

InfoSleuth Agents: The Next Generation of Active Objects

Darrell Woelk and Michael Huhns
Microelectronics and Computer Technology Corporation (MCC)
woelk@mcc.com, huhns@mcc.com

Christine Tomlinson
Express Star Systems, Inc.
tomlic@xstar.com

1. Requirements for a New Class of Applications

A new class of applications is evolving thanks to ongoing advancements in computer hardware and software. These applications have the following attributes:

- The applications solve a specific business problem by providing a user with seamless interaction with remote information resources, remote application resources, and remote human resources
- The identities of the resources to be used are mostly unknown at the time that the application is developed
- The pattern of interaction (workflow) among the resources is a critical part of the application, but the pattern might be unknown at the time the application is developed and might vary over time

The development of these new applications requires improved programming languages and improved system services. These should not be viewed as alternatives to such capabilities as OMG CORBA and OSF DCE, but rather as advanced features that are implemented at a higher level of abstraction and are useful across multiple heterogeneous distributed computing environments.

Since each application executes as a set of geographically distributed parts, a distributed object architecture is required. It is very likely that the objects taking part in the application will have been previously developed in various languages and will execute on various hardware platforms. A simple and powerful paradigm is needed for communications among these heterogeneous objects. Due to the distributed nature of the application, an object may not always be available when it is needed. For example, an object executing on a PDA may be out of physical communications with the rest of the application.

Since the identities of the resources are not known when the application is developed, there must be an infrastructure to enable the discovery of pertinent objects. Once an object has been discovered, the infrastructure must facilitate the establishment of constructive communication between the new object and existing objects in the application.

Since the pattern of interaction among the objects is a critical part of the application and may vary over time, it is important that this pattern (workflow) be explicitly represented and available to both the application and the user. When an object has been discovered to be relevant to an application, the language for interaction and the pattern of interaction with the object must be determined. This interaction becomes part of the larger set of object interactions that make up the application. The objects collaborate with each other to carry out the application task.

2. Agent-Based Software

A wide variety of software programs have been developed recently that are classified as software agents [1,2]. For our purposes here, we will define two major classifications of software agents. The first type of agent-based software is focused on the interaction between a user and a computer. These agents monitor the user interaction, initiate communication with the user, and may proactively initiate tasks on behalf of the user. The agent becomes more effective at assisting the user as it learns the user's habits and preferences.

Examples of this research by MIT, AT&T Bell Labs, CMU, and others can be found in [1].

The second type of agent-based software is focused more on the interaction among computing agents. The basic issues addressed are the interoperability among geographically distributed agents executing on heterogeneous hardware platforms. The choice of a communication language for these agents can be based on two approaches. The procedural scripting approach takes the view that the way to cause the execution of a remote task is to send a procedural script for interpreted execution at the remote site. Examples of this approach are General Magic Telescript [2] and Tcl [2]. The declarative approach takes the view that only a declarative description of the task should be sent to the remote site. An example of this approach is ACL [3].

The MCC InfoSleuth consortial project is conducting research in the area of agent-based software and is specifically developing the second type of agents described above. Both procedural scripting agents and declarative agents are under development. In particular, an agent-based infrastructure is being developed to address the requirements for distributed applications described in the first section of this article.

The InfoSleuth research is building on the base technology developed in the recently completed MCC Carnot research project. Before describing the InfoSleuth project in more detail, we will first review the architecture and accomplishments of the Carnot project.

3. MCC Carnot Project

The Carnot project [4] was initiated in 1990 with the goal of addressing the problem of logically unifying physically-distributed, enterprise-wide, heterogeneous information. A prototype has been implemented that provides services for enterprise modeling and model integration to create an enterprise-wide view, semantic expansion of queries on the view to queries on individual resources, and interresource consistency management. Carnot also includes technology for 3D visualization of large information spaces, knowledge discovery in databases, and software application design recovery. The Carnot prototype software has been used by the sponsors of the Carnot project to develop a number of applications. These applications have included workflow management, heterogeneous database access, knowledge discovery in large databases, and integrated access to both text databases and structured databases from a single initial query.

