the system but are rather dynamically generated at query time. This is done by building the expert finder on top of an information retrieval (IR) engine which would provide access to the organizational sources (databases, Intranets, etc.). The expertise modeling is then done dynamically on the sources returned by the IR engine (e.g. [11]). As shown in the contingency table, one can design the expert modeling to either be co-located with the IR engine on the server or, alternatively, to be done by a client at the user's computer (e.g. as a Java applet). Both options have the following advantages and disadvantages.

Advantages:

- Avoiding the storage and maintenance of large amounts of expertise information that is possibly only infrequently needed.
- Using the most recent information available to locate experts.

Disadvantages:

- High latency in query processing since all the expertise modeling operations are carried out once a user queried the system.
- Limited exploitation of information due to lack of persistent expertise models. This, for example, renders manipulations like the analysis, visualization, browsing and comparison of expertise data computationally expensive, if not impossible.
- Personal sources and non-document sources (e.g. recommendations from people, social relations, etc.) are hard to include.
- Full reliance on the availability of some search engine.

2.2 Distributed personal agents

If we adopt Chandy and Rifkin's [1] dichotomy of distributed systems as being *structured* or *anarchic*, personal agent-based expert finders can be viewed as distributed systems of the latter kind. Here, the whole system is a multi-agent system composed of self-managing, asynchronously operating "personal agents" residing in each expert's computer. These agents, which are endowed with capabilities to find and interact with each other, are the *loci* of all processing and control. Each agent carries out the dual tasks of modeling expertise (from authored documents and other personal sources) as well as assisting its owner in searching other experts. The fact that the principal entities in the expert-finding problem are experts (which themselves can be viewed as distributed knowledge sources) has made the agent-based approach a preferred solution by most works [6][13][10]. This approach has the following advantages and disadvantages.

Advantages:

- Locality and easier preservation of subjective privacy. Everyone's expertise model is locally determined and stored; the overall expertise analysis process is therefore highly decentralized.
- No single point of failure. The low degree of coupling among the agents allows graceful degradation in case of the unavailability of an agent. This decentralization also avoids the need for global system control.
- It allows the exploitation of personal information resources (e.g. emails, work-in-progress, etc.) which would not be made available to a centralized expertise modeling server due to privacy concerns.
- The expert may subjectively have the feeling of being more in control than in the case of the centralized approaches (although any access restrictions that may be imposed on an agent-based model can also be realized for centralized expertise models).

Disadvantages:

- In a strict realization, this approach has the limitation of relying solely on personal information sources of the individual, which excludes other organizational sources.
- The underlying assumption of this approach, namely that expert finders are meant only for supporting networking/sharing among experts, limits the accessibility and usability of the expertise information. Moreover, the fact that the expertise information is distributed results in its sub-optimal exploitation due to the difficulty of working on it in the aggregate. For example, it renders working on expertise data of multiple experts (like in analysis or visualization) inefficient as this would involve pulling expertise models from each agent.
- Coordination overhead may occur. In particular, getting agents to find and interact with one another presents enormous challenges. This situation is aggravated by the fact that agents might appear, disappear, or move at any time.
- In a knowledge-based implementation, this approach requires each agent to maintain its own copy of the central domain model (like in [13]) as well as a matching engine in addition to the expertise models.
- This approach also entails scalability problems as the number of experts grows larger since each query is forwarded to each expert's distributed personal agent – resulting in low query performance. (Foner [3][4] suggests clustering agents to alleviate this problem by limiting each search to a few potential clusters of agents only – an approach that is restrictive and hard to carry out in many cases).

