NetAgent: A Global Search System over Internet Resources by Distributed Agents

Taeha Park <taeha@cosmos.kaist.ac.kr>
Kilnam Chon <chon@cosmos.kaist.ac.kr>

Abstract

This paper describes a search system where users can do global searches in context. For this purpose, we propose a collaborative indexing model for gathering index data from users. In our model, various information servers and search systems are modeled as agents. Each agent defines its domain by the community and topic that it serves. Users can register associations to represent agent relationship on a given topic. To locate a resource, a user interacts with a series of agents by following a path that agents recommend. Agents gather users' feedback and use it to change weight values of associations. By interacting with agents, a user can avoid many of the search results that are irrelevant in the context of the user's community.

1 Introduction

In the Internet, people organize and collect information resources using many information tools like FTP [11], Network News [7], Gopher [1], WWW [3], WAIS [6] and so on. Due to the large volume of information resources, users often use search tools to locate resources [10].

Existing information tools do not provide a way to share users' experience on information resources. Imagine a user wants to find out resources of a certain criteria. The user will try various information tools and search tools to locate the resources. After some trials and possibly with the aid of human experts, the user will locate the necessary resources. Another user looking for similar resources will need to repeat the same sequence of trials.

In this paper, we describe NetAgent system, where users interact with distributed agents before actual searches. Each agent records a user's successful search patterns and keeps them for future usage. NetAgent allows users to search for information in context. Instead of a single indexing system, NetAgent deploys multiple topic-specific indexing agents. A community of users shares a common context over the global information space by interacting with common agents. A user can find a topic-specific indexing agent suitable for the user's community by following a path that agents suggest. By doing this, a user can avoid many of the search results that are useless in the context of the user's community.

The key idea of this paper is to enhance current indexing techniques by achieving the followings:

- *Collaborative indexing* for collecting index data from users. A user can explicitly register a new index or implicitly contribute to his or her community by generating feedback to the indexing system.
- *Context-sensitive searching* using organized index data. Community-specific search or topic-specific search is supported.

This paper is organized as follows. Section 2 presents the design of NetAgent, and Section 3 describes its implementation. Section 4 analyzes how NetAgent deals with the scalability problems. Finally, Section 5 concludes this paper.

2 Design

This section describes the design of NetAgent. The design is focused on collaborative indexing; provide a framework where index data are collected, organized and shared by multiple users.

The idea of deploying agents for collaborative indexing is introduced by the analogy of the real world. In general, when people have a question on a topic and cannot find the answer, they look for experts in that area and ask questions to them.

In the networked electronic world, users looking for resources on a particular topic would send messages to relevant mailing lists or news groups asking if anyone knows where to look. This means that significant parts of search information relies on users' experience.

2.1 Overview

In the NetAgent model, a collection of resources is defined as a unit of indexing. We define a *resource domain* as a collection of resources in an information server. A resource domain is represented by at least one specific form of an information tool. Examples of the resource domain are a directory in an anonymous FTP server, a subtree in a gopher server, a WAIS source, a

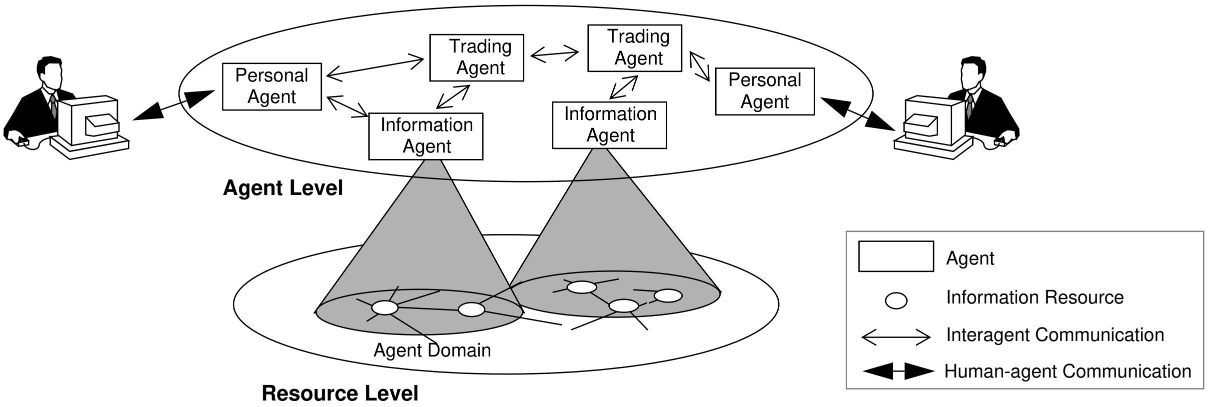

Figure 1: Two-level indexing model

USENET newsgroup, and a set of neighboring pages from a WWW page.

