Context-sensitive Searching over Heterogeneous Internet Resources by Distributed Agents

Taeha Park and Kilnam Chon

taeha@cosmos.kaist.ac.kr

Department of Computer Science

Korea Advanced Institute of Science and Technology

373-1 Kusong-dong, Yusong-ku, Taejon, Republic of Korea

ABSTRACT

In the Internet, two resource discovery paradigms are in common use: organizing/browsing and searching. Searching over organizing/browsing tools becomes important in large information space because the large amount of resources can easily overwhelm users while browsing. Users run search tools to find out relevant starting points for browsing. Many search tools were developed for this purpose. Each existing tool supports searching over a single tool, and returns the same result for different users. To relieve these problems, we propose Agent Model as a global search framework capable of context-sensitive searching and tool-transparent searching. In our model, both information on information resources (i.e. meta-information) and resource discovery tools are modeled as active agents. This model has the benefit of incorporating multiple users and multiple tools and supporting meta-information sharing. By context-sensitive searching, a user can have his/her own view to the global search space. A group of users sharing a common agent view shares feedback of resource discovery, resulting a common view to the information space. By tool-transparent searching, a user do not need to repeatedly try several tools to find out relevant resources. On behalf of the user, the underlying agents communicate with various tools also modeled as agents. More resource discovery tools can be easily incorporated as information agents. A subset of the model described in this paper has been implemented in a prototype.

INTRODUCTION

Increasingly, the Internet is regarded as communication media where users share information resources, rather than the original purpose of connecting computing resources. There have been significant efforts on organizing information resources scattered over the network so that users can find out what they want in an easy way. Many tools and protocols for resource discovery are currently in use, and each tool provides a view to the global information space. These systems also provide global access to resources across differing computing platforms, protocols and data formats. As protocols evolve, gateways can allow global access to remain possible. As data formats evolve, format conversion programs can preserve global access.

Future information systems can be characterized by large scalability including enormous data volume, rapid growth in the user base and burgeoning information system diversity as described in Bowman (1993). With the large scale, dynamic information resources, two serious problems happen in current resource discovery approaches. Firstly, when a new resource is put to the network without a well-organized framework, users are not informed unless an announcement like electronic mail posting is done properly. Secondly, users are overwhelmed by excessive amount of information resources newly generated and changed everyday. These problems are summarized as *information distribution problem* and *information overload problem* as described in Lovstrand (1991).

We see these problems happen since existing resource discovery tools do not provide a way to share user's experience on information resources, although resources are organized and shared by users. The purpose of this paper is to provide a framework that allows users to search for information in context. In our model, a group of users with a common view to some parts of the search space shares feedback of resource discovery. One user's feedback about searches is used by future users to help locate relevant resources. By doing this, users can avoid many of the keyword aliasing problems found in a flat search space, and get more closely related matches to the search request.

The NetAgent is designed to support global search over huge amount of resources distributed over the Internet. By global search, users will simply need to give a query, and using the intermediate results given by the NetAgent, users can continue browsing or searching by existing resource discovery tools.

What makes global searches difficult is search over the entire Internet resources is not possible, or at least not cost-effective. The costs for global search include communication costs among servers and computational search costs in each server. To relieve this problem, NetAgent views the entire information space as a collection of agents where each agent focuses on the organization of resources in a specific context. Agents will keep information on other agents also. In this agent-based resource discovery, only a small set of agents are activated for a global search query given by the user. NetAgent narrows the scope for resource discovery on behalf of users.

RESOURCE DISCOVERY PARADIGM

Various resource discovery tools appeared to help users locate and retrieve information over the Internet. In the Internet, two resource discovery paradigms are in common use: organizing/browsing and searching. Searching over organizing/browsing tools becomes important in large information space because the large amount of resources can easily overwhelm users while browsing.

Organizing/Browsing Paradigm

Resource discovery by tools like Gopher (McCahill (1992)), WWW (Berners-Lee (1992)), and Alex (Cate (1992)) fall in this category. A user first determines where to start browsing information, and follows links or the directory structure the tool provides. These tools let the

user a way to traverse distributed resources transparently. Users usually have knowledge on useful starting points for browsing, since the Internet resources are not well organized by nature. Although one portion is well organized for a user, another user with a different view may have problems in understanding the structure.