The implementation of the Carnot system has required unique advances in two technology areas. First, innovative techniques for knowledge representation have been developed to capture and maintain an enterprise model and to map operations between an enterprise model and the physical databases. Figure 1 illustrates the Carnot environment. The MIST tool, shown in the lower right hand corner, is a graphical based interactive tool that assists the user in the integration of each database with the common ontology, which serves as the enterprise model. The integration step results in the generation of articulation axioms that specify mappings between databases and the common model. These articulation axioms are stored in the Carnot repository.

The second unique advance is a flexible, dynamic, distributed processing environment consisting of intelligent, autonomous computing agents that enable the retrieval of enterprise information and control enterprise processes. For example, a client application on the left side of Figure 1 may specify a query with reference to the common model. An intelligent Carnot agent will receive the query, consult the repository to discover which databases should be accessed, and create other agents to execute this access. Each of these agents also is given the mappings necessary to translate information from an individual database into the correct format [5]. Client applications access the Carnot environment through either a PC Windows ODBC interface or a Unix X/Open SQL CLI interface. Various local corporate networks are utilized to connect hardware systems.

The Carnot technology is currently being commercialized by Express Star Systems, Inc. Products will be available in late 1995.

4. MCC InfoSleuth Project

The MCC InfoSleuth project [6] was initiated in January, 1995 and will extend the research agenda started

in the Carnot project. The InfoSleuth project will develop and demonstrate an agent-based architecture that will simplify the creation and maintenance of distributed applications. The general architecture of InfoSleuth is shown in Figure 2.

4.1 Collaborating Agents

The lowest layer consists of agents that collaborate to perform a task on behalf of a user. As mentioned earlier, there are two types of InfoSleuth agents. The procedurally scripted agents have been developed using the Rosette concurrent object language. The declarative agents have been developed using the RAD rule based language. By developing both types of agents, the InfoSleuth research staff is able to run practical experiments to determine the optimal conditions under which each type of agent should be deployed. We are also investigating hybrid agent implementations with optimal features based on the type of application.

Rosette Agents (Procedurally Scripted)

Rosette [7] is a high performance interpreter for the Actor model [8] which has been enhanced with object-oriented mechanisms for inheritance and reflection. It has been under development at MCC since 1988. With Rosette, the object-oriented programming model and the Actor concurrent execution model are combined to simplify the development of autonomous, distributed agents. The Actor model is ideal for the development of agents because:

- The basic semantics of the Actor model of computation is asynchronous communication among concurrently executing entities termed actors.
- The Actor model includes a very simple and powerful model for synchronizing and controlling interference among concurrently executing threads of control.

