# REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188



| 1. AGENCY USE ONLY (Leave blank) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | 10 Jan 1997 | Final |

**4. TITLE AND SUBTITLE**
Functional Description of A Command Agent

**5. FUNDING NUMBERS**
N61339-96-D-0002

**6. AUTHOR(S)**

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Lockheed Martin Corporation
Information Systems Company
12506 Lake Underhill Road
Orlando, FL 32825

**8. PERFORMING ORGANIZATION REPORT NUMBER**
ADST-II-CDRL-023R-9600237A

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

NAWCTSD/STRICOM
12350 Research Parkway
Orlando, FL 32826-3224

**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES**

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

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** *(Maximum 200 Words)*

This research activity examines the current state-of-the-art in modeling the command decision process and implementing such models in software. The primary and initial target application is in automated command agents for DIS/ADS. This report was prepared for the Command Decision Modeling ADST II Delivery Order in accordance with the following documents:

a) Command Decision Modeling Overview (ADST-II-CDRL-023A-9600236)
b) Functional Description of a Command Agent (ADST-II-CDRL-023A-9600237)
c) Rule Based Systems (ADST-II-CDRL-023A-9600238)
d) Genetic Algorithms and Evolutionary Programming (ADST-II-CDRL-023A-9600239)
e) Petri Nets and Colored Petri Nets (ADST-II-CDRL-023A-9600240)
f) Neural Networks and Bounded Neural Networks (ADST-II-CDRL-023A-9600241)
g) Case-Based Reasoning (ADST-II-CDRL-023A-9600242)

| 14. SUBJECT TERMS | 15. NUMBER OF PAGES |
|---|---|
| Command Agent; ADST-II; STRICOM; Simulation; DIS; DIS/ADS | 16 |
| | 16. PRICE CODE |

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UL |

# ADVANCED DISTRIBUTED SIMULATION TECHNOLOGY II (ADST II)

# DELIVERY ORDER #0018

# CDRL AB02

# FUNCTIONAL DESCRIPTION OF A COMMAND AGENT




FOR: NAWCTSD/STRICOM
12350 Research Parkway
Orlando, FL 32826-3224
N61339-96-D-0002
DI-MISC-80711

BY: Lockheed Martin Corporation
Martin Marietta Technologies, Inc.
Information Systems Company
12506 Lake Underhill Road
Orlando, FL 32825

# *DOCUMENT CONTROL INFORMATION*

| Revision | Revision History | Date |
|---|---|---|
| | Original release | 7/15/96 |
| A | Incorporated customer comments | 1/10/97 |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |
| - | | |

# Table of Contents

# PREFACE

This research activity examines the current state-of-the-art in modeling the command decision process and implementing such models in software. The primary and initial target application is in automated command agents for DIS/ADS. This report was prepared for the Command Decision Modeling ADST II Delivery Order in accordance with the following documents:

a) Command Decision Modeling Overview (ADST-II-CDRL-023A-9600236)
b) Functional Description of a Command Agent (ADST-II-CDRL-023A-9600237)
c) Rule Based Systems (ADST-II-CDRL-023A-9600238)
d) Genetic Algorithms and Evolutionary Programming (ADST-II-CDRL-023A-9600239)
e) Petri Nets and Colored Petri Nets (ADST-II-CDRL-023A-9600240)
f) Neural Networks and Bounded Neural Networks (ADST-II-CDRL-023A-9600241)
g) Case-Based Reasoning (ADST-II-CDRL-023A-9600242)

# 1.0 Functional Description of a Command Agent

A **command agent** is an intelligent agent that will act as a surrogate for a human commander in a military simulation. To develop command agents that are designed for change within the evolving simulation domain, requires an architecure which will allow the heterogeneous application of decision-making strategies, the management of myriad interfaces to simulation services and knowledge sources, and the proper expression of command behavior. The approach suggested is to develop an integrated collection of sub-agent **advisors** which is arbitrated and controlled by an overwatching **command agent**. This architecture can be applied to many different domains and be used to model different approaches to command decision making.

# 2.0 The Command Decision-Making Domain

Army commanders, by their very definition, make decisions. They make decisions on how to act in a given situation. They then formulate orders that are given to their subordinates who in turn can make decisions about how to act and give orders to their subordinates. This is the hierarchical structure of the military (refer to Figure 1, ADST-II-023A-9600236 Command Decision Modeling Overview) that allows a commander to control a large number of men and materiel without being overwhelmed with information. It also allows specialization in which one individual can gain deep knowledge about a specific aspect of the military domain, such as intelligence, logistics, artillery, etc.