In the NetAgent model, a user narrows down the resource domain by interacting with agents. Once a resource domain is selected, actual searches are done over the resources. At each stage of the interactions, the user decides which agents will be selected for further interactions.

In our model, an information space is composed of two levels (see Figure 1):

- In the agent level, an agent communicates with other agents to find out which agents have the index data on a given topic.
- In the resource level, resources are organized by tools in resource domains. An agent specifies a set of resources as its agent domain, and keeps index data for the resources.

A user can issue a query to search over the global information space. When a user issues a query to an agent, it responds with a list of other useful agents. The user can repeatedly send queries to the resulting agents until he or she is satisfied with the result.

Agents communicate by *message passing.* An agent can send a message to other agent only when there is an *association* with the target agent. An association is a tuple of a search key, a target agent, and a new query for the target agent. Agents determine where to propagate a query message by selecting associations.

The NetAgent design offers three principal features for resource domain selection: (1) registration of a new association, (2) trading of associations using weight values, and (3) feedback for changing the weight values of associations.

There are three types of agents according to their roles; personal agents, trading agents, and information agents. A *personal agent* interacts with a human user as well as other agents. A user can send a query to the network of agents by his or her personal agent. A query from a personal agent is propagated to a set of information agents via trading agents. A *trading agent* suggests which agents to follow for a given topic. The role of a trading agent is to support collaborative indexing by keeping the user-registered index data on a topic. An *information agent* is an agent that can access resources. An information agent performs resource retrieval by converting agent messages into the protocol of an information tool. As information tools are passive and can not generate agent messages, information agents retrieve resources actively using tools, and reply back to other agents.

2.2 Interagent Communications

Communications between agents are done by message passing. NetAgent system uses a client-server model for a single trading session. An agent becomes either a client or a server for a given session. In a trading path, a server agent for a session becomes a client agent for the next trading. The trading path ends with an information agent.

A message between agents is in Agent Access Format (AAF). The AAF is designed to be compliant to the Uniform Resource Locator format [4] to allow future information tools to access NetAgent. The syntax of AAF is:

$$\texttt{x-na://}host\texttt{/}agent\texttt{?}expr_1\&expr_2\&...\&expr_n$$

`x-na` is a temporally assigned name of URL scheme for NetAgent. *host* is a fully qualified domain name [9] of a network host where the agent resides. *agent* is a unique agent name in a host. *expr* is a simple expression composed of attribute name, operator and attribute value. The AND operator is assumed for all expressions in a message. Users can give multiple queries instead of using OR operators. Possible attribute names are shown in Table 1.

A trading agent may change the input expression and return it as an association. For example, if a "multimedia" agent gets an expression containing a term "multimedia", it will remove the term because it is no longer necessary.

Among the attributes, `area` is used to give a hint to information agents about finding resource domains. If `area` is not given, `subject` attributes are examined instead. Following is an example query expression for "find all resources on MPEG which is created by an author Kim"

subject="multimedia" & subject="mpeg" & author="Kim"

In addition to the expressions from the user, more parameters are included by [illegible] message. These parameters are hidden from the users. Message types and their parameters are listed in Table 2.

2.3 Three-phase query processing

A query issued by a personal agent is processed in three phases as shown in Figure 2.

1. Summary lookup phase
   Using a personal agent, a user interacts with a series of trading agents to find out target information agents. By using an initial query from a user, a personal agent sends a `QUERY` message to a trading agent, and gets `ASSOCIATION` messages back. The personal agent shows resulting associations to the user so that he or she can interact with resulting trading agents. If the user selects another trading agents, the personal agent sends messages to the selected trading agents.

   If an information agent gets a `QUERY` message, it will try to match the given query to the resources in its domain, and will send `REPORT` messages containing summary information. Summary information is a brief description about a resource composed of subject, date, author, and the identifier for retrieving the resource. The personal agent shows summary entries to the user so that the user can retrieve resources selectively.

