Mobile Agents on the Digital Battlefield

Martin O. Hofmann
Lockheed Martin
Advanced Technology Laboratories
1 Federal Street
Camden, New Jersey 08102
+1 (609) 338-3926
mhofmann@atl.lmco.com

Amy McGovern
University of Massachusetts
CS Dept., LGRC
Amherst, Massachusetts 01003
+1 (413) 545-1596
amy@cs.umass.edu

Kenneth R. Whitebread
Lockheed Martin
Advanced Technology Laboratories
1 Federal Street
Camden, New Jersey 08102
+1 (609) 338-4060
kwhitebr@atl.lmco.com

1. ABSTRACT

The Domain Adaptive Information System (DAIS) is a mobile intelligent agent system for information discovery and dissemination in a military intelligence network. DAIS is used reliably over low bandwidth military radio networks to facilitate Counter Intelligence/Human Intelligence (CI/HUMINT) operations. DAIS tested and validated two hypotheses: that agent technology could improve the efficiency of military tactical operations; and that mobile agents would outperform static agents in the unreliable, low bandwidth networks used in military operations. DAIS increased by two orders of magnitude military intelligence analysts' access to distributed information. This paper presents the design of the system, the experiments, and the results, as well as the future directions of DAIS.

1.1 Keywords

Mobile Agents, Agent Architectures, Information Agents, Middle Agents, Information Integration

2. INTRODUCTION

This paper reports the results of a series of experiments with the application of mobile intelligent agents to information networks used in military operations. Two hypotheses were evaluated, one functional, the other technical. Our functional hypothesis is that agent technology can improve the efficiency of military tactical operations. Our technical hypothesis is that mobile agent offer advantages over static agents in the unreliable, low bandwidth networks that are used in military operations. The results we observed support both hypotheses, but not entirely.

The success of military tactical operations depends on the accuracy and speed of information delivery. Critical information includes maneuvers, status, and plans of opposing forces and status of own forces. Counter Intelligence/Human Intelligence (CI/HUMINT) operations gather information from prisoners of war, cooperating civilians in the secured rear area, and long-range surveillance teams inserted deep into the enemy's territory. Effective CI/HUMINT operations are essential to force and asset protection in the theater rear area in war and in operations other than war.

Until recently, CI/HUMINT operations depended on paper reports, telephone messages, and transcription. Delays of several hours commonly occur between the collection of a piece of information and its integration into the status summarization and course of action planning process. Deficiencies caused by manual, error and delay causing procedures have prompted the U.S. Army to aggressively pursue integration of CI/HUMINT into the digital battlefield with the help of commercial off-the shelf (COTS) hardware and software [9]. Charged with inserting state-of-the-art computing technology into the CI/HUMINT process, we embraced the functional hypothesis that software agent technology can improve the efficiency of military tactical operations. We developed DAIS and fielded it with CI/HUMINT units in order to experiment and prove or reject this hypothesis. We looked for improvements beyond what could be accomplished with electronic mail and web-based client-server tools.

Information dissemination and discovery are the general problems addressed by our DAIS mobile agent system. Specifically, we focus on dissemination and retrieval of information in unreliable, low bandwidth networks of computer systems ranging from handheld personal computers to desktop personal computers and multi-user workstation. An autonomously executing process, moving between computers under its own control, appeared to be the best candidate to achieve reliable delivery of information-laden messages in this environment. Discovery of information in this maze of databases seems equally amenable to a mobile intelligent agent design. The belief that mobile agents solve the dissemination and discovery problem more effectively than stationary agents or client-server approaches constitutes our technical hypothesis.