Computer Generated Forces (CGF) must demonstrate realistic and valid behaviors of military entities in order to provide an effective training environment or predictive capability. To exhibit advanced coordination and cooperation of military units (which are found on a real battlefield) simulations turn toward modeling the commander of those forces to provide the Command, Control, and Communication ($C^3$) required.

The implementation of a model utilized to simulate the command decision making process will hereafter be referred to as a **command agent**. "An agent is a surrogate for a person or process that fulfills a stated need or action. [i]" A command agent provides a framework for the aggregation and abstraction of behaviors of various military echelons. It also allows the $C^3$ process to be closely patterned after the one utilized in the real world. The Stimulus, Hypothesis, Options, Response (SHOR) framework (see Figure 2, ADST-II-023A-9600236 Command Decision Modeling Overview) indicates a proven model for initiating and executing decision making processes.

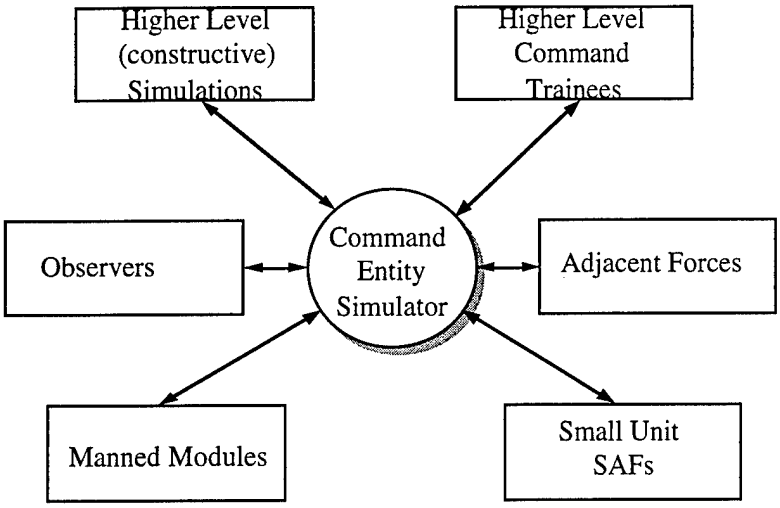


**Figure 1:** Command Agent Interfaces

Figure 3 of ADST-II-023A-9600236 Command Decision Modeling Overview shows the SHOR framework as applied within CGF. Decision making is divided into two phases: one for situation assessment followed by the assessment of options. Data received from the simulation environment stimulates the system to carry out assessments of situations and generate hypotheses. These hypotheses represent assessments of the situation. The option assessment phase uses these hypotheses to generate options (in military parlance: courses of action). Some decision criteria are generated and used to select an option for execution.

## 3.0 Implementation Implications

The general description of the command decision making domain described above and in ADST-II-023A-9600236 Command Decision Modeling Overview captures the process appropriately. However, for military simulations, there are many different implementation issues that must be addressed. In a military simulation there are a variety of interfaces that a command agent must be capable of utilizing. Figure 1 shows a representative collection of systems that are involved in a modern Distributed Interactive Simulation (DIS) environment.

The growing trend in the military simulation community is for greater degrees of system interoperability, computer resource distribution, and entity and aggregate behavioral complexity. Simulations of today must find some optimal solution to balance these contentious goals. A command agent must also be designed and implemented for these varying degrees of integration.

