Modeling Command Entities

Michael D. Howard*
Hughes Research Laboratories
3011 Malibu Canyon Rd, Malibu, CA 90265
howard@isl.hrl.hac.com

Abstract

A Command Entity (CE) is a goal-oriented intelligent agent specifically designed to command and control other agents. The CE niche, most often a military battlefield, is a complex, dynamic, adversarial environment in which a CE must balance the needs for thorough planning with the need for quick reactions to changing conditions. We describe the requirements of this environment, and how it constrains the design of CEs. Examples from several CE designs are used. The paper concentrates on the areas of planning, knowledge and teamwork.

1 Introduction

A Command Entity (CE) is a type of intelligent agent whose purpose it is to model the way a commander accomplishes an assigned mission by directing the actions of subordinate agents. A CE is "situated" in a simulation of a natural real-world environment, so the problem space of a CE is complex. The problem space includes goals, enemy situation, subordinates and peers, terrain, weather, and time. When tasked with a set of mission goals, the CE must analyze the mission in terms of these factors and plan a course of action that can accomplish the goals. Then it must direct the execution of the mission, adapting to changing situations. Application areas for CEs include training, development of tactics and doctrine, and mission rehearsal.

Following Hayes-Roth [1995], we define *niche* as "a class of operating environments: the tasks an agent must perform, the resources it has for performing tasks, the contextual conditions that may influence its performance, and the evaluation criteria it must satisfy." We further define *CE niche* as the niche occupied by a command entity. Although military CEs are the focus of this paper, the CE niche is not unique to the military. An example of a non-military agent in a CE niche is the Phoenix project [Cohen *et al,* 1989], which has applied a real-time, adaptive planner to fighting forest fires. The Phoenix command agent makes a plan for how to use it's assets (a hierarchical organization of fire-fighting agents) in a real-world environment, in the face of an adversary (the fire) whose actions are difficult to predict and can be deadly. Another example is Hayes-Roth's [1995] "Adaptive Intelligent Systems," applied to Intensive Care Unit-patient monitoring.

•Portions of this work were supported by DARPA under contract DAAE07-92-C-R007, administered through Army TACOM, and contract N66001-95-C-6008, administered through Navy NRaD

The outward actions of a CE are requests and commands in a hierarchical command and control structure. Although a lower echelon CE can actively sense the environment by reconnoiter or moving to good vantage points, most situational information comes to the CE via periodic situation and intelligence reports. This conceptual distance between the command agent and its sensors and effectors has implications for reliability of incoming data, and for the amount of time available for decision-making. Since the CE is not omniscient, and is competing against an adversary that may be actively trying to cause the failure of mission goals, a CE must be able to react to problems encountered by its subordinates [sec. 4]. A more advanced CE can compensate for failures of its peers to achieve team goals in a cooperative mission [sec. 6].

In the sections that follow, we investigate the niche of Command Entities to suggest constraints it offers to the computer modeler. Section 2 measures the capabilities required for a CE along dimensions of functionality commonly associated with "agents". Section 3 describes HRL's own Canonical Commander Model, an example of a generic military CE. Section 4 looks at planning architectures suitable for the command domain. Section 5 explores the structure of military knowledge, and section 6 deals with requirements of teamwork.

Goals	Time
Enemy situation	Terrain & Weather
Subordinates &, peers	

Figure 1: Problem space of a Command Entity

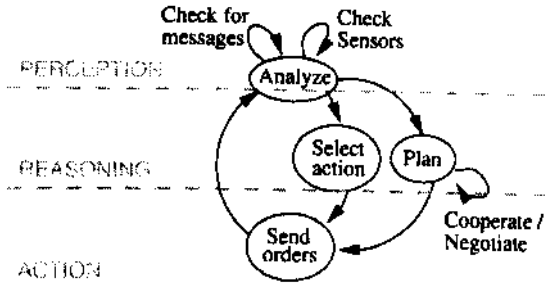

Figure 2: Command Entity's activities

2 Commanders are intelligent agents

Although very different types of tasks are given to commanders at different echelons (ranks) and branches of the service, the skills of analysis and decision-making are very similar. The process of commanding a unit is summarized in a number of Army field manuals as follows:

1. Receive the mission
2. Issue the warning order
3. Make a tentative plan
   a. Estimate the situation
      - Analyze mission
      - Develop Courses Of Action (COAs)
      - Analyze and compare COAs; War-game
   b. Expand selected COA into tentative plan
4. Initiate movement, and Reconnoiter
5. Complete the plan, and Issue the order

(3. Supervise, and Refine the plan

At all levels of command '.here are basic analysis, decision-making, command and control skills common to a wide variety of commanders. Figure 2 illustrates the activities of a CE. This matches Hayes-Roth's [1995] description of the functions continuously performed by intelligent agents ("typical AI agents"):

- Perception of dynamic conditions in the environment [i.e. read sensors, check for messages]
- Reasoning to interpret perceptions, solve problems, draw inferences, and determine actions [i.e. planning or action selection, if required]
- Action to affect conditions in the environment [i.e. issue orders and/or move]

Hayes-Roth characterizes a class of agent called "Adaptive Intelligent System (AIS)" that requires adaptation capabilities in addition to typical AI agent skills. AISs, like CEs, occupy a niche that presents "dynamic variability in ... required tasks, available resources, contextual conditions, and performance criteria". She concludes that AISs must be able to adapt their strategies. Two types of adaptation that are useful in a CE niche are:

- perception strategy - how often to read sensors, and when an active perception plan is needed (e.g. to reconnoiter or to move to a vantage point)
- control mode - whether to call a quick reaction drill or do a full replan (depends on time and situation)

Table 1 describes a CE in terms of such a list of properties commonly used to define agents from Franklin and Graesser [1996]. A CE, like a typical AI agent, is reactive, goal-oriented and communicative. Unlike "mobile" agents, it does not know about computer networks or network resources. The CE communicates for specific tactical reasons: to command, make requests, and report. Military communications are quite structured, and there is no need to engage in free-form discourse. The military is very interested in being able to task synthetic combatants by voice, as in the USMC's LeatherNet project [Clarkson and Yi, 1996]. But the allowable input is still very structured, and can be translated into something a CE can understand, as a pre-processor to the CE. One type of adaptation that is not required in a CE is learning of tactics [sec. 5]; standard knowledge acquisition methods are more effective. Finally, flexibility and personality are very desirable features in a command entity, to make it act more unpredictably. But it must be possible to turn such features off for certain types of validation tests where repeatability is desired.

The typical AI agent does not have the powerful knowledge-based planning power to act as a command entity. In turn, CEs are not mobile agents, and do not have special capabilities that would allow them to discover network resources. However, they are good at commanding other agents to do things, and could be used to command typical AI agents.

In conclusion, a CE is more than a "typical AI agent". It requires knowledge-based planning and reasoning, and can benefit from adaptive perception and reaction strategies. Distributed planning and negotiation capabilities may be required. Mobility (meaning the capability for the CE code to migrate between computers via a computer network) is not part of the job description.

AGENT PROPERTY	*CE*	*-NOTE-*
Reactive	yes	important
Autonomous	semi-	
Goal-oriented	yes	important
Continuously running process	no	
Communicative	yes	informative, not chatty
Adaptive	some	see below
Mobile (on network)	no	
Flexible	good	must be controllable
Personality	good	must be controllable

Table 1: Properties of Command Entities compared to those of "typical AI agents"

3 Canonical Commander Model (CCM)

The "Canonical Commander Model" is a synthetic commander developed by Hughes Research Labs for the DARPA Command Forces (CFOR) program [Hartzog and Salisbury, 1996]. The CCM can be configured with different knowledge bases of tactics and doctrine to simulate different types of commanders. In the last three years the CCM has been used to command simulations of a U.S. Army Company Team or a U.S. Marine Corps Rifle Platoon. The behavior of the CCM was validated by military experts for the Army and Marine Corps. When there were questions about the behavior, the knowledge base could be inspected and rules traced back to interviews with the SMEs.

The main principle guiding the design is that knowledge must be kept out of functional code, because CEs of all kinds use a common set of generic reasoning capabilities.

