NPS-SM-98-001

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DECISION SUPPORT FOR RECONNAISSANCE USING INTELLIGENT SOFTWARE AGENTS

by

Marcia R. Edmiston
Darrell R. Gregg, Jr.
David G. Wirth

March 1998

Thesis Co-Advisors: Tung X. Bui
Carl R. Jones
Suresh Sridhar

NAVAL POSTGRADUATE SCHOOL
Monterey, California 93943

Rear Admiral Chaplain
Superintendent

Reproduction of all or part of this report is authorized.

Released by:

REPORT DOCUMENTATION PAGE

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1998	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE
DECISION SUPPORT FOR RECONNAISSANCE USING INTELLIGENT SOFTWARE AGENTS

6. AUTHOR(S)
Edmiston, Marcia R., Gregg, Darrell R., Jr., Wirth, David G.

5. FUNDING NUMBERS

MIPR 97110-T

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER
NPS-SM-98-001

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Center for Reconnaissance Research, Naval Postgraduate School, Monterey, CA

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Research in reconnaissance traditionally focuses on data detection and discrimination methods. Less emphasis is placed on transforming the collected data into useful information and presenting it to key command and control nodes. Information not presented in a timely manner is excluded from the decision process. This thesis proposes a conceptual model of intelligent software agents to support the human decision process and reconnaissance-related tasks. The Mobile Agent Reconnaissance Kit (MARK) suggests a hierarchy of software agents to facilitate data integration and coordination in a network-centric multisensor environment. The model uses static and mobile agents to collect data from dispersed, heterogeneous data sources, process and fuse the data, and present the resultant information to the user in an HTML file. The authors explore applications of MARK in terms of the military Intelligence Cycle, the Joint Director of Laboratories (JDL) Technical Panel for C3I Data Fusion Model, and the Joint Operations Planning and Evaluation System (JOPES) Crisis Action Planning.

14. SUBJECT TERMS
Software Agent, Intelligent Software Agent, Reconnaissance, Decision Support

15. NUMBER OF PAGES
143

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFI- CATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

Approved for public release; distribution is unlimited.

DECISION SUPPORT FOR RECONNAISSANCE USING INTELLIGENT SOFTWARE AGENTS

Marcia R. Edmiston
Lieutenant, United States Navy
B.A., University of Kansas, 1989

Darrell R. Gregg, Jr.
Captain, United States Army
B.S., Northeast Missouri State University, 1989

David G. Wirth
Lieutenant, United States Navy
B.S., Pennsylvania State University, 1990

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
March 1998

ABSTRACT

Research in reconnaissance traditionally focuses on data detection and discrimination methods. Less emphasis is placed on transforming the collected data into useful information and presenting it to key command and control nodes. Information not presented in a timely manner is excluded from the decision process. This thesis proposes a conceptual model of intelligent software agents to support the human decision process and reconnaissance-related tasks. The Mobile Agent Reconnaissance Kit (MARK) suggests a hierarchy of software agents to facilitate data integration and coordination in a network-centric multisensor environment. The model uses static and mobile agents to collect data from dispersed, heterogeneous data sources, process and fuse the data, and present the resultant information to the user in an HTML file. The authors explore applications of MARK in terms of the military Intelligence Cycle, the Joint Director of Laboratories (JDL) Technical Panel for C3I Data Fusion Model, and the Joint Operations Planning and Evaluation System (JOPES) Crisis Action Planning.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

A.   AREA OF RESEARCH

Research in reconnaissance has traditionally focused on data collection, leading to new and improved methods of detection and discrimination. However, less effort has been bestowed in using information technology as an integrated means to transform the collected data into useful information and delivering it to key command and control nodes in time for operational use. The purpose of this thesis is to propose a model of intelligent software agents to support the decision process and reconnaissance-related tasks. The model can be used to assist warfighters in managing day-to-day activities and crisis action planning. A key objective is to facilitate data integration and coordination through intelligent agents in a network-centric multisensor environment. Using intelligent software agents, commercial-off-the-shelf (COTS) technology, and various network technologies, the research will define the architecture of an information system capable of collecting data from dispersed, heterogeneous data sources, processing and fusing that data, and presenting the resultant information to the decision-makers.

B.   RESEARCH QUESTIONS

1. What are the major characteristics of software agents?

2. What are the current techniques for developing and deploying intelligent software agents?

3. How can intelligent software agents be used to assist/support the warfighter in the decision process?

4. What is an application of an agent model for supporting reconnaissance related to current decision processes?

5. What government and commercial projects are being developed using software agents?

6. What issues are involved with agent management, maintenance and coordination?

C. DEFINITIONS AND SCOPE OF RESEARCH

The traditional definition of reconnaissance brings to mind a physical surveillance of an area, usually to obtain intelligence for military use. Reconnaissance assets are typically thought to be physical objects, such as people, cameras or satellites. The overall purpose of these assets is to collect information about the intent and capabilities of the opposition, and make the data available to the decision-maker. Advancements in technology, and improvements in communications, have created new methods of gathering data and new types of sensors. In order to encompass all assets available to provide data, the traditional definition of reconnaissance must be expanded beyond the realm of physical surveillance. Throughout this research, reconnaissance is defined to be the collection, analysis and dissemination of information. Whether the information is gathered by a satellite system or a computer program is immaterial.

The research will investigate how intelligent software agents can support reconnaissance-related tasks. The domain of this thesis is the development of a decision support model that uses intelligent software agents to assist with reconnaissance and provide decision support in a command and control system. The technology to create basic intelligent software agents exists in the marketplace. The authors take the currently available technology and propose new methods of application, which require the development of new capabilities. The thesis is conceptual in nature. The thesis also provides recommendations on whether and how intelligent agents can be used to improve command and control systems, and suggestions for further research, including the development of a working prototype.

D. RESEARCH METHODOLOGY

An extensive literature review was conducted to examine the background, concepts, and technology of intelligent software agents. Books, periodicals, the World Wide Web

and other library information resources were used. The information gathered was used to develop a taxonomy to classify agents based on multiple characteristics. The taxonomy was then used to explain how agents work, in concept. The authors created a decision support model. Multiple scenarios were developed to test the model:

- In a Joint Task Force environment.

- To support data integration and fusion.

- During a situation requiring crisis action planning.

The authors became familiar with various products using current agent technology prior to developing the model.

E.   ORGANIZATION OF STUDY

Chapter II is a literature review of software agent technology. It provides information on the terminology and issues associated with software agents. The taxonomy used to classify agents is explained, and the relationship between the characteristics is examined. Chapter III focuses on the development of intelligent software agents, to lay a framework for understanding the network environment and concepts used in creating the decision support model. Chapter IV introduces and defines security issues related to software agents. Threats to computer networks and software agents are identified, as are countermeasures that can be used to minimize the threat. A security protocol is proposed for use in agent based information systems. Chapter V describes the decision support model in terms of the hierarchy of agents, the role of each agent and the interaction between agents. The model includes a framework that integrates mobile intelligent software agents, resident intelligent software agents and decision support algorithms with appropriate user interfaces. A Joint Task Force level scenario is provided to illustrate how intelligent software agents can be used in a command and control system. Chapter VI explores connectivity considerations associated with an intelligent software agent architecture and various management issues concerning intelligent software agents.

Chapter VII includes a description of:

- The intelligent cycle using the intelligent software agent model

- A description of the Data Fusion Group of the Joint Directors of Laboratories (JDL) Technical Panel for C3I model for data fusion and how it can be adapted to use the intelligent software agent structure in the decision support model.

- The application of the decision support model to support crisis action planning.

Chapter VIII discusses research being conducted by government agencies and commercial companies into the use of software agents. Projects that use some form of intelligent software agents are described. The chapter contains a description of commercial products available to assist in developing software agents, and also discusses advanced search and retrieval technologies that use agents. Chapter IX provides the conclusions of the research and recommendations for future research.

II. LITERATURE REVIEW

A. INTRODUCTION

This chapter provides a comprehensive review of software agents, their history, components, characteristics, and considerations for using them. This chapter also explores the issues of mobile and distributed agents, text retrieval and document management, along with a discussion of the possible levels and types of intelligent software agents.

B. DEFINITIONS

1. Agent

A software agent is an object, complete with code, data, and execution state, running autonomously on behalf of a human or non-human user on one or more computer systems at different times in its life.

2. Mobile Agent

- Mobile agents are "objects consisting of code, data and execution state that may go beyond protection domains." [Ref. 1]

- "A mobile agent is a component containing at least one thread of execution, which is able to autonomously migrate to a different site. A site is a component execution environment inside which inter-component communication is less expensive than communication among components residing on different sites." [Ref. 2]

- A mobile agent is "a set of objects performing a computation on behalf of a user. This computation is performed within an agent execution platform that controls the execution of the agent. An agent may request to be moved causing its computation to be interrupted and resumed on another platform." [Ref. 3]

- "A mobile agent is a program: (i) that a person or organization vests with its authority, (ii) that can run unattended for a long time (e.g., a week), (iii) that can

meet and interact with other agents (iv) and that can execute on different computer systems at different times of its life" [Ref. 4].

3. Intelligent Agent

An intelligent software agent plays the role of an intelligent, dedicated and competent personal assistant. In the traditional approach, the computer is programmed to react to the user's instruction. The intelligent software agent approach is a proactive one, in that the user specifies what he/she wants the computer to accomplish, and the latter performs tasks on behalf of the user. An intelligent agent has the ability to learn from behavior, training and its environment.

C. HISTORY OF SOFTWARE AGENTS

Apple Computer demonstrated an application of user interface agents in a 1988 video about its vision of "Knowledge Navigator". The setting was a professor working on a laptop in 2010 that displayed a software agent named Phil - male human face with a bow tie. [Ref. 5] Phil was the professor's personal assistant who communicated through voice recognition technology. Other companies have recently contributed to agent technology including HP NewWave who developed an office automation system called Cooperation. IBM has developed "Charlie" and Microsoft has developed "Bob", which are both personal assistants for the desktop. The goal of these agents is to make computer systems more human-like and more natural to use, along with being able to tailor the agents to the user needs and preferences. The advent of Java opens new application opportunities for using software agents.

D. GENERIC SOFTWARE AGENT ARCHITECTURE

An agent possesses certain skills (intelligence) and knowledge (heuristics) to interface with the user or other applications. A generic architecture of the software agent is depicted in Figure 1.

User

**User-Agent Interface
or Adapter**

**Processing
Engine**

**Procedure
Repository**

View/Framework

Other Software Agents/Applications
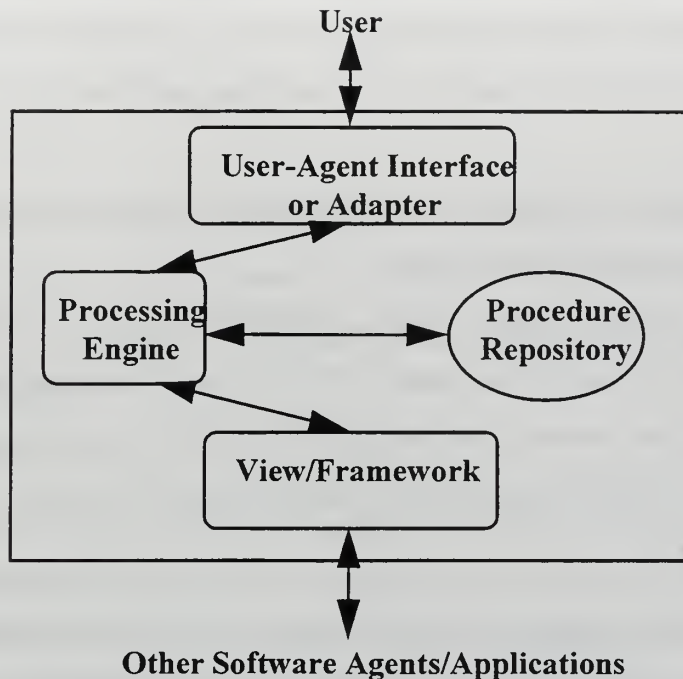
Figure 1. General Software Agent Architecture

From Ref. [6]

The user interacts with the agent via the User-Agent Interface or Adapter. This allows the user to retain the current application and add agent capability to the extent required. In other words, the interface role is to pass the agent's skills and knowledge in the user's format. The agent uses its embedded procedures (Agent's processing engine) and related data (Agent's Repository) to perform tasks and exchange information via the View/Framework. The latter defines the standards of interaction between the user and the agent, and provides the latter with a choice of views. The electronic mailbox is an example of the View/Framework.

The User-Agent Interface is comprised of a view framework and an adapter framework (e.g., IBM agent building environment, 1997). As its name suggests, the view framework visualizes the agent knowledge and skills to the user. The adapter framework is the agent's road map used to connect information.

7

The repository contains facts and rules allowing agents to reason and learn. It represents the persistent storage of knowledge. Facts are collected triggering events and sensors (short-term facts), from existing databases (long-term facts) and are derived from reasoning (beliefs).

The processing engine also contains the agent's current understanding of the user and the instructions received from the user.

Intelligent software agents could potentially have many elements in their composition. While not all of these parts will be required when programming an agent, they must be considered in context with the agent's mission. Some of these may be omitted without affecting the ability of the agent to do its task effectively. [Ref. 7]

1.   User

The user is the person or entity that tasks an agent. Users can purchase agents off-the-shelf or use pre-made agents from a library to accomplish their mission. Therefore, the agent must carry the identity of its user in order to provide feedback regarding its status and to report the results of its task. A mobile agent can have more than one user as it is tasked to serve on a distributed computing platform. The term user does not necessarily indicate a single human, but can represent a collection of individuals or other agents.

2.   Author

The author is the person who programs the agent. Should a problem arise, knowing who the author is will allow the user to contact them to work out a solution.

3.   Lifetime

The lifetime of an agent is also known as the time to live (TTL). Some agents will be given only a small task that can be done in a short amount of time. Once that agent has completed its mission, it will die gracefully. Other agents will require much more time to complete their missions based upon a variety of factors, such as computing resources, traveling to more than one location over an internet or intranet, or remaining persistent on a system for an extended period of time monitoring data. Determining the TTL for an agent

demands careful consideration of many factors. A short-lived TTL may not allow that agent to complete its task, while a long-lived TTL may tie up computing resources unnecessarily on the host system.

4.   Account

As a resource, it is possible that an agent be charged for the time and computing resources it uses while on the host system. If the agent is charged for whatever reason, the agent needs to have some form of owner identification inside it to charge the appropriate owner for resources used.

5.   Goal

The agent has to know when it has reached its desired end state. "Crisp statements of successful agent task completion will be necessary, as well as metrics for determining the task's completion and the value of the return." [Ref. 7] Assessing goal completion may be difficult. Finding the lowest price of a computer system from several different vendors is considered an easy task with a clear goal. The criteria, price and availability, are easily comparable. However, a task such as monitoring troop movement in Iraq necessitates using thresholds to trigger decision points in the warfighter decision cycle. The agent monitoring enemy troop movements is looking at more complex information and attempting to determine if the trigger threshold is exceeded prompting a response back to the user.

6.   Subject Description

The subject description helps identify what task the agent is attempting to accomplish. This short synopsis will provide the host system with a means to determine if the agent should be allowed access to information inside the host. The vulnerability in this is that a false subject description could mislead the host system into thinking that it should allow the agent access when it should not.

E. REASONS FOR USING INTELLIGENT SOFTWARE AGENTS

Intelligent agents offer tremendous potential in supporting intelligence analysts in tasks such as data fusion and in facilitating coordination. Intelligent agents are already being used in Internet applications to perform various functions such as customized search and information delivery. At least six general reasons are provided for using intelligent software agents. [Ref. 7]

1. Mundane Personal Activity

A considerable amount of time is spent doing mundane, routine personal actions, such as sorting through email. Reducing the amount of time spent on these actions is critical to individuals who need additional time to devote to more pressing issues. The goal should be to keep the amount of time spent doing these actions down to a minimum.

2. Repetitive Office Activity

Intelligent software agents have the ability to automate repetitive office activity such as scheduling by acting on the user's behalf. Repetitive office activity tends to be a major contributor to high labor costs. Reducing costs improves business operations. Automating these repetitive activities can increase the organization's productivity in addition to reducing costs.

3. Search and Retrieval

Current search and retrieval tools for information in a file or database on a computer or on a network basically require the user to constantly refine the search to locate the required information. Because of the time required with these searches, users spend way too much time directly involved in manipulating the search engines. Using agent technology to assist them would be more practical and save time. "These (intelligent software) agents perform tedious, time-consuming and repetitive tasks of searching databases, retrieving and filtering information, and deliver it back to the user." [Ref. 7]

4. Managing the Information Overload

The vast amount of information currently provided to users is overwhelming. Much time and effort is being expended to search this massive volume of information to find the specific information that the user needs. The user needs a smart mechanism (i.e., an assistant) to assist him/her in filtering all of the data and information sources to reduce the amount of time and effort required to obtain the desired information. Excessive time spent attempting to retrieve the correct information results in unproductive time and can cost the organization money.

5. Domain Experts

Human agents ranging from airline reservation receptionists to stock brokers are widely available but often at a high cost. Organizations that require certain domain experts (agents) could benefit by using intelligent software agents to do these same tasks for them at a greatly reduced cost.

6. Decision Support

The success of decision-makers is dependent upon their ability to obtain and use timely information to give them a competitive edge. Decision support systems have been proven to improve, at times dramatically, the productivity of decision-makers [Ref. 8]. Intelligent software agents can be added to decision support systems to reduce the amount of time and effort to obtain the desired information.

F. CHARACTERISTICS OF INTELLIGENT SOFTWARE AGENTS

Intelligent software agents are recent developments and many definitions have been given as to their abilities and functionality. Their characteristics have been narrowed down to help identify what makes them intelligent software agents. Intelligent software agents have the following characteristics: [Ref. 5]

1.   Independence

An agent has the ability to operate without user involvement once the user has sent the agent out on its mission. The user programs the agent with its parameters for obtaining the desired information then allows the agent to work autonomously until completion. The agent does not have to begin execution immediately after the user programs it. The agent may lay dormant until a specified time or event, then activate itself and carry out the intended task(s) independent of the user. For example, an agent may be programmed to search a stock market database right before closing time every day to see which stocks are extremely active for the day and notify the user through an email message. The user does not have to sit at his system everyday at this time to check the market. The agent does this independent of the user and allows the user to spend time doing more important things.

2.   Learning

Learning is the ability for an agent to modify its behavior in response to a changing environment. One method of learning is to replicate user actions when executing a particular task. This focuses on the personal assistant view of an intelligent software agent. A human personal assistant learns the patterns and traits of the person they are assisting and incorporates those into assisting that person. An agent does the same thing in a computer environment. It watches the user's actions, keeps track of those actions and modifies its behavior to its user preferences.

The learning usually occurs through observation, user feedback or training. [Ref. 5]

As mentioned above, the agent can learn through watching the user's actions. The user can modify the agent's behavior by providing feedback to the agent to improve its performance. If the information that the agent returns is considered useful in the context of the search, then the user can provide feedback to the agent to reinforce this type of search. Poor search results can be discouraged through user feedback as well. The user can also train the agent by running simulations of different scenarios to build a knowledge base for future use. Training an agent for a specific type of mission will improve the agent's

performance during real-world execution of that mission. User involvement will decrease and will focus on the information returned instead of making numerous refinements to reach the desired end state.

3.   Cooperation

When two humans cooperate to accomplish a task, they do so through communication and an understanding of their own and the other's mission. Programming cooperation between agents is highly complex. Independent cooperation requires dynamic modification of code at execution time in order to create an effective division of labor among the agents involved. Hierarchical cooperation may be simpler where an coordinating agent assigns individual tasks and controls, coordinates and synthesizes the results of their collective efforts. A communications protocol is needed to allow the software agents to communicate their information and work jointly on an unresolved issue.

4.   Reasoning

Reasoning is the ability to make a decision. The ability of an agent to reason or make inferences as to the best method to accomplish its task forces the agent programmer to develop an approach when designing an agent. There are three approaches to this issue: rule-based, knowledge-based, and learning. [Ref. 5] A rule-based approach refers to giving the agent certain rules or parameters to follow and is the easiest of the three to program. Knowledge-based approaches require an expert to compile vast amounts of information, which is subsequently given to the agent to determine any particular behavior or methods of the information. User or expert involvement is extremely high. Learning, as mentioned above, takes the information that the agent acquires from the user through repetitive tasking and feedback. The agent uses this information to modify its behavior for future tasks.

5.   Intelligence

The level of intelligence that an agent possesses is in direct correlation to the degree of use of independence, learning, cooperation and reasoning. The more of each of these areas that an agent uses, the more intelligent an agent's functionality is considered. As the

level of an agent's intelligence increases, though, the level of programming difficulty goes up dramatically. Agents should remain as simple as needed for their particular task. Agents that require a higher level of intelligence will take more thought and effort in their development. In a system that uses multiple agents, the degree of intelligence involved with the agents will vary based on the task of each one and must be kept as simple as possible to work effectively.

G.   SPECTRUM OF SOFTWARE AGENT CHARACTERISTICS

A software agent taxonomy classifies agents based on where they fall within seven different areas (Figure 2) [Ref. 6]. [1] This spectrum helps to identify the characteristics and abilities of agents performing various types of tasks.

1.   Intelligence

An agent's level of intelligence can be thought of in terms of whether the agent only executes a simple, specified task or has the ability to learn from the user and its own environment. The levels for this taxonomy are preference, rigid, reasoning, planning and learning.

2.   Rigid

A rigid agent is based on fixed, simple rules with no learning capability. It only executes specific instructions. The user must know exactly the information needed and, ideally, the source of that information to accomplish its task.

3.   Preference

Preference is based solely on evaluating decision criteria. The decision variables are built into the agent by the author or the user. It requires little intelligence on the part of the agent. An agent with little intelligence may have a problem in determining that two pieces of information are the same.

---

[1] Portions of the following section are taken verbatim from Dr. Bui's article, at his request, in order to ensure completeness in the technical report.

Software Agent Taxonomy

Intelligence					
Rigid	Preference	Reasoning	Planning	Learning	

Mobility	
Stationary	Mobile

Temporal		
Adhoc	Cloning	Persistent

Interaction		
Agent - Agent	Agent - Application	Agent - User

Task	
Specific	General

Environment	
Stable	Stochastic

Behavior						
Autonomy	Collaboration	Cooperation	Competitive	Champ	Relay	Crew

Figure 2. Software Agent Taxonomy

From Ref. [6]

15

4. Reasoning

An agent with reasoning ability can make decisions as discussed in previous sections.

5. Planning

An agent with an intelligence level of planning is able to plan actions independent of user input.