Searching Paradigm

Research discovery by tools like WAIS (Kahle (1991)) fall in this category. A tool maintains an index to the target resources and perform an efficient searching operation for a given query.

Search-then-browse Paradigm

With the increase of resource volume, search tools were developed to deal with the scalability problems of organizing and browsing tools. In the search-then-browse paradigm, users first narrow down the information space by using search tools, and then start browsing with the search result. Tools in this category are:

Archie (Emtage (1992)) and Veronica (Foster (1993)) access information servers periodically to gather central index data. Archie serves a collection of file names found at anonymous FTP sites, as well as other information found at anonymous FTP archives. Veronica maintains an index of Gopher titles, and provides keyword searches of those titles. Both Archie and Veronica use simple attribute-based expression to access index entries gathered by periodic retrieval on information servers. Since each search tool was developed for an organizing and browsing tools, there is no single tool which support searches over diverse tools. Furthermore, current search-then-browse paradigm does not consider different user preferences, since searching tools maintain all index data in a single database.

DESIGN

The goals of NetAgent design are to provide (1) context-sensitive searching, (2) tool-transparent searching, and (3) global resource registration method. In this chapter, we will describe NetAgent design issues based on the agent model.

Agent Model

In this paper, we propose a framework for meta-information sharing by agents. It's up to each agents how they process meta-information intelligently to generate intelligent behavior. Because each agents analyze and process meta-information in its own way (or own viewpoint), the meta-information is incorporated in one agent for incremental behavior. For example, an agent which belongs to a person will try to reflect the person's flavor as much as possible. Agents update weight values on meta-information by user feedback.

In the agent model, sharing is done in two levels: information resources shared by resource discovery tools, and meta-information shared by agents (see Figure 1). Agents translate a user's query into a set of resource discovery operations.

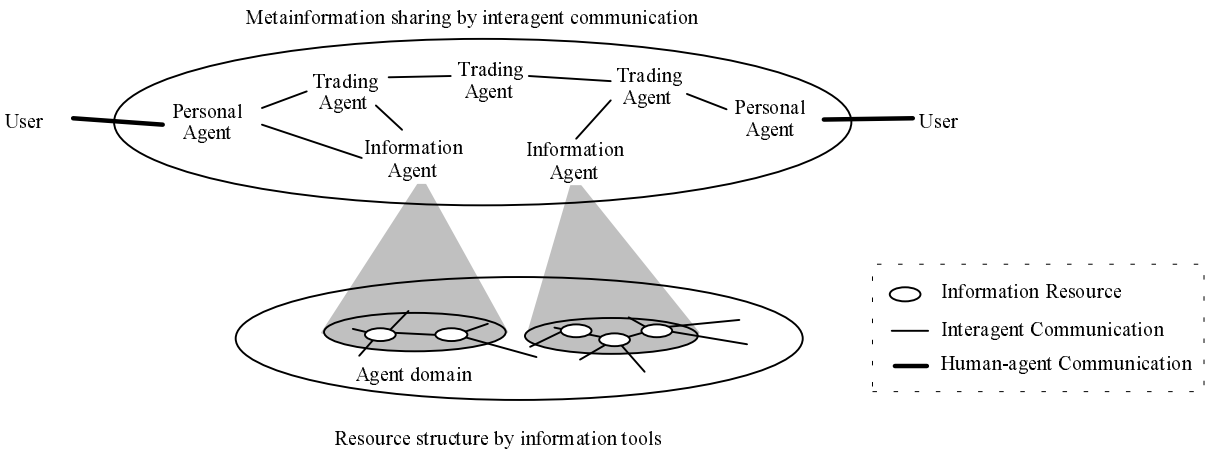

Figure 1. Agent Model

Agents behave adaptively to users' feedback by using meta-information, which is a set of

association and weight value pairs. Association is used to form a dynamic graph structure in the agent network for a topic. Agents may reorganize the graph structure by weight value changes from user's feedback. Meta-information itself is dynamic, and it is used for two purposes, i.e. (1) to associate an agent to other agents, or (2) to associate an agent to a set of resources. Association is a search key and access methods on the target agents. Agents determine where to propagate a query by the association. The number of messages between agents are controlled by the priority of weight values. Association may change as weight values change.