2. Retrieval phase
   If a user selects a summary item, the personal agent sends a `RETRIEVE` request to the corresponding information agent. The information agent translates the request into tool-dependent protocols and sends `RESULT` back to the personal agent. Further retrieval following a tool-dependent structure is possible by the interaction of the personal agent and the information agent.

3. Feedback phase
   After the retrieval phase, the information agent sends a `FEEDBACK` message to trading agents by following the reverse path of the summary lookup phase. On receiving a `FEEDBACK` message, the agent removes itself from the access path, and send `FEEDBACK` message to the precedent agent in the access path.

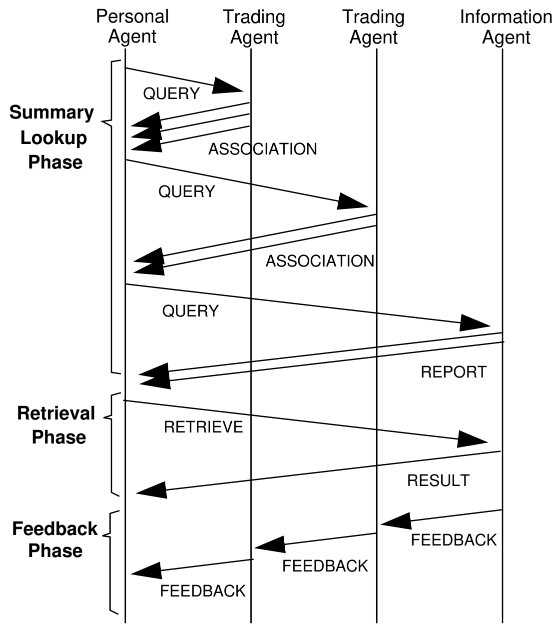

Figure 2: Three-phase query processing

2.4 Agent Domain

A trading agent aims to be specialized for a topic and a specific user community. Thus an *agent domain* is defined by the topic and the community of an agent.

A `topic` of an agent is the list of keywords the agent is supposed to index. The topic mechanism prevents users from registering irrelevant keywords to an agent. Narrowing down an agent's interest to a single topic makes topic-specific indexing more effective. For example, for a "multi-

Table 1: Attribute names in AAF expression

Name	Type	Description
subject	string	Title or subject of a resource
area	string	User-given area of a resource
content	string	Partial match for a string in a resource content
date	yy.mm.dd	Date appearing in a resource
author	string	The creator of a resource
source	string	Source tool (e.g. nntp, gopher, wais, mail, etc)
filetype	string	File type (e.g. text, postscript, gif, rtf, etc)
filename	string	File name for a resource
filesize	integer	Size of a resource in bytes
filedate	yy.mm.dd	Date of the file for a resource
path	string	Resource domain name where a resource is located

Table 2: Message types and parameters

Message Type	Parameters	Description
`QUERY`	community access-path	Request for summary information
`RETRIEVE`	community access-path local-id	Request for resource retrieve
`FEEDBACK`	community access-path	Request for changing weight values
`ASSOCIATION`	hit-count total-count owner	Reply on trading information
`REPORT`	subject date author local-id	Reply on summary information
`RESULT`	message	Reply on the content of a resource
`DEBUG`	message	Debugging messages
`ERROR`	message	Error messages

media" agent, a user can register a new association only if the association is about pre-defined keywords the agent permits (e.g. multimedia mail, CD-ROM, video, etc).

A personal agent has a `community` value and records it in all outgoing queries so that trading with community match will happen. A personal agent in an organization will commonly refer to an organization-specific trading agent, and will provide shared feedback to the agent. Thus the organization-specific trading agent (e.g. a laboratory agent) will have good knowledge on the organization members' preference.

A user who wants to be attached to more than one community can register new associations referring other organization's agents, although he or she can not offer feedbacks to those agents.

We chose domain names of the Domain Name System [9] to define the community. This has the benefit of simplicity. Another design alternative is to use a group of user names, but it is regarded to be of too fine granularity. An agent will accept a `FEEDBACK` message only if the agent's community is a superset of the message's community.