6. Learning

Learning is the highest level of intelligence. Learning means being able to learn and adapt behavior based on user's usage patterns and interaction with other agents.

7. Mobility

An agent has the capability to be either stationary or mobile. A stationary agent either resides on the client's machine or the server. Stationary agents do not move from their location but serve as an information-gathering source, typically on large-scale systems. These agents are monitoring sources, or agents whose mission necessitates a need to stay persistently on a system for extended periods of time. A mobile agent is able to package itself up, with all state information, and move from one location to another to execute its task. Mobile agents are not limited to the system on which they originate. They also have the ability to be called from a remote location to execute their task.

8. Temporal

With reference to the life of the agent, agents can be ad hoc, cloning or persistent. Ad hoc agents execute specific tasks then end gracefully. Cloning agents can replicate themselves. The ability to clone themselves allows faster search of multiple sources but can cause problems with agent coordination, collaboration and control. Persistent agents do not die after completing their tasks. They are allowed to live indefinitely. This allows constant monitoring to detect changing data.

9.   Interaction

Agents have the ability to interact with each other, with various applications and users. An agent that works with other agents may work in a peer-to-peer arrangement or a hierarchical one. This could require specific coordinating, facilitating or mitigating agents. An agent can work with applications such as databases, web browsers, spreadsheets, etc. A common interface language is essential to this type of agent because of the various operating systems and interfaces. Users can work with agents through graphical user interfaces or through the programming of the agents.

10.   Tasks

An agent's task is characterized as specific or general. An agent with a specific task is optimally designed for that one task only. An agent with a general task is a super agent, a jack-of-all-trades. It may be so general that it is not able to find a specific piece of information, but might discover information that the specific task agent missed due to its limited scope.

Two other types of task agents include information-specific and task-specific. [Ref. 9] An information-specific agent is programmed to deal with only a specific type of information. These agents know where to locate particular information in diverse networks or databases. A task-specific agent is designed to accomplish its mission regardless of the type of information requested. It can coordinate with other agents, if required to achieve its goal.

Front-end agents and back-end agents refer to two types of tasking for agents. Front-end agents directly interact with the user. The user interfaces with these agents in real-time, requesting information or providing guidance and direction for the agent. Back-end agents support the user but do not directly interact with him. These supporting agents do the behind-the-scenes tasks required by the user's system such as periodic updates and coordination. The user should not need to spend time directing or coordinating these agents.

11. Environments

During its TTL, the environment that the agent lives in is either stable or stochastic. A stable environment does not change. It is a secure environment. An agent that works in a stable environment is more likely to provide current, accurate data with little chance of providing wrong information. Little chance of attack, virus infection or interception exists for an agent in this particular environment. A stochastic environment for an agent is an insecure environment. Some probability of randomness and uncertainty exists. This environment would require additional skills and knowledge by the author when programming the agents. In this environment a much higher risk of attack, virus infection, interception plagues the agent.

12. Behavior

An agent can behave in many different ways depending on their task, intelligence and agency.

a. *Autonomy*

This agent works on its own. If an agent works on its own then it potentially does not have to worry about collaboration, cooperation or competition. The information it presents is a one-sided view of the information collected.

b. *Collaboration*

This agent works with other agents to solve a problem or complete a task. It has the benefit of additional sources to validate its information. The more sources that confirm the same information, the higher the likelihood that the information is valid and correct.

c. *Cooperative*

This agent assists other agents achieve their mission. It would not used primarily to get information, but is optimized to help other agents to work effectively.

d. *Competitive*

A competitive agent seeks to optimize itself even at the expense of degrading the performance of other agents.

e. *Champion*

A champion seeks to win no matter the outcome. This agent does not care what it has to do to complete its mission. It places itself at the top of the importance hierarchy.

f. *Relay*

A relay agent hands off to another agent when finished with its portion of the task. It passes state and information to another agent(s) to complete the final task.

g. *Crews*

Crews of agents work simultaneously with each other. This requires coordination among agents.

H. ISSUES OF MOBILE AND DISTRIBUTED AGENTS

Intelligent software agents can be of two types: static or mobile. Static agents reside on one computer system and never leave that system. Mobile agents have the ability to leave the system from which they originated and move to another system or possibly many different systems. Using mobile agents instead of static agents presents a number of different issues. [Ref. 6]

1. Programming

A mobile agent should be programmed in a language that allows the executing code to halt execution, preserving state and counter, and move to a different location and continue running. It must also be capable of running on a variety of systems. Multiple operating systems pose a significant challenge in programming an agent in a language understandable by all possible operating systems.

2.   Safety

If not controlled, an agent could potentially cause serious damage to a system. Protective measures and controls are required to prevent this from happening. An agent must not obtain access beyond what the host system allows for that particular agent. In this respect, the host is the vulnerable party and requires protection.

3.   Resource usage

An agent has the potential to monopolize the processor, hard drive and memory of the host system, if not controlled. However, the agent potentially needs some or all of the resources of the system to execute its task. The system should make allocations for the agent to execute, but not take over, the system resources in a manner that unfairly prevents anything else from using those resources. An equitable resource allowance for both the agent to accomplish its task and for the system to continue functioning is required.

4.   Navigation

In a networked environment, navigation presents more of a challenge than on a single system. Navigating to the right location to obtain the desired information is critical for the agent. An agent must understand the path(s) required to reach the intended location(s). In addition, the agent needs the path or means to send or provide the retrieved information back to the user.

5.   Privacy

A mobile agent travels to another system with its current state and programming code to be executed to obtain the desired information. The information inside the agent needs protection from outside sources. One school of thought is that all state information should be hidden from the host. The receiving system has no need to see any raw data carried by the agent or state information. Another viewpoint is that agents must be able to verify their state information and make necessary modifications to data carried within them [Ref. 10].

Another issue is that the information within the host system needs guarding against unauthorized disclosure. The host system protects itself by only revealing information to the agent that is allowed. The systems that the agent visits must be protected from unauthorized disclosure to the agent.

6.   Communication

A form of communication is needed between users and agents and possibly among agents. An agent that retrieves the desired information needs to communicate that back to the user in some format. An agent that cannot find the information has to inform the user of that as well. Based on the programming language, the communication medium has to allow for numerous types of interaction between the agent and the user. Agents may communicate with each other depending on their task. Sharing information between agents requires a communication protocol to effectively work together in a multi-agent environment.

7.   Control

Agents cannot run autonomously without some form of controls placed upon them. Controls can range from resource usage limitations, to authority to enter a particular system, to time allowed to live and execute once on a system. The challenge arises when the controls placed on the agent by the user/programmer conflict with controls placed upon the agent by the system. "The challenge today is to figure out how to best implement agents so they boost productivity rather than create chaos." [Ref. 7]

I.   TECHNIQUES FOR LAUNCHING INTELLIGENT SOFTWARE AGENTS

There are three techniques used to pass requests between an agent, a user and a server or host. They are remote procedure calls, remote programming and using middleware.

1. Synchronous communication-oriented remote procedure call (RPC). A remote procedure call is a traditional procedure call known as a request and reply cycle. The user sends out a request for another system to conduct a procedure. Once

21

the host system completes the procedure, it sends a reply back to the user with the requested information. This is the most common of the three types.

2. Asynchronous message-oriented agents. The messaging approach permits the distribution of data or control through the use of messages. This is also known as remote programming (RP). Message-oriented is more flexible and dynamic than RPC. It does not require a constant direct connection between client and server. RP can be accomplished using email.

3. Intermediaries or database middleware. Database middleware is a software layer that provides transparent access to homogeneous and heterogeneous relational or other databases across multiple protocol environments. Middleware is concerned with providing the agent access rather than passing messages between agents or with a server or host.

J. TEXT RETRIEVAL AND DOCUMENT MANAGEMENT ISSUES

Current computer systems have millions and billions of documents and files associated with them. Never in history have so many documents existed. Getting to the information needed requires modern tools, such as agents, to find exactly what the user wants and needs. Data filtering and data fusion both help the user get closer to the information they so desperately need. Search engines have helped with this process, but most do not allow the user to modify them in such a manner as to learn the user's preferences and desires.

1. Data Filtering

Data filtering is the process of sifting through volumes of data to determine the exact information being sought. With the huge amount of data readily available on today's computer systems, the ability to sift through it quickly and efficiently is crucial. Search engines used on the Internet, such as Infoseek and Excite, use data filtering to return the results of the search to the user.

Search engines currently on the market do a decent job of returning answers to search queries from large amounts of data in a timely manner. Keyword searches are the most prevalent types used. Some search engines use agent technology to assist them in their tasks. Most search results only have a few relevant hits concerning the information the

user actually wants. The user still is required to sift through the results, which is time consuming and potentially frustrating.

2. Data Fusion

Data fusion is analogous to the ongoing cognitive process used by humans to integrate data continually from their senses to make inferences about the external world. [Ref. 11]

Data fusion is much more involved than data filtering. The fusing of data requires a higher intelligence level in the software program. This issue is explored in much greater depth in Chapter VII.

III. BACKGROUND AND CONCEPTS

An agent enabled information system, tasked to support intelligence gathering, reconnaissance and operational planning, could consist of an organization of communicating software agents which gather, process, and distill data and information on behalf of the user.

The advantages to such a system are profound and far-reaching. The shear quantity of data available to human users makes the gathering of information almost impossible to accomplish exhaustively. Agents can help by gathering data and distilling it, removing redundant, or irrelevant data, and synthesizing it into a report or display a user can understand quickly. These systems can be built to run inside a user's Web browser, integrating all of the advantages of HTML. Browser technology allows for the display of text, graphics, and multimedia files.

In order to understand the way an agent based information system works, it is important to understand some key concepts in terms of their application to agent programming. These concepts are largely based on General Magic's work in the field [Ref. 4], as well as other researchers referenced in Chapter II. They are presented and expanded here to form a frame of reference for concepts presented throughout this thesis. The General Magic paradigm is referred to in this thesis because its mobile agent technology most closely fulfills the requirements of this project.

A.   REMOTE PROGRAMMING

Mobile agents work on a concept called remote programming (RP) where computer-to-computer interaction is accomplished by not only calling procedures in the remote computer, as in a remote procedure call, but by also supplying the procedure to be called. Each agent transported by the network includes a procedure to be executed on the host machine, and data that are its arguments. Therefore a client and a server can interact without ongoing communication. The architecture maps closely to the traditional

client/server architecture, but in this case the "client" is the user's local computer system, and the "server" is any remote system that provides mobile agent services.

The performance improvement of remote programming over traditional network communications depends on the network. It is more advantageous to use a remote programming paradigm on lower bandwidth networks like a wireless LAN at sea, than in high bandwidth fiber optic networks. A client need not be connected continuously to a remote server in order for the user's agent to gather information, storing it for transport to the user's computer the next time he or she logs on. The user's computer does not need to be connected while the agent carries out its assignment.

1. Mobile Agents

A mobile software agent is capable of interrupting its execution if certain user defined conditions are met, saving its current data and state information, and directing its own migration to another system or place. Once in the new place, the agent's execution picks up where it left off. In other words, the execution of a mobile agent appears continuous even if it migrates from one computer system to another.

2. Place

In mobile agent technology, a network of computers is a collection of places. A place offers a service to the mobile agents that enter it. The place is really a stationary agent designed to communicate directly with mobile agents, and provide a protected area in memory for the agent to execute. Agents, including stationary agents operating as a place, run within a virtual machine, and do not directly access the hardware, peripherals, or file systems of the host computer. This is primarily to prevent viruses, but it also insulates systems from poorly programmed or otherwise malicious agents.

In General Magic's Odyssey programming environment, for example, the agents run within the Java virtual machine, and are therefore removed from the operating system or hardware levels, enhancing the security and stability of the system as a whole. A further benefit of the abstraction is that, like other Java applications, an agent can run on different platforms.

Each place is programmed to provide a specialized service. These services can include file services, interfaces with local, third party search applications, or data base management systems. Thus, the programmer of a mobile software agent does not need to know the operating system, file structure, or database schema of the servers with which the agent will interact. He or she simply programs the agent to interact with a standard place, and the programmer of the place provides the database or file services, while the virtual machine provides an interface with the operating system.

A server may offer a place to get weather data, another place to obtain the latest intelligence data, and a third for geographic information for targeting. A host agent at the "Intel Place", for example, might check a user agent's credentials, security clearance, and access, then process its query, search the database, and return the requested information. Unauthorized agents would be refused and "destroyed," or deleted from memory. The host agent also knows about similar host agents, and can direct the mobile agent to them for additional information.

The Home Places on a client system are actually stationary agents that serve as the points of departure and return for agents that the user sends to remote places.
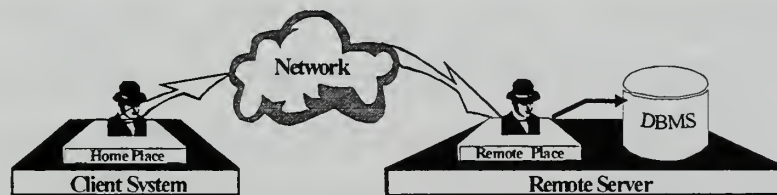
Figure 3. Agent Places

3. Travel

An agent can travel from one host to another *while it is running,* maintaining its procedure and state. If the trip succeeds, the agent's next instruction is executed at its destination. Thus, in effect, networking is reduced to a single instruction.

4. Ticket

A ticket is a network address, or a list of network addresses, to which an agent is authorized to travel. It may also specify the network the agent must use, restricting travel to the SIPRNET only, for example. The addresses on its ticket correspond to the systems on which the agent (user organization) has an account.

If a server place knows of another place that might have better information it can suggest that place to the agent. If the agent has an account on the suggested system, it can request permission from its user to change its ticket. If the suggestion comes from a trusted host the change may happen without human intervention. Otherwise, a server can create an agent of its own to retrieve the requested information, and leave behind a request to add an account for the new agent. Future searches will go directly to the new place, reducing the workload on the system as a whole. The agent learns. Alternatively, the agent can return home and inform its user of the new place. The user can request that an account for his agent be created on that machine and make the change to the agent's ticket himself.

5. Agent Accounts

An agent account is very similar to an account established for a user on any network, allowing the agent to logon to the remote server. The account identifies the authority the agent represents. Mobile agents must login just as human users must login, therefore agent accounts must be established in advance on systems to which a user might want to send his/her agents. An up-to-date account list allows the agents to visit all sites necessary to retrieve the information to answer queries. As new information sources are discovered, agent accounts must be established with those sources, and the account list updated. Agents can discover new sources through redirection by servers they have visited, or by user input. A system that does not have an updated account listing restricts the places where agents can go to those originally programmed.

There are various ways to manage the agent account listing. One method requires a human system administrator to establish accounts with each site that the mobile agents may

need to visit. Another way is to automate the system of account management so that a back-end agent can manage it.

a.   *Human Administrator*

One of the advantages of having a human administer the agent accounts is that the person can monitor and control where the agents visit. The human administrator can establish accounts on sites that meet all of the security procedures identified for that agent and prevent it from visiting sites that do not. The human can also prevent unknown agents from setting up accounts on his servers.

On the other hand, it can be very tedious and time consuming for a human to manage potentially thousands of agent accounts. Forcing an agent to return to its Home Place and wait for a human administrator to intervene and establish an account on a remote server negates the real-time capability of agents to locate and access new data sources. If the information is critical to the decision-making process, this can cause an unacceptable delay. It also restricts the learning of the system, in that the agents do not learn where to go on their own, but go only where told.

b.   *Agent Administrator*

Perhaps a better way to manage agent accounts is with another intelligent agent. This method relieves the human administrator of a tedious, repetitious task. A back-end agent can be developed to manage agent accounts. This agent can be programmed to establish a new account based on a set of rules established by a human system administrator or programmer. When an agent gets redirected, the back-end agent can determine if the new site can be trusted by using a trusted third party to confirm that the remote location has the safety properties required to run the agent [Ref. 10]. The trusted third party verifies that the mobile agent is properly configured and has the security clearance to visit the new site. The trusted third party also verifies that the server is who it says it is and can run the agent. Once satisfied on both accounts, the trusted third party creates a short program to send to the server. This program, when it is decrypted and run on the remote site, assures the server

29

that the agent is safe to run. Confident that the remote site is safe, the back-end agent can establish an account directly with the new server. The remote site can then run the mobile agent when it arrives and process the query.

There will be a delay as the back-end agent verifies the remote server with the trusted third party, but it will be much less than if the mobile agent had to return home and wait for a human administrator. The mobile agent can proceed to the next location on its ticket and await permission from the back-end agent via a connection, telling it that the requested site is safe.

c.   *Human User Accounts*

As in any other client-server system, the human users need to have accounts to access the system. These accounts provide an agent system with information regarding the clearance level of the user, which identifies the type of information that can be gathered and returned. The clearance level of the user is attached to each agent that is used to answer the query. This information is entered into the system when the system administrator establishes an account for the new user.

The user account also helps develop a profile on the user's preferences when receiving information. The profile is established through repetitive use of the system, which permits a user to personalize the presentation of information.

6.   Meetings

A meeting enables agents in the same computer call one another's procedures. These can be mobile agents sharing data, or more frequently, mobile agents meeting host agents, or places, on the host computer. The agent programming environment provides communications protocols for agent meetings. In the case of General Magic's Odyssey [Ref. 12], or IBM's Aglets [Ref. 13], these protocols are provided as Java classes.

7.   Connections

A connection allows agents in *different* computers communicate via messaging without traveling. Connections are useful in interactive applications where a mobile agent

may need to communicate with its human user. The agent that goes in search of weather information might send to an agent at home a small-scale map. The agent at home would present that map to the user via a graphical user interface and allow the user to specify more precisely the area he is interested in by drawing a box around it with his mouse. The home agent communicates that back to the remote agent who gathers the appropriate weather report and returns.

8. Authorities

The authority of an agent or place corresponds to the individual or organization in the physical world that the agent represents. The communications and security protocols (presented in Chapter IV) ensure that agents and places can reliably determine each other's identity. The protocol requires the verification of the authority of an agent each time it travels from one place to another. Places verify the authority of other places prior to transferring an agent to them. New places verify that an agent did in fact follow the path it reports as well as verifying the authority of the agents itself. In most cases, cryptographic forms of proof are required.

The process is analogous to a user logging on to a trusted system. The agent authenticates itself to the place, and the place authenticates itself to the agent. Agent or place identifications and passwords are exchanged and compared with the list of authorities with which the agent or place may communicate. Lack of anonymity also allows for auditing of agent or server activity. Users can know with confidence where their agent has been when it returns and any files it carries can be stamped with the originator's authority. Likewise, servers can track what agents have been serviced and charge accordingly. If hostile or unknown agents have attempted to penetrate the system, that event can be logged as well.

9. Region

A region is a collection of places provided by computers operated under the same authority. Usually the region would be the computer systems of a large organization maintained by the same Information Systems Department.

10.   Permits

A permit defines the set of capabilities that an agent can execute. Permits are granted to an agent by each authority with which the agent interacts. Its purpose is to limit the access and resource use of the agent. By itself, an agent cannot increase its capabilities.

Permits grant capabilities of two kinds:

1. The right to execute a certain instruction. For example, an agent's permit can give it the right to create other agents, or to clone itself. An agent can grant any agent it creates only capabilities it has itself.

2. A permit can also grant the right to use a certain resource in a certain amount. For example, an agent's permit can give it a maximum lifetime in seconds, a maximum size in bytes, or a maximum amount of computation resources.

Permits protect authorities by limiting the effects of errant and malicious agents and places. Such an agent threatens not only its own authority but also those of the place and region it occupies. For this reason the technology lets each of these three authorities assign an agent a permit. An agent can exercise a capability only to the extent that each of its three permits grants that capability. Thus an agent's effective permit is renegotiated whenever the agent travels. To enter a place or region an agent must agree to its restrictions. When the agent exits that place or region, its restrictions are lifted but those of another place or region are imposed. This feature provides some method of access control. If an agent must work only within a place, which in turn runs in the Java virtual machine, then conceivably at least, a single platform can service users of different security levels, and contain information at different classifications. Access can be controlled through the use of authorities and permits the way conventional systems use Access Control Lists.

11.   Allowance

An allowance is a portion of the permit that defines resource usage. An agent's allowance is renegotiated each time it travels and the agent's capabilities are limited by its most restrictive allowance. The allowance includes:

- **Time to Live**. Mobile software agents are persistent to the extent limited by their allowance. They may live indefinitely in the memory or file system of their current host. Systems administrators may not want agents to live indefinitely on their servers. They take up disk space and processor time and a busy server could become overloaded. For this reason, regions or places may limit an agent's time to live within that region or place.

- **Time Limit for Travel.** An agent can be configured to depart and return based on different criteria: completion of its mission; exhausting a list of network addresses in its ticket; if "emergency" or urgent information is discovered that corresponds with parameters set by the user; or some user defined time limit. Agents may be configured to go and live on a remote server indefinitely, watching for some parameter to be met, or some condition to become true, and then return and report. Agents can also be configured to search until a specified time and then return. Otherwise an agent may never come home, leaving the user to wonder if it was destroyed, captured, got lost, or is just still searching.

- **Maximum Size in Bytes.** There is a trade-off between a single large agent (carrying lots of data) and a smaller one that must make several trips to get all the information required. It is mostly a bandwidth issue. An agent may be programmed to spawn off another agent to carry data home while it continues its mission. Alternatively, data can be encrypted and e-mailed to reduce agent size. Max size may also be a parameter that dictates the time for an agent to return.

- **Processor Time Limits.** A place must be able to limit the system resources that agents can consume, otherwise they are vulnerable to a denial of service attack, or just greedy programming. Once an agent's allotment of processor time is exhausted it is sent on its way. Agents must be programmed with resource limitations in mind otherwise an exception may occur, or the agent might just die. Therefore, the agent itself must keep track of its processor time and ensure that it does not exceed its limits. Alternatively, a less robust solution would be to expect every place an agent visits to inform it that it has exceeded its resources and then cause the agent to travel to its next destination.

- **Agent Redirection.** As previously mentioned, places may know of other places where information can be found, and can refer an agent to them. If the agent carries with it a list of places that it is authorized to visit, then it does not need to request permission from its user to be redirected. When a referral is made the agent simply searches the list of places it may visit and if it finds a match, it goes. Otherwise, it sends a request to its user to be redirected to the new place. It may also request that the referrer send a mobile agent under its own authority to gather the information and then turn it over to the user's agent upon return.

This new agent can leave a request that an account be established on the new server.

- **Permissions to Execute Certain Instructions.** Permissions are granted to execute certain functionalities that are either programmed into the agent or are learned by the agent. Remote places could supply procedures for the agent to execute, or that are executed on behalf of the agent. Examples might be cloning, or extending the agent's list of trusted domains. The agent must then have a set of these instructions that it is allowed to execute. The list might also be region specific. Certain instructions can be executed in certain regions, or by certain regions. Again, if security is a concern (and it always is) one cannot allow agents to have their list of trusted domains extended by just any place.