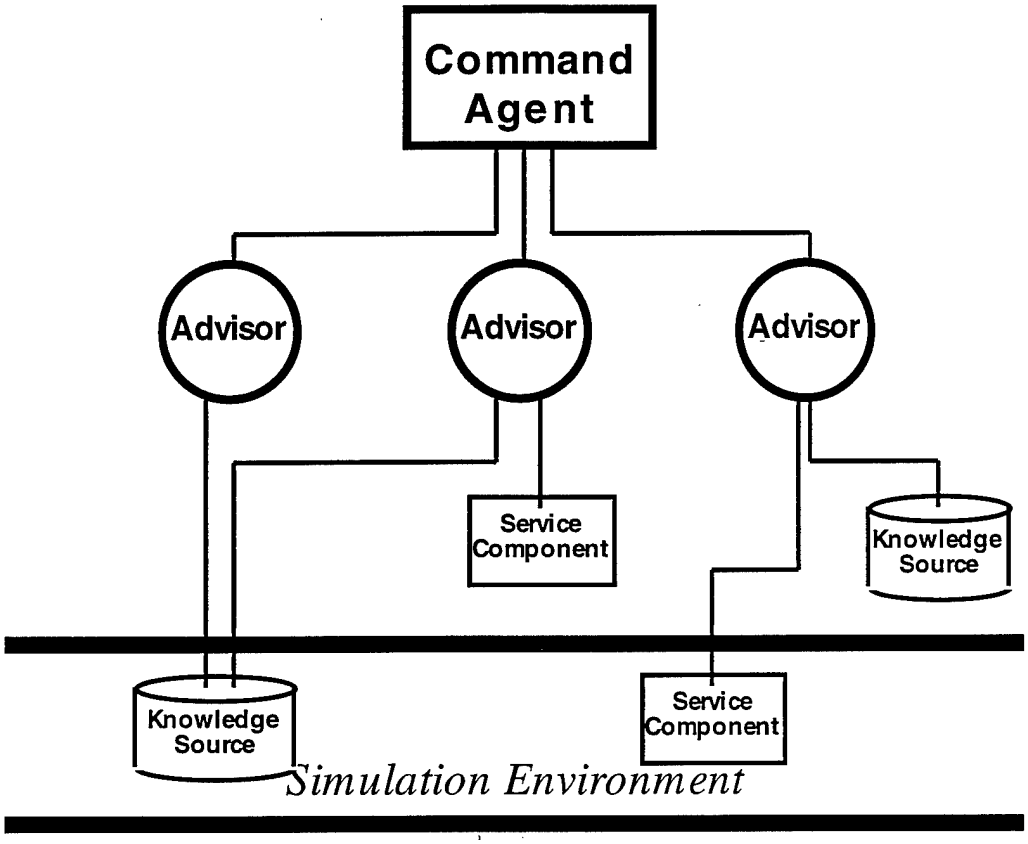


**Figure 2:** Command Agent Architecture

Complex decision making systems are typically a hybrid of artificial intelligence techniques. They deal with diverse types of information and knowledge that can be used in different contexts. A result of the work reported in [ii] was the need for a command agent to deal with environments rich with variegated information. The design of a command agent should reflect the need for heterogeneous integration both internally and externally. Figure 2 shows the generic layered approach suggested for constructing a command agent. A detailed application of this approach can be found in [iii] where "many different reasoning processes, societies of agents, are integrated in order to realize a software assistant capable of performing a broad range of tasks."

This design allows the implementation of different artificial intelligence techniques to be applied to their appropriate categories of problems. It also provides facilities to abstract the interfaces to external knowledge sources and simulations. The key to this approach is the implementation of advisors which are sub-agents constructed to monitor, interpret, and reason about information flowing into the command agent from various sources (i.e., terrain databases, sensor objects within the simulation, reports from subordinate units, etc.). The process modeled within the command agent can remain stable while new interfaces to various simulation components can be managed and translated by the advisors.

To correlate with the SHOR framework (Figure 3, ADST-II-023A-9600236 Command Decision Modeling Overview), stimulus is communicated from the simulation environment through the command agent's advisors. The advisors handle the Application Programmers Interfaces (APIs) to heterogeneous components within the simulation, which provide information and accept direction. Advisor's receive information, either passively or through query, and synthesize the information into situation hypotheses. These hypotheses are translated into the command agent's internal representation. The command agent acts to arbitrate advisor input and to control the services and interfaces that the advisors provide. The command agent has knowledge of how to utilize the analyses of its advisors to make decisions.

This implementation corresponds to a model that is very close to the way modern command decision-making is done. The commander receives knowledge through aides or advisors that analyze and interpret with specialized expert knowledge. The information is then presented to the commander as icons on a map or some other abstract representation that saves the commander from processing large amounts of irrelevant information. The commander receives only information appropriate to his decision-making process.

The command agent structure not only coincides with real-world models but it also provides many software engineering advantages as well. In order to add new services or other capabilities to the command agent, the developer need only extend an existing advisor or add a new one. This extension would not significantly impact the other components due to the layer of abstraction defined between the advisors and the command agent. This advantage applies, also, to advisors; whereby, they can supply the same services to the command agent but utilize new sources of information by switching APIs.

The command agent design provides a flexible architecture that allows modular functionality, extensibility, and interoperability. These requirements are necessary to perform within the diverse and plentiful simulation federations being fielded today. The implementation of a command agent can take many forms and depends on many factors derived from the environment in which command agent is intended to operate.

# 4.0 Command Agent Construction

The command agent is composed of a set of sub-agent advisors that specialize in generating hypotheses about the situation environment and the decision making component that supervises the advisors and makes decisions based on the information they produce.