Each trading agent can be regarded as an indexing server on a specific area. A *general agent* is a special kind of trading agent that does not limit its domain to certain keywords. Most organization-level agents would refer to the general agent for all unclassified keys. The role of a general agent is an operational issue, rather than a design issue. The registration on general agents would be performed intellectually by subject experts, or by automatic indexing methods studied in information retrieval area [12].

2.5 Feedback and Weight Value Changes

The dynamic behavior of an agent is based on weighted associations. A trading agent uses weight values for *relevance judgments* on how much its associations have been useful. We denote the weight values by two numbers: *hit count / total count*. Total count is the number an association was referred. Hit count is the number an association was actually used to retrieve resources by personal agents.

For example, if a weight value of an association is 50/100, it means that the agent suggested the association 100 times to users, but it was actually referred 50 times for retrieving resources. The total count is incremented by a `ASSOCIATION` message, and the hit count is incremented by a `FEEDBACK` message.

A personal agent copies its community to all messages it generates. A trading agent accepts a feedback only if the community of the message belongs to the same community that the trading agent serves.

For example, suppose a trading agent of community "mit.edu" suggests to a personal agent of community "kaist.ac.kr" to follow an association. If the personal agent retrieves a resource using the suggested association, a feedback message will be sent to the trading agent. But the feedback will be ignored since the community of the feedback message ("kaist.ac.kr") does not match the community of the trading agent ("mit.edu").

2.6 Registration of a New Association

A user can register a new association to a trading agent if he or she is allowed to do so by the agent's owner. A trading agent or an information agent on a resource domain can be registered. The registry program permits a new association if the topic and the registering person are acceptable for the agent. The registry checks the user's password to verify the registering person.

Figure 3 shows a registry user interface. In this example, a newsgroup is registered to the "multimedia" trading agent.

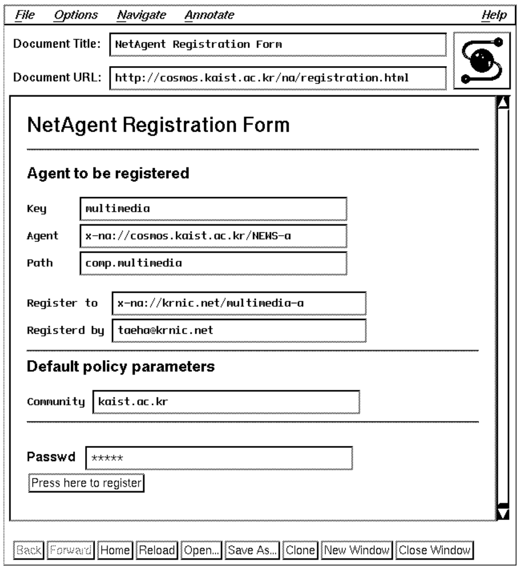

Figure 3: The registry in a Mosaic window

3 Implementation

A prototype system was implemented in Perl script language [13]. The user interface was done using Mosaic [2], a WWW client program. The authors configured a small number of trading agents like "multimedia" agent, and we are testing the implementation. Readers can try the NetAgent prototype at *http://cosmos.kaist.ac.kr/netagent.html*.

3.1 Personal Agent

A personal agent can be implemented in several ways such as a command language shell or a form-based language. In our prototype, personal agents are implemented in a Mosaic window. Figure 4 shows a query input of a personal agent.

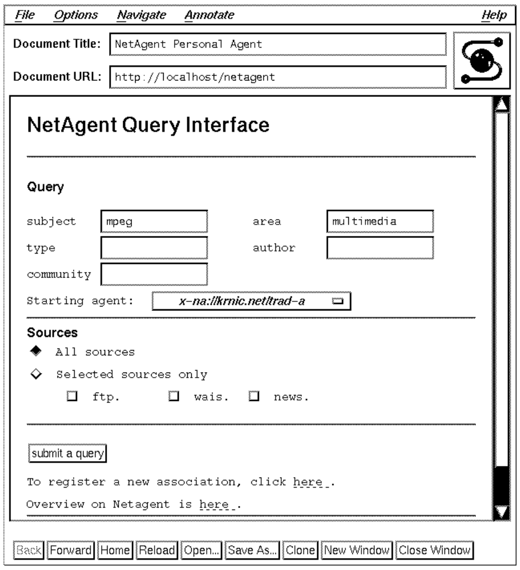

Figure 4: A personal agent in a Mosaic window

3.2 Information Agent

An information agent converts agent messages into tool-specific operations. We implemented information agents for network news, FTP, and WAIS system. More information agents might be added to support other information tools. Currently WWW is not supported, but the authors are planning to incorporate existing WWW indexing systems as information agents.