- **Authorities with Which the Agent May Interact.** An agent must carry with it, in addition to its list of trusted domains and network addresses, the electronic signatures (encrypted) of all of the Authorities with which it may interact. A meeting always begins with a challenge and reply routine. Each agent or place must identify itself to the satisfaction of the other before any other communication can occur. Alternatively, the list of electronic signatures may be kept at the home place and authentication may happen by establishing a connection between the mobile agent and its home place.

B.   THE QUERY FORMAT

One of the advantages of mobile agents is that they can be programmed to make queries without knowing the structure of the data sources they might visit. Each place is programmed to accept queries in a "standard agent format" and reformat them to whatever the legacy data source residing on that machine is expecting. Therefore, an agent with a single natural language query can request information from a database, a search engine, or a newsgroup. The place is programmed to accept the query, translate it as appropriate, and pass it on to the application running on the server, whether it's a data base management system, a third party search application like Verity's Search97, or just a simple file of word processor documents.

At some level the query format must be standardized and structured. Even if it is not necessary for an agent to be able to communicate directly with a remote database or file system because that functionality is provided by the place, it is still necessary for the agent

to be able to communicate reliably with the place in which it is running. Some form of standard ontology is required.

C. AGENT ENVIRONMENT

An agent based information system is physically no different from any other networked information center. It is a client/server architecture where agents representing users are free to move from local client machines to remote servers, execute their code and either return, move to another server, or send information back through a connection. The agent may either die in place, or become dormant, "living" in the host server's memory or file system, waiting for preset criteria to be met. The criteria are limited by the agent's permit.

A software agent really only exists as an agent within the CPU or memory of its host computer. Therefore agent technology is compatible with all network technologies. Whether the network protocol is Ethernet, TCP/IP, token ring, etc., the agent is transported over the network in packets just like any other data, and is reassembled in the receiving computer. In a TCP/IP network, the datagrams are labeled with the port number associated with the agent place. When packets arrive they are assembled in the place and the agent is run. Further, if software agents and places are programmed in Java, then they are compatible with virtually all computer systems currently in use.

IV. SECURITY

The more complex an information system becomes, the more complex the security requirements become. Network security is far more complicated than single platform security. The use of software agents on a network, intranet, or internet adds even more complexity to the problem. There are a seemingly inexhaustible number of security threats that an agent based information system must address in order to satisfy the security requirements of the users and administrators of the system. The first section of this chapter outlines the various areas of computer security. The following sections describe the major security concerns for agent based information systems in terms of authentication, integrity and secrecy. The final section of this chapter presents a security protocol for an agent based information system.

A.   DEFINITIONS

The security of an information system ensures the secrecy, integrity, and availability of the information stored and processed in that system. Computer security is primarily an exercise in identifying the vulnerabilities of a computer system and devising ways to protect against threats that may try to exploit those vulnerabilities. Security policies aim to reduce the likelihood that an attack on a system will be successful, or at least to make it prohibitively expensive for an attacker to successfully find and exploit any vulnerability that may exist. Further, security policies ensure the integrity and availability of the system and the information by protecting them from inadvertent or accidental damage by nature, legitimate users, programmers, or errant applications.

1.   Secrecy

The confidentiality of information in a computer system is maintained most often through controlling who has access to a system, or parts of a system, and through the use of encryption, which ensures that only those people who are authorized to access information can read it.

In networks, secrecy may also require protection from interception by unauthorized users during transmission. This may be accomplished by the physical protection of cabling and access controls; protection from traffic flow analysis where inferences can be drawn simply from analyzing patterns in the communication among parties; and selective routing of messages, which avoids threats by restricting communications to specific networks.

2. Integrity

A secure computer system must maintain the integrity of the information stored in it. Integrity refers to the accuracy and reliability of the software and information on a system. Secure computer systems must prevent the accidental or intentional corruption of information by preventing unauthorized *write* access to the information. Low assurance processes must not be allowed to write to high assurance data. In communications this ensures that messages are not forged or modified during transmission.

3. Non-repudiation

Repudiation is to reject the validity or authenticity of a communication. Computer security requires that parties not repudiate legitimate communications once they have occurred. For example, the author of an e-mail message may falsely claim not to have sent it. This is most often prevented through auditing, and the use of encryption and digital signatures.

4. Communications Security

Communications security is the protection of information while it is being transmitted. In computer systems this most often involves cryptographic protocols which rely on public and private key cryptography.

5. Availability

A secure computer system must keep information continuously available to its authorized users. Its hardware and software should continue working efficiently under various loads, and degrade gracefully, if at all, under extreme conditions of use. Processes

must not interfere with the efficient execution of other processes. Availability also refers to the graceful, reliable recovery of the system and data in case of a crash.

6. Authenticity

In networked systems, authenticity provides a means of ensuring that only authorized users have access to the system. The system verifies the origin or destination of data by recording who sent or requested it, ensuring that the subject has the proper authority to access the data, and recording the time and date that the access occurred. This is most often handled through *identification* and *authentication*. A user identifies himself to a server and authenticates his identity with some form of proof. To ensure the security of the session the server must also identify and authenticate itself to the user. This is most often handled through user names and passwords, but there are far more secure protocols available that can not only ensure the authenticity of the user and server, but can also protect the integrity and secrecy of the information passed in the session. The two most commonly used protocols for secure network communications are Secure Sockets Layer (SSL), common in internet communications, and third party certificates like Verisign's proprietary protocol for authentication and encryption.

a. Secure Sockets Layer

Secure Sockets Layer is the most widely used Internet security protocol. When a Web browser first connects to a secure Web server, the Web server sends a *hello request* to the browser. The browser responds with a *client hello* that contains a number called a session ID that uniquely identifies the current session. The *client hello* also tells the server which cryptographic algorithms, compression technologies, and SSL version the browser supports. Finally, it includes a random number that the browser generates. The server will respond with a *server hello* that includes the selected compression and cryptographic algorithm from the browser's list, the appropriate SSL version, another random number, and an acceptable session ID. This first set of communications is called the handshake, and it establishes the protocols that will be used through the rest of the

session. Once this is accomplished the server will send a digital certificate to the client which includes a public key to be used in the public-key encryption algorithm selected during the handshake. The public key protocol is used to securely pass a private key, which will be used during the rest of the session. Public key cryptography, while generally considered more secure than private key cryptography, is far more computationally expensive and is therefore used only to exchange the private key, which is generated for this one session, by the browser using the two random numbers exchanged in the handshake. As an added protection from eavesdropping, the true private key is not sent. Instead the browser sends a "premaster secret key". Using the random numbers generated during the handshake, and a predetermined algorithm, the server can then determine the true master key. Once this process is complete, both the browser and the server have copies of the master key and can communicate securely. [Ref. 14]

b. Digital Certificates and the Verisign Digital ID

Digital certificates are used as a method of distributing public keys in a way that ensures both authenticity and integrity. A public key encryption protocol is exceptionally resistant to compromise provided that private keys can be kept private, and public keys can be distributed in such a way that they can have only come from the party they claim to have come from. A certificate contains the name and network address of the person who owns it, as well as that person's public key. The certification agency that distributes the certificate guarantees its authenticity by digitally signing the certificate with their private key.

Verisign Digital ID was developed by RSA Data Security and works much the same way as the SSL protocol. This protocol also uses public key cryptography to exchange a private key. When a connection is established between a client and a secure server, the client software automatically verifies the server by checking the validity of the server's digital ID. The key pair associated with the server's digital ID is then used to encrypt and verify a session key that is passed between the client and the server. This session key is then used to encrypt the session. A different session key is used for each

client-server connection, and the session key automatically expires in 24 hours. [Ref. 15] Verisign provides a level of security that SSL does not. Recall in the SSL session described above, the server sends a digital certificate to the browser, but the browser does not authenticate itself to the server. By assigning digital IDs to individuals and organizations through the use of digital certificates, Verisign can guarantee the authenticity of the client and the server to each other.

B.   SECURITY IN AGENT BASED INFORMATION SYSTEMS

The use of software agents in an information system complicates the traditional client/server security problem. Mobile agents are software programs that run on host computers. They can come from anywhere on the network and can potentially be written by anyone. Their similarity to computer viruses is compelling, and the potential for a Trojan Horse attack in an agent based information system is significant.

1.   Threat to Servers

a.   Potential threat

The most obvious security threats in information systems are to the server's system and information. [Ref. 10] They include:

1. Damage to the host's file system. Modification or removal either of data files or of already-resident executables would violate system integrity.

2. Downgrading the system's availability. Excessive use of the host's resources, such as CPU time, main memory, or file systems can lead to the point where the mobile code effectively disables the other processes running on it. This can be either deliberate, as in a denial of service attack, or simply a result of sloppy programming. The most effective prevention of this type of threat is by empowering the server to limit an agent's allowance.

3. Compromise of secrecy, such as the leakage of private information belonging to the host's established users. Such leakage could be intentional and malicious (e.g., the mobile code could have been written to accomplish theft of information) or it may be unintentional and have unknown consequences (e.g.,

the mobile code or some other program with which it communicates could have a bug in its cryptography module).

4. Unauthorized access to the organization's network resources and nodes. Mobile code, once running on a server, may attempt to access private intranets or networks within the organization.

b.   *Addressing the threat*

Most of these threats fall into the categories of sloppy agent programming or malicious software. As mentioned previously, authentication is important for both the server and the client, which in this case is a mobile agent executing its code within the server's virtual machine. Fortunately the authentication of software agents to the server is really no different from authenticating a remote user to the server. The use of certificates, hashed checksums, and cryptography all lend themselves to the reliable authentication of software agents. Once an agent is properly authenticated, and its integrity verified, the threat of malicious subversion is significantly reduced. Servers can protect themselves from sloppy programming through the use of allowances and permits.

2.   Threats to Agents

a.   *Potential threat*

Because an agent relies on a host computer for its execution, malicious servers have the ability to alter the data or code contained in an agent. Users must be assured that their agents were not compromised while visiting a series of places. In another form of the denial of service attack, network devices can be programmed to watch for data packets from or to certain servers and intercept, redirect, or destroy them.

b.   *Addressing the Threat*

The obvious counter to these security threats is through authentication and encryption. But with mobile code that relies on a host computer for execution, the issue of authentication is problematic. The fundamental reality of mobile code is that a mobile software agent is *not* some autonomous entity that can travel the infosphere independently. In all cases, the agent must run in the memory and central processing unit of a computer

somewhere. For this reason, there is no way for an agent to reliably determine the identity of the server on which it is currently running. Even cryptographic forms of proof do not help because the agent must rely on the host computer to compute the cryptographic identity. Even if the agent only carries hashed passwords, and the server cannot determine the correct answer to the identification and authentication challenge, then the server can simply cause the program counter to bypass that code and take the conditional branches that correspond to a correct answer. Since it is possible for a malicious server to spoof the agent, it is important to audit the path an agent takes through the network.

When an agent arrives in a new Place, the agent can query not only the identity of the current Place, but also the identity of the Place from which it traveled. If server to server communications use authenticated and encrypted channels, then each server will know from where an agent traveled. If the agent is currently running on a hostile server, then the agent may be spoofed. But, if the location is logged, when the agent travels to an honest server, the prior location in the log will not match the location from which the agent traveled, as reported by the new Place. The agent will know that it was redirected.

3. Information Security

Any data carried by an agent is not necessarily secure for the same reasons. The information that an agent carries, including the read-write state of the agent's variables, is vulnerable to a malicious server. Therefore, an agent that has just left a malicious server cannot be trusted. There is no way to determine whether the agent's code, data, or execution state have been altered. All information collected prior to this point in time, including data from servers prior to visiting the malicious server, is now suspect. It may be preferable, therefore, for data to be encrypted and e-mailed back to the user's system from each host. This makes for a much smaller agent, but requires all data processing, fusion, redundancy elimination, etc., to be handled by the user's computer. It also increases network bandwidth requirements in that each e-mail carries a certain amount of overhead, and redundancies or irrelevant data cannot be eliminated "in the field". This is offset, however, by the fact that the agent itself does not ever return.

Alternatively, if each server on which an agent executes encrypts the results of the agent's query with the agent's public key, then the data from each server will be secure from manipulation or discovery from any other server the agent visits. When the agent returns home, each message can be decrypted using the agent's private key on the user's own system. This alternative does not have any obvious advantage over the e-mail alternative discussed previously. An agent still can't eliminate irrelevant information in the field because the information would have to be decrypted to be processed, and that must be done on a server. This scenario might work when an agent's mission is to collect reports from various Places. But if the agent's mission is to gather and compare data, for example, find the lowest airfare among all the available servers, then the most efficient method would be for the agent to carry the lowest fare with it from server to server, discarding data from higher priced servers.

4.   Transport Network

Classified information can be protected through encryption, but hostile agents can still conceivably capture or destroy friendly agents. Even though an agent's permit allows it to communicate only with certain other agents, it may be possible for a hostile agent to spoof friendly agents. Also, agents traveling over the public Internet are vulnerable to "hostile" routers or gateways. A router can be programmed to read the source and destination network addresses and route the packets to a hostile computer, or just destroy them as they arrive. Critical agents should probably only travel over trusted networks. Additionally, certain resources are accessible only over a specific network. Secure servers are not connected to the Internet, but may reside on the SIPRNET, while weather or news resources might be most easily accessed through the World Wide Web.

C.   A PROTOCOL FOR SECURE AGENT/SERVER INTERACTION

In order for a secure agent based information system to be constructed, a security protocol that reduces the threats discussed in the previous sections to an acceptable level must be developed. The technologies for such a protocol are already well established and

understood. By leveraging existing technologies like Secure Sockets Layer, digital certificates, public and private key cryptography, and hash functions, and applying the concepts of agent accounts, allowances, authorities, and permits to agent based information systems, a security protocol for agent/server interaction can be created.

1.   Server to Server Interaction

When a mobile software agent desires transport from one server to another, an SSL connection is established and certificates are exchanged. The process of using certificates and SSL technology reduces the risk that an agent will be intercepted or redirected. The model assumes that the agent's journey starts on an honest server. Each server positively identifies the next server in the route using digital certificates. The agent and its data are encrypted and signed, and then transmitted to the server which was just positively identified. Because public key cryptography is computationally expensive, it is desirable that a hybrid scheme be used where a one time private key is created and encrypted using RSA. The agent itself is then encrypted using the private key. This method is very much like the SSL protocol described earlier. Applied to an agent system, the protocol works like this:

1. Sender and Receiver establish a secure session by exchanging digital certificates using SSL.

2. Sender uses a mutually available hash function, H, to produce a hash $H(A)_{sender}$ of the original agent and data, A.

3. Sender signs $H(A)_{sender}$ by encrypting it with its private key $K_{sender-pri}$ producing $H(A)'_{sender}$.

4. The agent and data, A, and the signature, $H(A)'_{sender}$, are then encrypted using the session key established in step 1 producing a single encrypted message, M'.

5. Sender transmits M' to receiver.

6. Receiver decrypts M' using the session key created in step 1, obtaining the agent and data, A, and the signature, $H(A)'_{sender}$.

7. Receiver decrypts $H(A)'_{sender}$ using Sender's public key, $K_{sender-pub}$ producing $H(A)_{sender}$.

8. Receiver uses the same hash function to hash the agent and data, A, producing $H(A)_{receiver}$.

9. If $H(A)_{sender} = H(A)_{receiver}$ then the message came from Sender and has not been altered in transit.

Once the server has determined that the agent did in fact come from Sender, and that the integrity of the agent has been preserved, the agent enters the Place on the new server and executes.

2. Redirection and Interception of Mobile Agents

It is possible in TCP/IP networks for packets to be intercepted by a router and redirected to another server. To further prevent subversion by hostile servers, prior to leaving its current Place, an agent will log the *next* Place it intends to visit during a journey and hash the log entry to prevent tampering. The first instruction the agent executes in a new Place is to ensure that the *last* Place in its log matches the hash of the identity of the server from which it intended to travel. If the identity does not match, then the agent has been subverted and can handle the exception as its programmer sees fit, most likely by notifying its user that it has been compromised and dying. The protocol works with SSL as follows:

1. The mobile agent starts at the home Place and logs its Home Place identification, H, and that it is traveling to Place A. Both addresses are hashed producing H(H) and H(A), making up the first two entries in the agent's travel log.

2. H establishes an SSL session with A, exchanging certificates and verifying addresses, and then forwards the agent to A.

3. Agent arrives at A and checks that it did in fact come from H by hashing the address that server A reports it came from and comparing the result to H(H). If they match, then the agent has not been subverted.

4. Prior to departing Place A for B, the agent hashes the identification of Place B and adds that entry to its travel log.

5. Enroute to B, the agent is intercepted and redirected to a hostile server Z. Z reports correctly that the agent came from A and then causes the agent to believe that it is now on Place B. The agent and any information it is carrying have now been compromised, but neither the agent, nor its user would realize that at this time. At this point, Z may introduce misinformation to the agent or append a Trojan Horse and send the agent Home. The agent logs that it is returning to H.

6. Z establishes a connection to H and forwards the corrupted agent home. When the agent arrives on H, it requests the address of the Place from which it has traveled, hashes it and compares it to the address it logged while still on A, H(B). The result will not match and the user will know that his agent has been compromised and execution will be immediately stopped, preventing any further damage.

Note that this system relies on the integrity of the certificates passed between servers in the SSL protocol to ensure that servers cannot misrepresent themselves to each other. This is reasonable because the verification of the certificates happens on two independent processors, instead of in one processor, as is the case of an agent authenticating the server on which it is currently executing.

V. A FRAMEWORK FOR AGENT-BASED DECISION SUPPORT: THE MOBILE AGENT RECONNAISSANCE KIT (MARK)

In its initial concept, the proposed system supports the decision-maker by providing reliable information to aid in critical decision making. Intelligent software agents are not intended to replace humans in the decision process. The MARK system is intended to support the human user. It is also intended to work with existing systems to reduce the decision loop by providing accurate, detailed information in a timely manner. MARK acts as an intelligent personal assistant to the user, by doing the tasks described in Chapter II. The intelligent software agents can support the human by doing tasks that are repetitive and require searching through vast amounts of data. MARK can do computationally intensive tasks involving multiple variables faster than a human can. MARK can do data comparison and integration to assist the human user to develop a comprehensive picture of the situation. Computers are more efficient and faster at filtering through volumes of data and identifying possible trends than humans are. They are also less likely to discard a piece of data just because it does not fit a preconceived notion. One of the goals of intelligent software agents is to highlight options that the human might have overlooked or dismissed as insignificant because he/she did not see the cause-effect relationship that that option presents.

Humans generally prefer to make the ultimate decision when that decision involves human lives because few humans are willing to trust a computer to make the "best decision". Intelligent software agents can make decisions on which airline ticket to buy, for instance, because it involves a straight comparison of measurable variables. But an intelligent software agent should not replace a human regarding human life decisions. It is difficult to provide computers with intuition and program them to make value judgements. Intelligent software agents do not consider such things as fear, desperation, or greed. These things cause humans to react in ways that may be contradictory to the "logical action"

expected based on the facts. For this reason, MARK provides the human with information to support making a decision, but does not make the decision for the human user.

Humans and intelligent software agents can work as an interactive team, or intelligent software agents can be used to free the humans from time-consuming, repetitive tasks so they can concentrate on making decisions.

MARK consists of a hierarchy of agents that communicate with each other over a distributed computing environment. The system architecture will be examined from the points of view of the client machines, and the server machines, and will be illustrated with a scenario.

In the Anchor Desk paradigm introduced in Copernicus [Ref. 16], the information used to plan and execute a mission comes from a watchstander stationed at a specialized workstation called an anchor desk. This watchstander has access to the world of information through computer networks and internets. The information that he needs might include weather forecasts for the area of a strike, enemy strength estimates, geographic and targeting data, etc. The information may come from servers or other clients on the watchstander's local area network, or it may come from remote servers accessed through the SIPRNET, or the world wide Internet. The data sources may be unique, as in the case of intelligence estimates, or varied where any credible weather report will do.

Since the type of information required to plan and to execute a particular mission is usually the same each time, specialized applications can be developed and kept in a library for easy access by planners. There may be agent applications for non-combatant evacuation operations, strike missions, amphibious assaults, hostage rescue, etc. When a certain mission is identified, planners need only go to the Central Agent Repository (CAR) and load the appropriate application.

The overall architecture for MARK is shown conceptually in Figure 4, and discussed in terms of client and server functions in the following sections.
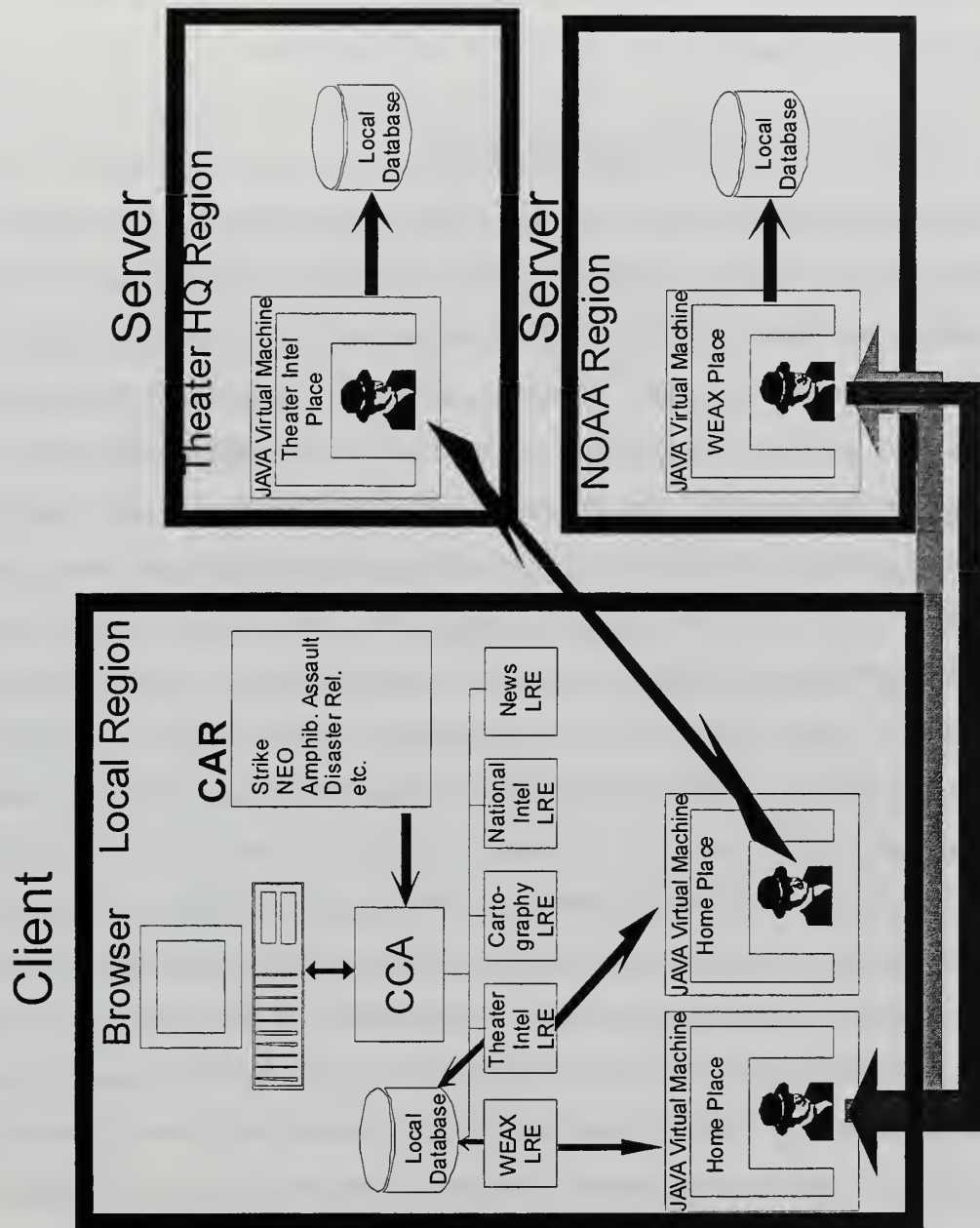
Figure 4. Mobile Agent Reconnaissance Kit

The terms client and server refer to agent clients and servers. The concept is analogous to the traditional client/server architecture in network computing. However, in agent computing, the term client refers to the user's system, and server refers to remote systems within a particular region.