The CCM simulates all the basic skills enumerated in sec 2. It can analyze the situation and terrain, and plan a course of action to accomplish the mission goals. An industry-standard entity-level simulation known as ModSAF is used to simulate the mission. The commander monitors the progress of the mission by reading situation reports, and can modify its plan in case of problems such as minefields or enemy ambushes. The CCM communicates in a structured command language called Command and Control Simulation Interface Language [Salisbury, 1995], which is similar to military messages.

4 Considerations for planning

"*// is a bad plan that admits of no modification.*"
(Publius Syrus. 42 B.C.)

CEs act in a niche with dynamic variability along every dimension of the problem space. The CE must plan to achieve multiple goals with real-world constraints in an adversarial environment. It must assume that initial assumptions about situations might prove wrong, and in addition that there exist opponents whose goal it is to actively work to cause failure of the CE's goals.

Ideally, planning in an unpredictable, real-time domain should be done with a *reactive planning system* that combines perception, planning and action and guarantees a response in bounded time. But in practice, a computer planner is so much faster than a human that planning speed for this niche is not an issue. In fact, the lowest echelons have Standing Operating Procedures (SOP) [USMC, 1995] for immediate reactions to such common problems as enemy ambushes and minefields. This *tactical* procedure is a drill that the troops know how to execute without direction from higher commanders, to get the troops into cover and returning fire. This gives the CE time to plan a *strategic* response. For example, if a USMC rifle squad is ambushed, it knows to immediately take cover and return fire. This gives the rifle platoon leader time to plan for reorganizing the squad, calling in indirect fire from support elements, and deciding whether to continue to engage the enemy or withdraw [USMC, 1986].

The *task* is the most primitive executable operation planned by a CE. A *mission* is a set of tasks that accomplish a strategic goal. When a mission is assigned to the CE, the order may specify one or more tasks to be accomplished during execution of the mission. For example, a unit might be ordered to take a certain approach route, or to pause and coordinate with another unit, during an attack mission.

All three of the development efforts in the DARPA Command Forces (CFOR) program [Hartzog and Salisbury, 1996; Calder *et al*., 1996; Gratch, 1996; Howard, 1996], and its precursor, the Eagle program [Salisbury and Tallis, 1993], independently produced mission planners that decompose a mission assignment into a sequence of tasks (i.e. a course of action). This type of planner manipulates points in a search space of partially elaborated plans, and is sometimes known as a Hierarchical Task Network (HTN) planner. The following very brief description of planning can be supplemented by an excellent review of planning techniques in [Hendler *et* al., 1990].

An HTN planner starts with a plan composed of nonprimitive *(abstract)* actions and *primitive* actions. Primitive actions are those that the system knows how to perform. The HTN planner's main transformation step is *task reduction,* in which an abstract action is expanded into a partial plan composed of abstract and/or primitive actions. Task reduction steps alternate with conflict resolution steps [Erol *et* al., 1995]. When the plan cannot be further reduced and all conflicts are resolved, the planning process is terminated.

The main alternative to HTN planning methodology is partial order (PO) planning. A PO planning problem consists of an initial state, a goal state, and a set of actions (or *operators).* The PO planner's main transformation step is to select a precondition of a step in the plan and add new steps or new constraints that satisfy it. A complete plan is a sequence of executable actions that can turn the initial state into the goal state.

Kambhampati [1995], comparing HTN planning with partial order planning, shows that HTN planning can be more efficient because it allows the user to control the solution space. This limits the search for acceptable plans.

The Canonical Commander Model (CCM) [sec. 3] uses the HTN planning approach. The rules that guide the task decomposition process in the CCM enumerate an AND-OR tree, in which the OR nodes require a decision. AND nodes may or may not prescribe a certain ordering of the child nodes. Should the planner reach a partial ordering of primitive tasks that cannot be resolved into a total ordering, it would have to backtrack to create a consistent plan. But in practice, it has been possible to structure the planning scripts for the CCM planner so that it has not been necessary to backtrack. In other words, using an HTN planner, we have been

able to control the solution space to such a degree that it has always been possible for the CCM planner to find an acceptable solution without backtracking.