An FTP agent was implemented using Glimpse (Global Implicit Search) [8]. Glimpse is an indexing and query system that allows users to search through lots of files in many directories by building a very small index (2-5% of the text). The FTP agent translates the given agent query into Glimpse search commands, and generates agent summary reports from Glimpse search output.

A news agent converts an agent query message into NNTP operations. Summary information is extracted from USENET news headers. Some file-related attributes like `filename` and `filedate` are ignored by a news agent.

A WAIS agent converts the message of an agent into the parameters of `waisq`, a WAIS client program. The output of `waisq` is translated back to agent summary reports.

Information agents are designed to make use of information inherent to information tools as much as possible for efficient searches. For example, for a given topic, an FTP agent tries file name matches first. If the matched file is a text file, then it executes full text searches. A news agent gathers information from USENET news headers such as subject and keyword for efficient searches. Among the news headers, news article fields of *From, Organization, Newsgroups, Subject,* and *Date* are used for match from the given query. *From* and *Organization* are used for matching the author. The WWW agent would navigate the hypermedia space by following a limited number of links from the starting node for full text searches.

The granularity of indexing is different for each information agent. Among the attributes in the query expression, only a subset of attributes will be supported by an information tool. For example, for a given topic, an FTP agent can try all attribute matches except *author*, since FTP agent cannot extract author information from an unstructured document.

Although a query does not specify the resource domain, the information agents should be able to suggest some domains. Information agents usually keep *seed index data* for this purpose. For example, a news agent tries to find candidate newsgroups by using a news system file that describes USENET newsgroups (e.g. `/usr/lib/news/newsgroups` in most Unix news server systems). Since the *newsgroups* file includes keywords of all news groups, it provides a good starting point for domain selection. Similarly, WAIS information agent refers to the WAIS *directory-of-servers* source to find out default starting points of a WAIS search.

3.3 Trading Agent

Behavior of each trading agent is specified in an agent configuration file. The agent configuration includes agent domain specification and trading specification.

Figure 5 is an example of a configuration file of a trading agent. Each entry represents an association by the key, hit count, total count, owner, and the AAF of the destination agent. In this example, hit count and total count are initially set to 50 and 100 respectively. If no matching associations are found for a given query, default associations are used only. A default association have a small hit count as its initial value to make it appear in a low priority in association reports.

In addition to the explicit registration by users, the owner of an agent can modify the agent configuration file manually to add heuristics to the agent. A trading agent loads its associations when it becomes active, and saves it back to the file when the trading is finished.

3.4 Run-time Architecture

Figure 6 represents a message flow among NetAgent system components. Each host invokes a process called *host server* for an incoming query. A host server distributes messages to agents in the host. Following is a sample scenario.

- **A**: A user registers an association on newsgroup comp.multimedia to the "Multimedia" trading agent using the registry as was shown in Figure 3.

- **B**: Another user using P_1 issues a query to "General" trading agent that has an association to "Multimedia" agent. The "General" agent would show candidate associations as shown in Figure 7.

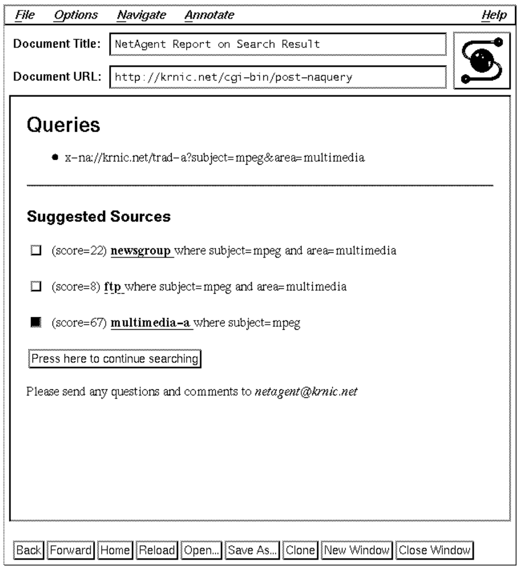

Figure 7: An example of a resource domain selection window

- **C**: If the user selects an association to "Multimedia" agent, P_1 will send a query to "Multimedia" agent. The "Multimedia" agent would show candidate associations as shown in Figure 8. The score of an association is the ratio of hit count and total count values.