A.   CONCEPT OF SYSTEM OPERATION

Information needed to plan a mission may come from a variety of sources, some local and some remote. The planner opens the CAR and loads a module. This Java application works inside a Web browser and displays information in HTML format, with hypertext links to the source data. Specific parameters are set, including, for example, the area of a strike, expected date and time, and any limits on the sources of information. The application is then executed and the CCA directs the Weather, Theater Intelligence, Cartography, National Intelligence, and News LREs to begin gathering the data required to execute the mission. Each LRE searches locally for data. In addition, the LRE may call a Home Place and pass query and agent configuration information to it with instructions to create a mobile agent to search for the information at remote places. For example, the Weather LRE receives weather reports from outside weather sources like the National Oceanographic and Atmospheric Administration (NOAA) and the National Weather Service, which it files for future reference. If these reports are sufficient, it retrieves the information locally and creates an HTML file, complete with weather maps, and makes it available to the CCA. If it does not have a suitable report, it calls a Home Place, which creates a mobile agent that travels to a weather place on a remote server and retrieves one. When the agent returns, its identity and authority are verified, and its data is passed to the weather place on the user's computer. The data is then passed on to the Weather LRE where it is formatted in HTML with links to the appropriate page on the remote server, and made available to the CCA which displays the report in the user's browser.

LREs routinely receive reports from external agencies, and file them for future use. The LREs are set up to monitor these reports as well as reports from its own agents for

52

certain parameters such as troop levels, rates of buildup or movement, types of threats present in an area, etc., and alert the CCA when the user defined parameters are exceeded.

The system can be programmed to handle natural language queries in a manner similar to Autonomy's AgentWare [Ref. 17]. Further discussion of agent products like Autonomy's AgentWare is provided in Chapter VIII. These queries can be refined by telling the system to "find more like this", where the application learns by example. As the system is used, user feedback and agent training will enable trend analysis of statistics concerning the quality of information returned from each source, quantity of redundant data, and irrelevant data filtered. Trends observed based on these statistics will enable the system to improve its performance over time.

1.   The Client

Each client platform will have a Central Coordinating Agent (CCA) that coordinates the activities of all subordinate agents in the system. Subordinate stationary agents, known as Local Resident Experts (LRE), are modules in the client application that specialize in collecting data from specific data sources. They may control subordinate agents of their own called Home Places which in turn create and control mobile agents. These mobile agents are specifically designed to search out and return specific types of information from remote places. The information is then passed up to the CCA, which formats it for display in the user's Web browser. Subordinate places are charged with controlling agents they spawn, setting authorities and permits of agents they send out, and verifying the security of those agents when they return with data. Figure 5 summarizes the roles of various controlling agents on a client machine. This figure provides a sample of potential LREs and is not intended to be all-inclusive. For the purposes of the thesis, MARK will illustrate how intelligent software agents can be used to support Intelligence related tasks. The model can also be used, however, for the day-to-day management of other areas, such as logistics or personnel, with other CCA and LRE modules. The CCA, LRE, and Home Places are examined in greater detail in the following sub-sections.

Central Coordinating Agent

- Coordinates all subordinate modules
- Interfaces with user through browser
- Collects information gathered by subordinate modules and passes to user interface.

Local Resident Experts

Weather LRE	**Theater Intel LRE**	**Cartography LRE**	**National Intel LRE**	**Commercial News LRE**
• Searches local data sources	• Searches local data sources	• Searches local data sources	• Searches local data sources	• Searches local data sources
• Receives and processes WEAX reports	• Receives and processes Theater Intel reports	• Receives geographic information	• Receives and processes National Level Intel reports	• Receives and processes news feeds. Stores in local database
• Controls mobile agents that query remote data sources through Home Place	• Controls mobile agents that query remote data sources through Home Place	• Formats maps and charts for display in user's browser	• Controls mobile agents that query remote data sources through Home Place	• Controls mobile agents that query remote data sources through Home Place
	• Alerts user when predefined conditions are met	• Controls mobile agents that query remote data sources through Home Place	• Alerts user when predefined conditions are met	• Alerts user when predefined conditions are met

Figure 5. Client System Controlling Agents

a.   *The Central Coordinating Agent*

The CCA is the heart of an agent application that is tailored to a specific mission. It is a top-level control module that coordinates the overall actions of the other modules in the application. The CCA consists of a list of predefined queries specific to the mission it was developed to support. These queries are common to most missions of a particular type. For example, the strike application would have queries to determine weather forecasts for the launch, route, and target areas; enemy troop and air defense estimates; and maps of the routes and surrounding areas. These queries are standard and are automatically passed with appropriate parameters to the cognizant LRE by the CCA. In addition, natural language, ad hoc queries can be entered by the user and passed to a designated LRE. The user interface relies on the elegant simplicity of a Web browser and passes commands to the CCA, and displays the HTML files the CCA returns as answers to the user's queries.

b.   *Local Resident Expert*

The CCA passes queries to the appropriate LRE. Ad hoc, natural language queries are also processed and interpreted and are then passed on to the appropriate LRE for action. The LRE interfaces with third party search applications and local database management systems. It may generate Home Places that in turn create mobile agents traveling and searching for information remotely.

c.   *Home Place*

The Home Place is a Java application that creates and executes mobile software agents. Queries and configuration parameters are passed to it by its associated LRE. The Home Place creates a mobile agent and issues it a ticket for travel, its authority, rights to execute special instructions, and lists of alternative locations the agent may be redirected to that are not on its ticket. The Home Place also includes the hashed electronic signatures of authorities the agent may interact with, including mobile and stationary agents, and the set of queries to be searched. The Home Place receives returning mobile

agents or messages they send back, decrypts and verifies their integrity, and passes any information they carry to the appropriate LRE for processing.

2. The Server

On server machines, each place verifies the authority of, and sets permits for, each of the agents that enter it. If an agent cannot prove its identity to the place's satisfaction, access is refused and the mobile agent handles the exception by returning to its home and reporting that access was refused. Since an agent carries the authority of its user, each agent that visits the server's place must have an account at that place. The system administrator sets up accounts. An account will have all necessary security information to control agents that arrive in a place, verify authorities, and set permissions. As long as an agent is running on a server, *it never leaves the place*. That is, it runs inside the Java virtual machine and cannot access hardware or files directly. All services are provided by the place through the Java virtual machine. This setup allows for a secure operating environment, and helps ensure that poorly written or hostile agents cannot cause damage to the server, or view files they do not have permission to see. Also, since all services are provided by the place, agents need not know anything about the structure of the database or file system on the host server. As long as the agent and the place can communicate, the data can be retrieved.

Once an agent's credentials are verified, it is allowed to run within the place. The agent executes within the Java Virtual Machine, and queries the place for the requested data. The place in turn verifies the authority of the agent based on the security classification of the requested data. If the agent's authority allows it to retrieve the information requested, the place queries the database on that computer, and provides the agent with the requested information. Alternatively, places on remote servers know about similar places and can refer a user's agent to those places if the agent's ticket allows.

B. INFORMATION FLOW

This section uses a scenario to examine the information flow between the stationary and mobile agent modules of the proposed system.

JTF 398 is established to coordinate operations against Islandia. JTF 398 is embarked in a ship at sea, away from its homeport, and is relying on satellite communications links. The Commander is tasked with developing several courses of action (COA) against possible targets in Islandia. The Operations Officer must develop a strike plan against an airfield and submit it to the Commander for inclusion in the COAs.

The information needed to plan the mission may come from a variety of sources, some local and others remote. The strike planner activates MARK and retrieves the Strike Module from the CAR. The top-level user interface is shown in Figure 6.
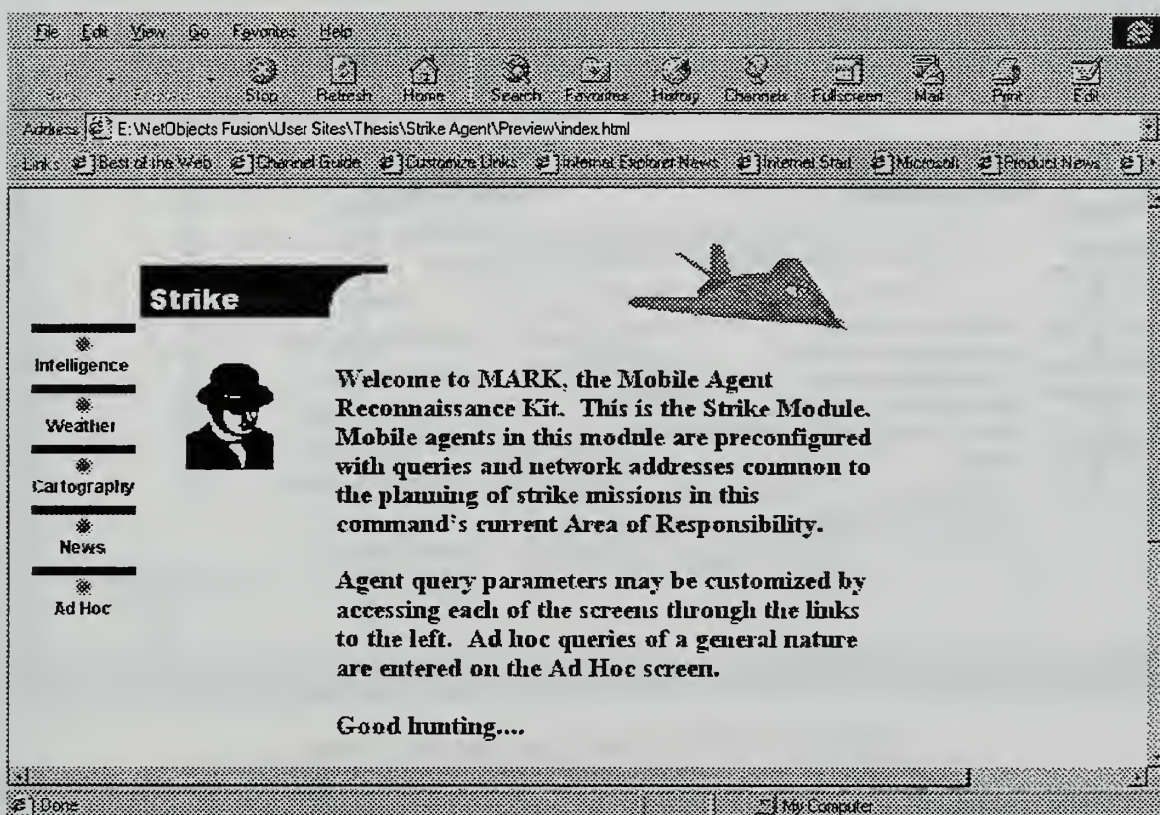
Figure 6. MARK Top Level Interface

The planner enters requests for strike information concerning a particular target on screens accessed from the links on the left. These screens correspond to the various LREs that are pre-packaged with the strike module and are shown in Figure 6. The CCA analyzes the request and provides tasking to the appropriate LRE. Each LRE is

preprogrammed and "trained" to locate information from particular locations, based on prior use and user feedback. However, the planner can modify the LRE parameters by selecting the link, if he has particular instructions for this mission.

The JTF 398 strike planner knows that he needs the Intelligence LRE to pay particular attention to the buildings north of the landing strip. After selecting the Intelligence LRE button, a MARK agent configuration screen appears, as shown in Figure 7.
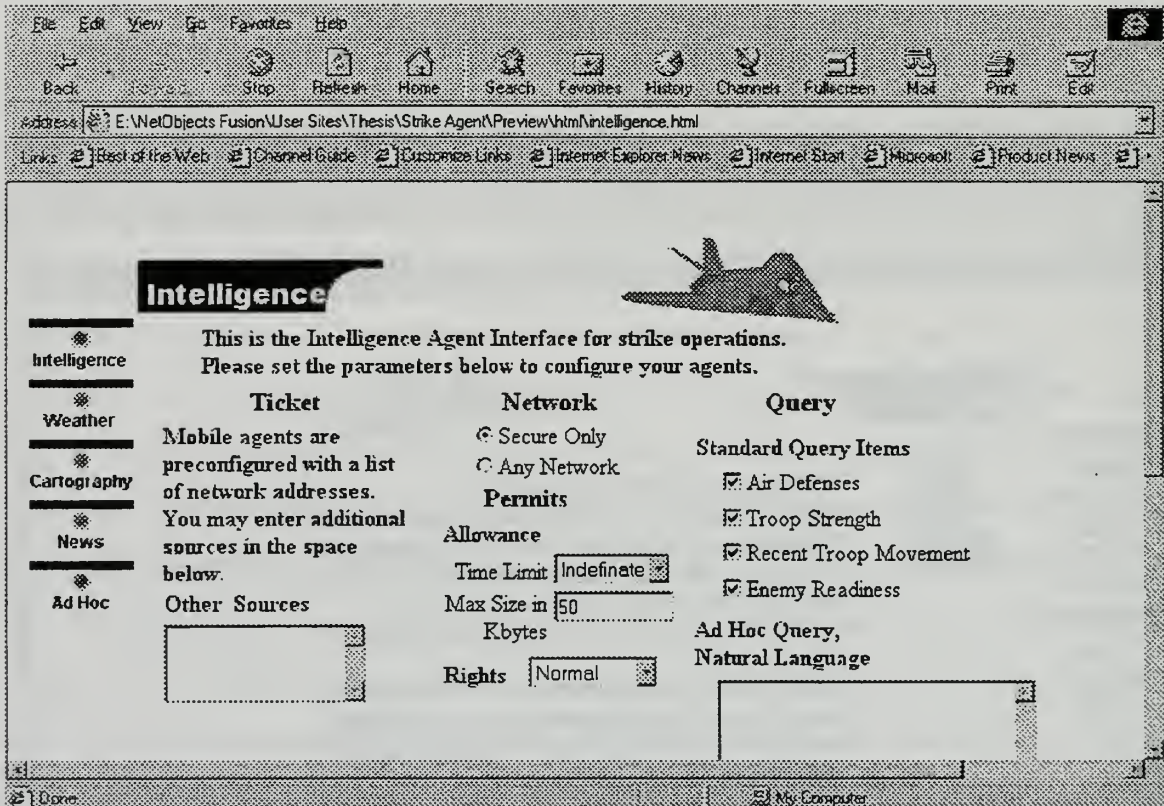
Figure 7. MARK Agent Configuration Screen

The standard settings for the LRE are displayed on the screen. The planner enters the ad hoc query in the dialog box and submits the change. The information is combined with the tasking that the CCA has passed to the Intelligence LRE based on the initial request for information. The Intelligence LRE searches the local databases for the

appropriate data to satisfy the request. If it is found, the LRE passes the data back to the CCA to be combined with the returned data from other LREs and formatted for the user.

If the data on the buildings north of the landing strip is not available locally, the Intelligence LRE calls a Home Place to spawn a mobile agent to search remote regions. The Intelligence LRE passes the query and configuration parameters based on the planner's input from the configuration screen. The Home Place creates a mobile agent and issues a ticket, the agent's authority, the permit and the search query, as shown in Figure 8.

Ticket
    IP address
    Transport network
    Time limit for trip
Authority
    Electronic signature/certificate
Permits
    Allowance
        Time to live
        Max size in bytes
        Processor time limits
    Right to execute certain instructions
    List of places it may be redirected to
    Authorities it may interact with
        Electronic signatures/certificates of all remote
        agents it may interact with
Query

Figure 8. Information Passed from Home Place to Mobile Agent

The Home Place establishes a Secure Sockets Layer session with the first server on the agent's ticket, in order to allow the agent to travel securely in accordance with the protocols presented in Chapter IV. When the protocols have been satisfied, the mobile agent leaves the local region and travels to the first ticket destination. Upon arrival at the server place, the mobile agent is executed within the Java virtual machine. The agent

passes its authority to the new place and receives the authority of the place in return. The authorities are hashed and compared to the certificates on file in the remote place or carried with the agent, as appropriate.

Once each party is satisfied that the other's authority is correct then the remote place issues the agent a local permit defining its capabilities on that server. The agent then passes it's the query about the buildings to the place, which examines it and marks it with a security stamp defining the level of access that agent has on the remote system. The place passes the query to a back-end agent, which formats the query for the local data source, runs the query, and returns the results to the remote place. The remote place delivers to the mobile agent the information that the north buildings are owned by a company that works for the military, but may have been converted recently to house an orphanage. The mobile agent puts a date/time/location stamp on the data, encrypts it and either encapsulates it to carry, or turns it over to the remote place to be sent back to the Home Place via e-mail. This exchange is shown graphically in Figure 9.

In this case, the mobile agent carries the information back to its Home Place. The mobile agent passes its authority to the home place, which verifies the agent's identity. The Home Place then validates itself to the agent and verifies the agent's list of certificates from remote places that the agent has visited. When the security checks have been completed, the agent passes the query result to the home place, which passes it to the Intelligence LRE. The LRE examines the data for relevancy and eliminates redundancies. The requested data about the buildings north of the field is combined with the information gathered from the standard queries and is passed to the CCA. The CCA combines the data from the Intelligence LRE with the information provided by the other LREs into the appropriate report form and passes it to the browser. The path of the query result is shown in Figure 10. Figure 11 illustrates the scenario in an information flow diagram.
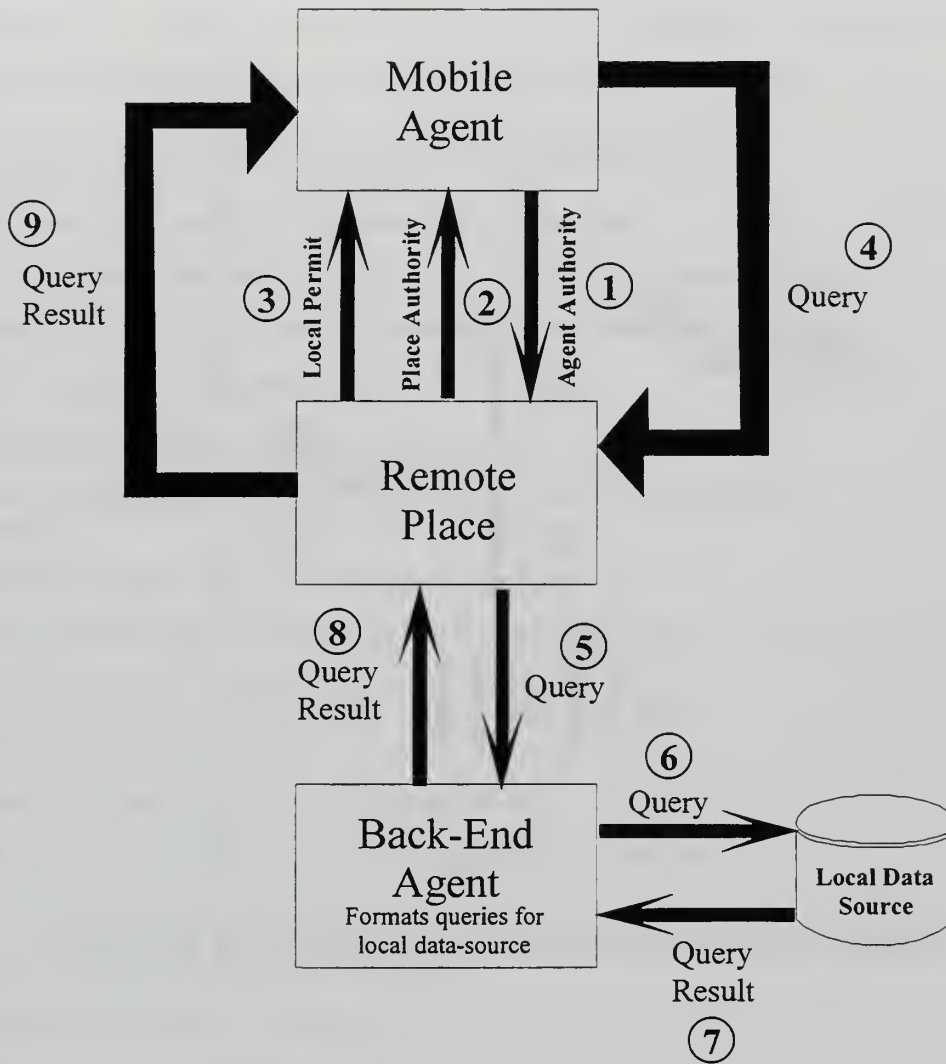
Figure 9. Information Transfer Between Mobile Agent and Remote Data Source
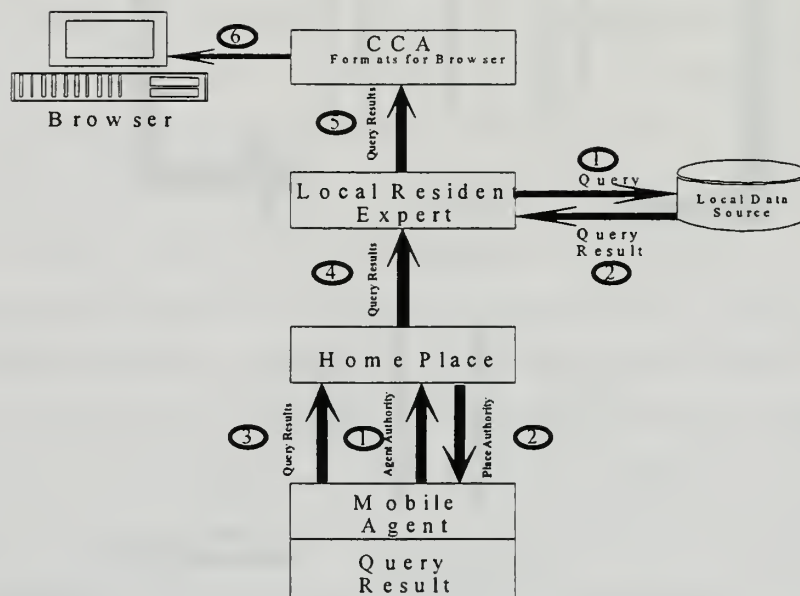
Figure 10. Information From Returning Mobile Agent to Browser

Request for strike data to Central Control Agent (CCA)

CCA does analysis of request

CCA calls Intelligence Local Resident Expert (LRE) Agent

CCA passes tasking to Intelligence LRE

Intelligence LRE searches local database for data ⟶ If finds data, LRE passes data to CCA

If fails to find data, Intelligence LRE calls the Home Place to spawn a mobile agent

The Home Place directs a mobile agent to pre-identified databases, issuing a ticket, authority and a permit. Passes the query to the mobile agent

Mobile Agent leaves local region and travels to 1st ticket destination

Mobile Agent passes authority to the server remote place inside Java virtual machine

Remote Place passes back a permit with local restrictions and its authority

Mobile Agent verifies and authenticates remote place authority

Mobile Agent passes query to Remote Place

Remote Place (front-end agent) passes query to Remote LRE (back-end agent) for Processing

Remote LRE formats query for local data source and executes query

Remote LRE takes response to query and hands it back to Remote Place

Remote Place delivers to Mobile Agent

Mobile Agent returns to Home Place

Mobile Agent passes its authority to the Home Place

Home Place verifies Mobile Agent authority and passes its own authority

Mobile Agent verifies and authenticates Home Place authority

Mobile Agent passes query results to Home Place

Home Place passes data to Intelligence LRE

Intelligence LRE passes data to CCA

CCA combines data with other LRE reports ⟵

CCA formats data for local browser

CCA passes data to local browser

Figure 11. Scenario Information Flow Diagram

VI. MANAGING MARK SYSTEM PERFORMANCE

A.   CONNECTIVITY MANAGEMENT

While high levels of software agent activity can increase the loads on agent servers, the limiting factor on the systems as a whole is the bandwidth of the connecting networks. Mobile agents provide advantages over stationary agents by traveling through the network and executing queries on remote servers, rather than requiring ongoing communication between clients and servers. Further, mobile agents follow a *pull* paradigm, relieving the network of the constant traffic from multiple push servers. There are obvious tradeoffs between the size and capabilities of software agents. The common sense assumption is that the more functionality and capability an agent has, the larger it will be. By further dividing the work between task specific, mobile agents and stationary, information specific agents, the system can maintain a high degree of functionality and expertise while still allowing for small mobile agents.