Each agent is eager to collect as much meta-information as possible. Since agents collect meta-information of some specific agent domain only, the same query to different agents may produce different set of resources. For example, two queries on weather information should result in different ways if the query is given from different countries.

A user can send a query to the network of agents by his/her personal agent. A query from a personal agent is propagated to a set of information agents via trading agents. Information agents perform actual retrieval of resources.

Agents collect index data from users, and manage themselves using the index data. Personal agents gather feedback information from user's actions during networked information retrieval to record the relevance of index data to a community-specific agent. NetAgent provides an clear registration mechanism that a user can register new information about resources to the network of agents.

Agents have capabilities of registration, trading, and information retrieval. Agents interact with human users and resource discovery tools as well as other agents according to the agent's role. There are three canonical agent types defined in the agent model by the agent's role.

A *personal agent* provides a human interface to the NetAgent system. The personal agent

communicates with its user and other agents. The personal agent keeps user's preference as weight values. The personal agents translate user's query (e.g. values in a form-based language) into agent messages. An *information agent* provides an interface to resource discovery tools. As resource discovery tools are passive and can not generate agent messages, information agents actively retrieve resources using tools, and reply back to other agents. A *trading agent* has the information on where to forward a query by community and topic.

Interagent Communication

Agents communicate by message passing. Agents exchange messages to register new information and to process queries cooperatively. Registration of a new association is propagated by REGISTER message until appropriate trading agents are found. The target trading agent is selected such that the new association most fits in the range of the agent's domain. An agent's domain is defined by its community and topic. A community is specified by a DNS name (Mockapetris (1987)) to represent the scope on the user community of an agent. For example, if the community of an agent is "KR (Korea)," the agent will serve all requests from community KR. Topic is a list of keywords an agent is proposed to support.

A query issued by a personal agent is processed in three phases as follow (see Figure 2).

- Summary lookup phase

  A query from a personal agent is propagated to information agents via trading agents. A trading agent forwards the query if the given query matches with its domain. An information agent will try to match the given query to resources in its domain, and will send REPORT messages containing summary information. Summary information is a brief description about a matching resource such as subject, date, creator name, and source tool, but not the actual content. On receiving summary information, the personal agent shows summary entries to the user so that the user can fully retrieve resources actually needed.

- Retrieval phase

  If a user selects a summary item, his/her personal agent sends a RETRIEVE request to the corresponding information agents. An 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 invoking the client program of the tool from the personal agent.

- 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. During this phase, an agent may change its association by its own policy, as will be described in the next section.

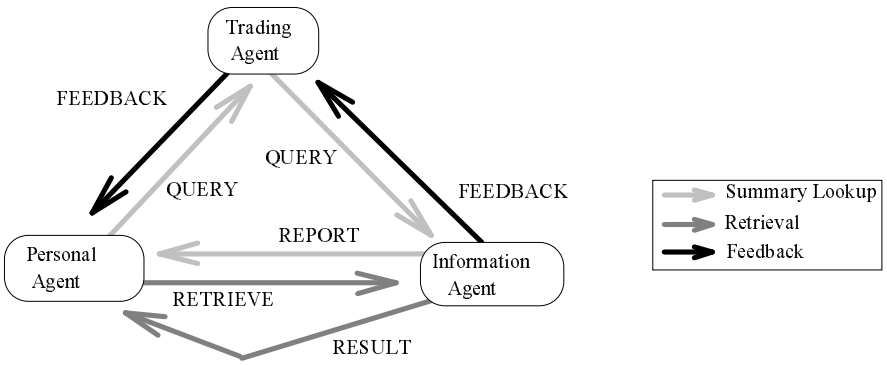

Figure 2. Three phase interagent communication for query processing

Management

Management issues are important in the agent model since there is no single unit to control the entire system. The goal of distributed management is two-fold. Firstly, agents reorganize themselves to be adaptive to user's actions. Secondly, number of agents activated from a query is restricted. There is a tradeoff between the overhead for invoking these agents and the benefit of increased number of REPORT's generated. Weight values are used by management operations for the first goal, and policy parameters are used for the second goal.