- **D**: If the user selects the association to the newsgroup comp.multimedia, P_1 will send a query to the "News" agent. Because "News" agent is an information agent, it will reply with summary information as shown in Figure 9.

```
topic: multi media hyper virtual
community: kaist.ac.kr
register: krnic.net

default          193 565  taeha   x-na://news.kaist.ac.kr/NEWS-a?path=comp.multimedia
default          168 371  taeha   x-na://krnic.net/WAIS-a?path=comp.multi.src
default          112 439  taeha   x-na://cosmos.kaist.ac.kr/FTP-a?path=/pub/mmc
default          132 604  taeha   x-na://krnic.net/WAIS-a?path=Digital-All.src
multimedia mail  52  112  taeha   x-na://news.kaist.ac.kr/NEWS-a?path=comp.mail.multi-media
virtual reality  54  106  taeha   x-na://news.kaist.ac.kr/NEWS-a?path=sci.virtual-worlds
hypertext        53  132  taeha   hypermedia
hypermedia       65  156  taeha   x-na://krnic.net/WAIS-a?path=SIGHyper.src
hypermedia       58  129  taeha   x-na://news.kaist.ac.kr/NEWS-a?path=alt.hypertext
```

Figure 5: An example of "multimedia" trading agent configuration

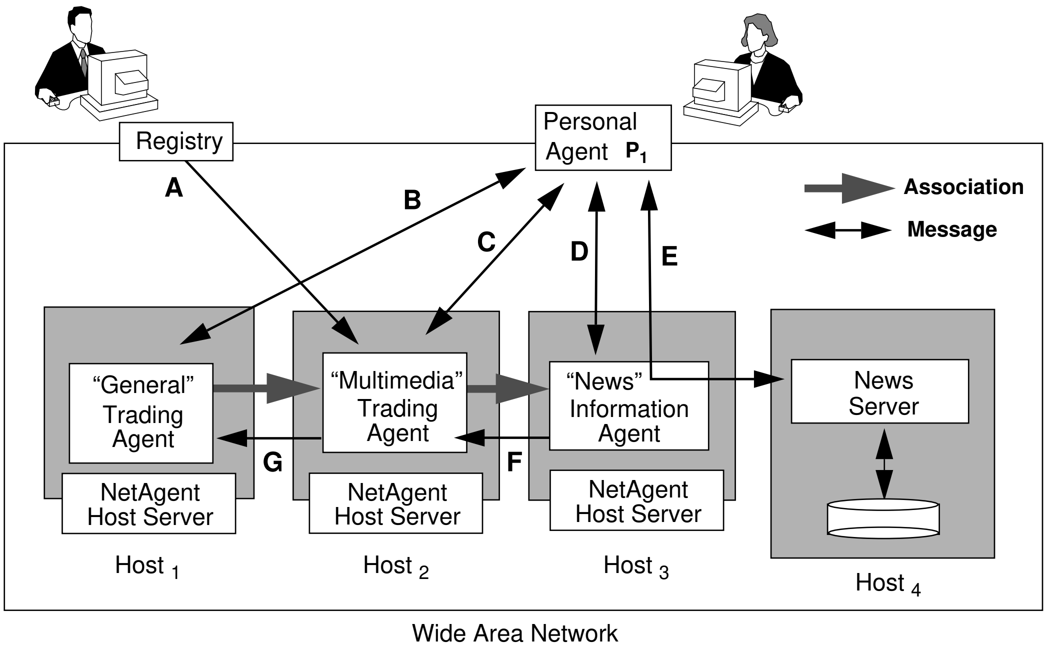

Figure 6: A sample interagent communications at run-time

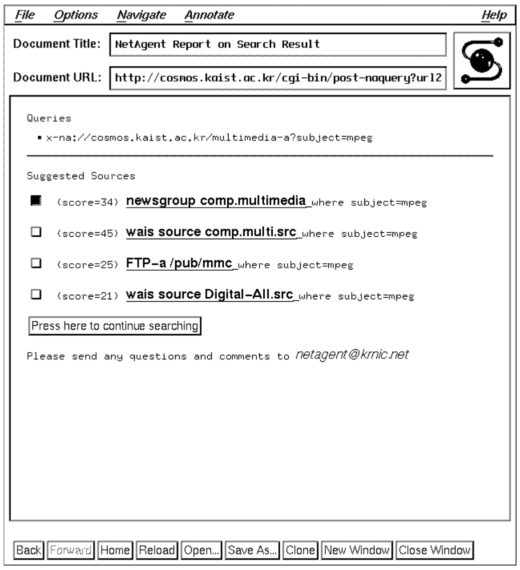

Figure 8: An example of selecting an association

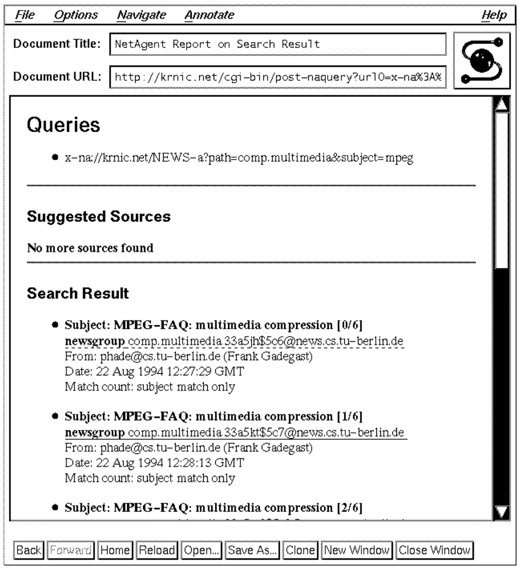

Figure 9: An example of summary information on a News source

- **E**: If the user selects summary information entries, P_1 connects the news server in $Host_4$ via "News" agent and retrieve resources.

- **F**: Since the trading in this example is determined to be useful, "News" agent generates a feedback message to "Multimedia" agent by referring the access path of the incoming `RETRIEVE` message.

- **G**: On receiving the feedback message, "Multimedia" agent increments the hit count of the association, and forwards the feedback message to "General" agent. "General" agent receives the feedback message, increments the association's hit count, and it stops forwarding the feedback message because it initiated the trading.

4 Analysis

Future information systems can be characterized by large scalability including enormous data volume, rapid growth in the user base and burgeoning information system diversity [5].

Among the scalability issues, we do not consider performance issues here. We will focus on how NetAgent deals with the scalability problems in user's context.

4.1 Information Overload Problem

In current information tools and indexing systems, users do searches in general context, and they get many irrelevant data also.