In general, a planner for a CE niche must be able to plan to achieve a set of goals. Approaches to solving such "conjunctive goal plans" are discussed in [Hendler *et* al., 1990]. In the CE niche, a set of goals can come from explicit tasking in orders in the form of multiple missions to be accomplished sequentially or explicit tasks that must be accomplished in pursuit of a mission. The order probably provides a sequence for the goals, or at least a priority may be implied by the wording of the order. Other goals and constraints are implied by the explicitly assigned ones; for example, a task of occupying a suppressive fire position implies a movement task to get there. These will also imply a sequential ordering that makes them easy to plan. The most difficult planning challenge is to accomplish goals and constraints related to standing operating procedure and the obligations of teamwork, because it is more likely that the planner cannot use the *linearity assumption.* That is, the planner cannot plan for one goal at a time, and assume that the solution for one goal will not conflict with the solution of any other. *Nonlinear* planners are able to deal with goal conflicts explicitly; Chapman [l990] pulled together various approaches into a simple, provably correct domain-independent planner called "TWEAK". The CE niche is a constrained domain, and if goals and constraints from all sources can be expressed in the same format, they can be seamlessly integrated into the multi-goal planning and monitoring process.

5 It's how you play the game: Military Knowledge

DARPA's Joint Simulation System (JSIMS) mission statement includes training and use of computer simulations to help define doctrine, tactics, and operational requirements. This implies that it is important that a CE not only achieve its assigned goals, but also conduct the mission in a doctrinally correct manner. If a human company commander is using simulations of several subordinate platoon commanders to rehearse a mission and evaluate different courses of action, and the simulated subordinates do something unexpected, the human commander will want an explanation - not only for the human's understanding and training effect, but to maintain confidence in the simulation system. Accurate acquisition of relevant tactics and doctrine is essential for both realistic behavior and meaningful explanation.

Military doctrine and tactics have been developed over centuries and codified in a simple form that can be taught to soldiers of widely varying abilities. At the lowest echelons, reaction drills like those in the USMC Battle Drill Guide [USMC, 1986] have been formulated for squad and platoon leaders. Soldiers are trained in these basic skills until they become second nature. The battle drills are grouped into scenarios called "Situational Training Exercises". But while the information is very helpful to the developer in understanding the military problem space, it is difficult, and undesirable, to develop rules directly from the manuals. While the manuals do have a disciplined structure, the organization is not always logical from a programmer's perspective. The programmer must study the manuals and become expert enough to reformulate the knowledge as needed for the application.

The Command Forces (CFOR) program provides developers with military experts to identify the appropriate doctrine and tactics, and to validate the simulated behaviors. There are no established objective performance tests for this type of behavioral representation. In the CFOR program, behaviors are evaluated subjectively on a PASS/FAIL system, by military experts watching CE performance on different test scenarios.

AI planners work to refine a plan down to primitive actions or tasks, so it is natural for a developer to want an enumeration of these primitive tasks, along with allowable parameters. But military tasks are greatly dependent on context (1). For example, it is natural for a designer of CEs to want to use "MOVE" as a primitive operation, parameterized by formation type and route. But the military experts we have dealt with have felt very uncomfortable when asked, for example, to relate a movement behind friendly lines to a movement across a "danger area". They prefer to talk about situations, or sets of tasks in context. This difference in conceptualization makes knowledge acquisition in this domain especially challenging.

The holy grail of CE designers is a system that can learn tactics and doctrine by watching an expert work. Some recent attempts are in [Krozel *et* al., 1994; Hille *et* al., 1996; Rajput *et* a/., 1996]. But the complexity of the CE niche makes it difficult for a program to discern which contextual features contribute to making a decision, in all but the simplest scenarios. For example, although built on a well-developed learning architecture (Soar) [J.Laird *et* a/., 1987], learning of tactical/strategic knowledge has not been attempted in the Rotor-Winged Air (RWA) CE [Gratch, 1996] built for the CFOR program. A good interviewing technique will more efficiently elicit the important decisions and the pertinent factors.

In the CFOR program, developers used different strategies to incorporate the knowledge into their simulations. The Army developers [Calder *et* al., 1996] embed knowledge derived from military experts into instances of object-oriented "constraint sets" (CS), where each CS represents an abstract or primitive task. The RWA CE developers build the knowledge into the plan refinement operators used to refine its plan.