DAIS has evolved during a period of intense research on intelligent and mobile agent methodologies. Some of its features were inspired by other work, others resemble competing solutions because of the shared basic assumptions and goals, still others are extensions of common mobile agent system features. We developed DAIS as a mobile agent-based information infrastructure instead of a decision support tool or human behavior emulation. Our initial implementation was based on Dartmouth's AgentTcl [6] layer. The agent itinerary is inspired by parallel work on IBM's Aglets [3]. We have developed a collaboration scheme which generalizes the "meet" concept in General Magic's Odyssey [2] environment. Like ObjectSpace's Voyager [7] technology, we are moving to provide CORBA compliant agent services. See [5] for a comparison of these agent systems. DAIS agents exploit mobility and make intelligent decisions about network resource usage and information retrieval, but de-emphasize decision making about the user tasks supported by this information. Our most recent efforts, based on the DAIS system described here, are producing a taskable, collaborating, reactive agent infrastructure.

In the following sections we present the features of mobile intelligent agents which gave credence to our hypotheses and contrast them to other possible approaches. Next, we characterize our mobile agent system design, followed by a description of the experimental evaluation of our design in real field training exercises of a MI Brigade. We then discuss the results and lessons learned and close with a presentation of ongoing and future work.

3. MOBILE INTELLIGENT AGENTS

A mobile agent, traveling in a distributed system, moves the activity specified by its creator or controller close to the data. DAIS exploits the fact that in many applications, the code that specifies the activity is smaller than the data to be operated on. Moving the agent instead of the data has the potential to reduce the communications load on the network connecting the distributed systems. A theoretical analysis of the trade-offs between mobile agent migration and the remote procedure call paradigms can be found in [8].

A mobile agent operates autonomously, without maintaining a continuous connection to its controller or user. The agent performs its task and saves any results until its connection to the controller is re-established. This "Launch & Forget" capability is essential for users who can only temporarily establish connections to the network which contains their chosen data repositories.

DAIS mobile agents are persistent, waiting for unreliable data sources and links to become available, and waiting for dynamic information to be made available by a source. Agents allow the delivery and retrieval of data to complete without user monitoring, trouble-shooting, or recovery actions. Agents relieve users of tedious data monitoring tasks.

Traditional database client-server solutions off-load user interface processing from the server to the client. Clients need a continuous connection to the server to receive the results of a request. Clients are frequently customized to connect to one particular server.

Web client-server approaches are attractive because web clients are becoming ubiquitous and many services are being equipped with web-server access. As in traditional client-server approaches, most of the processing is done at the user's end and large amounts of data have to be downloaded. Web page sizes are growing because pages increasingly incorporate multi-media contents and fail to provide compact text-based alternatives. Frequently, an extended dialogue between user and server is required to accomplish information retrieval tasks.

Electronic mail primarily serves to disseminate information, but is not equipped to deal with delivery problems. It operates as an effective communications medium between human users only, and even in relatively small networks incompatible mail services may be found.

Mobile information agents clearly emerge as the most promising of the alternative schemes. The following section describes our mobile agent design.

4. DAIS: MOBILE INTELLIGENT AGENTS AND DOCKS

DAIS is a system for information discovery in low bandwidth networks of large volumes of rapidly changing data, as described in [10].The DAIS approach separates mobile task agents from stationary service agents. Task agents travel through the network and execute tasks on behalf of the user. Service agents remain at one location and pre-process information available locally and mediate access to this information. Service agents and the basic agent life cycle services collectively constitute the extended agent dock, see Figure 1.

DAIS agent intelligence strictly focuses on effective exploitation of the information infrastructure in disseminating and discovering information. Agent intelligence does not necessarily focus on intelligent or expert problem solving in the application domain, such as MI analysis. During information dissemination, DAIS relies on agent intelligence to ensure delivery of messages despite temporary network and host outages. During information