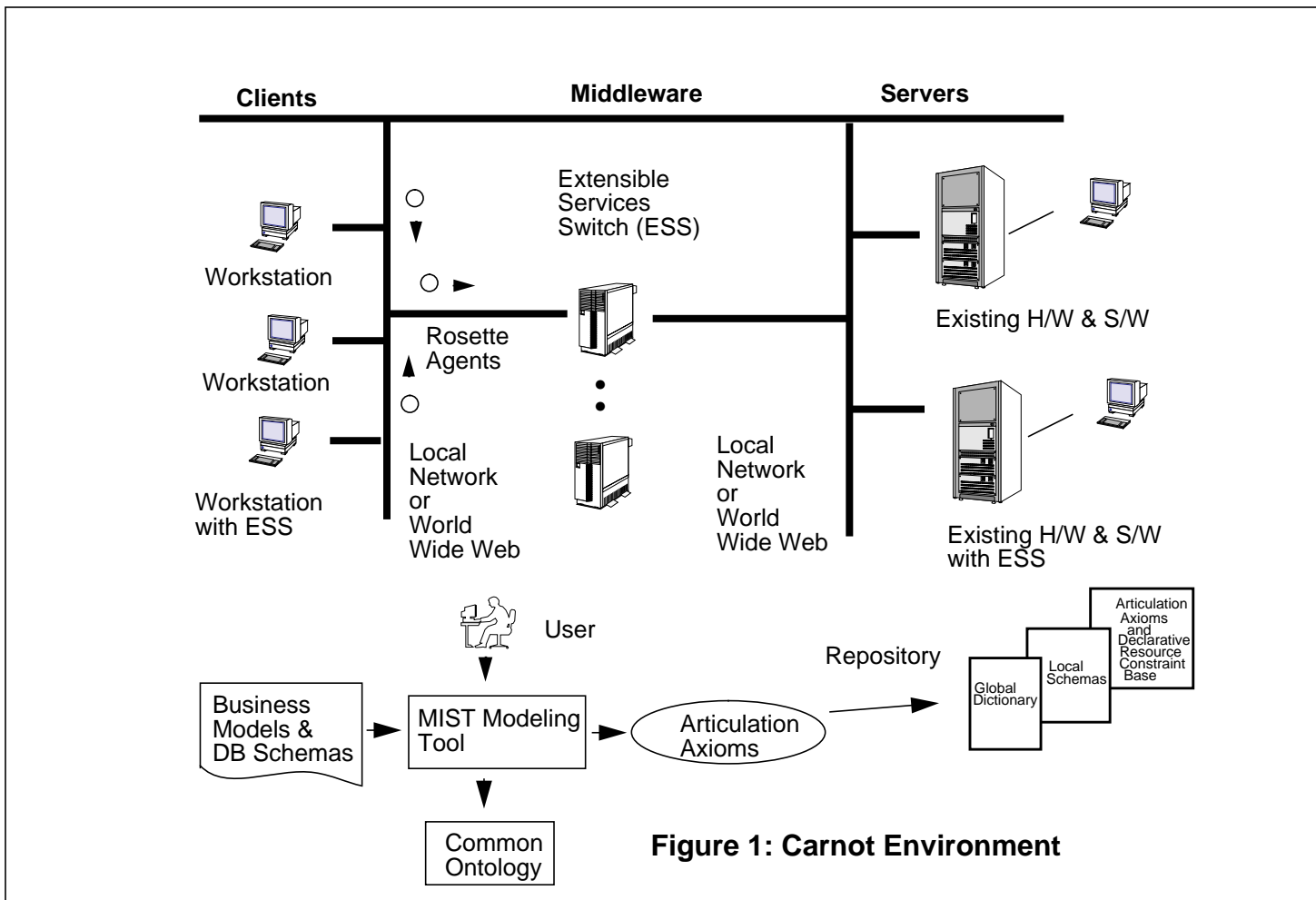


Figure 1: Carnot Environment

Most scripting languages are sequential languages with some process ideas grafted on. Rosette, on the other hand, is an inherently concurrent language framework that makes it natural and simple to express actions that are inherently distributed and can be evaluated concurrently. If a set of operations are not explicitly sequentialized in the Rosette language, they will be executed concurrently by the interpreter.

Rosette also incorporates sophisticated support for the dynamic definition of foreign function calls to C and C++. It then provides typechecked access to C and C++ procedures that have been compiled and linked dynamically into the Rosette runtime environment. These features make it possible to rapidly integrate new facilities into Rosette. Rosette executes as a single process on Unix platforms, with each actor executing as an ultra-lightweight thread within the single Unix process.