NetAgent provides *navigate-then-search* instead of blind search. By navigating among trading agents, user can find a limited set of resource domains for a topic and perform effective search on the selected resource domains. Because users can get response from selected trading agents that has index data on a topic, they can avoid the overhead on information filtering.

The use of community also filters out many irrelevant data. Community-specific trading agents isolates different set of index data although they share same keywords.

4.2 Information Distribution Problem

Using existing information tools, changes on information space are distributed directly or indirectly to the users.

In direct information distribution, users usually send a message to a common interested group identified by mailing lists or news groups. The problem of direct information distribution is that reaching to the right set of users is not always possible.

Another problem is that individual users have to be aware of the changes in the global information space, or have to configure an information tool manually to keep his or her personal view up-to-date.

In indirect information distribution, the changes are done in the global information space without sending notice to users. The problem is that only a small subset of the entire related associations will be affected by the manual changes. Users would not notice the changes if they are accessing unaffected information spaces.

In NetAgent approach, trading agents guide a user to the relevant resource domains. An active user will register a new information as an association to a topic-specific trading agent. Other casual users will benefit by the new registration from the active user. If a new association is determined to be useful for a trading agent by the feedback of users, the association will be broadly used by trading agents. Thus the users don't have the overhead of direct information distribution, but still be able to find out new resource domains by active users' contributions.

4.3 Information System Diversity

During resource discovery, users have to choose the right information tool and switch to other tools whenever necessary. To relieve this load, existing tools provide a way to access resources of other tools. Current approaches are categorized by operation mapping and data mapping [5]. Operation mapping is done by using gateways to translate operations in a system into the ones in another system. Data mapping is done by collecting data from diverse resource domains by agreement protocols.