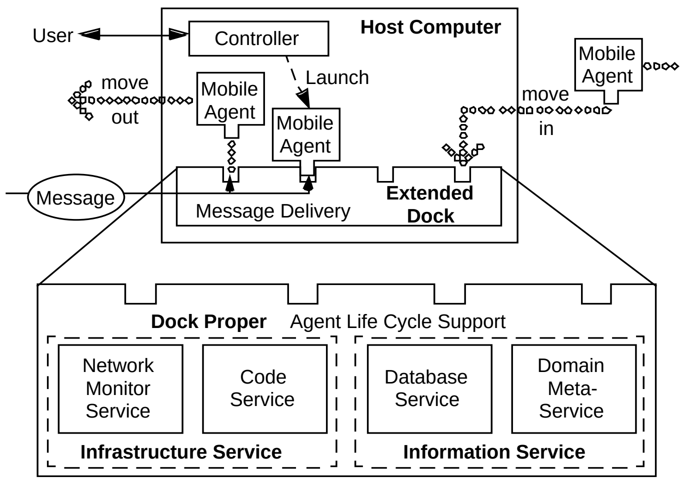

Figure 1. Mobile Agent Architecture. Mobile agents use services provided by the dock to efficiently travel through the network and, at their destination, manipulate data.

discovery, DAIS harnesses agent intelligence to locate appropriate data sources, extract the requested information, and translate the information into the requested format.

The extended agent dock consists of the dock proper, infrastructure service agents, and information service agents. The dock proper provides agent mobility and life cycle services, accepts and manages incoming mobile agents, and channels incoming messages to their intended receiver. In the following, the term dock will refer to the extended dock unless otherwise noted. The dock permits or denies mobile agents to move to its host. When the dock detects that its host is overloaded with visiting agents, it prevents additional agents from moving in and ensures sustainable agent load on the machine.

The dock proper is implemented as a CORBA object in Java, and the dock services and mobile agents are implemented in a mixture of Java and Tcl. However, our next-generation implementation of DAIS in 100% Java is nearing completion.

The infrastructure service agent pool contains the Network Monitor Service and the Code Service; the information service agent pool consists of the Database Service and the Domain Meta-Service. The communication and migration layer, which includes all the failure and exception handling, was initially implemented using AgentTcl and has now been re-implemented in Java using Iona Corp.'s OrbixWeb CORBA tool.

The **Network Monitor Service** maintains the status of the connections to other hosts on the network. Before it attempts to move to another host, a mobile agent can verify that the host is reachable. If the host is not reachable, the agent can request that the dock notify it when the host becomes available again, or can decide to move to an alternate destination. In any case, multiple mobile agents will behave in an organized manner, avoiding repeated wasted efforts by each agent to contact the remote host.

The **Code Service** represents our efforts to find an optimal compromise between a mobile agent system and a system of communicating agents. The Code Service allows an agent to discard most of its code before moving to another host and to re-acquire it from the Code Service at the destination host. Even the code used to interact with service agents other than the Code Service, is available from the Code Service. The agent only carries enough code to facilitate its correct start-up and reconstitution at the destination. In the extreme, this scheme can reduce the agent's move to a transfer of the agent's state and the results it has already collected.

The **Database Service** is tantamount to a uniform database client (or wrapper) which interacts with one of several database management systems (DBMS). A unique database ser-vice agents exists for each DBMS, such as Microsoft Access or the Warrior/Warlord MI custom sys-tem. Mobile agents do not need to know how to interact with each of the different data sources distri-buted in the network. Instead, the agents request database services via the database service agents.

The **Domain Meta-Service** pro-vides mediation between user tasks and information sources. A basic ontology of domain concept terms and a representation conventions model are used to construct mappings from general ontology terms to specific database schema field labels. Once these mappings have been established, users can interact with multiple heterogeneous data sources through a uniform, task-oriented interface which hides the diversity of the underlying data formats. The mobile agent architecture naturally supports both the discovery and characterization of the data sources and the collection and aggregation of information from multiple sources.

Mobile agents exploit the dock services. We developed a set of mobile agents which address the information dissemination, retrieval, and discovery needs of the MI user community. Information dissemination or "push" agents include the database *Add* and the record *Forward* agents, which visit the Database Service agents on the distributed hosts to insert records into their databases. The *SQL query* agent performs information retrieval or "pull" on databases via the Database Service agent. The *Abstract query* agent discovers relevant information from multiple distributed, heterogeneous databases, guided and supported by the Domain Meta-Service agents. The *Modify* and *Delete* agents perform database utility operations on any of the distributed databases. All mobile agents use the Network Monitor and Code Services to efficiently travel through the network.