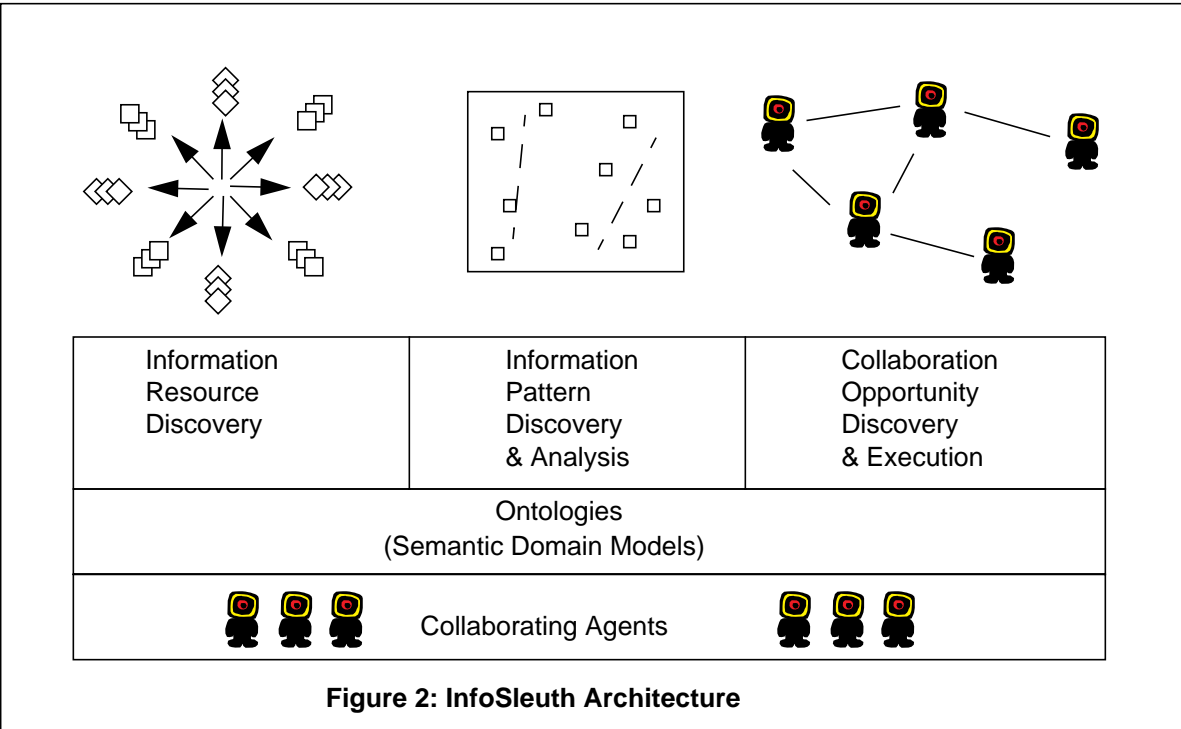
Rosette uses a form of remote evaluation to cause the execution of a script on a remote computing system. Most of the distributed coordination among Rosette agents, however, is through a communication mechanism termed a Tree Space. A Tree Space provides a set of functions similar to the Linda Tuple Space [9], and uses an addressing scheme that is similar to the directory system in the Unix operating system. It can be used for pattern directed retrieval of messages by anonymous Rosette agents, since an attempt to retrieve a message via a pattern for which there is no match leads to the requestor to block until a message is deposited by another Rosette agent.

RAD Agents (Declarative Rule-Based)

The InfoSleuth project will also utilize rule-based agents that communicate through declarative messages. These agents are based on the Reasoning Architecture for Design (RAD) agents developed at MCC and enhanced in the Carnot project [10].

RAD agents have a high-performance forward and backward reasoning engine that uses Warren Abstract Machine technology, a frame system integrated into a typed unification algorithm with multiple inheritance, a distributed truth-maintenance system, and a contradiction resolution mechanism. The actual communication among the RAD agents is enabled by the interaction of Rosette agents within the distributed Rosette Tree Space.

The communication among RAD agents uses a speech-act foundation based on a proprietary set of



performatives. These are being replaced with a standard set of performatives based on KQML. The KQML performatives will be forwarded through the Rosette Tree Space. For example, a RAD agent's proprietary distributed truth-maintenance system already performs the default reasoning necessary to implement the untell, flush, and trash performatives.

The design of the RAD agents is based on our observations of how a user interacts with a single, conventional knowledge base. A user can query and make assertions to the knowledge base, to which the system responds with either an answer or an acknowledgment, as appropriate. As a side effect of forward or backward inferencing, the system can independently inform or make a request of a user, who must answer all requests. Interactions can thus be either user-initiated or system-initiated.