Information that the agent carries in the form of data and reports also adds size to the agent. There are several ways to limit network demand by reducing the number of agents on the networks and reducing the size of the agents that do travel. This section explores some of those possibilities.

1.   Region Coordinating Agents

The coordination provided by the hierarchy of Central Coordinating Agents, Local Resident Experts, Home Places, and mobile agents allows for more efficient use of the resources available to each individual MARK application. If regions are also arranged hierarchically, then the concept can be abstracted to another level where Regional Coordinating Agents (RCA) coordinate the efforts of all mobile agents running within their region. If two agents, from different clients arrive at a higher level server carrying similar queries, the RCA can reduce traffic on the network by only forwarding the first agent to arrive. Agents with similar queries are stored locally until the first agent returns with the

requested information. That agent then shares the results of its search with the other agents waiting at the region server. The mobile agents then return home. Traffic in the upper regions will be limited and network performance will improve at all levels. This arrangement is illustrated graphically in Figure 12.

2. Strategies to Limit Bandwidth Requirements

There are a number of strategies that can be employed to reduce the bandwidth requirements that MARK puts on the system as a whole.

a. *Consolidation of common code*

There is a division of labor between the cooperative agents in the proposed architecture. Mobile agents are task specific, carrying specific queries and processing specific types of information. They may carry some level of domain knowledge with them to enable remote processing of data, which allows agents to eliminate redundant data in the field, without carrying it across the network. Remote places are information specific agents, specialized in handling the information available at that node. It is possible to further reduce the size and bandwidth requirements of mobile, task specific agents by eliminating common modules from the programming of the mobile agent and storing those modules on remote agent servers. When an agent needs to execute instructions within a given module, it invokes the module by means of a remote procedure call. For example, communications among mobile agents and between mobile agents and places is a capability shared by all agents. If the programming modules that control communications are stored at each place, then there is no need for mobile agents to carry the code. Only high level controlling modules and modules that are unique to the tasks of a particular agent should be carried across the network.
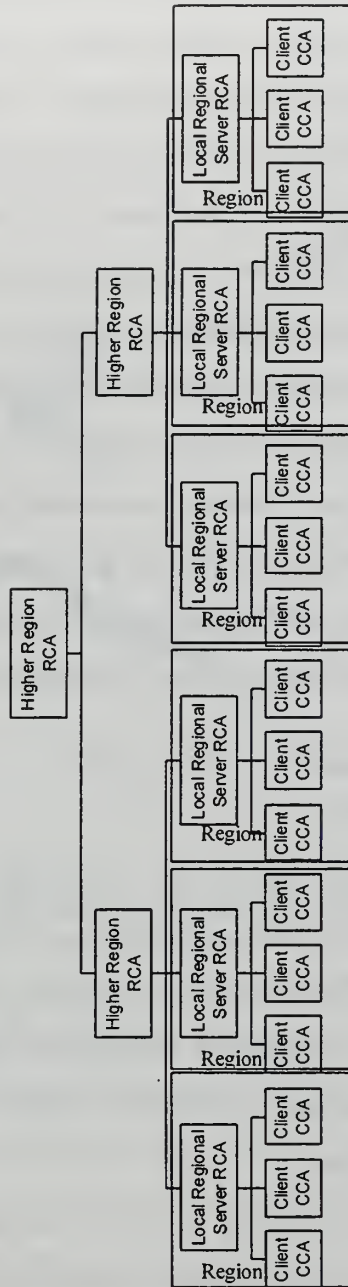
Figure 12. Regional Coordinating Agents

b. Encapsulation versus agent connections

As mobile agents gather data, encoding and encapsulating it for transport, the size of the agent grows with the quantity of data collected. This can be limited by taking advantage of agent expertise and processing capabilities. As information is gathered it is processed on remote servers. New data is compared with data the agent has obtained from other sources, and redundant information is eliminated. This reduces the quantity of data the agent must carry, but opens the system to the information security threats described in Chapter IV. If it is not safe to process information on remote servers, then there is no reason for the agent to encapsulate it and carry it across the network. MARK agents can be programmed with the capability to encrypt reports and send them to their Home Place via connection services provided by each server. This reduces the peak bandwidth requirements of the system by spreading the traffic over time. Distributing the tasking over several nodes improves the survivability chances of the data getting through by reducing bottlenecks in the network. This strategy also reduces the load on the individual servers by spreading it over multiple nodes.

c. Proxy Agent Servers

If the agent's Home Place is at the end of a small pipe (e.g., a wireless LAN at sea), then it may be advantageous to do some processing of information on a secure server attached to the higher bandwidth networks ashore. The use of a proxy agent server can reduce the quantity of traffic on low bandwidth networks by processing information remotely, and forwarding only that information which is unique. Figure 13 shows a conceptual diagram of a Proxy Agent Server.

In this arrangement, the proxy agent server receives queries from a client at sea. The proxy agent server runs a MARK application complete with a CCA, LREs, and mobile agents on behalf of the client. All processing is done remotely, and only the results are returned to the CCA running on the client machine. This also reduces connectivity requirements, as the client does not need to communicate with mobile agents that are running from the proxy.
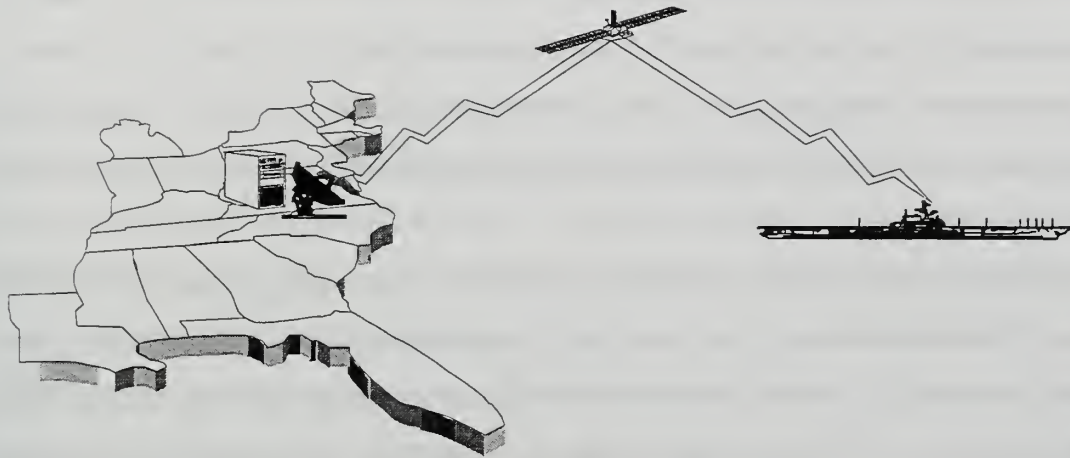
Figure 13. Proxy Agent Server

d. *Persistence and cloning of Mobile Agents*

Software agents can be programmed to monitor information systems for certain conditions, then respond when that condition becomes true. For example, an LRE may be specifically tasked to locate information on an enemy tank battalion. The mobile agent tasked to find the data might need to search communications, imagery, and electronics intelligence sources. The agent should monitor them all, but doing so requires continuous movement over the network, checking each source periodically for changes. Instead, when the agent arrives on the first server and finds the condition to be false, the agent can clone itself, forwarding the clone to the next server, then "sleep" on the server where it currently resides, waking periodically to see if the condition is true. Each new clone now periodically monitors the conditions at each server without moving across the network, and when the

69

tank battalion moves the agents send reports to their Home Place. The reports can be updated periodically and when there is no longer a need for the information, the Home Place sends a message to the agents and they die in place.

B. AGENT SYSTEM TRAINING

Intelligent software agents can learn on their own by following and repeating operations done by the user and tailoring their behavior [Ref. 18]. The MARK system, being composed of intelligent agents, will also learn as it is used. Initially, however, the system needs to be trained. Each module needs to be run through scenarios targeted at that module's mission.

During the initial agent training, the human user can use historical data to verify the accuracy of the returned report. The user should concentrate on the *recall* and *precision* of the MARK output. Recall measures how well the agents locate and return all available data regarding the query. By using local databases in a controlled environment, the user can isolate data located by the LRE and the mobile agents and provide feedback accordingly. Precision measures how well the agents eliminated irrelevant data before forwarding the returned report [Ref. 19]. Again, using a controlled environment for the initial training allows the human user to isolate those parts of the system that need direct feedback or additional training/programming.

Once MARK is deployed, training is conducted each time the system is used. The more often it is used, the more the agents will learn. The human users should provide feedback on every report generated by the system. MARK has the capability of being tailored to each specific user, based on the profile developed every time the user logs on. The agents will learn that person's preferences and desires of where and how data is retrieved. There is a danger in this, however, that the system will also learn that user's prejudices and biases, thereby not realizing the full potential of MARK. For example, a human user may have worked at the National Security Agency (NSA) and be familiar with the material they produce. The human user may train the agent to always accept material

provided by NSA over material provided by another source (i.e., the Defense Intelligence Agency), even though NSA may not be the best source of information for some queries. In other words, the human trains the system to develop bad habits. Improper training may cause MARK to limit its scope of resources and miss vital pieces of information.

Perhaps a better method of training MARK is to promote global training based on the entire user base, rather than on personalized training. Feedback from each user is not necessarily tied to the user profile, but used to train the entire system. MARK learns from each user's feedback, but applies that training across all users, rather than only to that specific user. This allows MARK to increase the knowledge base of the entire system, and perhaps compensate for individual biases. The global training approach provides MARK the opportunity to use collaborative filtering to predict what items a new user might like based on the preferences of similar users. [Ref. 18] As a new user develops a profile, MARK compares the items that the user requests, or enters, to those of other users. MARK then tries to predict other items the new user might want to see. When using collaborative filtering, MARK learns based on the community of users.
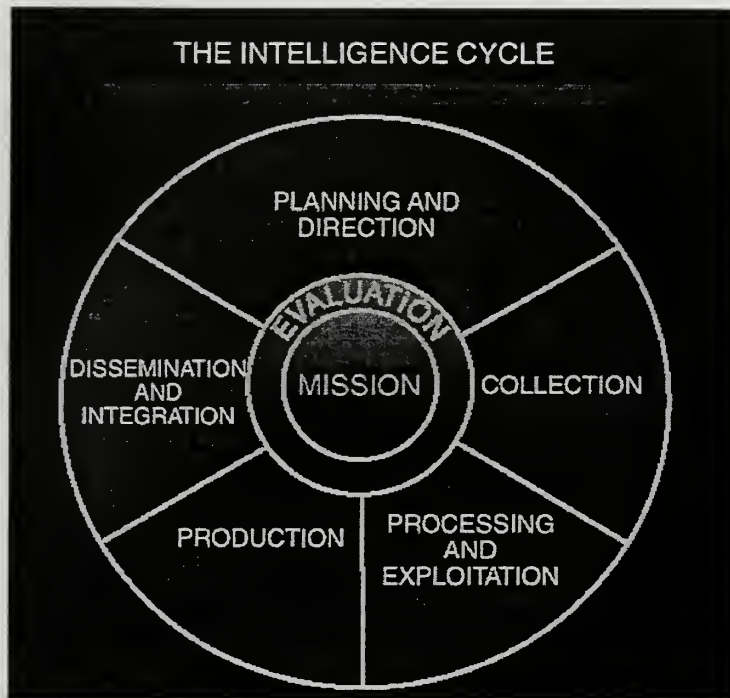
VII. APPLICATIONS OF MARK

There are potentially many applications of MARK in the military and commercial sectors. This chapter explores one possible application in terms of military intelligence problems.

Preparation of the intelligence battlespace is essential to ensure operational success. MARK can be used to assist in the intelligence and reconnaissance efforts that support the decision-maker. The processes described below are representative of areas that have repetitive actions or that require the user to search large amounts of data in a very short amount of time. MARK can be used to perform preliminary data fusion and data integration of large amounts of data. During crisis situations, the intelligent agents can provide multiple courses of action to the user, based on the objectives and constraints provided.

A.   THE INTELLIGENCE CYCLE

Intelligence is defined as the product of the collection, evaluation, analysis, integration and interpretation of all available information regarding a foreign country or area [Ref. 20]. The activities required to process information and transform it into intelligence can be represented by a continuous cycle, as shown in Figure 14. The United States intelligence cycle has six steps: planning and direction, collection, processing, production, dissemination and evaluation [Ref. 21].

MARK follows the same steps when answering a request for information from the user. The CCA interprets the request and decides which LREs are best able to satisfy the tasking (planning). The CCA identifies the requirements, prioritizes them and provides direction to the LREs on their assigned tasks to complete the mission. The CCA supervises the collection efforts of the LREs to ensure that the intelligence requirements are being met. The LREs search their databases in an effort to satisfy the tasking. If the requested data is not readily available, mobile agents are tasked to locate the data (collection).

THE INTELLIGENCE CYCLE

PLANNING AND DIRECTION

EVALUATION

DISSEMINATION AND INTEGRATION

MISSION

COLLECTION

PRODUCTION

PROCESSING AND EXPLOITATION

Figure 14. U.S. Intelligence Cycle

From Ref. [21]

Processing of data takes place at all levels. Agents conduct preliminary processing of the data to determine if it satisfies their goal. The LREs process data as it is returned from various agents, looking for redundancies or unnecessary information. The CCA receives and processes reports from all LREs. The CCA is then responsible for integrating all the data into a finished product. The CCA disseminates the final report to the user and to a database that tracks each query and stores information on the returned answer for future use. A human user, before dissemination or after, can evaluate the information to ensure that the product of the intelligence cycle is meeting the needs of the decision-maker. The evaluated feedback is incorporated into the intelligence cycle, training the CCA and LREs and improving the finished product.

The finished product must be tailored to the user's needs, which the CCA has learned through multiple previous tasks. The intelligence report provided by the CCA must

be objective and unbiased, not influenced by preconceived ideas, as can happen with human analysts. The report, however, can be swayed or slanted in a particular direction if the user's request is not carefully stated. Thorough and frequent training of both the user and the agent can minimize this problem.

The agents do not have to move sequentially through the five steps. When the LRE receives tasking from the CCA, it does not always have to initiate collection through mobile agents. If the information is already held in local databases, the agents in the model skip the collection step and proceed directly to processing or production. This saves processing time and decreases the decision cycle time of the user.

B. SUPPORT FOR GENERAL MILITARY INTELLIGENCE AND ESSENTIAL ELEMENTS OF INFORMATION

Two types of intelligence essential to preparing the battlespace for decision-makers include general military intelligence and essential elements of information (EEI). General military intelligence includes information that can be used to provide background information on a country, or detailed information about a specific area. EEIs provide critical information about the opponent or the environment that a decision-maker requires to combine with other information when planning a particular operation. MARK can be used to manage both types of intelligence requirements.

1. General Military Intelligence

Updating general military intelligence can be very time-consuming, requiring someone to research the area of interest and update numerous databases. For example, MARK, running at a theater level, can maintain databases with updated information on a continuous or periodic basis, depending on user requirements. Information concerning some areas of the world may only require updating every week, while others require information to be collected every hour. When users at the tactical, operational or strategic level need the information, agents can be sent to retrieve the general military information from the theater level databases.

75

The CCA, programmed to coordinate the intelligence collection and dissemination for a specific country or location, can direct the LREs that monitor a particular area of general military intelligence. One LRE, for instance, may be responsible for monitoring the armed forces capabilities of a country. The LRE may use mobile agents to search open source material (books, newspapers, journals), intelligence reports based on signals intelligence, communications intelligence, human intelligence, etc., and imagery to update information on order of battle, organization, training, doctrine and strategy. Another LRE may maintain information on terrain intelligence, meteorological data, geological and oceanographic information. Other LREs can track transportation, communications capabilities, economic developments, political and sociological trends, and other topics that might be of interest to the U.S. or its allies during military operations.

When not receiving direct collection guidance, the CCA follows the steps of the intelligence cycle, looking for gaps in information that may generate new requirements. Instead of presenting a finished product to the user, MARK can store the information for future reference. The general military intelligence can be presented in a variety of formats, based on the needs of the user. For example, geographical and terrain information can be presented in chart or picture form, depending on user interface capabilities.

2. Essential Elements of Information

Essential elements of information represent the "critical items of information regarding the enemy and the environment needed by the commander by a particular time to relate with other available information and intelligence in order to assist in reaching a logical decision" [Ref. 20]. MARK can assist the intelligence staff in locating the information to answer critical intelligence requirements.

Essential elements of information (EEI) are generally associated with a particular mission, and so are more specific than the aforementioned general military intelligence. An example EEI may look like the following: "Determine whether the enemy will reinforce Objective X. If so, when and in what strength." LREs and CCAs may use the general military intelligence to answer the specific question.

Operational plans may have standing EEIs, addressing intelligence necessary prior to the initiation of operations. MARK can be tailored to the intelligence staff, to be used to provide updated information to planners almost immediately. At the theater or strategic level, each OPLAN can be assigned a CCA that is stored in the CAR. The CCA can be programmed to update the OPLAN EEIs on a periodic basis, determined by the threat level in the area. Each LRE can be assigned a particular EEI to manage, using local databases and agents as described in Chapter V to locate the required intelligence. It may not be worthwhile in terms of processor time and storage space to have all OPLAN EEIs updated all the time. It may be very beneficial, however, to have the OPLAN CCA available in the CAR and to run scenarios to train it occasionally. The CCA and LREs learn through repetition and feedback.

As the EEIs are updated, the CCA can format the information to complete the Intelligence Estimate for the intelligence staff (see Appendix A). The Intelligence Estimate template can be stored in the browser, allowing the CCA to present the requested information in a format useful to the user. The Intelligence Estimate is continually revised and updated as the situation changes. MARK can provide continuous information flow and updates to the Intelligence Estimate. This allows the intelligence staff and the force commander to have a timely, up-to-date report at their fingertips to aid in decision making.

C.   DATA FUSION AND INTEGRATION

Fusion is defined in Webster's Dictionary as the merging of different elements into a union. Data fusion attempts to take information from multiple sources and use it to make inferences about the environment external to the sensors, creating a single picture. Humans do highly complex data fusion all the time, assimilating input from the senses to create an understanding of events happening around them. Humans are also able to take information they have accumulated, compare it to the current situation, and make an inference or a prediction of future events.

Data integration is slightly less complex than data fusion. Webster's defines integration as the act of joining an object with something else. Data integration tries to form a complete picture from the information gathered from multiple sources, but it does not make inferences based on the data. Data integration joins all of the pieces of the picture in some logical manner, such as time of occurrence or time of reception. Computer search engines perform data integration when they attempt to answer search queries. Key word searches find data on each part of the search query and then join the data in a logical manner. The data is not evaluated and merged into a single, complete picture, as it is in data fusion.

Data correlation involves finding some form of relationship between two pieces of data. The relationships may be causal, complementary or reciprocal. Data correlation can be done based on mathematical formula or by comparison with information already stored in a database. For example, if a sensor picks up a particular electronic signal, it can search its database for something comparable. The database may contain information that tells the sensor that that signal is associated with a particular emitter on a specific class of ship.

Biological systems provide many examples of data fusion, through the senses. Animals are able to receive input from eyes, ears and nose, evaluate it and fuse the data to create situational awareness. Creating a computer system that can conduct data fusion from multiple sensors with often contradictory or incomplete data is not as easy. Emulating biological systems requires in-depth and complex programming. The first step has been to model the data fusion process and create algorithms to support the model.