The abstract query mechanism introduces a small-scale information integration capability into the DAIS system. The Abstract query agent translates a user query, expressed in general domain terms, into multiple SQL queries, geared toward database-specific record schemata. It selects from all data sources that are capable of holding relevant data in their schemata. It creates and sends out one SQL query agent for each database type and subsequently collects the results from the selected sources. The abstract agent eventually presents the results to the user in a uniform format.

The Domain Meta-Service provides the general domain concept terms from which the user constructs the abstract query. It maintains an index from these abstract concepts to the schema fields of the distributed databases. The index implicitly encodes whether a database is capable of holding data relevant to a term.

The index is constructed when DAIS starts and is updated when new databases are created or when new hosts with additional databases are added to the network. The Domain Meta-Service associates common labels and label fragments with concept terms. Index entries are constructed when concept term labels match the field names of any of the reachable databases. The abstract query mechanism is conceptually simple but effective and greatly reduces the skill required to discover information in distributed heterogeneous networks of the kind set up for MI operations.

In the CI/HUMINT domain, our knowledge acquisition work revealed that the ubiquitous SALUTE format (Size, Activity, Location, Unit, Time, Equipment), augmented with a Person concept, would be appropriate as a general information retrieval format. As a small example, assume we are trying to determine if a suspect named Kim has been reported at a location named Tai. The analyst creates an abstract query for a "Person *Like* Kim at Location *Like* Tai." Among four databases, CIBlackGreyWhiteList, CISpot, Salute, and CICasingReport, the abstract query mechanism recognizes that the schemata of CISpot and Salute support this query. It generates four SQL queries, shown below. Activity and Remarks fields often contain descriptions which include information on participant persons.

SQL: SELECT * FROM CISpot WHERE (Location LIKE '%tai%' AND Subject LIKE '%kim%')

SQL: SELECT * FROM CISpot WHERE (Location LIKE '%tai%' AND Activity LIKE '%kim%')

SQL: SELECT * FROM Salute WHERE (Location LIKE '%tai%' AND Remarks LIKE '%kim%')

SQL: SELECT * FROM Salute WHERE (Location LIKE '%tai%' AND Activity LIKE '%kim%')

5. DAIS EXPERIMENT IN MI OPERATIONS

The most effective way to test our hypotheses was to install the DAIS system on the computer networks of a MI unit and observe its use and effect during field training exercises. In field training exercises, the DAIS system was exposed to realistic operating conditions which would be nearly impossible to simulate in the laboratory.

We had the opportunity to cooperate closely with the experimental users of the DAIS information dissemination and discovery system throughout the development cycle. Initial requirements were collected at the 525th MI Brigade at Ft. Bragg, NC and at the 201st MI Bgde at Ft. Lewis, WA. The final, most in-depth experiments were conducted in the context of field training exercises of the 210st MI Bgde. The 14th CI/HUMINT Battalion was the most enthusiastic adopter of this technology: DAIS became an integral part of their standard mode of operation.

So far we have participated in thirteen MI exercise over the course of the DAIS project. At each exercise, we installed DAIS in the MI host computers and administered user training. Unfortunately, workstations are not available to MI personnel between training exercises to practice using the system. Therefore, ease of use was a critical requirement for DAIS development.

Figure 2 shows a typical architecture of the computer network supporting CI/HUMINT brigade operations during an exercise. Every solid box corresponds to a cluster of activity. Each cluster provides high-speed (10 Mbps) ethernet connections between its hosts. Radio links, such as the MSE (Mobile Subscriber Equipment) or SINCGARS radios, connect clusters. The MSE implements a common ethernet network of all hosts at 16 to 64 kbps. The SINCGARS link establishes point-to-point voice and digital wireless communications between remote teams and the operations centers (Ops) over the Cybernet CIGAR modem at 4.8 kbps. The aironet system implements a wireless TCP/IP connection at 2 Mbps but only up to 1000 ft.