Similarly, RAD agents communicate at a fundamental level by exchanging messages, specifically, queries or assertions. The agents can be either passive or active, i.e., they can either respond to questions and commands from another agent or initiate dialogs with another agent. A problem is solved by a RAD agent in the following way. At any given time there will be a number of computational agents, each having its own domain specific knowledge, and a number of users, each assisted by an interface. A user or other agent gives a problem to one of these agents, which

- solves the problem itself, using local knowledge and internal reasoning mechanisms, or
- decomposes the problem into subproblems, distributes each one of these subproblems to an appropriate agent, and then integrates the solutions returned by these agents. This cooperation enables the solution of the problem.

4.2 Ontologies (Semantic Domain Models)

The ontologies layer of the InfoSleuth architecture in Figure 2 provides collaborating agents with a common vocabulary and a common semantic model for interaction in some application domain. For example, a set of collaborating InfoSleuth agents in a medical application may have access to a healthcare ontology. To revisit the example used earlier in the review of the Carnot architecture, there might be an agent A that has been provided with rules that represent mappings between the healthcare ontology and two different hospital legacy databases. Agent B can request information from agent A in terms of the healthcare ontology. Agent A can map the query into the proper subqueries to access the two legacy databases and create agents to execute the access and map the resulting data into the proper format to be returned to agent B.

There are two important points that need to be made with respect to the two lower layers of the InfoSleuth architecture. First, the ontologies layer itself will be implemented as a set of collaborating agents, thus providing scalability and extensibility of the ontology itself. Second, as in the Carnot architecture, applications may be developed that do not require the use of an ontology. In these applications, either all of the individual agents share a common internal structure for information or the individual agents are programmed to translate from their internal information structure to the internal information structure of another specific agent.

The InfoSleuth project will investigate three categories of applications as shown in the top layer of Figure 2: Information Resource Discovery, Information Pattern Discovery & Analysis, and Collaboration Opportunity Discovery & Execution. The common theme of each of these application categories is that there is a discovery process that requires (or allows) interaction among human users and computing agents. Each discovery process itself must be explicitly represented so that it can be documented, modified, reasoned about, and possibly aggregated with other discovery processes.

4.3 Information Resource Discovery

Information Resource Discovery applications have increased in visibility and importance with the explosive growth of the Internet and the World Wide Web. A distinguishing characteristic of these applications is that new information sources are constantly being added, and there is local autonomy, i.e., no network-wide control of the registration of new information sources and their content.

In this type of environment, traditional techniques for expressing and optimizing database queries are

inadequate because of the rapidly changing schema information and the fuzzy nature of the queries. Text search techniques and interactive navigation techniques are also inadequate because of the immense size and (potentially) remote distribution of the available information.

Research projects and commercial products related to the World Wide Web have focused on the development of spiders that traverse the URL references in World Wide Web pages and build a keyword index that can be used to find pages of interest. Examples of spiders are [11,12]. The Harvest system at the University of Colorado [13] and the WebAnts project at Carnegie Mellon University [14] are developing systems that utilize multiple coordinated spiders to traverse parts of the Web and merge the results into single or multiple indexes. The InfoSleuth project is investigating the use of InfoSleuth collaborating agents to coordinate the traversal of the Web to build the index and to coordinate the use of multiple indexes for responding to user queries.

The InfoSleuth project is also investigating techniques [15] to improve the indexing of information resources in the World Wide Web by using the ontologies layer of the InfoSleuth architecture.

- Providers of information will advertise its availability by relating the information to the ontology.
- Clients that are searching for information will discover the availability of potentially useful advertised information. Autonomous InfoSleuth agents will be deployed to search for information, remaining active to monitor for the addition of pertinent new information.
- InfoSleuth agents will cooperate to integrate the information from the various resources.

The use of a common ontology by both providers of information and clients searching for information will enable an InfoSleuth agent for a information provider to search for clients that might be interested in the information.

4.4 Information Pattern Discovery & Analysis

Once a pertinent information resource has been discovered in the network, there is typically a phase in which the contents of the information resource are analyzed in more depth. In the case of World Wide Web pages, this phase consists mostly of browsing pages and possibly creating a new hypermedia document with references to existing Web pages.

However, agents may also assist in the discovery of patterns in the relationships among elements of information in an information resource. This is particularly true when the information resource is a structured database (such as a relational database, object-oriented database, etc.), but analysis of text, audio, image, and video databases is also possible.