## *4.1 Command Agent*

The command agent's main role is to control the operation of its advisors and make decisions based on the results of these advisors. The command agent is made up of the following components:

- Internal Representations
- Inference Engine
- Command Behavior Model
- Agent Arbitration and Control Knowledge
- General Sub-Agent Interface

The command agent is an intelligent agent. "An intelligent agent is composed of the represented knowledge that drives its functions. This knowledge is described in the form of goal(s) and process(es) that lay out both the objectives for the agent and manner in which it will perform its tasks." [iv]

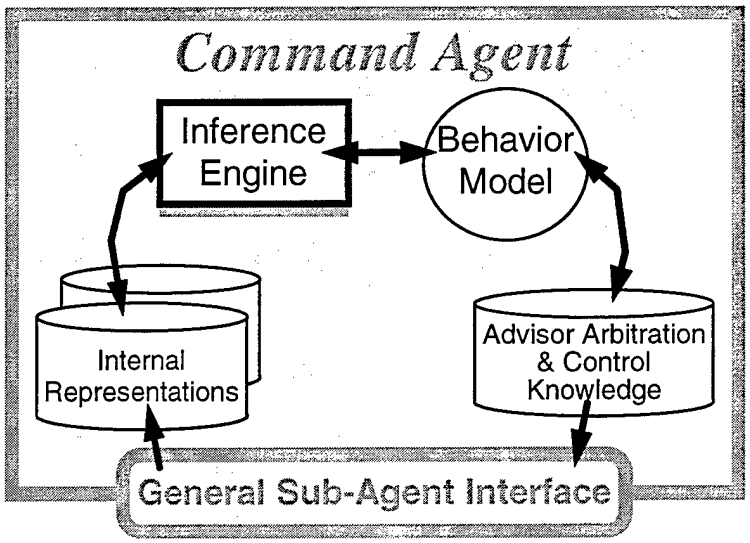


**Figure 3:** Command Agent Internals

## *4.2 Internal Representations and Inference Engine*

The Command Agent retains internal representations that represent its perceptions of the battlefield environment. These internal representations are updated and maintained by the flow of information from the Command Agent's Advisor sub-agents. Within [ii] the authors chose to use two representations to store battlefield knowledge: a semantic

network and a tessellated terrain map. The choice of internal structures can depend on the level of abstraction required in order to accomplish the command decision-making process efficiently and effectively.

Frame-based systems appear to be the best representation for the internal knowledge of the command agent. This technique follows an object-oriented strategy to depict the concepts and relationships that the command agent must handle. It provides a degree of flexibility and resolution for organizing battlefield information, and it requires a simple inference engine processing control rules to utilize the information. This corresponds to the human commander who does not cognitively handle the vast amounts of data that flows around a battlefield. Instead, the commander relies on trusted advisors to fuse, filter, and extrapolate important information from this flow of data. Decisions are made based on these abstract views.

The frame-based system is recommended because it provides a general structured approach to modeling battlefield knowledge that applies at different echelon levels and different levels of complexity; it can be as simple or complex as the developer wants to make it. It is, as pointed out above, cognitively similar to the way human commanders conduct their decision-making process. Inference engines derived from expert system shells are also readily available, and there is prevalent legacy experience in the engineering of these kinds of systems for decision-making. Other features such as decision explanation, case-based reasoning, and machine learning can be easily value-added to a frame-based system.

### *4.3 Behavior Model*

The Command Agent has a model of behavior which it uses to control its operation. These can be represented as constraints on its decision-making capability. These are usually defined by the higher-level controller of the command agent: a human operator or higher-echelon command agent. These constraints can be represented in a variety of ways, but the most easy to implement would be as meta-rules within the inference engine. These rules would be differentiated from the static doctrinal rules by their dynamism. The implementation would be based on the capabilities of the expert system shell, but could include either newly written (marked) rules or existing rules that depend on defined data within the command agent's representation.

### *4.4 Advisor Arbitration and Control Knowledge*

Arbitration and Control Knowledge defined for the command agent's advisors is of extreme importance to the design of the command agent. This component is where the command agent's knowledge of its advisors is stored. Utilizing the arbitration and control knowledge, the command agent is able to handle inconsistencies that may occur when two or more agent's produce conflicting hypotheses about the environment. In spite of the very careful compartmentalization of each agent's role in the simulation domain, agents can produce information that does not correlate either directly or by

inference from that of other agents, especially as the system's components grow in complexity.