The algorithm on how the weight values will be used are dependent on each agent, i.e. determined by the policy of an agent. Conservative agents would prefer old agents as they are regarded to be more stable. Radical agents would prefer agents with high hit ratio although it is a new one. In general, "old and wise" associations are graded to be of high priority.

Weight Values

A trading agent keeps weight values on their associations for the adaptive behavior from user's feedback. To record the feedback from the requesting agents, agents keep two values on each association: *trial count* and *hit count*. Trial count is total numbers an association was referred. Hit count is the number an association was actually used for RETRIEVE by personal agents.

When a user selects an entry from the candidate resource list, the corresponding information agent sends the resource content to the personal agent, and generates a positive feedback to change weight values of all trading agents involved in associating the information agent. These changes are done by following the query path in reverse. As weight values of associations change during registration and query processing, agents may reorganize themselves.

A user can see the information space in his/her own view using agents since a personal agent will keep strong association with trading agents that had good matching results to the person's queries. Agents will compete to become more popular in the agent society with high hit ratio.

Policy Parameters

Policy parameters control each agent's behavior in two ways. Firstly, it limits the number of agents activated. If trading agents keep propagating queries to other trading agents instead of information agents, the number of queries will be ever increasing without doing real things useful. Secondly, agent associations are reorganized by policy parameters and feedback from other agents. If policy parameters are missing in a query, default values are provided by trading

agents. Each trading agent has policy parameters as shown in Table 1 and Table 2. When a query is being forwarded by a trading agent, policy parameters can be overridden by the trading agent.

Name	Type	Description
community	domain name	Preferred community of this query
maxagent	integer	Maximum number of agents that can be activated by this query
maxwait	integer	Maximum time an agent waits for reply messages

Table 1. Query policy parameters

Name	Type	Description
community	domain name	Perferred community of an agent
topic	string	List of search keys an agent can support
maxagent	integer	Maximum number of agents activated by a query
maxwait	integer	Maximum time an agent waits for reply messages
maxfanout	integer	Maximum number of associations in an agent
minfanout	integer	Minimum number of associations in an agent
maxidle	integer	Maximum idle time of an agent before expiration
minhit	integer	Minimum hit ratio of an association before expiration
refresh	integer	Refresh period to decrease hit count of each association

Table 2. Agent policy parameters

When a personal agent issues a query, it will wait for reports from information agents. The personal agent sets `maxwait` value in a query and simply ignores delayed reports exceeding the timeout. `maxagent` value is used to detect the termination. A personal agent will put `maxagent` value to prevent queries floating around agents. When a query is forwarded, the forwarding agent will decrease the `maxagent` value, and agents receiving queries of `maxagent` value zero will not try to generate additional queries.

A user can create an agent, and specify how the agent will behavior. An agent is expired when it is ignored by other agents longer than the period specified by `maxidle`. Hit count of an association is decreased periodically, and an association expires when its hit ratio becomes smaller than `minhit`. An agent dies when all of its expression entries are expired.

When there is no response from the requested agent, the requesting agent weakens the

corresponding association. Thus agents with frequent failure will be ignored gradually in the agent society. In addition, slow responses might be regarded as failure since agents can constrain the response time by `maxwait` parameters.

Agent Domain

Since an agent processing all kinds of input queries would not be actually handle any queries, it is necessary for an agent to have a domain, which constraints its capability. We use `topic` and `community` to define an agent's domain for a specific task. An agent will handle a query only if the agent's domain is a superset of the query domain. For example, if the topic of an agent is specified as "multimedia," only association containing the word multimedia will be allowed for the registration from other agents (e.g. multimedia mail, multimedia standards, etc).

An agent can be regarded as directory servers themselves for a specific area. There can be some generic agents that most agents like to forward queries that not classified. A generic agent would have better algorithms and more associations to other agents. Popular agents have rich meta-information on a specific area, possibly by monitoring and manual specification changes by expert users.

`community` denotes what community the agent is for mostly. A personal agent can have a `community` value and may put it to queries so that trading with community match will happen. Similarly, an organization-specific agent (e.g. a laboratory agent) will have knowledge on the organization's preference. A personal agent in the organization will commonly refer and provide shared feedback to the organization-specific agent.

