Centralization vs. Decentralization Issues in Internet-based Knowledge Management Systems: Experiences from Expert Recommender Systems

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

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

Abstract

In this paper we report on the issues pertaining to centralized vs. decentralized architectures that we encountered at various points of our research on a particular Internet-based knowledge management and sharing system, namely expert finders/recommenders. First, we briefly describe and analyze our design experiences in expert recommender systems, emphasizing the architectural challenges and decisions. Then we extrapolate our experiences and insights to highlight some of the centralization vs. decentralization issues that we think need to be considered in systems design process. We point out the need to also include other relevant dimensions of a system's architecture into these design considerations, and suggest isolating centralization and decentralization factors from the problem space of an application rather than trying to adopt one or the other approach for the entire system.

1. Introduction

The increasing adoption of Internet technologies and online information repositories in organizations has spawned Internet-based knowledge management and knowledge sharing systems. Isolating the factors and requirements that influence optimal architectures for these systems is an active research area. Such systems need to handle a diversity of knowledge sources, computing resources and system users all of which are often distributed throughout an organization and beyond [4] [1]. In these systems, the access to and integrated representation of knowledge from different sources and domains across space and time is a primary requirement.

Expert recommenders are knowledge management and sharing systems that aim at helping users to trace human information and expertise sources rather than documents. What is more, these systems aim to achieve this by mining implicit sources of expertise data rather than explicit input. Like all other knowledge management and sharing systems, these systems, too, involve distributed knowledge sources (i.e. people), extract expertise data from distributed information resources, and cater for a variety of users (both human users and system clients).

From our domain analysis [11], we have learned that expert finding systems generally comprise a number of operations which can be grouped into three major activities: (1) expertise data source recognition and extraction mechanisms, (2) expertise modeling (including expertise indicator extraction, expertise model representation), and (3) expertise model deployment (query mechanisms, matching operations, output presentation, adaptation and learning operations).

These features can be implemented and organized in a variety of topologies. As is the case with all systems that have to exploit distributed sources and cater to distributed users, the decision whether to centralize or decentralize these operations pervades the architectural decision process.

In the following, we report our experiences in few of these approaches. We first present a purely centralized implementation of query-time generated expertise modeling system and discuss our experiences with it. Next, with the aim of designing a system that resolves the shortcomings of this approach, we compare centralized and decentralized alternative architectures. Finally, we demonstrate how our hybrid approach, called DEMOIR[1], addresses the requirements of distributed and heterogeneous, organizational and personal expertise data sources on the one hand, and centralized access to extracted expertise information on the other.

---

[1] DEMOIR stands for "Dynamic Expertise Modeling from Organizational Information Resources".

2. A centralized query-time expertise modeling system

Our initial approach to tackling the problem of modeling expertise of people from implicit sources was making use of a Web indexing and search engine. Such systems are *de facto* the standard mechanisms for exploiting distributed information sources both on the Internet and on intranets, especially in most Internet-based knowledge management applications. Commonly, these systems build a centralized global index for the designated information resources and provide a search engine to enable access to it.

As shown in Figure 1, we used a web indexing system called Glimpse (which is also the default search engine used in the well-known Harvest Web indexing system [2]) and built an expert modeling and recommender module on top of it. For gathering the personal web pages and related pages from a department's (GMD-FIT) web site, we used the spider called WebGlimpse. The documents gathered by this spider are indexed and searched by Glimpse. When a user submits a query for the Expert Query Interface (which is accessible using any browser), the query is passed to Glimpse which returns the passages in documents that contain the given query terms. The output from Glimpse is then fed to the expertise Modeler & Tracer co-

Figure 1. IR engine based query-time expertise modeling and recommender system

mponent that uses various statistical and heuristic methods (analyzing proximity and document structure) to associate occurrences of names with query terms in the passages. The expertise modeling is hence dynamically performed on the sources (passages) that are returned by the retrieval engine. Finally, experts are ranked by the overall statistical correlation between names of experts and query terms.

In general, one can design the expert modeling to either be co-located with the indexing and retrieval engine on the server (as we did) or alternatively do the expert modeling in clients at the users' computers (e.g. as a Java applet)[2]. However, the latter involves a large amount of data transfer over the network at the time of retrieval and hence is less viable. Moreover, the decentralization achieved in this approach is currently limited since the clients would still need to interact with a centralized indexing and retrieval engine. Current search engines typically require a centralized index of the distributed sources of expertise data. Normally, the web spiders pull documents and store them in a centralized server to be indexed by the search engines. The need for distributed indexing and search of the World Wide Web documents that can cope with the exponential growth of documents in organizations as well as the Web is long known [6]. There are also research works on distributed indexing and query systems (e,g., [3] [9]). However, most currently available systems make use of a centralized global index, and so far query processing using decentralized indexes is consistently outperformed by a global (albeit distributed in a tightly coupled manner at a lower level) indexes [8]. This is mainly due to the overhead incurred by the query routing involved in the distributed approach.