Arbitration and Control Knowledge also aid the command agent in directing the operation of its advisor sub-agents. This stores knowledge for the command agent to direct when and how the advisors should be utilized for efficiency. The behavior model greatly affects the utilization of this meta-knowledge for directing the decision making process. This knowledge could be stored as a procedural construct (e.g. a finite-state machine) in which advisor management strategies would be based on the context or node of the command agent. This provides a straight-forward and simple approach to handling advisor control and arbitration.

### *4.5 General Sub-Agent Interface*

The General Sub-Agent Interface provides a layer of abstraction between the advisors and the command agent. This allows there to be a layer of abstraction between the command agent and its advisors. The agent can call on the services encapsulated by the advisors through the sub-agent interface. The advisors also provide information to the command entity through this interface. This layer of abstraction allows the different pieces of the command agent to be functionally distinct but their combined behavior to act homogeneously.

A layered approach allows for independent and concurrent development of the interior processes of the advisors and the command agent. The different advisors and command agent can be tested without the other components being complete by simulating (stubbing) their behavior. The interface through which these pieces communicate must be maintained as well, but it provides flexibility required for the evolution of the design and implementation.

This layered approach can be seen in the technical reference model and implementation of the Command Forces (CFOR) system [v]. Figure 4 shows the high-level layered model used as a basis for the CFOR design. There is a defined API, composed in an Interface Design Language (IDL), between the command entity application and decision processes, the information services and utilities, and the SAF baseline infrastructure (which in this implementation contains ModSAF and the CCSIL network protocols). This approach is extended in that advisor agents are the shepherds of a myriad number of interfaces, services and utilities that will support the command decision-making process.

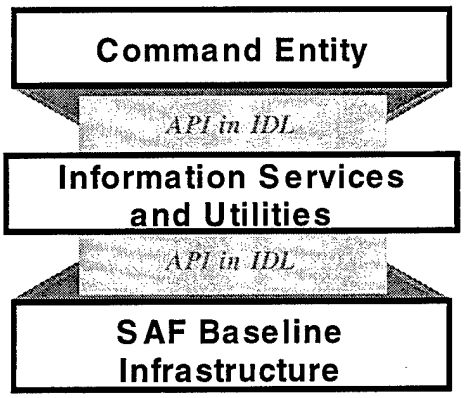


**Figure 4:** CFOR Reference Model

### *4.6 Advisors*

Advisors are sub-agents of the command agent that sift the extreme amount of battlefield data in a simulation and synthesize the data into information useful to the command agent's decision-making process. The agent's compartmentalized operation allows its internal process to be implemented using heterogeneous methodologies that are best-suited for that particular agent's specialized sub-domain of the battlefield environment. "In general, multiagent systems are computational systems in which several semi-autonomous agents interact or work together to perform some set of tasks or satisfy some set of goals." [vi]

As can be seen from Figure 5, the advisor sub-agent consists of the following pieces:

- Interfaces
- Command Agent Directives
- Domain Knowledge
- The Reasoning Mechanism
- Provided Services.

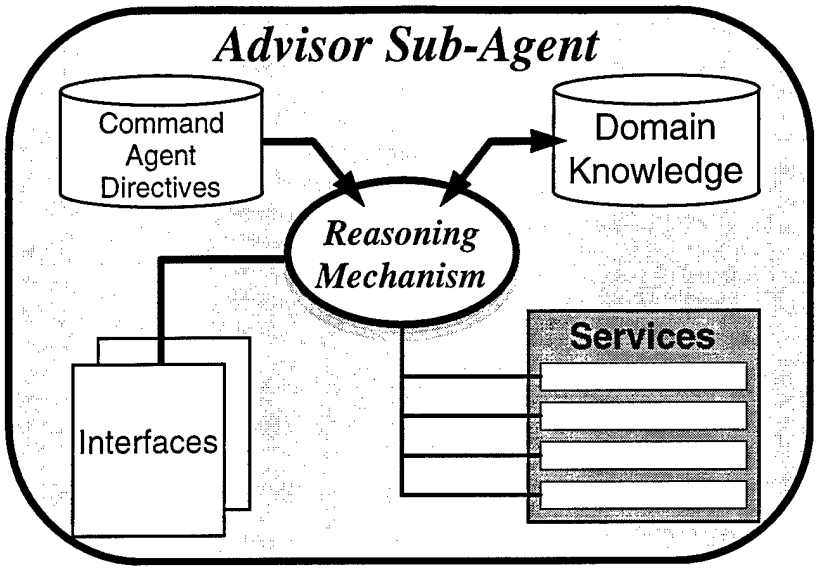


**Figure 5:** Advisor Sub-Agent Model

The advisor controls the interfaces to components within the simulation that provide data and services. This can be terrain algorithms, $C^3$ equipment, or vehicle primitives, etc. The advisor also receives and stores command agent directives that controls the advisor's operation. This can take the form of queries of data or services the command agent requests, constraints on the services of the advisor sub-agent, or tasks that the command agent has set for the advisor.