2.3 Aggregated expertise modeling

In this approach, expertise models of individuals are either dynamically aggregated to create a central store of expertise models, or linked to a pre-constructed central expertise model which can be a kind of knowledge structure (ontology), organizational structure, etc. (rudimentary versions of this approach are reported in [9] [7]). While this approach fails to provide the locality, easier security, graceful degradation and related advantages of the personal-agent based approach, it however has the following compelling advantages:

- It makes the expertise models widely accessible and easily amenable for efficient multi-purpose exploitation and analysis. For instance, aggregated expertise models make it easy to present each expert as an element within a broad and structured expertise framework which, in turn, can be used for efficient identification and selection of experts through browsing. It also permits interactive refining of queries through browsing, since the expertise models form a closed data set that can be visualized.
- It opens the opportunity to monitor a wide range of expertise data feeds (repositories, databases, Web/Internet resources, etc.)
- It offers better performance due to a single location of information.
- It facilitates using both knowledge-based and statistical expertise modeling. For instance, it allows for the analysis and grouping of experts based on their expertise models, and it facilitates inferences on the basis of relationships among experts and their expertise.

3. Hybrid architecture in DEMOIR

3.1 Requirements

DEMOIR is a framework to develop and test expertise modeling algorithms. Based on the above analysis, we

have outlined the following sets of requirements that need to be met by DEMOIR.

1) Distributed monitoring of expertise data sources

We have two broad categories of expertise data sources in organizations: (1) organizational information resources, and (2) experts and their personal resources. Both are essentially *distributed* and *heterogeneous*. Hence, monitoring them requires handling these properties. Here we focus on the distributedness of the sources and we will come back to the heterogeneity aspect later.

So as to be able to exploit both of the above sources, we suggest the use of configurable gatherers for organizational sources while providing expert controllable gatherers (in the form of lightweight agents) for personal sources. Note that the first part of the expertise modeling process is identifying and selecting those sources that can provide expertise evidence from the large mass of sources available. And designing the gatherers in the above fashion decentralizes this particular operation.

The provision of expert controllable gatherers would allow experts to control how their personal sources are mined and let them approve (or specify) what is extracted from their resources. This addresses the privacy issues that arise in monitoring personal resources belonging to experts. Expert finders must find a way to achieve their goal, namely enhancing visibility and collaboration, without compromising the privacy and security of experts' information resources. This is particularly important since many valuable sources of expertise indicators (like emails) are commonly private. As it turns out, privacy and security considerations rank high as factors necessitating a decentralized approach to monitoring private sources, unlike the case of conventional distributed systems where reliability and performance requirements are the prime concerns. In addition to controlling which private sources are used for expertise modeling, experts may also require the expertise data extraction from some of their resources to be done on their own machines. Hence, the architecture should be flexible enough to meet such requirements, too.

2) Centralized expertise information server

This relates to providing a centralized server that aggregates, stores and retrieves the extracted expertise information. In a sense, we can see this as motivated by the need to extend the purpose of expert finders beyond the one advocated in the distributed agents-based approach (i.e. supporting networking among experts) by also incorporating the concept of users (people, organizational functions, etc.). Having the expertise information on a centralized server would make the expertise information readily accessible and useable by these entities. To allow this, the server should provide a generic low-level application programming interface (API). Moreover, this server should provide a "second line of defense" to protect privacy and security of expertise information (i.e. in addition to enabling experts to control how expertise data is extracted from their personal sources) by providing a mechanism to assign access privileges to others. For example, it should allow organizational role-based access control [12].

3) Potentially distributed clients

The approach should delegate application specific exploitation of the expertise information (e.g. searching, browsing, analysis and visualization) to clients. Hence, the system should allow a broad range of possibly distributed clients to interact with the centralized expertise information server. It should also allow integration of expert finding capabilities with other services in the organization (e.g. knowledge management, information retrieval, groupware, recommender systems, collaborative information filtering systems, expertise markets, etc.).

None of the alternative approaches discussed in the previous section can address these three requirements alone. Rather, these requirements suggest a hybrid architecture composed of centralized and decentralized processes. DEMOIR is organized in such a manner to satisfy the above three requirements. Specifically, it is a hybrid of two of the above four alternatives, namely distributed and centralized expertise modeling. It offers a centralized expertise modeling server while also permitting decentralized expertise indicator source gathering, expertise indicator extraction, and distributed clients.