Information enters the system when IPW, CI, or LRS teams file reports in response to simulated events generated by the Battle Simulation Center. DAIS Add agents assist the users in entering the reports in one of the approved report formats and transport the reports to the IPW, CI, and LRS Company operations centers (Ops). The Add agents enter the new reports into the local databases in the Ops centers and signal the arrival of new information to the Ops center operators. Operators perform preliminary analysis and quality control measures on the reports, then task Forward and Modify agents to move the verified reports to the Battalion TOC, the Bgde TOC, the Rear Area Operations Center (RAOC), and the single source CI/HUMINT operator in the ACE (Analysis and Control Element) for further analysis.

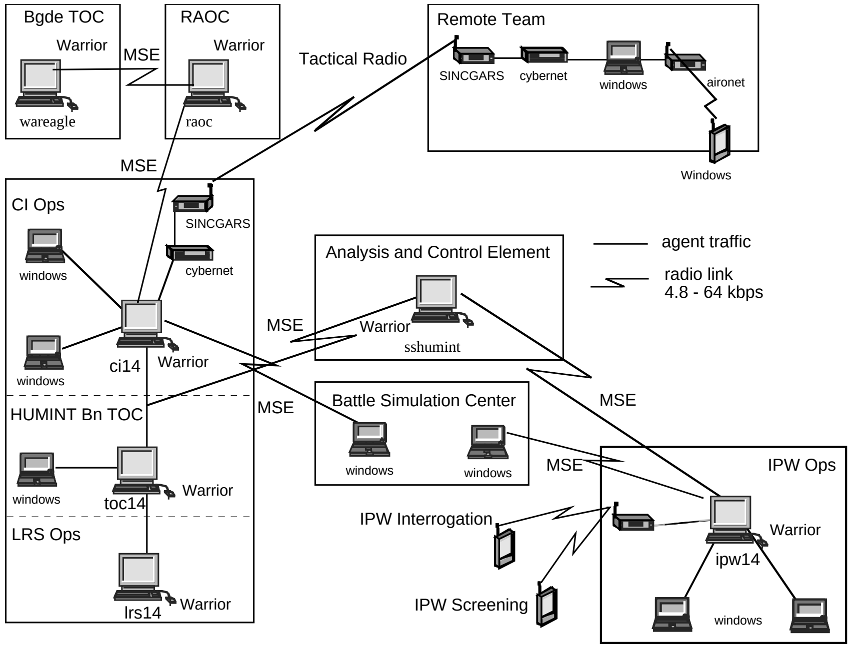

Figure 2. MI Exercise Automation Plan. Information is pushed and pulled from CI, IPW, and LRS collection teams to analysts.

Operators in the Ops centers and at higher echelons generate summaries and control measures periodically and in response to situation developments. Operators task SQL queries to summarize the contents of specific databases, such as the SALUTE database of critical messages. Operators task Abstract query agents to find information related to a specific topic of interest, such as the whereabouts and activities of a dangerous insurgent.

The operational goals of the MI operation, particularly the MI exercises, are to ensure that critical information is collected, represented correctly, and rapidly disseminated to the appropriate decision makers. MI analysts strive to create accurate and timely situation summaries from individual reports and to discover enemy courses of action. MI commanders strive to control MI assets so that they are optimally effective in the collection and interdiction of enemy activities. Many of these operational goals can only be evaluated qualitatively, but we identified several quantitative measures of operational success:

1. Percentage of reports which reach their intended recipients
2. Time interval from report entry until it has reached all recipients
3. Time interval from report entry until it has been acknowledged by all recipients
4. Percentage of reports formatted correctly
5. Amount of training required for automation users

In a typical exercise about two dozen (24) operators used DAIS as their main automation tool. Operators were divided into two shifts and functioned as report collectors, quality controllers, and analysts. Improved operator effectiveness validates our functional hypothesis.