The domain knowledge element denotes the advisor's special capabilities regarding the context of its specialty. Domain knowledge can take many forms, some that encodes knowledge implicitly. Domain knowledge can be captured within rules, a training of a neural net, or within the cases of a case-base. "Agents could represent the same knowledge differently to optimize their particular use of it, or agents could obtain knowledge from different sources..." [vii]

The reasoning mechanism of the advisor describes the operations that are performed utilizing the encoded domain knowledge. The reasoning mechanism could be a forward- or backward-chaining inference engine. For neural nets in the form of the training and retrieval algorithms with which the net has been constructed. Case-based systems would employ key-matching and case adaptation as its reasoning mechanism.

Finally, the services that the advisor may be defined as part of its interface to the command agent. Data-driven techniques could be employed to allow the advisors to "register" its services with the command agent describing it capabilities and purpose. If the advisors conform to the protocol of the General Advisor Interface then this could facilitate an automated way of integrating altered or new advisors to the command agent's repertoire. This concept is a smaller-scale implementation of the ideas set down in the High-Level Architecture (HLA) specification.

# 5.0 Command Agent Implementation

Employing the layered concepts above allows the implementation of the command agent to take many forms while retaining the same decision-making process described in the SHOR framework. The flexibility of the design allows the implementation of command decision making agents to take any form which is appropriate to the problem domain. The developer is constructing an ontology of SHOR components. The following sections describe examples of command agents that may be constructed for different echelon levels and different strategies of conducting decision-making.

## *5.1 Judgmental METT-T Command Agent*

Command agents must be capable of rapid decision-making in constantly changing domains. Knowledge acquisition to apply to automated decision-making is of great difficulty because of the lack of canonical literature. To aid in this process the command agent can be modeled around doctrinal epistemology.

Situational awareness is paramount to the a battlefield commander. Army doctrine captured from Subject Matter Experts (SMEs) and training literature promotes the acronym METT-T to describe a process for an army commander to analyze the battlefield environment. METT-T stands for Mission, Enemy, Terrain, Troops, and Time, and it is utilized at all levels of the U.S. Army command structure. As can be seen in Figure 6, the construction of the Command Agent's advisors corresponds with these subjective areas. As described above, the advisors act as the command agent's proxies and filters to the information sources that exist inside and outside of the simulation environment.

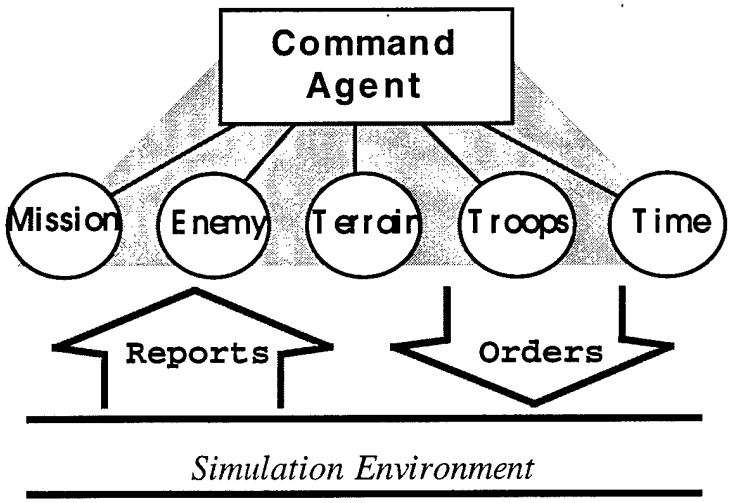


**Figure 6:** Command Agent with METT-T Advisors

For example, the Terrain Advisor controls the interface to the terrain database of the battlefield. The Terrain Advisor has rational methods of analyzing this data and deriving knowledge that is useful to the Company Team Command Agent. The other Advisors have similar interfaces to other information sources necessary to conduct their specialized purposes. The strength of this approach is that various AI techniques may be encapsulated within these Advisors providing a flexible, extensible framework in which

to construct hybrid systems. On top of this design is the Command Agent that acts as arbiter and director of the outputs of each advisor's analyses.