In developing the CCM agent (sec 3) for the Marine Corps, we used an expert system approach. The CCM uses the Modular Knowledge Acquisition Tool (M-KAT) [Goldman, 1996] to acquire knowledge and turn it into a rulebase known as "fuzzy tables". The M-KAT approach to knowledge acquisition features an interviewing style that helps the expert to express the knowledge in

	CE type	*Knowledge Format*
HRL (CCM)	Army Co Tm USMC Pit	M-KAT fuzzy tables
SAIC	Army Co/Bn Army FIST	constraint sets
ISI	RWA Co	plan refinement operators j

Table 2: Knowledge formats used in Command Entities

a form that can be directly encoded. This can improve the efficiency of the knowledge acquisition process. The interviewing approach helps an expert to formulate the knowledge in a form that can be put directly into the knowledge base.

6 Teamwork and Theories of Joint Action

Military organizations are specialists in teamwork. Breakdowns in teamwork can and do happen, and a breakdown in teamwork in a military mission can be fatal if team members are not able to compensate. Most CE's are not capable of reasoning about teamwork, which require an explicit model of the key tasks each team member is going to accomplish in support of the team goals. With such a model it would be possible to detect failure in one of the key tasks, and reason about an alternate course of action.

If orders are well-written, they will spell out the commander's intent for the mission (that is, the desirable outcome) and list any constraints on each team members' movement, weapons fire, and time. As long as it doesn't violate the constraints, each team member can move and shoot at will, confident that it won't hurt another team member. A good order does not "micromanage" the mission, but there will be explicit information about when and where any team interactions will take place. If something goes wrong with one team member, the others have the information they need to reason about how to compensate.

An example of a breakdown in teamwork that was experienced in the CFOR program was when a Marine platoon leader commanded an attack on a prepared enemy position. The assault was to be supported by three fire support assets that were to begin firing upon command by the platoon leader. When the suppressive fire began, the leader's plan was to order the assault. However, if one of the fire support assets was unable to get to its support position, whether due to difficult terrain or attrition, the leader was unable to reason about a way to compensate; the rules required that all three fire support assets be in place and firing before the assault could be ordered. It is possible to get around this problem by adding more rules to the commander's knowledge database, but it is difficult to write the rules so as to generalize to other situations. Furthermore, if the leader were to be isolated because of a breakdown in communication, more specific rules would have to be added so that the subordinates could adapt.

It is desirable to have a general method of reasoning about teamwork, with rules for adjusting the plan when teamwork breaks down. There are few implementations of theories of joint action. Jennings' [1995] implementation of the "joint intentions" framework in an industrial multi-agent setting was the model for Tambe's [1996] implementation in the air combat domain. If a team is to be able to act in a coherent manner even during unplanned events, Jennings asserts that team members need to have sufficient knowledge of the problem-solving process. They must understand the mission goals and the roles of all team members. The common understanding of the domain, guidance given in the order, and rules for handling specific situations would all be used to define the latitude team members have in adjusting to difficulties in the mission.

7 Conclusions

A Command Entity (CE) acts in a very challenging, dynamically changing domain. The "CE niche" is more complex than the niche occupied by "typical AI agents". To act as a CE, a typical agent would probably have to have a more powerful planner, capable of working with a knowledge base. It is possible, and desirable, to use a Hierarchical Task Network (HTN) planner in the design of a CE. HTN planners offer the user some control over the solution space, which can make them more efficient than Partial Order planners. The CE niche is mostly managerial, and does not require a planner with guaranteed results in bounded time. We characterized military knowledge, which is not easy to build into a simulation of this type. Finally, the requirements for teamwork were briefly discussed. The Canonical Commander Model (CCM) was used throughout the paper to illustrate some of the concepts.

8 Acknowledgments

The interest and support of CDR Peggy Feldman (DARPA Synthetic Forces PM) and Susie Hartzog and Jeff Clarkson (NRaD Project Managers) is greatly appreciated. HRL's command entity is the work of a talented team which includes Craig Lee and Seth Goldman, under the guidance of our program manager, David Tseng.