1. Current Data Fusion Model

The Data Fusion Group of Joint Directors of Laboratories (JDL) Technical Panel for C3I created a high-level functional model of the data fusion process (Figure 15). The purpose of the model was to create a common framework for all of the researchers, technologists, developers and users interested in data fusion automation [Ref. 22].
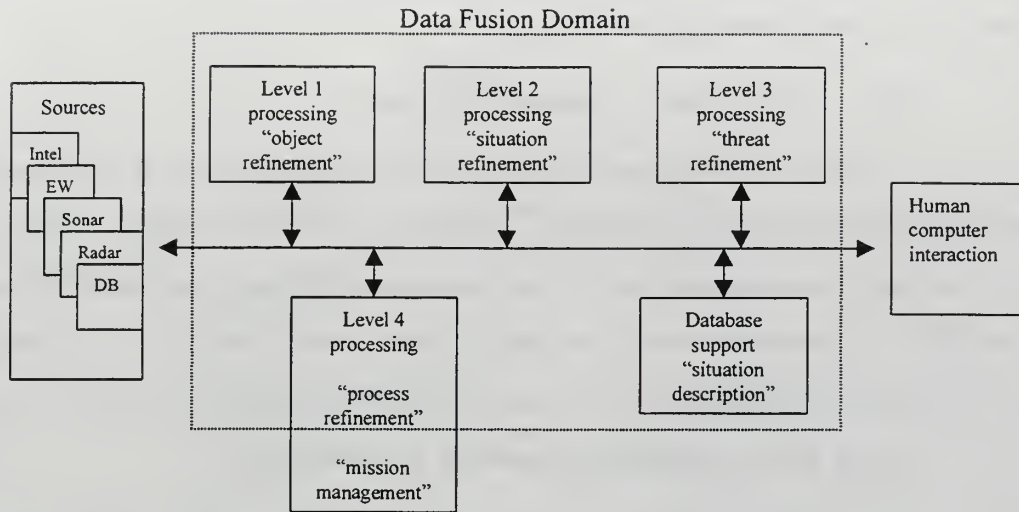
Figure 15. JDL Data Fusion Model

From Ref. [22]

a. *Level One – Object Refinement*

Level One is concerned with object refinement. It strives to process data in four areas: object detection; the association of the detected object with data previously collected; the establishment of information such as velocity and position; and object identification and classification.

b. *Level Two – Situation Refinement*

Level Two fusion uses the information produced by Level One to create a clear picture of the situation. In order to conduct the higher level analysis, Level Two must have historical information stored from previous tasks and intercepts.

c. *Level Three – Threat Refinement*

Level Three attempts to do threat refinement, fusing the information from Level Two with information available in the supporting databases. The goal is to develop

79

potential actions or maneuvers that the detected object might execute, making inferences concerning intent based on evaluated data.

### d.   *Level Four – Process Refinement*

Level Four tries to assign the best assets to conduct required collection based on pre-identified objectives and priorities. Level Four plays a role outside the data fusion domain, constantly assessing the status of various sensors and monitoring the quality of the data collected. Within the data fusion domain Level Four is tasked with focusing the reasoning efforts of the other three levels, based on prioritized requirements [Ref. 11, 22].

2.   Data Fusion Model Supplemented With MARK

The agent model presented in Chapter V can be broken down to meet the functions described above. Figure 16 combines elements of the JDL Data Fusion Group model and a model developed by Hall and Llinas [Ref. 11] with the capabilities of intelligent software agents found in MARK.

a.   *System Input*

The data fusion system can receive inputs in several ways. It can come directly from data observed by the agents. Static agents that monitor day-to-day activity can input data if the activity they monitor exceeds an established threshold. The system can receive input from data and commands initiated by human operators or users. The data or request would be passed from the user to the Central Coordinating Agent (CCA) and from the CCA to the Local Resident Expert (LRE). The system can also receive inputs from previous data stored in pre-established databases. The user can train the agents by inputting known data and initiating a query. The model will initiate a search for data and provide the information back to the user, allowing the user to verify the accuracy of the returned report. In this way, the system continually updates its data and checks its reasoning algorithms.

Data Fusion Environment
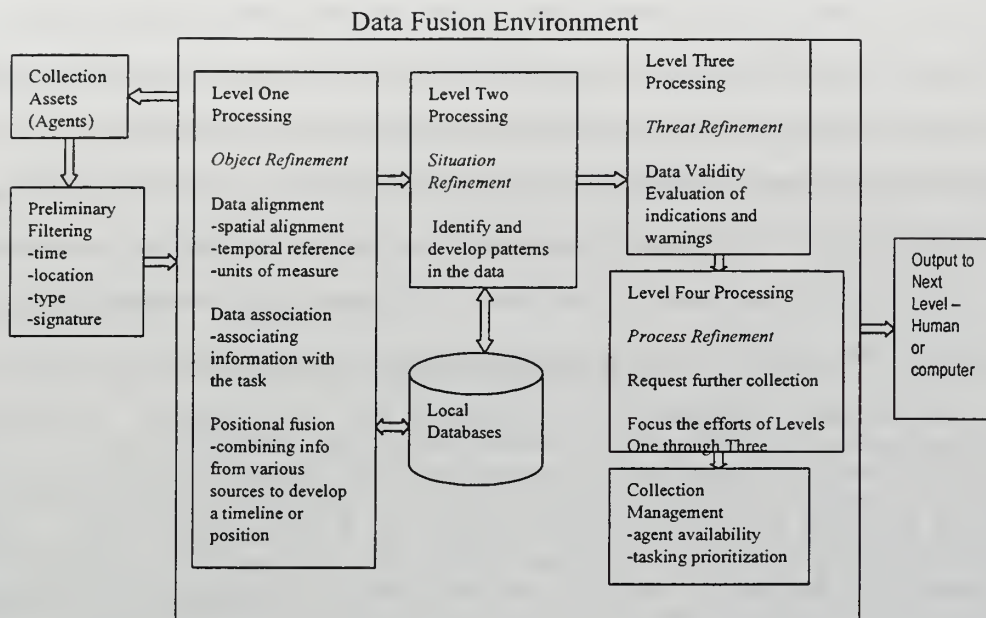
Figure 16. Agent Supported Data Fusion Model

The data fusion environment can occur at the LRE level, as it receives its tasking from the CCA and analyzes data from the agents, or at the CCA level, as it receives its tasking from the human and analyzes data from the LREs. For purposes of this example, assume that the LRE is receiving tasking from the CCA and that the agents have the ability to do some preliminary filtering.

b.   *Preliminary Filtering*

The preliminary filtering includes making sure the information fits within the time and location identified in the query. The agent ensures that the event described in the query response answers the question asked (i.e., ships are moving north, not south). The agent also ensures that there is a signature on the data, identifying the source of the information and the location from which it was retrieved. For the agent to do the filtering, it must be able to buffer data and have access to the server's computing power to do the comparisons. This concerns the issues of agent learning and intelligence. For the agent to do a comparison between data, it must have the ability to determine which information is more relevant to its mission.

81

There is also a security issue. If the mobile agent does the filtering, it will have to "open" the information it received from one server on another to do the comparison. This can be a problem in that the second server can now read what the agent is carrying, since the agent is using that server's processor. This might not be desirable if the agent is carrying "need-to-know" information or information that is classified or from a classified source. If the mobile agent is on a trusted server with the same level of classification, it can do basic level data analysis on the spot and determine if the information is worth sending home. The amount of analysis the mobile agent can do is based on its intelligence. Performing the analysis immediately conserves bandwidth and processing time on the home system. If the agent is not on a trusted network, or it is on a server with a different classification level than the material it is carrying, then all information can be returned to the home system. The programmer and user have to determine the acceptable levels of tradeoff between bandwidth/processing time, security and agent size.

c.   *Level One – Object Refinement*

In Level One, tasking is received from the user, whether it is a human or another agent. The LRE checks the local databases for information and spawns agents to get the latest data. As the data arrives, the LRE does data alignment, putting it into a common frame of reference. It transforms map coordinates and establishes a common time reference. Coordinate transformation and data unit conversion requires putting the information into a standard format, no matter the source. For instance, the Army and the Navy use different methods of recording location (grid coordinates vs. lat./long). How will an agent know that the data it received from one server fulfills its mission better than the information it received from another server? One alternative is to have most data conversion happen at the LRE level, when the agent sends the information home. That prevents the code of the agent from becoming too large. The LRE can also do some analysis of the information and determine if it is redundant or not. Another alternative is to have the agent do the conversion immediately, before it leaves the server. This increases the code of the agent, but saves some processing time by the LRE.

Data association refers to combining all of the information that belongs to a particular problem or query. The LRE performs data association by taking the data returned from multiple mobile agents and forming it into intelligence that can be associated with a particular object or target. The LRE also does some positional fusion, combining data from various sources to develop a timeline of events or location of an object. Redundant information is identified and discarded.

d.   Level Two – Situation Refinement

In Level Two, the LRE evaluates the data that has been returned by the agents and from the local databases. It tries to put it into an order (by time or by subject) that is applicable to the situation. The LRE also looks for patterns in the data that may identify relationships between entities. As the LRE combines the data from various sources, it is doing "situation generalization," which Antony defines as "a bottom up abstraction of information for the purpose of situation awareness with respect to higher level-of-abstraction entities" [Ref. 22]. For example, an agent returns imagery of an LCAC heading away from the beach. The LRE knows that LCACs are typically associated with ships. Given the fact that the LCAC was seen on the water, the imagery indirectly provides evidence of a higher level-of-abstraction entity (a ship).

The LRE is also doing "situation specialization," which is top down reasoning for the purpose of deducing or inferring subordinate elements or entities. For example, the agent returns information about the location of an aircraft carrier. The LRE knows that aircraft carriers carry airplanes. Although the LRE does not have specific confirmation, it can infer the existence of lower level-of-abstraction elements (the airplanes). The airplanes may be unobserved or unobservable because they are in the hanger bay.

Depending on the sophistication of the coding, the LRE can also attempt situation abstraction, where it attempts to fill in missing information based on reasoning. The combined output of situation generalization and situation specialization may not provide a complete picture, as they use reasoning based on directly observed objects [Ref.

22]. Situation abstraction uses more complex inference and reasoning algorithms in an attempt to "mimic the reasoning performed by human experts." [Ref. 11]

e. Level Three – Threat Refinement

The LRE conducts threat refinement for its area of expertise. It looks at historical expectations, objectives, intents and capabilities. Threat refinement provides information on possible enemy intent and friendly force vulnerabilities. For example, the weather LRE receives information about multiple bad weather areas in the South Pacific. It can review historical trends and the characteristics of the weather in that area for a particular time of year and project the likelihood that a typhoon will form. It can also provide projected information about paths that the typhoon may take, including which areas may expect the most damage.

The CCA does threat refinement by combining all of the data from the LREs into one picture.

f. Level Four – Process Refinement

The LRE does process refinement by redirecting agents to gather more data if it needs further collection for analysis. If the LRE knows that it will need a particular piece of information to answer its tasking, it can control the number of agents spawned and direct their actions based on a global collection strategy. This helps prevent the waste of resources on unnecessary data collection (i.e., shotgunning agents to news agencies that concentrate on the Middle East when the LRE needs information about Greenland).

Collection management responsibilities reside at each level of the model. The LRE prioritizes its tasks based on input from the user and the CCA. The LRE passes the prioritization to the Home Place through the time to live allowance in the permit for mobile agents. The CCA manages tasking prioritization for the LREs based on the preprogrammed modules, direct user input and agent availability. If the user enters an ad hoc query that does not directly translate to a specific LRE, the CCA will assign tasking

based on which LRE can provide a portion of the answer. The CCA is then responsible for assembling the parts of the answer and including it in the final report.

The intent of the MARK supported data fusion model is not to replace the human decision process, but to augment it with accurate inferences based on sensor input.

D. CRISIS ACTION PROCEDURES

The intent of this section is to step back from the technical aspects of MARK and illustrate how it can be integrated into the human decision making process. The Crisis Action Procedures (CAP) used by joint staffs contain several phases that require accurate information and swift decision-making. These are areas where MARK can be applied. Each phase within CAP will be discussed along with a proposal of how MARK could assist the user to better reach the decision or resolution at the end of that particular phase.

On a joint staff, the planners follow the deliberate planning process when time allows. When a situation or crisis develops that is time sensitive, the Joint Planning and Execution Community (JPEC) is forced to follow CAP. Because of the time sensitive nature of a crisis, having flexible procedures that capitalize on previous planning and ensure rapid and effective communications throughout the planning and execution phases are critical.

The CAP is comprised of six distinct phases, each beginning with a specific event and ending with a decision or resolution. Figure 17 depicts a summary of the CAP phases. [Ref. 23]

Phase I Situation Development	Phase II Crisis Assessment	Phase III Course of Action Development	Phase IV Course of Action Selection	Phase V Execution Planning	Phase VI Execution
Event					
• Event occurs with possible national security implications	• CINC'S REPORT/ • ASSESSME NT resolved	• CJCS sends WARNING ORDER	• CJCS presents refined and prioritized COAs to NCA	• CINC Receives ALERT ORDER or PLANNING ORDER	• NCA decides to execute OPORD
Action					
• Monitor world situation • Recognize problem • Submit CINC's ASSESSMENT	• Increase awareness • Increase reporting • JS assesses situation • JS advises on possible military action • NCA-CJCS evaluation	• Develop COAs • CINC assigns tasks to subordinates by evaluation request message • Create/mo dify TPFDD • USTRAN SCOM prepares deployment estimates • Evaluate COAs	• CJCS advice to NCA • CJCS may send PLANING ORDER to begin execution planning before formal selection of COA by NCA	• CINC develops OPORD • Refine TPFDD • Force preparation	• CJCS sends EXECUTE ORDER by authority of SECDEF • CINC exercises OPORD • JOPES database maintained • JPEC reports execution status • Begin redeployment planning
Outcome					
• Assess that event may have national implications • Report the event to NCA/CJCS	• NCA/CJCS decide to develop military COAs	• CINC sends Commander's Estimate with recommended COA	• NCA select COA • CJCS releases COA selection by NCA in ALERT ORDER	• CINC sends OPORD	• Crisis resolved • Redeploym ent of forces

Figure 17. Summary Of Time-Sensitive Planning Phases

From [23]

Figure 18 shows the crisis action planning process and its relationship with the deliberate planning process and the Joint Deployment System (JDS). [Ref. 23]
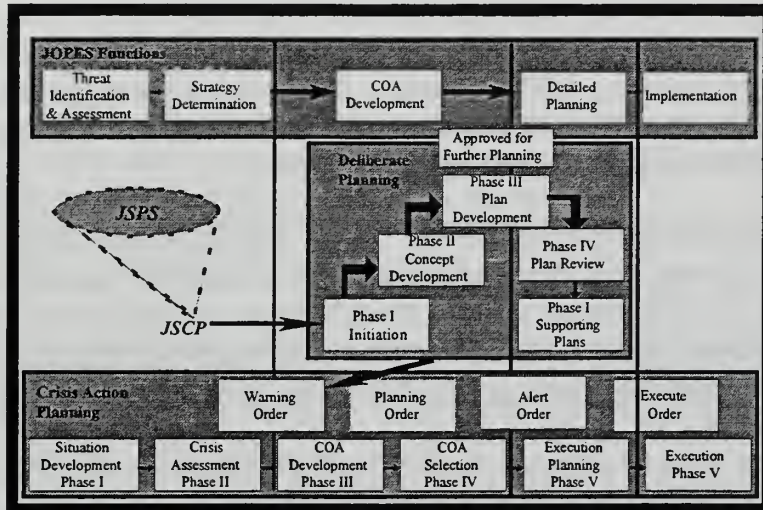
Figure 18. JOPES Functions and Joint Planning

From Ref. [23]

1. Situation Development

Situation monitoring is an ongoing vigilant effort to study and analyze events happening throughout the world. All types of resources, such as TV, radio, intelligence, imagery, etc., are used to assist in this process. These resources are constantly monitored to look for events that have the potential to have an adverse impact on the national security or national interests of the United States. When a potentially adverse event is recognized, it is reported by means of a Critical Intelligence Report (CRITIC) or OPREP-3 PINNACLE (OPREP-3P) to the National Military Command Center (NMCC). Once the event has occurred and has been reported to NMCC, this signals the end of the situation development phase.

> *Scenario*: JTF 398 is currently involved in operations against Islandia. At 0200Z, the intelligence officer, COL Smart, while using MARK to update his intelligence picture of the Islandia operation, happens across information showing that Pacifica is showing indications of supporting Islandia. Pacifica has been an ally of the US for a long time, and a staging base for logistics trains in support of the strike on Islandia, even though Pacifica is a close neighbor of Islandia. Realizing this could seriously hinder strike efforts currently underway on Islandia, COL Smart queries MARK to validate this piece of information. MARK returns a confirmation from various sources that Pacifica has switched loyalties to Islandia and is in the process of standing up their coastal defense force in support.
>
> COL Smart notifies the CINC of the "event" happening on Pacifica. COL Smart continues to collect information from MARK on the Pacifica situation. MARK returns a report from Reuters, the global news agency, that Pacifica no longer supports US forces in the region and demands that all American personnel leave at once, but that they will not provide any transportation.
>
> The CINC's staff immediately notifies the NMCC...

2.   Crisis Assessment

During crisis assessment, the National Command Authority (NCA) and the Chairman of the Joint Chiefs of Staff (CJCS) analyze and evaluate whether a military option should be pursued in response to the reported event. Timely, accurate information concerning the event is critical to these decision-makers. Depending on the event, special teams are assembled to assist in the planning process for countering the event through military courses of action. The CJCS assesses the situation, reviews current OPLANs and makes recommendations as to military options. The reporting CINC continues to monitor the situation, to evaluate the status of his own forces and provides status reports to the joint staff.

This phase ends with a decision from the National Command Authority (NCA) to have military courses of action developed to counter the situation.

> *Scenario*: ...Upon receipt of the report from the field, the joint staff launches their version of MARK to assist in the intelligence and reconnaissance gathering process. They also brief the CJCS and the NCA on the situation. With both the joint staff and the CINC's staff running MARK simultaneously, the intelligence returned verifies Pacifica's intentions. With US forces becoming increasingly vulnerable in the region, the NCA decides to pursue military options to conduct noncombatant evacuation operations (NEO) to get the approximately 2500 Americans out of Pacifica before further military action can be taken.
>
> The CINC continues to monitor the situation through MARK, looking at all available sources of intelligence, weather, news, etc...

3. Course of Action Development

The reporting CINC is responsible for developing and submitting COAs to the NCA as military options to counter the situation. During this phase CONPLANs and OPLANs developed as part of the deliberate planning process and stored in the Joint Deployment System (JDS) are examined to determine if they can be used to assist in the COA development. Large amounts of information are passed between the players involved during the COA development. Time is critical during this phase.

This phase is completed once the CINC prepares and submits his Commander's Estimate along with his recommended COA to the CJCS.

> *Scenario*: ...The CINC's staff begins to develop COAs for the NEO. In the development of the COAs, the staff planners pull the NEO module from MARK's CAR and load it into the system. The planners input data into MARK, such as the location and approximate time of execution. MARK automatically launches the appropriate agents to find information concerning aspects of this particular NEO. COL Smart decides to send MARK to look at Pacifica's newspaper source, Libertad, to see if any recent news events point to this switch in loyalties. MARK returns a list of articles that show the leader of Pacifica recently had a somewhat discrete visit with the leader of Islandia.
>
> MARK also looks at CONPLANs and OPLANs in the JDS database to see if any of them closely resemble the upcoming operation based upon the input from the planners. As MARK returns the required information to the planners, it feeds that information into a database that is used to assist in the

development of COAs. MARK continues to do situation monitoring on the Pacifica situation, providing updates as other events occur, or pre-established thresholds are exceeded.

Once the COAs are finalized and the CINC determines the best COA, his Commander's Estimate is forwarded to the CJCS...

4. Course of Action Selection

The CJCS presents the COAs listed in order of preference to the NCA for their decision. The joint staff has evaluated the recommendations from the CINC. The COAs were modified or revised as necessary. New COAs may have been required due to changing factors concerning the situation.

This phase ends with the NCA selecting a COA. Their decision to begin planning for its execution is sent to the CINC as an Alert Order.

Scenario: While the NCA is deciding which COA to choose from the recommendations, Pacifica intensifies their efforts to support Islandia by loading 1000 members of their elite coastal defense special operations force onto a ship in their main harbor of Corona. MARK's persistent agent on the satellite imagery database captures this and forwards the information to COL Smart. COL Smart notifies the CINC, who in turn notifies the CJCS. This activity is verified concurrently by the joint staff's version of MARK. Since the threat to US forces now is even greater, the NEO operation must be coupled with an aggressive counter against this potentially dangerous action by Pacifica.

A new COA is developed that includes an offensive action against Pacifica's force in Corona while still conducting the NEO in Pacifica's capital city of Tequila. This new COA is accepted by the NCA and planning begins to execute the COA. MARK still continues to monitor the situation...

5. Execution Planning

During the execution phase, the COA chosen by the NCA evolves into an operation order (OPORD) at the CINC's level.

Three major tasks occur in the execution planning phase: execution planning, force preparation and deployability posture reporting. Execution planning is directly concerned

with preparing the OPORD so that when the NCA directs, the CINC can execute. An OPORD can be developed by manipulating an OPLAN from JDS, by expanding a plan that already exists or by writing a new plan.

Information flow between planners and staff is extremely heavy during execution planning. Time is usually of the essence, as well.

This phase ends when the NCA decides to proceed with the execution of the OPORD, move into a holding pattern, or cancel the OPORD due to the use of non-military means to resolve the situation.

> *Scenario*: ...It's now 0500Z and the CINC's staff has just received the Alert Order from the NCA. The planners update MARK with the expected time of execution of the NEO and the addition of another strike module against the force in Corona. All players (i.e., supported or supporting elements) from all levels have initiated their MARK systems in support of the pending operation. MARK continues to gather information on weather, intelligence, force readiness, etc., based on inputs from the planners and the pre-determined criteria from each module. Information returned is used to prepare the OPORD for the operation. MARK returns that the OPLAN from the NEO conducted two years ago on the island of Bastonia in the JDS database provides enough similarities that it can be modified to fit the NEO at Pacifica. The planners agree and modify that OPLAN into the OPORD for the NEO and use the strike plan, with modifications from the strike at Islandia, for that phase of the operation. Information moves rapidly between the planners, with MARK continuing to monitor the situation and gather information as it changes. At 0700Z, the planners have finished the OPORD. The CINC notifies the NCA that they can begin execution at 1100Z. The NCA acknowledges and places the OPORD execution in a hold pattern until 0900Z...