A detailed description of this model as applied to a Company Team Command Agent utilizing METT-T advisor sub-agents can be found in [ii]. Table 1 shows some possible AI techniques that may be applied to the different areas of METT-T.

**Table 1:** Techniques applied to METT-T

| | |
|---|---|
| **Mission** | *Objectives-based Planning* |
| **Enemy** | *Case-Based Reasoning (CBR)* |
| **Terrain** | *Neural Nets* |
| **Troops** | *Rule-based Systems* |
| **Time** | *Temporal Reasoning* |

### *5.2 BOS Command Agent*

The Battlefield Operating System (BOS) is a large connection of computers and intercommunication components that is used to disseminate and fuse large amounts of battlefield information. The BOS is divided into several different areas of command and control that is under a high-level echelon commander's purview. The example, shown in Figure 7, is meant to illustrate how a domain model of decision-making can be constructed for command agents that differs from the one presented above.

The same SHOR framework (see above) concept is employed, distributed through the different advisors. Data flows from the simulation environment and spawns one or a concurrent number of stimuli within the advisor environs. The advisors generate hypotheses that are incorporated into the BOS Command Agent's internal representations. The BOS Command Agent determines how to use these hypotheses. It can call on advisor services in order to make a decision or continue to collect information.

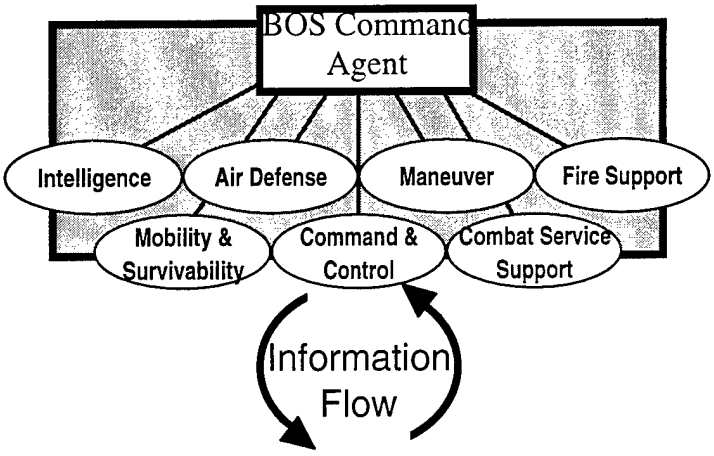


**Figure 7:** BOS Command Agent with Critical Combat Function Advisors

The advisors specialize in critical combat functions necessary to particular areas of a military operation. The significant aspect of this engineering structure is that each advisor controls the information flow from one channel implemented within the BOS. The interfaces to these different parts of the BOS are encapsulated within each specialized BOS advisor sub-agent. In this way, the construction of the command agent can be incremental and only concern specific aspects the designer wishes to implement. This design also shows how this approach provides an easy facility to integrating with existing military systems while employing hybrid AI techniques.

### *5.3 Commanding Officer Agent*

Commanders from Battalion and above have staff positions held by subordinate officers that are responsible for different areas of an Army organization. At the battalion and brigade levels these positions are called Staff (S) positions. At division to army level they are held by General (G) Officers.

The positions and their designations are described in Table 2.

**Table 2:** Staff/General Positions

| Designation | Position |
|---|---|
| S/G 1 | Personnel |
| S/G 2 | Intelligence |
| S/G 3 | Operations |
| S/G 4 | Logistics |
| S/G 5 | Civil Liaison |

The S/G1 (Personnel) handles discipline, awards, and assignments. This position assures that men are available and placed properly within the organization, whether it is a battalion or an army. The S/G2 is the intelligence officer responsible for security, counter-intelligence, and the Intelligence Preparation of the Battlefield (IPB). The IPB is basically the proper presentation of mission materials to indicate where enemy emplacements are believed to be and how the enemy is expected to behave. The S/G3 is the highly important operations officer. His responsibilities encompasses the planning of the movement and disposition of all forces in the organization. This includes training, detailed mission plans, and movement of all equipment, men, and their support. The S/G4 deals with the logistics of his unit. He facilitates the supply and maintenance of all things the organization will require.

The S5 is a position that is generated based on mission requirements. This position manages the politics, public relations, and interface to civil authorities sometimes required by Operations Other Than War (OOTW), such as humanitarian aid. At higher echelons this position becomes a necessary part of the organization otherwise it is filled as needed.