Apart from the above architectural requirements, DEMOIR is also motivated by a fundamental approach to expertise modeling where the heterogeneity of sources is not just a situation to be normalized (homogenized), but rather a source of valuable meta-information that must be exploited. For instance, we want to explicitly capture how the sources relate to experts and where the expertise evidence stems from, and factor this information into the expertise modeling process. For instance, knowing the relation of sources to experts can allow us to weigh the importance of data extracted from each source based on how it relates to the expert. As an example, the occurrence of a phrase in an expert's resume implies a stronger relation to her expertise than its occurrence in one of her articles.

Based on this observation, we can introduce the concept of *source type* to denote the type of relation a given source (e.g. document, database, individual, etc.) has with an expert (or groups of experts). The *format* of a source refers to its syntactic form (e.g. web page, relational database record, etc.). Note that sources of different formats may fall under the same source type and vice versa. Once we identified source types, we can further suggest carrying out differentiated extraction of expertise evidence from each source type. Notice that this approach allows the inclusion of people as distinct source types. This would be valuable in cases that require human input

to augment expertise that had been determined by mining documentary sources.

This emphasis on dealing with heterogeneity of expertise indicator sources implies a need to support diverse expertise modeling algorithms. This, in turn, motivates a design goal of developing a flexible modular software architecture. DEMOIR aims to achieve this goal and satisfy the above requirements by partitioning the system into composable modules and defining appropriate interfaces for each module.

3.2 DEMOIR components

We designed the DEMOIR architecture, shown in Figure 1, as a flexible architecture composed of modules for source gathering, expertise modeling and expertise model exploitation. The DEMOIR modules can be implemented separately and readily combined to suit an application environment. These modules, while conceived to meet the aforementioned requirements, also draw from the features we isolated in the domain analysis and listed in section 2. What is more, they are aimed at allowing multiple instantiations of these features and an easy extension of their functionality. Defining appropriate interfaces, these modules can also be distributed across computers if necessary. Below, we briefly describe the components of DEMOIR, grouping them into three general functions.

Expertise indicator source gathering. The gatherers are essentially robots and personal agents which have expertise data source recognition logic built into them. They are envisaged to be independent, adaptable to local needs and constraints of the expertise indicator sources and are not controlled by any central system. At the same time, these agents should have the common goal of acting as "informers" to the centralized server wherein the expertise data is aggregated and made available for easy exploitation and analysis by users and client systems.