6.   Execution

Phase VI, Execution, begins with the execution of the OPORD. Inevitably, changes to the original plan will occur. All possible variables (forces, transportation assets, logistics, the enemy, weather, etc.,) require constant monitoring throughout the execution. Adjustments and modifications must happen through close coordination and monitoring to ensure success due to these changing variables.

Scenario: ...At 0845Z, the NCA gives the green light to execute the operation at 1100Z. The planners and MARK have continued to monitor the situation and have noticed that the elite coastal defense special operation force has been pulled off the ship. They are staging outside Corona to counter the mobs gathering around the capital in opposition to Pacifica siding with Islandia instead of the U.S. MARK feeds information into the OPORD process to account for this change of threat. The strike will now be tailored back into a supporting role for the NEO. The operation begins precisely at 1100Z, with MARK in full operation to monitor any possible changes and to feed the information to the right element. The weather data provided by MARK confirms an upcoming storm in the area, but should not affect the operation. MARK displays a digital picture taken by an unmanned aerial vehicle five minutes after the operation commences showing the demise of the 1000-man elite force and the successful entry into the country by U.S. forces. MARK continues to provide information as the operation unfolds...

VIII. CURRENT INTELLIGENT SOFTWARE AGENT RESEARCH

Interest and research in software agent technologies are on the rise. Several government projects using software agent technology have been completed or are in progress. Many commercial products have been developed to assist in software agent development and implementation. The first section of this chapter will discuss commercially available development tools for building mobile agent applications. It will also explore issues involved with the selection of a programming language and development environment. This section will also explore some commercial applications and how they can interface with MARK. The second section will highlight three government projects using software agent technology.

A.   RELATED INDUSTRY PRODUCTS

The cost effectiveness of any information technology project relies heavily on the availability of an established developer base. There are a number of commercial products that support the development of agent based information systems. These developer's tools make rapid system development possible and cost effective. Other products may extend the functionality of MARK systems by providing plug and play interfaces between MARK agents and third party information discovery products. This section will explore various commercial developer's tools and third party products that can support and extend the development, deployment, and functionality of MARK.

1.   Developer's Tools

The programming language used for the development of mobile agents as presented in this thesis must support the concepts presented in Chapter II, the security requirements and protocols of Chapter IV, and the platform independence assumed throughout the thesis. Throughout this study, Java has been presented as the language of choice for mobile agent development primarily because the language can support all of the concepts presented and also because Java has a large developer's base.

a.   *General Magic's Odyssey*

General Magic's development of mobile agent technology most closely maps to the requirements for MARK as discussed in this thesis. For their initial implementation of agent technology General Magic created a new programming language called Telescript [Ref. 4]. It was built from the ground up to support the creation of mobile software agents and enabled the mobile agent concepts presented in this thesis. Telescript is not a scripting language. Rather it is a complete object oriented programming language that allows developers to implement the major components of mobile agents.

The Telescript engine is a software program similar in concept to the Java virtual machine. It provides a protected area for agents to run. Like the Java virtual machine, it is an abstraction layer that interfaces with the operating system of the host computer. It does not allow direct access to the hardware, peripherals and storage of the host machine.

General Magic was unable to sustain support for Telescript, and its development environment, Tabriz AgentWare, primarily due to a lack of developers interested in building applications in another new programming language. "General Magic recognized that widespread adoption of Java prevented general acceptance of Telescript. Tabriz was a product based on Telescript and designed to supplement web servers. General Magic has withdrawn both products from the market." [Ref. 12] General Magic is taking advantage of the large installed developer base of Java and has implemented their agent technology in "100% Pure Java" using Java classes. Java provides most of the functionality of the Telescript language, but some capabilities are not supported in version 1.1 of the Java virtual machine. Specifically, Java does not currently provide a way to capture the state of an executing program. Odyssey agents must restart at each destination, or execute only specified methods at each destination. In an attempt to overcome this weakness, General Magic developed the idea of an Odyssey worker. The worker class is a subclass of the agent class that runs one task per destination. A worker is a set of tasks and a set of destinations. At each destination the worker executes to completion all of the tasks on its

list for that destination. These tasks and destinations can be modified as information is gathered during the agent's travels. [Ref. 12]

b.   *IBM Aglets*

IBM has done a significant amount of agent research and development and offers a number of demonstrations through the Internet and in applications such as Lotus Notes. For the most part, IBM's agent research has focused on stationary agents that operate on either the client or the server. One area of research with direct implications for MARK however, is the Aglets Workbench, and Aglets Building Environment. Like General Magic's Odyssey, these tools are used by developers to create mobile software agents.

The Aglets Workbench is a visual environment for creating agent-based applications. It consists of the following components: [Ref. 13]

- Aglets, Java class libraries and tools to enable objects to move

- Jodax, a high level Java library to IBM's DB2 database

- JDBC, and ODBC-style library to RDBs

- Tazza, a visual GUI builder for Java

These tools provide a useful development environment for creating agent applications. Like the General Magic implementation, IBM's Aglets are Java threads that are capable of running on any Java enabled browser. Also like General Magic's implementation, IBM's Java Aglets are not capable of maintaining state when traveling. A work around like General Magic's worker class may be useful in partially overcoming this weakness.

2.   **Commercially Available Plug-ins**

At least two products on the market now appear to have significant potential to extend the functionality of MARK by providing a plug and play interface between mobile software agents or places, and third party search utilities. Verity's Search'97 is an

enterprise information discovery tool. Excalibur's Visual RetrievalWare is a developer's tool for creating search applications capable of retrieving image files.

a.   Verity's Search'97

Search'97 [Ref. 15] is a search application that automatically catalogues all of the information available within a network. The application can catalogue and retrieve information from over 100 different applications and databases. The application has the following functions:

- A browser based user interface that allows users to search locally and across the network

- An Information Server that indexes information in hundreds of different applications and databases

- An agent server capable of servicing over 100,000 concurrent agents.

- Advanced search and query enhancements including query by example, natural language parsing, and spelling override.

Search'97's agent server and enhanced query features indicate that it can be used to extend the functionality of MARK by providing a third party search application that can fulfill queries passed through the places serving mobile agents.

b.   Excalibur Visual RetrievalWare

Excalibur Visual RetrievalWare [Ref. 24] is an application development tool for creating search applications capable of retrieving multimedia files such as photographs. These media management systems can automatically index and retrieve visual information based on its native content. The search works on the binary pattern of the files in the index and provides feature extraction, analyzing, indexing, and retrieving of digital images based on their color, shape, and texture. The applications are capable of query by example, where the user asks "Have you seen anything that looks like this?" In a demonstration from the Excalibur web site (www.excalib.com), users can click on a random image in a set of twelve

drawn from a catalog of 28,613 images. The application then returns other images from the catalog with similar shape, color, and texture.

Such an application could be very effective in retrieving satellite imagery from a database. Visual RetrievalWare applications can be created as plug-ins to a MARK system that interface with agent places where the agent requests imagery similar to a sample either carried by the agent, or stored locally in an index of samples.

B.   RELATED GOVERNMENT PROJECTS

1.   Intelligent Decision Aids (IDA)

The Intelligent Decision Aids project [Ref. 25] is a joint effort between the Army Research Laboratory and GTE Laboratories. Since the Army's communication architecture is changing from primarily voice dominated to more data/information dominated, a need was recognized to provide an automated decision support service to better assist commanders.

Their architectural concept is to separate service control functions (i.e., decision support applications) from that of resources provided by the existing communications networks, multimedia servers and information servers. These decision support applications will be split between Service Control Nodes (SCN) directly attached to the network and Service Clients residing in the client terminals at the commander's location. The key point of their architecture is the new concept of an SCN. The SCN is a network-based *intelligent agent* located between, and having access to, existing information servers and network resources. It acts as a gateway or filter between clients and the information they seek to eliminate redundancy and fuse data from various information servers. This new approach attempts to reduce bandwidth requirements by consolidating and fusing data from multiple information servers and providing that data to clients.

The SCN monitors the network status (bandwidth usage, throughput, etc.) and can reallocate bandwidth to clients since the network protocol used between the SCN and the client is asynchronous transfer mode (ATM). This is particularly important today because

the amount of data, video, audio, etc., going across networks continues to increase drastically while bandwidth limitations remain as a major constraint.

2. Fire Engagement Analysis Tool (FEAT4)

FEAT4 is a prototype, collaborative planning system developed by the Marine Corps, as part of the Sea Dragon Program, with assistance from the CAD Research Center of California Polytechnic State University. The goal of this system is to provide for real-time decision support on the battlefield. In addition to decision support, the framework provides for situational awareness, mission analysis, intelligence preparation of the battlefield, and cooperative planning across functional areas through the use of *intelligent software agents*. The agents residing on the FEAT4 workstations in the Experimental Combat Operations Center (ECOC) of the Marine Corps Commandant's Warfighting Lab (CWL) continuously monitor factors related to the planning and execution environment and provide that information to the user.

FEAT4 has service agents, mentor agents and human agents. The system uses a total of seven service agents: engagement, mission, weather, terrain, movement, logistics and network. Agent interactions are initiated and coordinated through an agent kernel that allows the agents to receive and post information on what is described as a semantic network. A semantic network in this context contains the current state information of the agent in object form. The agent kernel also coordinates the communication between agents. Mentor agents represent soldiers, weapon systems, tanks, etc. These agents provide information back to the FEAT4 workstations as to their status. The human agent interfaces with the system through the workstations. [Ref. 26]

3. Intelligent Information Dissemination Server (IIDS)

The Intelligent Information Dissemination Server (IIDS) project, sponsored by DARPA, is an enhancement of an earlier project known as the Battlefield Awareness and Data Dissemination (BADD) Information Dissemination Server (IDS). The original IDS goal was to "to automatically filter and package information and to anticipate future Warfighter's Associate (WFA) information needs" [Ref. 27]. The IIDS project goal is to

improve the accuracy and timeliness of this "smart information push" through the use of *intelligent software agents*.

IX. CONCLUSIONS

The goal of this thesis was to propose a conceptual model of intelligent software agents to support the human decision process and reconnaissance related tasks. For the purposes of the thesis, reconnaissance was defined as the collection, analysis and dissemination of information.

The Mobile Agent Reconnaissance Kit (MARK) was proposed to conduct reconnaissance tasks and to facilitate data integration and coordination in a network-centric multisenor environment using a hierarchy of intelligent software agents. MARK consists of several mission modules that are stored in the Central Agent Repository (CAR). The decision-maker retrieves the appropriate module from the CAR and provides the Central Coordinating Agent (CCA) basic information about the mission to be performed. The CCA interprets the information and assigns tasking to various Local Resident Experts (LRE) to begin reconnaissance tasks. If the LREs are not able to provide the information the decision-maker needs from local databases, a mobile agent is created to travel to remote servers. When the data is located, the mobile agent returns it to the LRE, which reviews it for relevancy and redundancy. If the information is pertinent to the queried task, it is combined with other information found by the LRE and passed to the CCA. The CCA integrates the data with that of the other LREs into an HTML file and displays it on the user's browser.

A. REVIEW OF RESEARCH QUESTIONS

The following section addresses each research question.

1. What are the major characteristics of software agents?

Software agents can be characterized by their level of intelligence, independence, reasoning, learning and cooperation. The software agents in MARK incorporate these characteristics and additional attributes shown in the taxonomy of software agents in Chapter II. The taxonomy provides a sliding scale to classify agents based on their

characteristics. The primary characteristics of software agents developed in MARK include the intelligence of agents, their mobility, the life span of an agent, the ability of agents to interact with other agents and applications, the types of agent tasking, the environment of an agent; and the behavior of agents, from autonomy to teamwork.

2. What are the current techniques for developing and deploying intelligent software agents?

There are several technologies in the marketplace that can be used to program agents. The authors recommend using Java as the programming language for agents. Currently, it is a very flexible language that can be used across multiple platforms. Java provides some security for the agents and the servers through the Java virtual machine. The Java virtual machine provides a place for agents to execute their instructions that is separate from the operating system and the hardware of the server. The benefit of this separation is that the server is protected from poorly programmed or malicious software agents. General Magic and IBM, leaders in mobile agent technology, have elected to use Java for their agent programming language.

Software agents can be deployed using synchronous communication-oriented remote procedure calls (RPC), asynchronous message-oriented remote programming (RP), and middleware. The MARK system uses remote programming concepts to control agents. Remote programming is more flexible than RPC or middleware because it does not require a continuous connection between the client and server. The agent carries the procedures it requires to execute on the remote server. This is particularly advantageous on networks that are constrained by bandwidth and connectivity, as the user's computer does not have to be connected to the network while the agent is completing its assignment.

3. How can intelligent software agents be used to assist/support the warfighter in the decision process?

The MARK system uses intelligent software agents to gather and process information to support the decision process. Intelligent software agents have the ability to sort through vast amounts of data on multiple systems simultaneously. They can perform

data filtering to eliminate redundant or irrelevant information, thereby reducing the information overload experienced by the user. Agents improve the process of human knowledge discovery by providing only information relevant to the situation, allowing the human user to focus.

Intelligent software agents have the ability to learn from the users. This can be instrumental in decreasing the decision process cycle time because the agents do not have to be told specifically what to do every time they are used. The agents are able to predict the types of information the user needs based on previous results and collaborative filtering techniques. Intelligent software agents can help improve the quality of the decisions made by expanding the sources of information available to the user on which to base the decision. Intelligent software agents place the information into a format preferred by the user, be it textual, graphical or imagery, so that the information is easily absorbed.

4. What is an application of an agent model for supporting reconnaissance related to current decision processes?

The authors applied MARK to reconnaissance tasks in several decision processes related to military intelligence. For example, MARK would be able to support the decision-maker by conducting a timely, comprehensive update of two types of intelligence essential to preparing the intelligence battlespace. General military intelligence and essential elements of information (EEI) can be gathered and provided to the decision-maker as soon as the need is identified. MARK could also be used to update operation plan (OPLAN) EEIs on a periodic basis. This allows the CCA to produce an Intelligence Estimate within a matter of minutes, rather than the hours it takes a person to do it from scratch.

The intelligent software agents in MARK could perform data fusion to present a complete, comprehensive picture of a given situation to the user, based on multiple sensor inputs. The objective of data fusion is to take information from multiple sources and use it to make inferences about the environment external to the sensors, creating a single picture. MARK could conduct data fusion on multiple levels at the same time, providing updated analysis of the situation as it changes.

MARK could also be applied to the Crisis Action Procedures (CAP) used during a time sensitive situation. CAP is comprised of multiple phases, each of which require accurate information to assist in swift decision-making. MARK could monitor the situation, provide immediate updates, and point out changes that will affect current operational plans.

5. What government and commercial projects are being developed using software agents?

Two commercial development tools for agent programming were discussed. General Magic's Odyssey and IBM's Aglets programming environments contain concepts that support MARK. In addition, Verity's Search'97 and Excalibur's Visual RetrievalWare were presented as examples of potential products that can extend the functionality of MARK.

Three government projects were reviewed that explore the use of intelligent software agent technology. The Intelligent Decision Aids, FEAT 4 and the Intelligent Agent Dissemination Server projects emphasize different aspects of software agents. Intelligent Decision Aids uses a software agent to control bandwidth usage by consolidating queries and consolidating data from multiple sources into one response. FEAT 4 uses persistent agents to monitor a situation and provide continuous updates. Intelligent Agent Dissemination Server uses intelligent software agents to improve the accuracy and timeliness of information dissemination.

6. What issues are involved with agent management, maintenance and coordination?

The thesis discussed several issues that are involved with agent management, maintenance and coordination. Some of the most important include bandwidth constraints; the training of agents; and the security of an agent supported information system.

Bandwidth considerations affect the size of an agent and the location where processing of information gathered by agents is conducted. The authors proposed development strategies that limit the bandwidth requirements of MARK.

Agent training must be conducted every time the agents are used. The user should provide feedback on the relevancy of the information returned and the format in which it is presented. Frequent use of the MARK system ensures that the agents continually learn and enhance their coordination skills. MARK agents should be trained by multiple users to prevent them from learning the biases of one person.

It is critical to ensure the security of the agents and the systems they visit. Mobile agent security issues include protection of the agent from hostile servers, protection of the server from rogue agents and protection of the information in both the server and the agent. The authors proposed a security protocol for mobile agents that combines security techniques found in Netscape's Secure Sockets Layer, digital certificates, public and private key cryptography, and hash functions.

B.   RECOMMENDATIONS FOR FUTURE RESEARCH

The ideas in this thesis are structurally sound in a conceptual environment. The next step is to develop a prototype of MARK and validate it in an operational environment. Agent technology exists to fulfill most of the functions described in the MARK system. As the capabilities of agents continue to develop, the MARK model should be built and deployed in a JDISS or GCCS environment for a proof of concept study.

The further application of MARK in the intelligence community is an another area that should be examined. The methods by which intelligence is conducted may be changed with the advent of intelligent software agents.

The organizational implications of agents should be further explored. Intelligent software agents can have a tremendous impact on the way business is conducted on a day-to-day basis. Agents can assist their human users in some areas and replace them in others. The breakdown of tasks that can be performed by an intelligent software agent instead of a human should be studied, with a view to optimizing the use of both the agent and the person. The trust relationship that must be developed before the user will allow the agent to make decisions that impact human life is an area that should be examined.

Commercial products exist that can be used to extend the functionality of agent based information systems by providing advanced search services to mobile agents. Verity's Search'97 and Excalibur's Visual RetrievalWare are examples of such products. Other products should be identified, tested and evaluated in terms of their performance, how well they interface with MARK and what additional services they can provide.

APPENDIX INTELLIGENCE ESTIMATE

This appendix provides an example of the Intelligence Estimate used by the intelligence staff to provide information to the decision-maker. It is an appraisal of the information related to a specific situation, developed to assist in the determination of potential courses of action. It provides general military intelligence about the adversary and information regarding the adversary's capabilities.

The information found in this appendix is from Joint Pub 2-01, Appendix D.

SAMPLE INTELLIGENCE ESTIMATE FORMAT

INTELLIGENCE ESTIMATE

SECURITY CLASSIFICATION

Originating Section Issuing Headquarters*
Place of Issue
Day, Month, Year, Hour, Zone

INTELLIGENCE ESTIMATE NUMBER**

( ) REFERENCES: a. Maps and Charts.
                 b. Other relevant documents.

1. ( ) Mission. State the assigned task and its purpose. The mission of the command as a whole is taken from the commander÷s mission analysis, planning guidance, or other statement.

2. ( ) Adversary Situation. State conditions that exist and indication of effects of these conditions on adversary capabilities and the assigned mission. This paragraph describes the operational area, the adversary military situation, and the effect of these two factors on adversary capabilities.

    a. ( ) Characteristics of the Operational Area. Discuss the effect of the physical characteristics of the operational area on military activities of both combatants. If an analysis of the area has been prepared separately, this paragraph in the intelligence estimate may simply refer to it, then discuss the effects of the existing situation on military operations in the area.

        (1) ( ) Military Geography

           (a) ( ) Topography

* When this estimate is distributed outside the issuing headquarters, the first line of the heading is the official designation of the issuing command, and the ending of the estimate is modified to include authentication by the authorizing section, division, or other official according to local policy.

** Normally, these are numbered sequentially during a calendar year.

SECURITY CLASSIFICATION

1. ( ) Existing Situation. Describe relief and drainage, vegetation, surface materials, cultural features and other characteristics in terms of their effect on key terrain, observation, fields of fire, obstacles, cover and concealment, avenues of approach, lines of communications, and landing areas and zones.

2. ( ) Effect on Adversary Capabilities. Discuss the effect of topography on broad adversary capabilities such as attack and defense, describing generally how the topography affects each type of activity. The effect on employment of nuclear, biological, and chemical (NBC) weapons; amphibious, airborne, or air-landed forces; surveillance devices and systems; communications equipment and systems; electronic warfare; psychological operations, operations security and military deception; logistic support; and other appropriate considerations should be included.

3. ( ) Effect on Friendly Course of Action (COA). Discuss the effects of topography on friendly forces÷ military operations (attack, defense) in the same fashion as for adversary capabilities in the preceding subparagraphs.

(b) ( ) Hydrography

1. ( ) Existing Situation. Describe the nature of the sea and the coastline within the amphibious objective area; adjacent islands; location, extent, and capacity of landing beaches and their approaches and exits; nature of the offshore approaches, including type of bottom and gradients; natural obstacles; surf, tide, and current conditions.

2. ( ) Effect on Adversary Capabilities. Discuss the effects of the existing situation on broad adversary capabilities.

3. ( ) Effect on Friendly COAs, Discuss the effects of the existing situation on broad COAs for friendly forces.

(c) ( ) Climate and Weather

1. ( ) Existing Situation. Describe temperature, cloud cover, visibility, precipitation, light data, and other climate and weather conditions and their general effects on roads, rivers, soil trafficability, and observation.

2. ( ) Effect on Adversary Capabilities. Discuss the effects of the existing climate and weather situation on broad adversary capabilities.

3. ( ) Effect on Friendly COAs. Discuss the effects of the existing climate and weather situation on broad COAs for friendly forces.

(2) ( ) Transportation

(a) ( ) Existing Situation. Describe roads, railways, inland waterways, airfields, and other physical characteristics of the transportation system; capabilities of the transportation system in terms of rolling stock, barge capacities, and terminal facilities; and other pertinent data.

(b) ( ) Effect on Adversary Capabilities. Discuss the effects of the existing transportation system and capabilities on broad adversary capabilities.

(c) ( ) Effect on Friendly COAs. Discuss the effects of the existing transportation system and capabilities on broad COAs for friendly forces.

(3) ( ) Telecommunications

(a) ( ) Existing Situation. Describe telecommunications facilities and capabilities in the area.

(b) ( ) Effect on Adversary Capabilities. Discuss the effects of the existing telecommunications situation on broad adversary capabilities.

(c) ( ) Effect on Friendly COAs. Discuss the effects of the existing telecommunications situation on broad COAs for friendly forces.

(4) ( ) Politics

(a) ( ) Existing Situation. Describe the organization and operation of civil government in the operational area.

(b) ( ) Effect on Adversary Capabilities. Consider the effects of the political situation on broad adversary capabilities.

(c) ( ) Effect on Friendly COAs. Consider the effects of the political situation on broad COAs for friendly forces.

(5) ( ) Economics

(a) ( ) Existing Situation. Describe industry, public works and utilities, finance, banking, currency, commerce, agriculture, trades and professions, labor force, and other related factors.

(b) ( ) Effect on Adversary Capabilities. Discuss the effects of the economic situation on broad adversary capabilities.

(c) ( ) Effect on Friendly COAs. Consider the effects of the economic situation on broad COAs for friendly forces.