This pattern discovery and analysis phase can also operate across multiple discovered information resources. It should be noted that using Carnot for enterprise information integration is a special case of information resource discovery followed immediately by information pattern discovery and analysis.

The InfoSleuth research is using the Logic Data Language (LDL++) [16], developed at MCC and enhanced in the MCC Carnot project, as a language for pattern discovery and analysis. LDL++ is a deductive database system that integrates concepts of logic programming, relational database technology, and object-oriented programming. LDL++ provides rule-based interaction with and integration of existing databases.

4.5 Collaboration Opportunity Discovery & Execution

Agents can collaborate on tasks other than discovery and analysis of information. The InfoSleuth project will also investigate a broad range of other applications and develop prototypes for a few applications. These other applications can be characterized by the requirement to dynamically discover the requirement for collaboration and then dynamically discover the pattern of collaboration that should be followed.

5. An Example Application using InfoSleuth Agents

An example application using InfoSleuth agents is illustrated in Figure 3. A marketing manager needs to determine the optimal geographical location for test marketing a new product. To make this decision, she

will need the following types of information:

- Have other marketing managers in the company test marketed similar products?
- Have competitors test marketed similar products?
- Have similar products sold well?
- What would be the best geographical location for a test market?

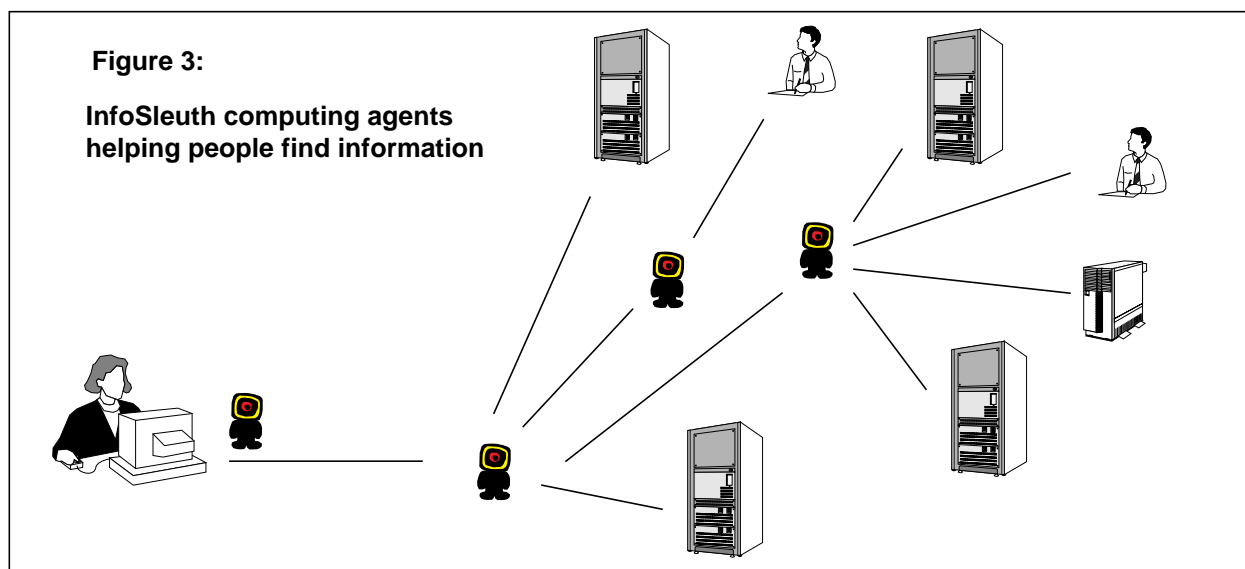
In the past, the marketing manager might have conferred with a marketing analyst who would access information in a company database. But the analyst would also depend heavily on interaction with others to find needed market information. It was not necessary for the analyst to know of the existence of the databases containing the information and how to physically access the databases. The analyst only had to know someone who knew how to find it.

Each of the other people helping in this search for information had some specialized knowledge of an area, including knowledge of the specialities of other people. Two people could adapt their interactions to handle slight variations in the way a questions is asked or to handle confusion on the meaning of information that is found.