In the expert finding application, this architectural limitation imposed by the underlying indexing and search engine is compounded by the fact that expertise modeling is done only when a list of documents (or passages from them) that match a given query are returned by the search engine. These outputs need to be temporarily stored in a cache at query-time and analyzed in real time. We found that this very approach resulted in a number of shortcomings which affect both performance and expertise model usability. These shortcomings included:

- high latency in query processing;

- limited exploitation of the expertise data due to the absence of a persistent storage of all expertise data extracted from the sources (only the sources returned as matching a given query are available at a time);

- inherent unsuitability of the indexes for expertise modeling (since the indexes are originally intended for keyword based retrieval of documents, and not for supporting data extraction tasks like expertise model-
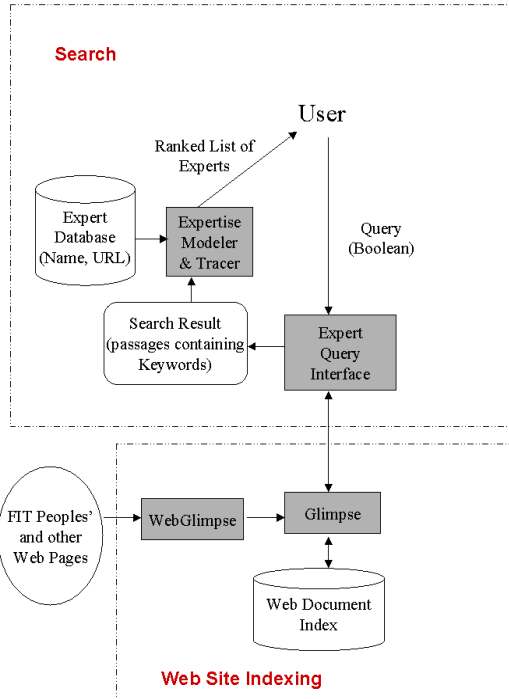
---

[2] In Fig.1, these clients would sit on the computers of the department members.

ing, they are inefficient in supporting queries like "what are the expertise of Mr. X ?");

- limited types and coverage of documents that can be made accessible to the centralized indexing system (eg., emails and other personal sources are difficult to include);

- difficult to incorporate sources of expertise data other than documents (eg. recommendations from people, social relations, feedback, etc.);

- full reliance of the expertise modeler on the availability of the search engine.

In general, the above shortcomings called for two remedial actions: (1) removing the indexing and search engines and coming up with better ways of exploiting the organizational information resources for expertise modeling, and (2) generating expertise models in the background and in advance (in anticipation of future queries) and support expertise information services based on the resulting models.

3. Pre-generation of expertise models: to centralize or decentralize ?

While analyzing alternative approaches that can satisfy the above requirements, the question of adopting a centralized or decentralized architecture became a prime focus of our analysis. In particular, we identified and evaluated two alternative approaches: (1) distributed expertise modeling using personal agents, and (2) centralized models with decentralized gathering. Below, we will briefly describe each of these approaches and point out their merits and demerits (see [12] for a more detailed treatment).

In the decentralized approach, expertise modeling is distributed to self-managing agents belonging to each individual expert, and reside on his/her machine [eg. 10]. As such, the whole system becomes a multi-agent system composed of personal agents which carry out the dual tasks of modeling expertise (from authored documents and other sources) as well as assisting their owners in searching other experts. These agents can be endowed with autonomy to do all the processing and control of expertise modeling on their own and with mobility or pseudo-mobility to interact with one another.

In the centralized approach, individual experts are linked to a pre-constructed or dynamically generated central expertise model which can be a kind of knowledge model (ontology), organizational structure, etc. (rudimentary versions of this approach are reported in [7] [5]).

Each of these two approaches have their own merits and demerits. The decentralized approach permits locality of expertise modeling, lower control complexity of the expertise modeling process and privacy, and graceful degradation of the overall performance if one or more local components become unavailable. It suffers, however, from the limitations of relying on personal sources of information only (e.g., it does not consider documents *about* experts that are not located in the personal spheres of these experts), of limited accessibility and sub-optimal utilization of expertise data by others as well as scalability problems as the number of experts grows large. In contrast, the centralized approach has the advantage of allowing organization-wide and multi-purpose exploitation of the expertise information, allowing the system to manipulate the expertise information in aggregate, and being able to monitor a wide range of organizational sources for up-to-date expertise data. However, this approach fails to afford local control by experts, access to personal information sources of individual experts, and must include additional backup mechanisms in case of breakdown and security mechanisms to guarantee the privacy of the expertise data.