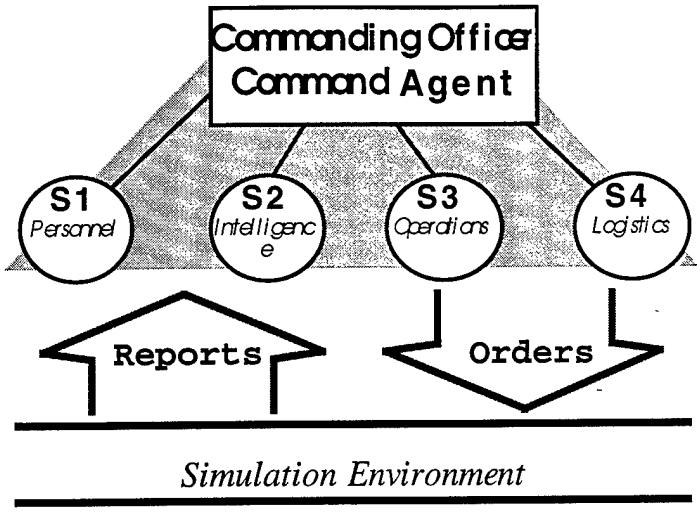


**Figure 8:** Commanding Officer Agent with Staff Advisors

Figure 8 shows the organization of the command agent architecture based on the commander and staff model. This shows the advisors being divided along the staff specializations: personnel, intelligence, operations, and logistics. The civil liaison duties would not commonly be required in a battle simulation. Figure 8 again shows the common theme of information flowing to and from the simulation environment which results in command behavior interaction with the simulation entities.

### *5.4 Command Agents at Multiple Echelons*

An interesting aspect of this problem is the creation of multiple levels of command agent interaction along echelon levels. The developer can create a hierarchy of command agents that act as both advisor and command agent. However, creating a stovepipe connection between command agents, as in Figure 9, can be detrimental to efficiency. This is true for real life military operations where informal, non-doctrinal communication relationships arise to assure the effective accomplishment of objectives, as in Figure 9.

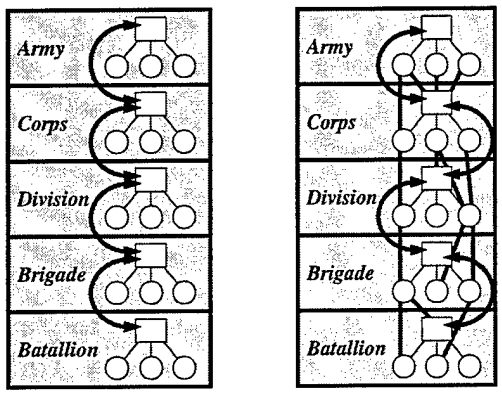


**Figure 9:** Multi-echelon Command Agents

## 6.0 Conclusion

The implementation of a command agent must be designed for evolution in the changing simulation environment in which it will be employed. This requires a special architecture that can provide flexibility and scalability. The architecture described above can be used to design a command decision-maker that accounts for the highly dynamic domains of a simulated battlefield environment, but also allows for the implementation of heterogeneous strategies for reasoning about the environment. This architecture also allows for different models of command decision-making to be implemented. The proper implementation of this architecture can develop a command agent that functionally acts as a surrogate for a human commander in a military simulation.

## 7.0 References

[i] **King**, James A. "Intelligent Agents: Bringing Good Things to Life". *AI Expert.* February 1995.

[ii] **Mall**, Howard; **Bimson**, Kent; **McCormack**, Jenifer; and **Ourston**, Dirk. "Command Entity Cognitive Behaviors for SAF and CGF". *The Fifth Conference on Computer Generated Forces and Behavioral Representation.* May 9-11, 1995.

[iii] **Riecken**, Doug. "M: An Architecture of Integrated Agents". *Communications of the CDM.* Vol. 37. No. 7. July 1994.

[iv] **King**, James A. "Intelligent Agents: Part 2". *AI Expert.* March 1995.

[v] Contacts: **Calder**, Robert B., and **Salisbury**, Marnie. "DARPA Command Forces (CFOR) Technology Program". Company Presentation Slides. MITRE Corporation. April 1996.

[vi] **Lesser**, Victor R. "Multiagent Systems: An Emerging Subdiscipline of AI". *ACM Computing Surveys.* Vol. 27, No. 3, September 1995.

[vii] **Adler**, Mark; **Durfee**, Edmund; **Huhns**, Michael; **Punch**, William; and **Simoudis**, Evangelos. "AAAI Workshop on Cooperation Among Heterogeneous Intelligent Agents". *AAAI Conference.* Summer 1992.