(6) ( ) Sociology

(a) ( ) Existing Situation. Describe language, religion, social institutions and attitudes, minority groups, population distribution, health and sanitation, and other related factors.

(b) ( ) Effect on Adversary Capabilities. Discuss the effects of the sociological situation on broad adversary capabilities.

(c) ( ) Effect on Friendly COAs. Discuss the effects of the sociological situation on COAs for friendly forces.

(7)   ( ) Science and Technology

(a) ( ) Existing Situation. Describe the level of science and technology in the operational area.

(b) ( ) Effect on Adversary Capabilities. Discuss the effects of science and technology on broad adversary capabilities.

(c) ( ) Effect on Friendly COAs. Discuss the effects of science and technology on broad COAs for friendly forces.

b. ( ) Adversary Military Situation (Ground, Naval, Air, Other Service)

(1) ( ) Strength. State the number and size of adversary units committed and adversary reinforcements available for use in the operational area. Ground strength, air power, naval forces, NBC weapons, electronic warfare, unconventional warfare, surveillance potential, and all other strengths (which might be significant) are considered.

(2) ( ) Composition. Outline the structure of adversary forces (order of battle) and describe unusual organizational features, identity, armament, and weapon systems.

(3) ( ) Location and Disposition. Describe the geographic location of adversary forces in the area, including fire support elements; command and control facilities; air, naval, and missile forces; and bases.

(4) ( ) Availability of Reinforcements. Describe adversary reinforcement capabilities in terms of ground, air, naval, missile, and NBC forces and weapons, terrain, weather, road and rail nets, transportation, replacements, labor forces, prisoner of war policy, and possible aid from sympathetic or participating neighbors.

(5) ( ) Movements and Activities. Describe the latest known adversary activities in the area.

(6) ( ) Logistics. Describe levels of supply, resupply ability, and capacity of beaches, ports, roads, railways, airfields, and other facilities to support supply and resupply. Consider hospitalization and evacuation, military construction, labor resources, and maintenance of combat equipment.

(7) ( ) Operational Capability to Launch Missiles. Describe the total missile capability that can be brought to bear on forces operating in the area, including characteristics of missile systems, location and capacity of launch or delivery units, initial and sustained launch rates, size and location of stockpiles, and other pertinent factors.

(8) ( ) Serviceability and Operational Rates of Aircraft. Describe the total aircraft inventory by type, performance characteristics of operational aircraft, initial and sustained sortie rates of aircraft by type, and other pertinent factors.

(9) ( ) Operational Capabilities of Combatant Vessels. Describe the number, type, and operational characteristics of ships, boats, and craft in the naval inventory; base location; and capacity for support.

(10) ( ) Technical Characteristics of Equipment. Describe the technical characteristics of major items of equipment in the adversary inventory not already considered (such as missiles, aircraft, and naval vessels).

(I1) ( ) Electronics Intelligence. Describe the adversary intelligence-gathering capability using electronic devices.

(I2) ( ) Information Warfare. Describe the adversary offensive and defensive IW capabilities.

(I3) ( ) NBC Weapons. Describe the types and characteristics of NBC weapons in the adversary inventory, stockpile data, delivery capabilities, NBC employment policies and techniques, and other pertinent factors.

(14) ( ) Significant Strengths and Weaknesses. Discuss the significant adversary strengths and weaknesses perceived from the facts presented in the preceding subparagraphs.

c. ( ) Adversary Unconventional and Psychological Warfare Situation

(1) ( ) Guerrilla. Describe the adversary capability for, policy with regard to, and current status in the area of guerrilla or insurgent operations.

(2) ( ) Psychological. Describe adversary doctrine, techniques, methods, organization for, and conduct of psychological operations in the operational area.

(3) ( ) Subversion. Describe adversary doctrine, techniques, methods, organization for, and conduct of subversion in the operational area.

(4) ( ) Sabotage. Outline adversary organization and potential for and conduct of sabotage in the operational area.

3. ( ) Adversary Capabilities

a. ( ) Listing each adversary capability that can affect the accomplishment of the assigned mission. Each adversary capability should contain information on the following:

(1)   ( ) What the adversary can do.

(2)   ( ) Where they can do it.

(3)   ( ) When they can start it and get it done.

(4)   ( ) What strength they can devote to the task.

b. ( ) In describing adversary capabilities, the J-2 must be able to tell the commander what the adversary can do using its forces in a joint environment. First, of course, the J-2 must assess the adversary÷s ground, naval, and air forces. It is customary to enumerate separately the NBC and unconventional warfare capacities. Hypothetical examples follow.

(1)   ( ) Ground Capabilities

(a) ( ) The adversary can attack at any time along our front with an estimated 6 infantry divisions and 2 tank divisions supported by 24 battalions of artillery.

(b) ( ) The adversary can defend now in its present position with 7 infantry divisions supported by 2 tank divisions and 16 battalions of medium and light artillery.

(c) ( ) The adversary can reinforce its attack (or defense) with all or part of the following units in the times and places indicated:

UNIT	PLACE	TIME
315th Airborne Div	Vic RESOGA	8 hrs after starting time
41st Motorized	Vic CARDINAL	6 hrs after starting time

(2)　　( ) Air Capabilities

(a) ( ) Starting now, and based on an estimated strength of 300 fighters and 100 medium bomber aircraft, the adversary can attack in the operational area with 240 fighter sorties per day for the first 2 days, followed by a sustained rate of 150 sorties per day, and 60 bomber sorties per day, for 1 day followed by a sustained rate of 48 sorties per day.

(b) ( ) Using airfields in the vicinity of , the adversary has sufficient transport sorties to lift one regiment in a single lift to airfields in the vicinity of ____ and ____ within 4 hours÷ flying time.

(3) ( ) Naval Capabilities. Starting now, the adversary can conduct sustained sea and air operations in the entire area with 6 DDs, 4 FFs, 1 CV, 7 SSNS, a mine force of 20 craft, and 70 gunboats and smaller craft now on station in the area.

(4) ( ) Nuclear Capabilities. The adversary can employ at any time and in any part of the operational area an estimated 40 to 60 nuclear weapons of yields from 2 to 50 kt delivered by cannon and rocket artillery, guided missile, and aircraft.

(5) ( ) Biological and Chemical Capabilities. The adversary can employ the biological and chemical agents , , and in the operational area at any time delivered by air, cannon, and rocket artillery and by guided missile.

(6) ( ) Unconventional Warfare (UW) Capability. The adversary can conduct UW operations in the area within 10 days after starting the operation using dissident ethnic elements and the political adversaries of the current government.

(7) ( ) Joint Capabilities. The adversary can continue to defend its present position with 6 infantry divisions, supported by 16 artillery battalions and reinforced by 3 mechanized divisions within 8 hours after starting movement. Adversary defense also can be supported by 150 fighter sorties daily for a sustained period and by continuous naval surface and air operations employing 6 DDs, 4 FFs, 7 SSNS, and 1 CV.

4. ( ) Analysis of Adversary Capabilities. Analyze each capability in light of the assigned mission (considering all applicable factors from paragraph 2 above) and attempt to determine and give reasons for the relative order of probability of adoption by the adversary. Discuss adversary vulnerabilities. In this paragraph, examine the adversary capability by discussing the factors that favor or militate against its adoption by the adversary. When applicable, the analysis of each capability should also include a discussion of adversary vulnerabilities attendant to that capability; i.e., conditions or circumstances of the adversary situation that render the adversary especially liable to damage, deception, or defeat. Finally, that analysis should include a discussion of any indications that point to possible adoption of the capability, as in the following:

    a.   ( ) Attack now with forces along the forward edge of the battle area ....

        (1)   ( ) The following factors favor the adversary÷s adoption of this capability:

(a) ( ) ....

(b) ( ) ....

        (2) ( ) The following factors militate against the adversary÷s adoption of this capability:

        (a) ( ) Road and rail nets will not support large-scale troop and supply movements necessary for an attack in the area.

        (b) ( ) Terrain in the area does not favor an attack.

        (3) ( ) Adoption of this capability will expose the adversary÷s west flank to counterattack.

        (4) ( ) Except for minor patrol activity in the area, there are no indications of adoption of this capability.

    b.   ( ) Delay from present positions along the River line ....

(1) ( ) The following factors favor the adversary÷s adoption of this capability:

        (a) ( ) There are several excellent natural barriers between the River and the Mountains.

        (b) ( ) The effectiveness of the water barriers will improve, and trafficability on the upland slopes of the terrain barriers will deteriorate with advent of the monsoon.

(2) ( ) The following factors militate against the adversary÷s adoption of this capability:

(a) ( ) ....

(b) ( ) ....

(3) ( ) In the adoption of this capability, the adversary÷s lines of communications will be restricted by a limited road and rail net that can easily be interdicted.

(4) ( ) The following facts indicate adoption of this capability:

(a) ( ) Aerial photography indicates some preparation of barriers in successive positions.

(b) ( ) Considerable troop movement and prepositioning of floating bridge equipment along the water barriers have been detected.

5. ( ) Conclusions. Conclusions resulting from discussion in paragraph 4 above. Include, when possible, a concise statement of the effects of each capability on the accomplishment of the assigned mission. Cite adversary vulnerabilities where applicable. This paragraph contains a summary of adversary capabilities most likely to be adopted, listed in the order of relative probability if sufficient information is available to permit such an estimate. If appropriate, it should also include a concise statement of the effects of each adversary capability on the accomplishment of the assigned mission. Exploitable vulnerabilities should also be listed, where applicable.

a. ( ) Adversary Capabilities in Relative Probability of Adoption

(1) ( ) Defend in present locations with ....

(2) ( ) Delay from present positions along ....

(3) ( ) Reinforce the defense or delay with ....

(4) ( ) Conduct UW operations in the area ....

b. ( ) Vulnerabilities

(1) ( ) Adversary left (west) flank is open to envelopment by amphibious assault ....

(2) ( ) The adversary÷s air search radar coverage is poor in the left (west) portion of its defensive sector ....

SECURITY CLASSIFICATION

(Signed) _____
J-2

(The staff division chief signs the staff estimates produced by that division. If the estimate is to be distributed outside the headquarters, the heading and signature block must be changed to reflect that fact.)

ANNEXES: (By letter and title) Annexes should be included where the information is in graphs or of such detail and volume that inclusion makes the body of the estimate cumbersome. They should be lettered sequentially as they occur throughout the estimate.

DISTRIBUTION: (According to procedures and policies of the issuing headquarters)

SECURITY CLASSIFICATION

LIST OF REFERENCES

1. Kato, Kazuhiko, University of Tsukuba. http://www.softlab.is.tsukuba.ac.jp/OS/kato/index.htm, 1998.

2. Picco, Gian Pietro, Politecnico di Milano http://www.polito.it/~picco, 1998.

3. Vitek, Jan, University of Geneva http://cuiwww.unige.ch/~jvitek, 1998.

4. General Magic, *Telescript Technology: Mobile Agents.* 1996, http://www.genmagic.com/html/ telescript.html

5. Kalakota, Ravi and Whinston, Andrew B., *Frontiers of Electronic Commerce*, Addison Wesley, Reading, MA, 1995.

6. Bui, Tung and Sankaran, Siva, "Group Decision and Negotiation in Telemedicine: An Application of Intelligent Mobile Agents as Non-Human Teleworkers", Proceedings of the 30th International Conference on System Sciences, Maui, Hawaii, January 1997.

7. Kalakota, Ravi; Stallaert, Jan; Whinston, Andrew B., "Mobile Agents and Mobile Workers", Proceedings of the 29th Annual Hawaii International Conference on System Sciences, 1996.

8. Turban, Efraim. *Decision Support and Expert Systems: Management Support Systems,* 4th Edition, Prentice Hall, January 1995.

9. Sycara, Katia and Zeng, Dajun, "Cooperative Intelligent Software Agents", Carnegie Mellon University, March 1995.

10. Feigenbaum, John and Lee, Peter, "Trust Management and Proof-Carrying Code in Secure Mobile Code Applications", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 March 97.

11. Hall, David L., Mathematical Techniques in Multisensor Data Fusion, Artech House, 1996.

12. General Magic, *Odyssey FAQ* 1996. http://www.genmagic.com/html/odyssey-faq.html

13. Gilbert, Don, and Janca, Peter, *IBM Intelligent Agents*, IBM Corporation, March 1996, http://www.raleigh.ibm.com/iag/iagwp1.html

14.   Rosen, Michele "Internet Security Standards." *PC Magazine*, January 20, 1998, p 242.

15.   Courtot, Philippe, SEARCH'97 White Paper, A New Personal and Enterprise Application. Verity, www.verity.com/products/prd.html

16.   CNO N6, *Copernicus Document*, 1997.

17.   Autonomy, *AgentWare Version 1.5L*. http://www.agentware.com, 1996.

18.   Dragan, Richard V., "Future Agent Software", *PC Magazine*, 25 March 97.

19.   Finnie, Scot, "Information Retrieval for Your Intranet", *PC Magazine*, 6 May 97.

20.   DoD Dictionary of Military and Associated Terms, Joint Publication 1-02, 3 Nov 97.

21.   Joint Intelligence Support to Military Operations, Joint Publication 2-01, 20 Nov 96.

22.   Antony, Richard T., *Principles of Data Fusion Automation*, Boston, MA, Artech House, 1995.

23.   *Joint Staff Officer's Guide*, Armed Forces Staff College Publication 1, Jan 97. www.afsc.edu/pub1.htm

24.   Excalibur Technologies. "Excalibur Visual RetrievalWare." http://www.excalib.com/products/vrw/vrw.html, February 1998.

25.   Bitar, Nabil, Hinnawi, Nabil, Rivera, Brian, Voruganti, Ram, Yu, Che-Fn, "Intelligent Decision Aids for 21st Century C4I Architectures", MILCOM Conference, Monterey, CA, 2-5 Nov 97.

26.   Pohl, J. *Fire Engagement Analysis Tool (FEAT4)*, CAD Research Center, California Polytechnic Institute, 1997.

27.   Dukes-Schlossberg, Jon and Lee, Yongwon and Lehrer, Nancy, "IIDS: Intelligent Information Dissemination Server", MILCOM Conference, Monterey, CA, 2-5 Nov 97.

BIBLIOGRAPHY

"A Collaborative Planning System for the US Marine Corps Sea Dragon Program", CAD Research Center, California Polytechnic State University, 1997.

"Agents to Represent Traders", *Wall Street & Technology,* 1 Jul 96.

Alvarez, Cesar, "Intelligent Agents Help to Minimize Bandwidth by Browsing Offline." *LAN Times,* 20 Jan 97.

Anderson, Jon R., "Hunter Warrior: Testing New Ideas/Exercise Will Pit Speed, Stealth Against Armor", *Navy Times,* 3 Mar 97.

Antony, Richard T., *Principles of Data Fusion Automation,* Boston, MA, Artech House, 1995.

Autonomy, *AgentWare Version 1.5L.* http://www.agentware.com, 1996.

Bitar, Nabil, Hinnawi, Nabil, Rivera, Brian, Voruganti, Ram, Yu, Che-Fn, "Intelligent Decision Aids for 21st Century C4I Architectures", MILCOM Conference, Monterey, CA, 2-5 Nov 97.

Bui, Tung, "An Agent-based Framework for Building DSS", Proceedings of the Fourth International Conference on Decision Support Systems, ISDSS '97, University of Lausanne, July 1997.

Bui, Tung and Sankaran, Siva, "Group Decision and Negotiation in Telemedicine: An Application of Intelligent Mobile Agents as Non-Human Teleworkers", Proceedings of the 30th International Conference on System Sciences, Maui, Hawaii, January 1997.

Canamero, Dolores, "Plan Recognition for Decision Support", *IEEE Expert,* 1996.

CNO N6, *Copernicus Document,* 1997.

Courtot, Philippe, *SEARCH '97 White Paper, A New Personal and Enterprise Application.* Verity, www.verity.com/products/prd.html

Crowston, Kevin, "Market Enabling Internet Agents", Proceedings of the 17th ICIS, 16-18 December 96.

DeVoe, Deborah, "SRI Distributed Agents Promise Flexibility", *Infoworld,* 23-30 December 1996.

Dean, Drew and Felton, Edward W., "Secure Mobile Code: Where do we go from here?", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

Doctrine for Planning Joint Operations, Joint Publication 5-0, 13 Apr 95.

DoD Dictionary of Military and Associated Terms, Joint Publication 1-02, 3 Nov 97.

Dragan, Richard V., "Future Agent Software", *PC Magazine*, 25 Mar 97.

Dukes-Schlossberg, Jon and Lee, Yongwon and Lehrer, Nancy, "IIDS: Intelligent Information Dissemination Server", MILCOM Conference, Monterey, CA, 2-5 Nov 97.

Excalibur Technologies. "Excalibur Visual RetrievalWare." http://www.excalib.com/products/vrw/vrw.html, February 1998.

"Exploiting the Revolution", *InfoWorld*, November 1996.

Feigenbaum, John and Lee, Peter, "Trust Management and Proof-Carrying Code in Secure Mobile Code Applications", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

"Final Report for the Intelligence Analyst Associate (IAA), Engineering Change Proposal (ECP)", Systems Research and Applications Corporation, September 1996.

Finnie, Scot, "Information Retrieval for Your Intranet", *PC Magazine*, 6 May 97.

Focardi, Riccardo and Gorrieri, Roberto, "Non Interference: Past, Present, Future", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

Geddings, John C., *Intelligent Agents*, Faulkner Information Services, 1996.

General Magic, *Tabriz White Paper. Transforming Passive Networks into an Active, Persistent, and Secure Business Advantage*, March 1997. http://www.genmagic.com/html/tabriz.html

General Magic, *Telescript Technology: Mobile Agents*,. 1996. http://www.genmagic.com/html/telescript.html

General Magic, *Odyssey FAQ* 1996. http://www.genmagic.com/html/odyssey-faq.html

Gilbert, Don, and Janca, Peter, *IBM Intelligent Agents*, IBM Corporation, March 1996. http://www.raleigh.ibm.com/iag/iagwp1.html

Gong, Li, "Surviveable Mobile Code is Hard to Build", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

Gunter, Carl A., Homeier, Peter and Nettles, Scott, "Infrastructure for Proof-Referencing Code, DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

Hall, David L., *Mathematical Techniques in Multisensor Data Fusion*, Artech House, 1996.

Huhns, Michael N. and Singh, Munindar P., "Agents on the Web", *IEEE Internet Computing*, May/June 1997.

Hwang, Diana, "Intelligent Agents Seek Place on Web – Technology Crucial to Managing Internet" *Emerging Technologies*, 12 Feb 96.

Intelligence and Communications Architectures (INCA) Project Office, *Communications Support to a Joint Task Force (JTF). A Framework for Planning.* 6 May 94.

"Intelligent Agents for Real?", *Computer Reseller News,* 21 Oct 1996.

Jameson, Steve and Whitebread, Kenneth R., "Information Discovery in High-Volume, Frequently Changing Data", *IEEE Expert*, October 1995.

Joint Doctrine for Intelligence Support to Operations, Joint Publication 2-0, 5 May 95.

Joint Intelligence Support to Military Operations, Joint Publication 2-01, 20 Nov 96.

Joint Staff Officer's Guide, Armed Forces Staff College Publication 1, Jan 97.
www.afsc.edu/pub1.htm

Joint Warfare of the Air Force of the United States, Joint Publication 1, 10 Jan 95.

Kalakota, Ravi and Whinston, Andrew B., *Frontiers of Electronic Commerce*, Addison Wesley, Reading, MA, 1995.

Kalakota, Ravi; Stallaert, Jan; Whinston, Andrew B., "Mobile Agents and Mobile Workers", Proceedings of the 29th Annual Hawaii International Conference on System Sciences, 1996.

Laudon, Kenneth C. and Laudon, Jane P., *Management Information Systems, New Approaches to Organization and Technology*, 5th edition, Prentice Hall, 1998.

Lidsky, David, "The Web Delivers", *PC Magazine*, 18 Feb 97.

Meadows, Catherine, "Detecting Attacks on Mobile Agents", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

Pohl, J. *Fire Engagement Analysis Tool (FEAT4),* CAD Research Center, California Polytechnic Institute, 1997.

Powers, Michael, "The Intelligence Fusion Center (IFC): A COTS-Based Information Retrieval, Processing, and Archiving System", MILCOM Conference, Monterey, CA, 2-5 Nov 97.

Radosevich, Linda, "Intelligent Agents Arrive for Data Management", *Infoworld,* 23-30 Dec 96.

Ram, Sudha; Venkatsubramanyan, Shailaja; Marsh, Stuart; Ball, George, *Resource Discovery and Intelligent Image Retrieval in a Distributed Environment,* IEEE (HICSS), 1997.

Rosen, Michele "Internet Security Standards." *PC Magazine,* January 20, 1998, p 242.

Schemmer, Benjamin F., *The Raid,* Harper & Row, Publishers, 1976, pages 210-213.

Snell, Monica, "Oracle Adds CDPD* Support to Mobile Agent Technology", October 97. http://www.uworld.com/lantimes/archive/505b014c.html

St. Clair, William G., "Agents Ready to Tackle Mobile Web Task", *Electronic Engineering Times,* 18 Nov 96.

Spitzer, Tom, "Needles in Document Haystacks: Text retrieval and Management Technologies Come of Age", *DBMS,* 1 Jan 96.

Swarup, Vipin, "Trust Appraisal and Secure Routing of Mobile Agents", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

Sycara, Katia and Zeng, Dajun, "Cooperative Intelligent Software Agents", Carnegie Mellon University, March 1995.

Toomey, Christopher, "Satellite Image Dissemination via Software Agents", *IEEE Expert,* 1995.

Turban, Efraim. Decision Support and Expert Systems: Management Support Systems, 4th Edition, Prentice Hall, January 1995.

Yee, Bennett S., "A Sanctuary for Mobile Agents", DARPA Workshop on Foundations for Secure Mobile Code Workshop, Monterey, CA, 26-28 Mar 97.

INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Professor Carl R. Jones, Code SM/Jo 1
   Systems Management Department
   Naval Postgraduate School
   Monterey, CA 93943

4. Mr. Franklin White 1
   Naval Command, Control and Ocean Surveillance Center RDTE
   53140 Gatchell Road
   San Diego, CA 92152-5001

5. Professor John Powers..6
   Center for Reconnaissance Research
   Naval Postgraduate School
   Monterey, CA 93943

6. Superintendent 1
   (Attn: Code EC/LT)
   Naval Postgraduate School
   Monterey, CA 93943-5121

7. Naval Information Warfare Activity 2
   (Attn: LT Marcia R. Edmiston)
   9800 Savage Road
   Fort George G. Meade, MD 20755-6000

8. CPT Darrell R. Gregg, Jr. 2
   1701 Darr Street
   Hannibal, MO 63401