In order to evaluate our technical hypothesis, we measured the required DAIS system effort, i.e. the demands DAIS agents imposed on the users, the host systems, and the network connections. Our technical hypothesis is validated if we can show that superior operational performance is achieved with equal or lower effort than with competing methods. We evaluated user efforts qualitatively, measuring system efforts by the processor resources consumed by DAIS, and measuring communications efforts by the number and size of the agents and inter-agent messages sent across the communication links.

6. RESULTS AND LESSONS LEARNED

6.1 Operational Goals

The best indication of the measure of DAIS' operational success happened during the "Griffin Genesis" exercise. As a test of the MI Brigade's ability to recognize and disseminate a critical message, exercise evaluators inserted a report in the simulation which indicated that an airfield in the rear area was to be raided by enemy special operations force commandos in less than two hours. Aided by the DAIS system, the IPW interrogator who collected the information was able to pass the report up the MI CI/HUMINT chain to the RAOC within minutes. Because of the effective information dissemination process of the DAIS mobile agent system, the report reached the RAOC in time to interdict the enemy raid.

Collections of quantitative operational measures was hampered by the sensitivity of the data. We were limited to a quick categorical analysis at the exercise location during and immediately after the exercise. Therefore, in Table 1 we report results in rough categories, averaged over all exercises. Historical data for operations without DAIS automation support are unavailable. The values listed are based on our own observations and the subjective assessment of the user community.

In summary, the operational goals of the MI units were substantially achieved by the automation provided by DAIS. Speed of information dissemination increased by one to two orders of magnitude with the use of DAIS. Quality and usability of information increased substantially. Our functional hypothesis is thus verified.

Goal	w/o DAIS	with DAIS	Comments
1. Delivery Rate	< 50%	80-100%	Near 100% after additional 1 hr on the job user training
2. Delivery Delay	4 hr avg.	10 min	Remaining delays are due to human report review process
3. Acknowledge	> 4 hr	10 min	Same as 2 because DAIS persistently alerts users to reports
4. Correct Formats	< 50%	80-100%	Some user errors are not detected by DAIS
5. Training	?	ca. 4 hours	Additional 1 hr on the job training required for some users

Table 1. Categorical Summary of Operational Resultsdata files, locked up the network for tens of minutes. The ability of DAIS agents to extract just the requested information at the source has proven very effective.

6.2 Technical Goals

User effort in learning and utilizing DAIS to perform their MI collection and analysis duties proved to be minimal. Users control the activities of the agents but are not aware of the details of agent mobility. DAIS effectively presents every data source in the whole distributed MI network as one step away from the user's console. Users receive notification when agents complete their tasks and are alerted when agents expect user instructions. The DAIS mobile agent system provides simpler access to all heterogeneous databases on the tactical network than the standard MI-specific Warrior database system provides for the local databases on the user's computer.

The image of a DAIS mobile agent was much larger than desired. Each agent contains its own Java virtual machine and a Tcl interpreter in addition to the agent specific code and data. Some of these elements can be shared among agents, but we determined that a minimum of 32 MBytes of memory was required on a personal computer to run DAIS at a reasonable speed. We are addressing this issue by moving to a thread-based implementation for the dock, the dock services, and the agents. We are also moving to a pure Java implementation.

The footprint of a typical agent on the network ranges from 8 to 15 KBytes. On standard ethernet, an agent move appears instantaneous, but on the 4.8 kbps SINCGARS link, the move consumes 30 seconds. Agents send back results via data messages whenever they have a connection to their controller available. Data messages have very small overhead and their size is approximately equal to the size of the data sent. We have thus significantly reduced data traffic at the small expense of sparse mobile agent traffic. The agent traffic on the tactical network has been unnoticeable in all the exercises we attended. Other activities proceeded unimpeded while DAIS agents traveled through the net. On the other hand, we witnessed several instances where the transfer of large amounts of data, packed in web pages or data files, locked up the network for tens of minutes. The ability of DAIS agents to extract just the requested information at the source has proven very effective.