Sample Scenario

Following is a sample scenario on using agents for global searching. This example shows how a user can locate a set of resources to efficiently start browsing.

1) If a user want to find out information about "video boards," he/she asks his/her personal agent to search for video or board or multimedia. The search key "multimedia" is added to give the agent a clue for related areas.

1) The personal agent searches its association to find out if there are matches on any of the search keys. If any associations are found, the personal agent forwards the query to the corresponding trading agents.

1) A trading agent finds an association on multimedia, so it replies to the personal agent to contact multimedia agent with search key "video board"

1) The personal agent send a query to multimedia agent. Multimedia agent searches its association to find if there are matches on "video" or "board." Since there are no matches, it replies with the default association, news agent.

1) The personal agent send a query to news agent with path parameter comp.multimedia. The news agent accesses a news server using Network News Transfer Protocol (NNTP) to search for news articles of subject including "video" and "board." On getting the search list, the news agent replies to the personal agent with the summary information list.

1) The personal agent display the summary information entries to the user.

The User can select one or more summary information entries to get the actual contents. By following the resources, users can start browsing. The selection of summary information entries are recorded by the personal agent, and the result is sent back to all trading agents participated in getting the summary information.

IMPLEMENTATION

The prototype system was implemented to support search-then-browse on a few resource discovery tools. Perl script language (Wall (1990)) was used for the prototype development. Currently the prototype system provides indexing to ftp servers and network news servers. By adding information agents for other resource discovery tools, the capability of the prototype system can be extended incrementally.

Context-sensitive Searching by Trading Agents

The role of a trading agent is to get `QUERY` messages, to translate them, and to generate QUERY messages. Query expression is a list of attribute name and value pairs. Possible attribute names are shown in Table 3.

Name	Type	Description
community	domain name	Preferred community name
subject	string	Title or subject of the resource
content	string	Partial match for the string in resource content
date	yy.mm.dd	Date appear in the resource
author	string	Creator of the 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	Name of the file for the resource
filesize	integer	Size of the resource in bytes
filedate	yy.mm.dd	Date of the file

Table 3. Attributes for a query expression

Behavior of each trading agent is represented by agent specification. The agent specification is composed of policy specification and trading specification. The policy specification is a list of policy parameter values. Trading specification is a list of meta-information of the following form:

```
search-key hit-count trial-count association
```

`search-key` determines the keyword that matches the entry. `hit-count` and

`trial-cont` shows the history how many times the entry was referred and actually used for RETRIEVE. When a query is given to an agent, the agent tries to match `search-key` with the given query. `association` represents how to forward the query. It is composed of an agent identifier and optional path values to refine the agent access method. association is in the form of Uniform Resource Locator (Berners-Lee (1993)) so that existing tools can be also incorporated to the NetAgent system.

Association represents how to access an agent with parameters. As the status of an agent is stored in agent specification that users can understand, the owner of an agent can change the specification to add his/her heuristic knowledge to the agent.

User Interface by Personal Agents

Personal agents interact with users to provide access to the NetAgent system. A personal agent can be implemented in several ways such as a command language or a form-based language. In our prototype, personal agents are implemented as an active node in a WWW client program as shown in Figure 3. Personal agents let users to interact with other agents. Personal agents generate feedback messages for dynamic changes on trading agents that were activated for a query.