The best solution, thus, seems to be a mixed architecture. The distributed data sources, privacy and control issues motivate decentralization. On the other hand, the need to have a single-point of access to the expertise information (which facilitates exploitation by several systems and users), integrate the information for efficient and multi-purpose exploitation and the need to more easily mine a variety of organizational sources thrust towards a centralized approach.

At this stage, so as to determine the exact architecture of our system, we had to look into the expert finding problem domain in more detail and try to decompose the solution space with the aim of coming up with modules that are loosely coupled with one another. But how to create these modules turned out to be a complex decision involving factors other than centralization vs. decentralization. For example, we wanted to base the architecture on our hypothesis of taking note of expertise data source types as an integral part of the expertise modeling process (see [11] [12] for details). Moreover, the decision involved comparing the benefits of organizing on different bases (e.g. location of expertise data source versus location of expertise data use). We also had the options of creating modules that carry out identical functions at distributed locations (e.g. distribution of expertise modelers at different hosts of data sources and then bringing the data to a central repository) or decomposing the expertise modeling into modules that can be distributed across computers.

The resulting DEMOIR architecture, shown in Figure 2, is a modular architecture consisting of centralized as well as decentralized components. DEMOIR dissociates functions like source gathering, expertise modeling and expertise model exploitation and delegates them to specific components which can be implemented separately
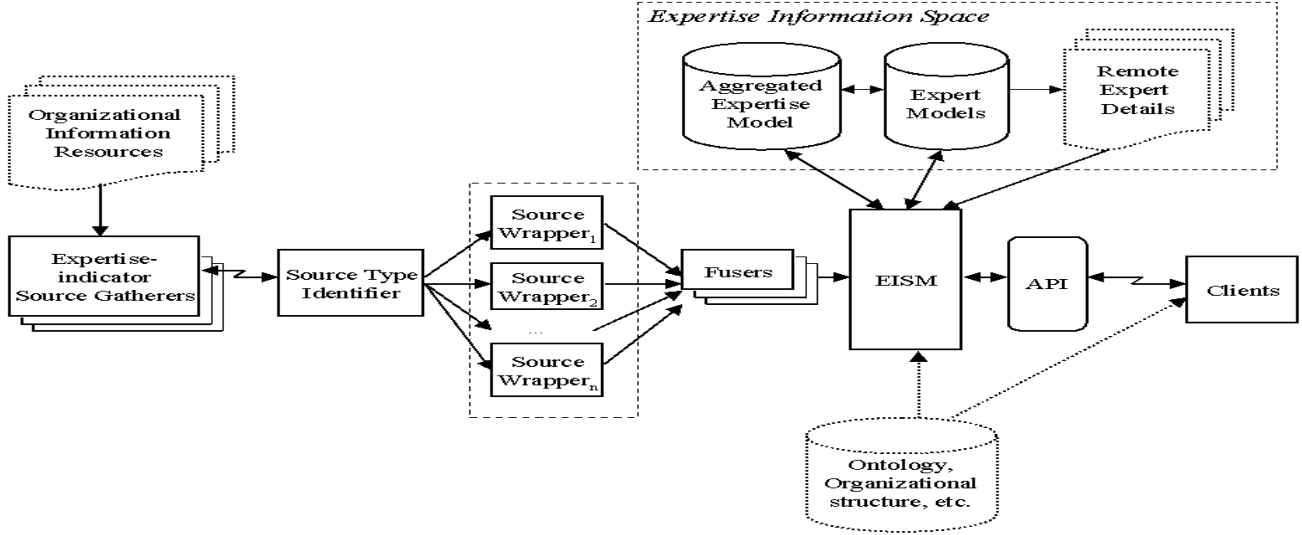
Figure 2. The DEMOIR Architecture

and readily combined to suit an application environment. Designing appropriate interfaces, these modules can also be distributed across computers.

The Expertise Indicator Source Gatherers which serve as "informers" for the expertise modeling components are decentralized in that they are independent (not controlled by a central control) and adaptable to local needs and constraints of the expertise indicator sources. In a way, the gatherers in DEMOIR are architecturally akin to the distributed gathering and pre-processing approach of the Harvest system [2] (cf. the presently predominant technique of using web robots to copy documents for centralized indexing). However, DEMOIR's gatherers have a different motivation as well as manner of operation.