Our experimental results indicate that the biggest impact of the mobile agent approach was due to their ability to operate independently of the user and persist in their determination to execute their assigned task in the face of network and host outages. DAIS also persistently reminded users of their charge to acknowledge and process new information. Assured and rapid information push and pull through the tactical network was a direct and most valuable result of the mobile agents' ability to navigate through the network autonomously and to interact appropriately with the digital and human entities they encountered along their path. We attribute the operational success of DAIS as reflected in Table 1 to the agent task persistence. We had expected the biggest advantage of the mobile agent approach to derive from the reduced load on the network, since the small agent moves to large amounts of data.

Given that we achieved the operational goals with relatively little demand on the MI network resources, our technical hypothesis has been corroborated by the experimental results for the most part. We have identified agent memory image size as the weakest aspect of our implementation. The DAIS design has proven itself, however, and enhancements to its implementation will increase DAIS system efficiency.

6.3 Lessons Learned

Agent migration across low bandwidth communication channels is more challenging than we expected. The problem is compounded because low performance handheld computers are typically connected via such links. An alternate strategy, which is functionally equivalent to an agent move, is to create or maintain idle agents at the handheld computer which only receive their tasks via the low bandwidth communications channel. However, the burden of creating and/or maintaining idle agents at a low performance computer makes this alternative less attractive. Our future strategy combines the mobile and tasked agent models in this highly constrained case. Tasks which are performed frequently will be supported by idle agents. Unusual tasks are carried by the mobile agents described in this paper.

Redundant activities are frequently performed in a network of multiple users and data sources. Users often re-execute an exhaustive retrieval task just to get the latest information. Two users sometimes request the same or similar information from a source. If the agents and results must travel over the same low bandwidth link, the information should be shared instead of queried and delivered multiple times. These lessons of the DAIS project have prompted us to initiate a project on general information sharing and agent collaboration support mechanisms [4].

The abstract agent mechanism worked very well, despite its limited knowledge about the MI domain. It did not see as much use as we had hoped, because multi-source combination and in-depth analysis of MI information happens in the all-source section of the ACE. Current military doctrine does not call for the CI/HUMINT units we worked with to perform analysis which would benefit from the abstract query mechanism. When we tested the abstract query mechanism in the MI network, one abstract query for the name of a known insurgent using the abstract MI concept "Person" spawned 55 SQL queries on 120 Database fields against 65 distributed databases. Fields included "Name", "Last_Name", "Associate", and "Remarks". Another query for an Artillery Battalion using the "Unit" concept searched fields, such as "Unit", "Unit_Name", and "Unit_Size", of 150 databases. We are currently starting a follow-on project in intelligent information integration funded by DARPA.

Our rapid prototyping approach to DAIS system development ensured that successive versions rapidly addressed the concerns of the MI users. We needed to gain user acceptance to measure system success. We chose the scripting language Tcl as primary implementation language because it supports rapid prototyping. Tcl has also been recognized as appropriate for agent development [1]. Portability concerns led us to add a Java layer underneath the Tcl agent behavior code. The resulting architecture has become cumbersome and we are working towards a pure Java implementation.

7. FUTURE WORK

The major focus of our continuing work on DAIS is on more general intelligent agents and on agent communication and collaboration. Instead of having a specific agent for each task, the next DAIS system is based on the Universal Mobile Agent (UMA) concept. A UMA is a taskable mobile agent which contains an itinerary and a task schedule. Each itinerary element consists of a task, the destination host, and the reaction to task success and failure events.

In the UMA system, the dock is extended to include a task server. The task server contains agent tasks and a network-wide directory of services which are able to perform each task. Tasks are parameterized. For example, the *SQL Query* task takes the place of the SQL Query agent and carries a SQL statement to the selected database. Permanent service agents and mobile agents can register and unregister the tasks they are able to perform with the task server on any host machine. Task servers distribute their local registries throughout the network.