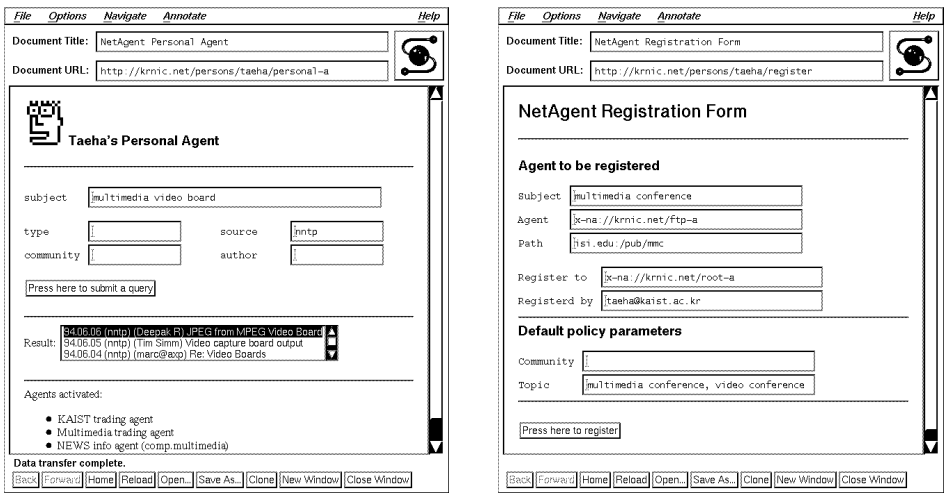

Figure 3 An example of the NetAgent user interface in a WWW client window

By the weight values of the associations from a personal agent, a user can reflect his/her preference on resource discovery without extra efforts. NetAgent is not a replacement of the existing resource discovery tools. Rather, the NetAgent user interface provides summary information and switches to a client of a resource discovery tool when the user want to see the resource. By doing this, NetAgent is seamlessly integrated with the existing tools. Users can navigate the information space of a resource discovery tool while keeping a global view on the entire information space by NetAgent. Although NetAgent does not replace existing resource discovery tools, it can emulate Archie or Veronica system by simply specifying the *source* to a single protocol (FTP or Gopher in this case). Since NetAgent does not keep index data, the performance should be lower compared to the dedicated search tools.

Tool-transparet Searching by Information Agents

An information agent supports resource retrieval in a tool-specific way. An information agent may translate an agent message into tool-specific operations, or may access resources directly without the mediation of a resource discovery tool. A number of information agents exist such as the FTP agent, the news agent, the WAIS agent, the WWW agent, and the Gopher agent.

Information agents are designed to make use of information inherent to resource discovery tools as much as possible. 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. The WWW agent would navigate the hypermedia space by following a limited number of links from the starting node for full text searches.

ANALYSIS

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.

(1) Information overload problem

The information overload problem happens when too many information resources are given to the user as not organized by user's context. Current resource discovery tools and directory systems support searches in general context, but do not support searches in context of a person or a community. In existing approaches, users do flat searches against the global information space, which is not well organized by nature. As a result, the large volume of irrelavant data may overwhelm the user. In NetAgent, users can get response from a community-specific or topic-specific trading agents, focusing the view in user's context.

(2) Information distribution problem

In existing approaches, changes on the information space are distributed directly or indirectly to the users.

In direct information distribution, users send a message to a common interested group identified by mailing lists and news groups. The problem of direct information distribution is that reaching to the right set of users is not always possible. If it is distributed to broad range of users, some users will be receiving irrelevant information, and the problem similar to the classical junk mail problem happens. If it is done to too narrow range of users, the user who really need the information would not be able to get the information. Another problem is that individual users have to memorize the change of the global information space, or have to configure a resource discovery tool manually to sustain his/her personal view in a tool.

In indirect information distribution, the changes are done in the global information space. Users search or navigate the information space and notice the change when the user actually retrieve relevant information. The problem is that manual changes to all relevant parts of the global information space is not possible. Only a small subset of the entire related association will be

affected by the change. Users would not notice the changes if they are accessing still unaffected information subspace.

In NetAgent approach, the changes are propagated automatically to community-specific and topic-specific trading agents. 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 the trading agent.

(3) Information system diversity

During resource discovery, users have the load to choose a right resource discovery tool and switch to other tools whenever necessary. To relieve this load, existing tools provide a way to access resources of other tools.

Bowman (1993) categorizes current approaches by operation mapping and data mapping. Operation mapping is done by gateways to translate operations from one system into operations in another system. Data mapping is done by collecting data from diverse sources 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 first, and if users want to get full content of the resource, then information agents rely on existing tools to transfer the content In NetAgent approach, heterogeneous tools are seamlessly integrated with the yellow pages services, while not losing the functionality of the existing tools.