The expertise modeling is carried out by the four components, namely source type identifier, source wrappers (specialized extractors of expertise indicators from sources), 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.

Distributed clients are supported by the application programming interfaces (APIs) 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 access the expertise information. Besides, applications which have appropriate interfaces to integrate other applications can also 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 exploita-

tion 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 to enable experts control the extraction of expertise data from their personal resources. 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. Summary

As shown, analyzing the tradeoff between centralized and decentralized architectures was a major concern in our design process. One observation from our work was that centralization/decentralization is only one dimension of a system's architecture. In designing a system, one also needs to deal with different, though related, dimensions like complexity (number of different parts, heterogeneity of data and their sources), permissions (eg. accessibility/ privacy constraints, manner of use), communication patterns, etc. Centralization/decentralization options need to be analyzed keeping such relevant dimensions in mind. Not only should one keep these dimensions in mind, but should also analyze how they affect the centralization vs. decentralization decision.

Our experience also shows that, at least in applications like ours, both centralized and decentralized options have their merits and demerits. Moreover, an Internet-based application domain would very likely involve both centralizable and decentralizable tasks. The difficult challenge then becomes isolating such tasks and analyzing the tradeoff between centralizing and decentralizing their operation. Also, in cases where an application has to be composed of some modules that work in a centralized manner and others that are decentralized, how to get these modules work together is a non-trivial task.

In retrospect, we can divide the process we followed in handling this issue into three steps:

1. Identifying system requirements/tasks (to do this, we built upon our domain analysis as well as experiences from our initial search engine based expertise modeling system).

2. Identifying and analyzing the centralized and decentralized architectural alternatives of meeting these requirements. Doing this gave us a clearer picture of the centralization and decentralization factors involved in our problem domain. It also has helped us gain a better understanding of our problem domain in general.

3. The final specification of the central features that should constitute our system then resulted in the realization that a hybrid approach was necessary. We then built the overall system architecture (DEMOIR) to be flexible enough to accommodate varying degrees of centralization and/or decentralization depending on the particular demands of a deployment. This also involved, *inter alia*, handling the new levels of coordination and control that ensue as a result of the decentralized approach.

References

[1] M. Bonifacio, P. Bouquet and A. Manzardo. "A distributed Intelligence Paradigm for Knowledge Management". In *Bringing Knowledge to Business Process workshop in AAAI Spring Symposium Series 2000*, 20 –22 March, 2000.

[2] C. M. Bowman, P. B. Danzig, D. R. Hardy, U. Manber, and M. F. Schwartz. "The Harvest Information discovery and Access system". *Computer networks and ISDN Systems*, 28, 1995, pp. 119 – 125.

[3] J. C. French, A. L. Powell and W. R. Creighton. "Efficient searching in distributed digital libraries" *Proc. of the third ACM Digital Libraries Conference*, 1998, pp. 283 – 284.

[4] B. R. Gaines and M. G. Shaw "A Networked, Open Architecture Knowledge Management System". *Proc. of 10th Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Alberta, Canada, 1996. V. 2, pp. 1 - 22.

[5] 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.

[6] T. Koch, A. Ardö, A. Brümmer and S. Lundberg. "The building and maintenance of robot based internet search services: A review of current indexing and data collection methods" Report to meet the requirements of Work Package 3 of EU Telematics for Research, project DESIRE. London University Library, NetLab, 1996. http://www.lub.lu.se/desire/radar/reports/

[7] B. Krulwich and C. Burkey, "The ContactFinder Agent: Answering bulletin board questions with referrals". in *Proc. of the 1996 National Conference on Artificial Intelligence (AAAI-96)*, vol. 1, 1996, pp. 10 – 15.

[8] B. Ribeiro-Neto and R. A. Barbosa. "Query performance for tightly coupled distributed digital libraries*" Proc. of the 3rd ACM Digital Libraries Conference*, 1998, pp. 182 – 190.

[9] B. Ribeiro-Neto, E. S. Moura, M. S. Neubert, N. Ziviani, "Efficient distributed Algorithms to build inverted files" *Proc. of the 22nd ACM SIGIR Conference, 1999, pp. 105 – 112.*

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

[11] 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*. Boston, MA: MIT Press, 2000 (under review).

[12] D. Yimam and A. Kobsa. "DEMOIR: A Hybrid Architecture for Expertise modeling and recommender systems". To appear in: *IEEE WETICE Knowledge Media Networking Workshop*, 14-16 June 2000, National Institute of Standards & Technology (NIST), USA.