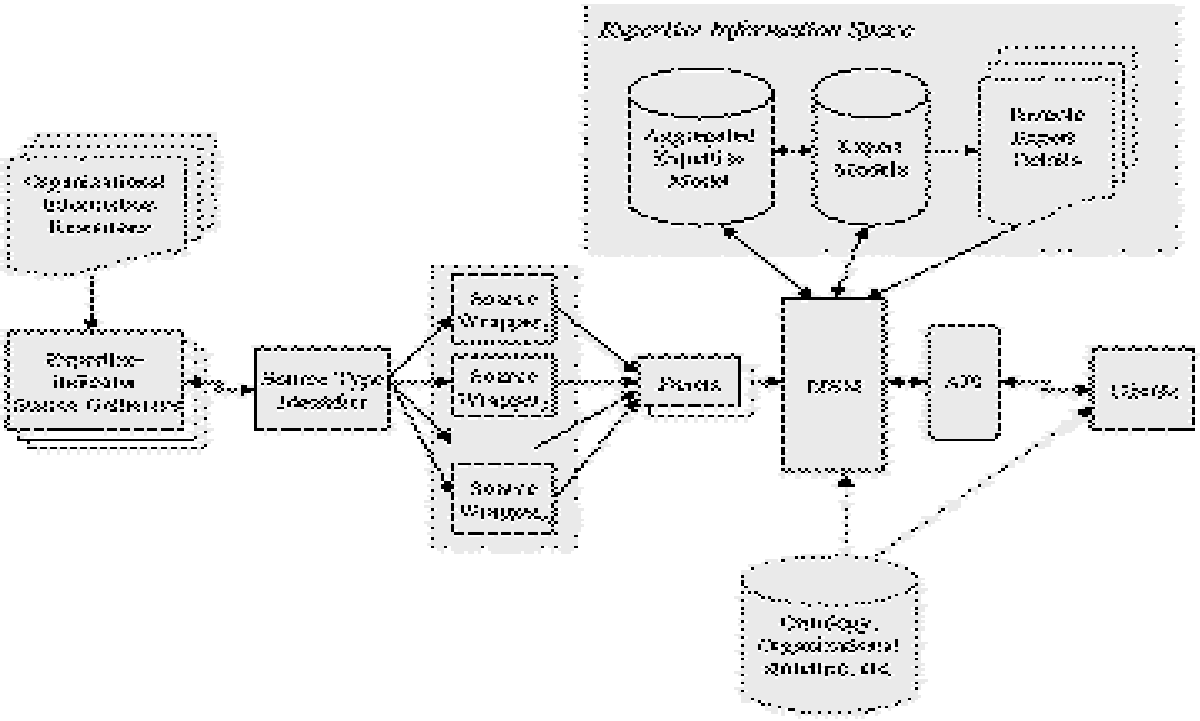

Figure 1. The DEMOIR architecture

Expertise modeling. For this, DEMOIR provides four components that are aimed at meeting our desire to factor provenance, expertise indicator source types, etc. into the expertise modeling. As shown in Figure 1, these components are:

- a *source type identifier* (which determines with which experts each gathered source relates to, and in what way),
- *source wrappers* (source type specific extractors of expertise indicators),
- *fusers* (that aggregate expertise indicators from the wrappers for storage as expertise models),
- and the *expertise information space manager* (EISM) which handles the storage and retrieval of the expertise information.

The source type identifier, wrappers, and fusers are configurable/pluggable and extensible. Hence, one can add new source wrappers and new rules for source type identification or fusing.

Expertise model exploitation. The DEMOIR server is designed as a collaborative system without anticipating particular types of clients. This means that the exploitation manner of expertise information is not limited. This is achieved by *API*s that can be used to

access the expertise information. Any kind of *client* which can be located anywhere in the organization can use the API to provide customized usage of the expertise information. Besides, applications which have appropriate interfaces to interoperate with other applications can make use of the server.

Furthermore, DEMOIR permits flexible partitioning of its modules as server and client operations. Currently we are adopting an approach where the expertise modeling is implemented as a server operation while the expertise indicator source gathering and expertise model exploitation can be distributed. These two client processes differ in their interaction with the server. While the expertise indicator source gatherers need to register with the server, the user clients don't need to do so.

But in cases where privacy issues prohibit the gathered sources from being brought to the expertise modeling server, some portions of the modeling modules (for example wrappers) can also be included in distributed gatherers. Furthermore, one can also envisage a deployment where the document type identifier, wrappers, fusers and the EISM are distributed across computers on a network interacting through a middleware like CORBA's ORB.

4. Current implementation status

As we indicated earlier, our implementation is at the moment mainly focused on the expertise modeling modules of DEMOIR. Specifically, we are implementing and experimenting with source wrappers, fusers and the EISM.