NetAgent users do not need to worry about what tools to use. Users only need to concentrate on the topic the user wants to get. In existing approaches, if a user wants to find a specific set of information by searching, the user will need to try searches on various information servers using several resource discovery tools such as WAIS, Veronica, archie, and so on. Agents do a sequence of actions on behalf of the user. In this context, agents are similar to a macro processor that expands a given search query to a set of search operations on several resource

discovery tools.

The drawback of NetAgent is that the searching over diverse tools is less efficient than the dedicated search tools. This problem can be avoided by creating an information agent for each dedicated search tool.

CONCLUSION

This paper has described NetAgent system that provides global search capability over various resource discovery tools for search-then-browse paradigm. In our approach, index data are organized by the association of distributed agents. The organization of agents is changed dynamically to be adaptive to the user's feedback during resource discovery. NetAgent provides user's explicit registration method. By NetAgent registration method, meta-information on information resources can be propagated to the global information space. NetAgent is designed to be extensible so that a new resource discovery tool can be incorporated as information agents.

We believe that future information systems will allow users to search for information in context, and our model based on the agent provides a solution for global search in context. Further studies are required to extend the agent model to consider caching and replication issues. 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 access pattern to its domain and suggest how to do caching and replication for efficient use.

Functionally, we would like to enhance NetAgent to support browsing and monitoring in user's context. An agent can enhance browsing in two ways. First, the agent might provide a view to the information space by filtering out unnecessary links and adding shortcut links. This could be done by configuring resource discovery tools using agents. Second, the agent might recommend which path the user will follow. In conventional resource discovery tools, monitoring is usually done by users. Users repeatedly check information sources to find out if

there is any new information interesting. An agent might be configured to do monitoring on behalf of users, and to report periodically. Agents of this functionality are similar to the agents in network management systems and office information systems.

REFERENCES

Anklesaria, F., McCahill, M., Lindner, P., Johnson, D., Torrey, D. & Alberti, B. (1993) "The Internet Gopher Protocol (a distributed document search and retrieval protocol)", *Internet RFC1436*, March.

Berners-Lee, T., Calliau, R., Groff, J. & Pollerman, B. (1992) "World-Wide Web: The Information Universe," *Electronic Networking: Research, Applications and Policy*, Vol. 2, No. 1, Meckler Publications, Westport CT.

Berners-Lee, T., Calliau, R. & Pollerman, B. (1993) "Uniform Resource Locators," *Internet Draft,* IETF URI Working Group, October.

Bowman, C.M., Dansig, P.B. & Schwartz, M.F. (1993) "Research Problems for Scalable Internet Resource Discovery," *Technical Report CU-CS-643-93*, University of Colorado, March.

Cate, V., (1992), "Alex - A Global File System," Proc. USENIX File Systems Workshop, May.

CCITT/ISO (1988) "The Directory, Part 1: Overview of Concepts, Models and Services," *ISO DIS 9594-1*, December.

Danzig, P.B., Li, S.H. & Obraczka, K. (1992) "Distributed Indexing and Autonomous Internet Services," *Technical Report USC-TR-92-519*, University of Southern California.

Emtage, A. & Deutsch, P. (1992) "Archie - An Electronic Directory Service for the Internet," *Proc. USENIX Winter Conference*, January.

Foster, S. & Barrie, F. (1993) "Veronica Frequently Asked Questions," *Article posted to USENET newsgroup comp.infosystems.gopher*, August.

Harrenstien, K., Stahl, M. & Feinler, E. (1985) "NICNAME/WHOIS," *Internet RFC954*, October.

Kahle, B. & Medlar, A. (1991) "An Information System for Corporate Users: Wide Area Information Servers," *ConneXions - The Interoperability Report*, November, Vol. 5, No. 11.

Lovstrand, L. (1991) "Being Selectively Aware with the Khronika System," *Proc. ECSCW'91 Computer-Supported Cooperative Work*, September.

Mockapetris, P. (1987) "Domain Names Concepts and Facilities," *Internet RFC1034*, USC/Information Sciences Institute, November.

Schwartz, M.F. & Pu, C. (1993) "Applying an Information Gathering Architecture to Netfind: A White Pages Tool for a Changing and Growing Internet," *Technical Report CU-CS-656-93*, University of Colorado, December.

Wall, L. & Schwartz R.L. (1990) *Programming Perl,* O'Reilly and Associates Inc.