Developing InfoSleuth Agents Using Rosette: An Actor Based Language

Darrell Woelk
Microeclectronics and Computer Technology Corporation (MCC)
3500 Balcones Center Dr.
Austin, Texas 78759

1. InfoSleuth Architecture

The MCC InfoSleuth project [1] [2] is developing an architecture for and prototypes of an agent infrastructure for a variety of applications. The InfoSleuth research is building on the base technology developed in the recently completed MCC Carnot research project. The Carnot project [3] [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. 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 InfoSleuth project will investigate the use of Carnot technology in a more dynamically changing environment, such as the Internet, where new information sources are constantly being added and for which there is no formal control of the registration of new information sources. 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. InfoSleuth will build on Carnot semantic modeling capabilities to enable "deep" descriptions of available information sources. InfoSleuth will deploy semantic agents to carry out distributed, coordinated, self-adapting search algorithms. The general architecture of InfoSleuth is shown in Figure 1.

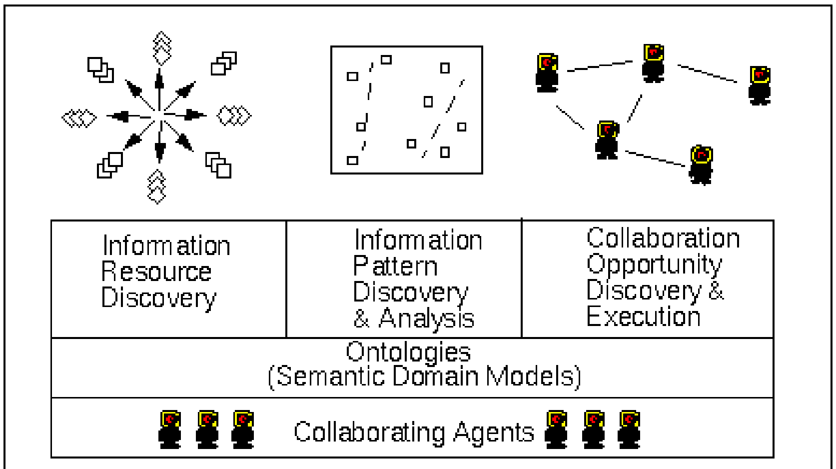

Figure 1: InfoSleuth Architecture

1.1 Collaborating Agents

The lowest layer consists of agents that collaborate to perform a task on behalf of a user. There are two types of InfoSleuth agents. Rosette agents are based on the Actor model and are scripted with a procedural scripting language. They will be discussed in more detail in the next section. The other types of agents are 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 [5]. Selected features of the RAD agents are being re-implemented using CLIPS [6].

1.2 Ontologies (Semantic Domain Models)

The ontologies layer of the InfoSleuth architecture in Figure 1 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. 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, 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.

1.3 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.

The InfoSleuth project is investigating techniques [7] 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, re- maining 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.

1.4 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.

1.5 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.

2. Implementation of Rosette Agents

Rosette [8] is a high performance interpreter for the Actor model [9] 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 execut- ing entities termed actors.
- The Actor model includes a very simple and powerful model for synchronizing and controlling interference among concurrently executing threads of control.

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 [10], 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.

3. Developing a World Wide Web Application Using Rosette Agents

A Carnot-based application [11], previously developed for use at Eastman Chemical Company in Kingsport, Tennessee has been modified to support remote access via the World Wide Web. The original application provided a forms-based interface and a natural language interface to multiple Eastman Chemical Company databases. These Digital Equipment RDB databases contain historical information on chemical experiments and the manufacturing of chemical compounds.

The interfaces were developed using C++ and Motif. The forms-based interface was developed using LDL++, a deductive database system based on the integration of logic programming with relational database technology [12]. LDL++ generates SQL operations that are executed by the RDB DBMS. The natural language interface was developed using the MCC Knowledge Based Natural Language (KBNL) [13] software. KBNL was used to interpret English queries and translate them into SQL. LDL++ and KBNL sent messages to Carnot semantic agents which provided access to the databases. The semantic agents also translated semantic differences among databases for the natural language queries.

The C++/Motif interfaces used in the Eastman Chemical Company application have now been replaced by HTML forms. The HTML forms can be executed by World Wide Web client software running on a variety of platforms (Unix, PC, Macintosh) and they enable remote access through the World Wide Web. When a user has filled out a form and submitted the form, an HTTP message is sent to the site where the Rosette software is executing. The HTTP server executing at that site routes the messages containing form data to a Rosette agent through a gateway module that conforms to the WWW Common Gateway Interface. The Rosette agent then expands the form data into messages to an LDL++ system or a KBNL system.

Information returned to the Rosette agent by LDL++ or KBNL is translated into HTML format by the agent. Only a specified amount of information is returned by the agent to the World Wide Web client. The HTML page returned to the World Wide Web client contains a hidden session identifier which can be used in subsequent HTTP messages to retrieve further information resulting from the original query.

The Eastman Chemical Company application is being further expanded to take advantage of the development of InfoSleuth agents with specialized knowledge of various chemical and business domains that can collaborate to accomplish design and marketing tasks. These agents are part of a growing class hierarchy of Rosette agents that are programmed to communicate with each other through the Rosette Tree Space.

References

[1] http://www.mcc.com/projects/infosleuth

[2] Woelk, D., M. Huhns, and C. Tomlinson. "Uncovering the Next Generation of Active Objects," Object Magazine, July-August, 1995. pp. 33-40.

[3] http://www.mcc.com/projects/carnot

[4] 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.

[5] 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.

[6] http://www.cosmic.uga.edu/pub/CLIPS.html

[7] Woelk, D. and C. Tomlinson, "InfoSleuth: Networked Exploitation of Information using Semantic Agents", COMPCON, March, 1995.

[8] 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.

[9] Agha, G. "Concurrent Object-Oriented Programming", Communications of the ACM, September 1990, pp. 125-141.

[10] Carriero, N. and D. Gelernter. "Linda in Context", Communications of the ACM, Vol. 32, No. 4, 1989.

[11] Woelk, D. and C. Tomlinson, "Carnot and InfoSleuth: Database Technology and the World Wide Web", Proceedings of ACM SIGMOD Intl. Conference on the Management of Data, May, 1995.

[12] 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.

[13] Singh, Mona, Munindar Singh, and D. Woelk. "Knowledge-Based Natural Language Interfaces for Information Access", MCC Tech. Report in preparation.