NetAgent approach is a combination of operation mapping and data mapping, that we call *context mapping*. NetAgent does not try to fully translate operation and data formats. Instead, information agents provide summary information from various information tools first. Users can determine whether to access a resource by the summary of the resource, rather than having to switch to a specific information tool. Thus NetAgent users only need to specify the resource, and agents perform actions on behalf of the user to access various information tools.

5 Conclusion

In existing indexing systems, the resources are mostly indexed by their attributes, and the organizational information of each information tool is lost. In addition, there has been no support to incorporate vast amount of users' knowledge on the index data. Our collaborative-indexing system is a new way to collect and distribute networked information resources. Since anyone can register a new indexing information, other users can locate the wanted resources without the same navigation overhead. Furthermore, since the indexing information is graded by weight values from users' feedback, the user can determine which resource domains are useful in the user's context. NetAgent is designed to be extensible so that a new information tool can be incorporated as information agents.

Future research areas include the followings:

- Access to heterogeneous information tools. More information tools and database systems can be incorporated as information agents.

- Replication and caching. Agent model would be extended for massive caching and replication of information resources since agent domain can be a unit for this purpose. An agent might monitor resource usage pattern of its domain and suggest how to cache and replicate its domain for efficient use.

- Monitoring in a user's context. An agent might be extended to do monitoring on behalf of users, and to report periodically.

References

[1] F. Anklesaria, M. McCahill, P. Lindner, D. Johnson, D. Torrey and B. Alberti, "The Internet Gopher Protocol (a distributed document search and retrieval protocol)," Request For Comments 1436, 1993.

[2] M. Adreessen, "NCSA Mosaic Technical Summary," Technical Report, NCSA, University of Illinois at Urbana-Champaign, 1993.

[3] T. Berners-Lee T, R. Calliau, J. Groff and B. Pollerman, "World-Wide Web: The Information Universe," Electronic Networking: Research, Applications and Policy, vol. 1, no. 2, pp 52-58, Westport CT, Meckler Publications, 1992.

[4] T. Berners-Lee, L. Masinter and M. McCahill, "Uniform Resource Locators," Request For Comments 1738, 1994.

[5] C.M. Bowman, P.B. Dansig and M.F. Schwartz, "Research Problems for Scal-

able Internet Resource Discovery," Proc. INET'93 Conference (San Francisco), 1993.

[6] B. Kahle and A. Medlar, "An Information System for Corporate Users: Wide Area Information Servers," ConneXions - The Interoperability Report, vol. 5, no. 11, pp 2-9, 1991.

[7] B. Kantor and P. Lapsley, "Network News Transfer Protocol – A Proposed Standard for the Stream-Based Transmission of News," Request For Comments 977, 1986.

[8] U. Manber and S. Wu, "GLIMPSE: A Tool to Search Through Entire File Systems," Proc. USENIX'94 Winter Conference (San Francisco, CA), pp 23-32, 1994.

[9] P. Mockapetris, "Domain Names Concepts and Facilities," Request For Comments 1034, 1987.

[10] T. Park and K. Chon, "Collaborative Indexing over Networked Information Resources by Distributed Agents," Distributed Systems Engineering Journal, pp. 362-374, January 1995.

[11] J. Postel and J. Reynolds, "File Transfer Protocol," RFC 959, October 1985.

[12] G. Salton, Automatic Text Processing, pp 275-312, Addison-Wesley Publishing Company Inc., 1989.

[13] L. Wall and R.L. Schwartz, Programming Perl, O'Reilly and Associates Inc., 1990.

Author Information

Taeha Park is a senior engineer at I.Net Technologies, Inc., Delta Bldg., Yoksam-dong 732-21, Kangnam-ku, Seoul 135-080, Korea. He has special interests on resource discovery tools and autonomous agents. He has been working for establishing KRNIC, the national network information center of Korea.

Kilnam Chon is a professor at Computer Science Depart, KAIST, Kusong-dong 373-1, Yusong-ku, Taejon 305-701, Korea. He has special interests in computer networking and distributed systems. He has been the chair of the Asia Pacific Networking Group and Korea Networking Council. He is also a co-chair of Coordinating Committee for Intercontinental Research Networking.