As shown in the DEMOIR architecture, classifying expertise sources based on how they relate to experts permits one to apply tailored extraction of expertise data using the wrappers. But, this is not all there is to it. The goal is to infer the expertise models by combining the inferred relation of experts to sources (i.e. source type) with expertise descriptors contained in, or implied by the sources. One strategy to achieve this goal is to employ matrices as representations of the above relations and develop an algorithm to drive the expertise models through a series of matrix merging. Hence we call this algorithm *matrix fusion*. Crudely put, this algorithm involves the following five steps:

1. Using expertise indicator source gatherers which are presently implemented as web robots, sources (documents) that are deemed to contain expertise data are gathered and a source-to-expert matrix that maps these sources to associated experts is generated. This matrix contains source type codes as its elements (note that a single source may be mapped to multiple experts).
2. For the time being, the gathered documents are then classified into three source types, namely documents produced by experts, document profiling experts and documents mentioning experts (this step is presently done manually).
3. Then, each source is passed to the respective wrapper that employs a set of heuristic algorithms to analyze its content and structure to select/identify key expertise descriptors and note their statistical features. The wrappers output these data as descriptor-to-source matrices.
4. The matrices generated by the wrappers are then fed into a fuser that merges them applying appropriate weighting which takes their source type into account. This is done by parameterizing the term weights by heuristically determined coefficients that quantify the relative importance of the source types.
5. Finally, the descriptor-to-source matrix produced by the fusers is merged with the source-to-expert matrix that was generated by step 1. This merging generates a direct mapping of expertise descriptors to experts. This mapping, which we call the *expertise model*, is also stored as a matrix (a descriptor-to-expert matrix).

How does this algorithm map onto the expertise modeling components of DEMOIR? The source type identifier is implemented to contain an *Expert-to-Source matrix generator* (that presently relies on manually tagged source types) and a *wrapper launcher* that traverses the expert-to-source matrix and passes the source to the respective wrappers. Two types of fusers are used to generate the expertise models from the wrappers' outputs. *A Descriptor-to-Source Matrix Fuser* (DSM Fuser) merges the descriptor-to-source matrices generated by the wrappers, while a *Descriptor-to-Expert Matrix Fuser* (DEM Fuser) performs step 5 of the algorithm described above. Presently, the expertise information space manager (EISM) performs basic matrix storage and access functions, and stores and retrieves the expert models (stored as LDAP directories as discussed below) as well as the archives of sources. The API merely provides access to EISM's functionality. Note that the matrix representation of the expertise model only contains plain terms as descriptors of expertise and hence the conceptual relation among these terms needs to be established later (e.g. by a mapping to an ontology, by a client that does dynamic clustering, etc.)

The implementation also aims to fulfill DEMOIR's goal of flexibility and configurability. To this end, we are currently making use of design patterns [5], specifically factory and prototype patterns, to develop the wrappers as pluggable components that are registered to, and invoked by, a central control. This has enabled us to configure these components to be loaded at runtime. It also has made it possible to control the behavior of wrappers through configurations and to add new wrappers with minimum change needed. The source type identifier components and the fusers, too, are based on configuration files enabling extensibility and

reconfiguration (for example, using different parameterization strategies during fusing). In general, our preliminary results in this regard are encouraging.

The *Expert models* that presently contain limited details of individual experts (like name, URL, etc.) are stored as LDAP directories and linked to the expertise model. This approach, in addition to benefiting from the features of LDAP directories, is expected to facilitate the integration of expertise models to directories already available in organizations as well as other related systems like user modeling servers [2]. These directories can also be distributed. The expertise models, too, can be placed on a separate host than the one storing the LDAP directory.

5. Future work

Once we completed our implementation and evaluation of the above components, the next step will be to automate the source type identification part. A candidate approach to performing this is to apply categorization techniques that are based on supervised learning. This will also include the consideration of more fine-grained source type as well as source format classifications that reflect the wide variety of sources commonly encountered in real world organizations (eg. human resource databases which can indicate experts' formal educational backgrounds, professional careers, current job description, etc.; skill databases profiling people's competency areas; employee performance evaluations; e-mails/discussion groups; queries for IR systems/document repositories; expertise recommendation by people; etc.).

Once the expertise models are properly generated, a range of interesting possibilities of exploitation arise. For instance, we plan to develop a client that integrates an ontology definition framework [8] being developed in our research department to dynamically categorize experts. We also plan to investigate the application of clustering techniques to visualize both expert clusters as well as clusters of expertise descriptors. We also intend to incorporate mechanisms to refine the expertise models based on social feedback/ranking.