References

[Calder *et* al., 1996] R. Calder, R. Carreiro, J. Panagos, G. Vrablic, B. Wise, F. Chamberlain, and D. Glasson. Architecture of a command forces command entity. In *Proc Sixth Conf on CGF & BR.* IST-TR-96-18, July 1996.

[Chapman, 1990] D. Chapman. Planning for conjunctive goals. In *Readings In Planning.* Morgan-Kaufmann, 1990.

[Clarkson and Yi, 1996] J. Clarkson and J. Yi. LeatherNet: A synthetic forces tactical training system for

the usmc commander. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996.

[Cohen *et al,* 1989] P. Cohen, M. Greenberg, D. Hart, and A. Howe. Trial by fire: Understanding the design requirements for agents in complex environments. *AI Magazine,* Fall 1989.

[Erol *et al,* 1995] K. Erol, J. Hendler, D. Nau, and R. Tsuneto. A critical look at critics in HTN planning. In *Proc IJCAI-95*. Morgan Kauffman Publishers, Inc, August 1995.

[Franklin and Graesser, 1996] S. Franklin and A. Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Proc Third Intl Workshop on Agent Theories, Architectures, and Languages,* 1996. www.msci.memphis.edu/franklin/AgentProg.html.

[Goldman, 1996] S.R.Goldman. Knowledge acquisition and delivery: Constructing intelligent software command entities. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996.

[Gratch, 1996] J. Gratch. Task-decomposition planning for command decision making. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996.

[Hartzog and Salisbury, 1996] S. Hartzog and M. Salisbury. Command forces (CFOR) status report. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996. http://ms.ie.org/cfor/cgl9607/cgf9607.html.

[Hayes-Roth, 1995] B. Hayes-Roth. An architecture for adaptive intelligent systems. *Artificial Intelligence,* 72:329 365, 1995.

[Hendler *et al,* 1990] J. Hendler, A. Tate, and M. Drummond. AI planning: Systems and techniques. *AI Magazine,* Summer 1990.

[Hille *et al,* 1996] D. Hille, G. Tecuci, and M. Hieb. Training a ModSAF command agent through demonstration. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996.

[Howard, 1996] M.D. Howard. *Software Design Document for the Canonical Commander Model version 1.0.* Hughes Research Laboratories, February 1996.

[Jennings, 1995] N. R. Jennings. Controlling cooperative problem-solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence, 75,* 1995.

[J.Laird *et al,* 1987] J.Laird, A. Newell, and P. Rosenbloom. Soar: An architecture for general intelligence. *Artificial Intelligence,* 1987.

[Kambhampati, 1995] S. Kambhampati. A comparative analysis of partial order planning and task reduction planning. *SIGART Bulletin,* 6(1), January 1995.

[Krozel *et al,* 1994] J. Krozel, D. Keirsey, D. Payton, and D. Tseng. Case-based computer generated forces. In *Proc Fourth Conf on CGF & BR*. IST-TR-94-12, May 1994.

[Rajput *et al,* 1996] S. Rajput, C. Karr, J. Cisneros, and R. Parsons. Learning the selection of reactive behaviors. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996.

[Salisbury and Tallis, 1993] M. Salisbury and H. Tallis. Automated planning and replanning for battlefield simulation. In *Proc Third Conf on CGF & BR*. IST-TR-93-07, March 1993.

[Salisbury, 1995] M. Salisbury. Command and Control Simulation Interface Language (CCSIL): Status update. In *Proc Twelfth Workshop on Stds for Interoperability of Defense Simulations,* March 1995. http://rns.ie.org/cfor/diswg9503/diswg9503.html.

[Tambe, 1996] M. Tambe. Flexible teamwork for intelligent simulated pilots. In *Proc Sixth Conf on CGF & BR*. IST-TR-96-18, July 1996.

[USMC, 1986] USMC. *Battle Drill Guide and Individual Training Packages.* Marine Corps Institute, Arlington, Va, 1986. MCO 1510.35A.

[USMC, 1995] USMC. *Basic Officer Course.* The Basic School, Marine Corps Combat Development Command, Arlington, Va, December 1995.

PLANNING AND SCHEDULING

Planning 3: Planning under Uncertainty