The UMA can intelligently schedule the tasks on its itinerary. The itinerary can be divided among a number of children UMAs in order to get the work done most efficiently. The UMA can create children to query in parallel across multiple machines instead of querying in series. This capability is enhanced above the current capabilities by storing estimates of how long each task will take and scheduling based on that as well as where each machine is. Also, the scheduling algorithm takes into account the maximum number of agents allowed within a system at one time. This capability will increase the efficiency of the UMA system over the current specialized task agents.

The UMA itinerary elements specify how the agent should react to events related to the execution of its task. When a task completes successfully, the agent reacts by selecting the next element on the itinerary. When a host specified as a destination is unreachable, the agent spawns a child UMA which waits for the destination to become available and the agent continues the remaining itinerary. Waiting is implemented efficiently by registering a wake-up event with the dock. Results from agent activities, such as records retrieved from databases, are returned to the requestor asynchronously by the parent and child agents, and are collected into a common result set. Agent reactions to other events can easily be specified. Our goal is to avoid forcing the user to specify the details of agent behavior in every circumstance. Instead, we are working on defining goal-based classes of agent behavior, which imply specific reactions.

In order to facilitate agent collaboration and communication, we have developed a Postmaster mechanism [4] and are integrating it into DAIS. The Postmaster will allow the UMA agents to cache the results of queries and to share data between agents. Caching the results of previous queries will speed execution time and reduce resource usage by not repeating identical or similar queries. Agents can also share results with other agents by leaving a copy of their results with the Postmaster. This also improves efficiency. Lastly, the Postmaster provides a method for updating urgent data by allowing the agent to mark queries as urgent. When new information is reported to the Postmaster that matches the urgent query, the Postmaster will forward the requested information to the original requester.

8. ACKNOWLEDGMENTS

This work has been supported by DARPA Contract N66001-95-C-8636. Important contributors include Julius Etzl, Greg Jorstad, Russ Lentini, and Jennifer Kay. We thank the commander and personnel of the 201st MI Bgde, 525th MI Bgde, and the 14th MI Bn, in particular, for their valuable contributions and their patience during early prototype testing..

9. REFERENCES

[1] Cost, R.S.; Lakhani, J.; Finin, T.; Miller, E.; Nicholas, C.; and Soboroff, I. Agent Development Support for Tcl. Proceedings of the Fifth Tcl/Tk Workshop, (Boston MA, July 1997).

[2] General Magic Odyssey. http://www.genmagic.com/agents/

[3] IBM Aglets Workbench, Programming Mobile Agents in Java. http://www.trl.ibm.co.jp/aglets/whitepaper.htm

[4] Kay, J.; Etzl, J.; Thies, J.; and Rao, G. The ATL Postmaster: A System for Agent Collaboration and Information Dissemination. Proceedings of the Second International Conference on Autonomous Agents (Agents '98), (Minneapolis/St. Paul, May 1998).

[5] Kiniry, J. and Zimmerman, D. Special Feature: A Hands-On Look at Java Mobile Agents. IEEE Internet Computing, 1(4) (July 1997), 21-30.

[6] Kotz, D.; Gray, R.; Nog, S.; Rus, D.; Chawla, S.; and Cybenko, G. Agent TCL: Targeting the Needs of Mobile Computers. IEEE Internet Computing. IEEE Internet Computing, 1(4) (July 1997), 58-67.

[7] ObjectSpace Voyager Core Technology, http://www.objectspace.com/voyager/index.html

[8] Straßer, M. and Schwehm, M. A Performance Model for Mobile Agent Systems. Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications PDPTA'97, Volume II, (Las Vegas NV, 1997), 1132-1140.

[9] U.S. Army Intelligence Center and Fort Huachuca, Directorate of Combat Developments, ASAS Support Division, All Source Analysis System (ASAS) Counter Intelligence (CI) and Human Intelligence (HUMINT) Subsystem User Function Description (UFD), Coordination Draft, January 7, 1997.

[10] Whitebread, K.R. and Jameson, S. Information Discovery in High-Volume, Frequently Changing Data. IEEE Expert, 10(5) (October 1995), 51-53.