Extending the current expert models/profiles to incorporate more sophisticated representations of additional attributes of individual experts is also necessary. For example, we should be able to represent attributes like social, organizational, co-authorship, citation, and other relations among experts, which would be valuable augmentations of the expertise models.

6. Summary

In this paper we identified and analyzed four approaches to expertise modeling. With particular reference to three of these alternatives, we have shown that each has its advantages and disadvantages. Consequently, we have proposed an architecture called DEMOIR that tries to combine the advantages of two of the identified alternatives. DEMOIR combines centralized and decentralized processes to create a hybrid approach to expertise modeling. In addition to this architectural requirement, DEMOIR is also motivated by the goal to perform expertise modeling that takes into account the differing importance of various sources as suppliers of expertise evidence.

We also described the approach we adopted to implement DEMOIR using a matrix-based representation of the relations between experts and documents, and the fusion of these matrices to generate the expertise models.

References

[1] K. M. Chandy and A. Rifkin, "Systematic Composition of Distributed Objects: Processes and Sessions", *The Computer Journal*, 40(8), 1997, pp. 465 – 478.

[2] Fink, J., *User Modeling Servers – Requirements, Design, and Implementation*, Ph.D. Thesis, Dept. of Mathematics and Computer Science, University of Essen, Germany, forthcoming.

[3] L. Foner, "A Multi-Agent Referral System for Matchmaking". in *The First International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technology (PAAM96)*, London, UK, 1996.

[4] L. N. Foner, *Political Artifacts and Personal Privacy: The Yenta Multi-Agent Distributed Matchmaking Systems*, Ph.D. Thesis, Massachusetts Institute of Technology, 1999 (http://foner.www.media.mit.edu/people/foner/PhD-Thesis/Dissertation/)

[5] Gamma, E., R. Helm, R. Johnson and J. Vlissides, *Design Patterns: Elements of Reusable Object-oriented Software*, Addison-Wesley, Reading, MA, 1995.

[6] H. Kautz, B. Selman and A. Milewski, "Agent Amplified Communication", in *The Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, Portland, OR, 1996.

[7] H. Kautz and B. Selman, "Creating Models of Real-World Communities with ReferralWeb", In *Working Notes of the Workshop on Recommender Systems, held in conjunction with AAAI-98*, Madison, WI, 1998.

[8] R. Klemke and A. Nick. "*brokersLounge: An Ontology –based Knowledge Management Environment*" (in preparation).

[9] B. Krulwich and C. Burkey, "The ContactFinder Agent: Answering Bulletin Board Questions with Referrals". In *Proceedings of the 1996 National Conference on Artificial Intelligence (AAAI-96)*, vol. 1, 1996, pp. 10 – 15.

[10] H. Libermann, A. Vivacqua, "Agents to Assist in Finding Help", *Proceedings of the CHI 2000*, The Hague, The Netherlands.

[11] D. Mattox, M. Maybury and D. Morey, "Enterprise Expert and Knowledge Discovery", In *Proceedings of 8th International Conference on Human-Computer Interaction (HCI International'99)*, 1999, pp. 303-307.

[12] Schreck, J., *Security and Privacy in User Models*, Ph.D. Thesis, Dept. of Mathematics and Computer Science, University of Essen, Germany, forthcoming.

[13] A. S. Vivacqua, "Agents for Expertise Location", in *The Proceedings of the AAAI Spring Symposium on Intelligent Agents in Cyberspace*, Stanford, CA, March 1999.

[14] D. Yimam and A. Kobsa, "Expert Finding Systems for Organizations: Problem and Domain Analysis and the DEMOIR Approach", In M. Ackerman, A. Cohen, V. Pipek and V. Wulf, eds.: Beyond Knowledge Management: Sharing Expertise, forthcoming.