Today, as computers and communication among computers have become more ubiquitous, more of the information that the marketing analyst is seeking is likely to be accessible directly online. Finding this information remains difficult, however, since the typical search techniques available today on the Internet and the World Wide Web are static links and syntactic keyword indexes. People still remain a part of this process, but they can be augmented and represented by InfoSleuth agents as shown in Figure 3.

InfoSleuth will provide software tools that assist people in expressing their knowledge about information sources in a clear and concise manner. An InfoSleuth agent can represent the marketing analyst who is assisting the marketing manager. The marketing manager requests information from the agent through a natural interaction. The agent then independently sets out to find the information, possibly returning to the manager for clarification or to report status. The agent also interacts with other InfoSleuth agents to find the proper sources of information, to retrieve the information from those sources, and to present the information to the manager in an understandable manner. The marketing manager can request that the InfoSleuth marketing analyst agent continue to monitor information sources to identify changes in the requested information that might represent trends in the market.

The InfoSleuth marketing analyst agent might contact an InfoSleuth real estate investment agent to better



understand changes in the real estate market in various geographical locations. The real estate investment agent may access information in a number of different online databases. Just as with a human agent, the agent will receive compensation for its assistance in addition to the compensation to the owner of the online database. A good real estate investment agent will have many customers and owners of databases will seek to have their databases accessed by that agent.

6. Summary

The emergence of a ubiquitous world wide communication infrastructure is enabling a new class of applications that are characterized by the requirement for intelligent collaboration among independently developed applications executing in a dynamically expanding, geographically distributed environment. The MCC InfoSleuth research project is developing agent-based infrastructure software that will meet these requirements. The software agents are being deployed and tested in real-world applications by MCC and the sponsors of the InfoSleuth project.

References

- [1] Riecken, D. "Introduction", *Communications of the ACM (Special Issue - Intelligent Agents)*, Vol. 37, No. 7, July, 1994, pp. 18-21.
- [2] Indermaur, K. "Baby Steps", *Byte*, March, 1995, pp. 97-104.
- [3] Genesereth, M. and S. Ketchpel, "Software Agents", *Communications of the ACM*, Vol. 37, No. 7, July, 1994, pp. 48-53.
- [4] <http://www.mcc.com/projects/carnot>
- [5] Woelk, D., P. Cannata, M. Huhns, W. Shen, and C. Tomlinson. "Using Carnot for Enterprise Information Integration". Second International Conference on Parallel and Distributed Information Systems. January 1993. pp. 133-136.
- [6] <http://www.mcc.com/projects/infosleuth>
- [7] Tomlinson, C., P. Attie, P. Cannata, A. Sheth, M. Singh, and D. Woelk, "Workflow Support in Carnot", *Data Engineering Bulletin*, Vol. 16, No. 2, June, 1993, pp. 33-36.
- [8] Agha, G. "Concurrent Object-Oriented Programming", *Communications of the ACM*, September 1990, pp. 125-141.
- [9] Carriero, N. and D. Gelernter. "Linda in Context", *Communications of the ACM*, Vol. 32, No. 4, 1989.
- [10] Huhns, M. and D. Bridgeland, "Multiagent Trutch Maintenance," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 21, No. 6, November/December 1991, pp. 1437-1445.
- [11] "Lycos", <http://lycos.cs.cmu.edu>
- [12] "WebCrawler", <http://webcrawler.cs.washington.edu/WebCrawler/WebQuery.html>
- [13] "Harvest", <http://harvest.cs.colorado.edu>
- [14] "WebAnts", <http://thule.mt.cs.cmu.edu:8001/webants>
- [15] Woelk, D. and C. Tomlinson, "InfoSleuth: Networked Exploitation of Information using Semantic Agents", *COMPCON*, March, 1995.
- [16] Ong, K., N. Arni, C. Tomlinson, Unnikrishnan, and D. Woelk, "A Deductive Database Solution to Intelligent Information Retrieval from Legacy Databases", *Proc. of 4th International Conference on Database Systems for Advanced Applications*, April, 1995.