A Multi-Agent Decision Framework for DDD-III Environment*

Candra Meirina, Georgiy M. Levchuk, and Krishna R. Pattipati[1]

University of Connecticut, Dept. of Electrical and Computer Engineering
Storrs, CT 06269

Abstract

In this paper, we present techniques for modeling the decision-making processes of a team of synthetic agents operating in task selection and resource allocation settings within the third generation distributed dynamic decision-making (DDD-III) paradigm. The DDD-III simulator provides a controllable, multi-player, multi-platform organizational environment. The paper provides two major contributions. First, motivated by the need for a network of intelligent agents within C2 experimental settings, a brief overview of modeling techniques for the design of a network of collaborating agents is provided. Second, techniques for modeling the decision-making processes of synthetic agents in task selection and resource allocation settings are presented. In the proposed framework, the decision-making processes of a network of intelligent agents are addressed via limited look-ahead, auction-based scheduling and resource allocation algorithms from the phase I of the three-phase organizational design process. Preliminary results of operationalizing the DDD-based multi-agent-network paradigm are presented in two different mission scenarios. A coordination-free scenario illustrates the basic structures of the intelligent agent design, in terms of stimulus-hypothesis-option-response (SHOR) model-based three-stage decision-making process. The second example, derived from the A2C2 Experiment 8, highlights the potential of utilizing the agent framework in C2 experiments.

I. INTRODUCTION

OVER the years, C2 researchers have come to appreciate the roles of war games to provide a simulated, highly dynamic, creative and challenging environment, in which military organizations can explore their strategic thinking and tactical planning, in order to meet the challenges of the increasingly agile environments in which they operate. A war game provides valuable insights into the possible strategies that the organization can employ, as well as develop and gauge the impacts of alternative strategic responses of the adversary. The safe, but highly-charged, competitive environment fostered by the war game often results in some surprisingly creative and innovative thinking.

The scope of problems that the military organizations need to address, however, call for dynamic simulation of large-scale organizations. This is because the interesting insights of how human teams coordinate cannot be revealed within teams of mere 6 or 10 individuals, as is being done in current A2C2 experiments. Often times, however, the cost and availability of subject matter experts prevent larger team experiments from being operationalized. Synthetic agents, which can reasonably mimic the decision-making processes of human players in such war games, have the potential to facilitate large-scale experiments.

This paper serves two purposes. First, motivated by the need for a network of intelligent agents within C2 experimental settings, presented in section II, we provide a brief overview of modeling techniques for the design of a network of collaborating agents. This is the subject of section III. The second purpose is to present techniques for modeling the decision-making processes of synthetic agents in task selection and resource allocation settings within the DDD-III [24] framework. We propose to employ limited look-ahead, auction-based scheduling and resource allocation algorithms from the phase I of the three-phase

*This work was supported by the Office of Naval Research under contract #N00014-00-1-0101.
[1]To whom correspondence should be addressed: krishna@engr.uconn.edu

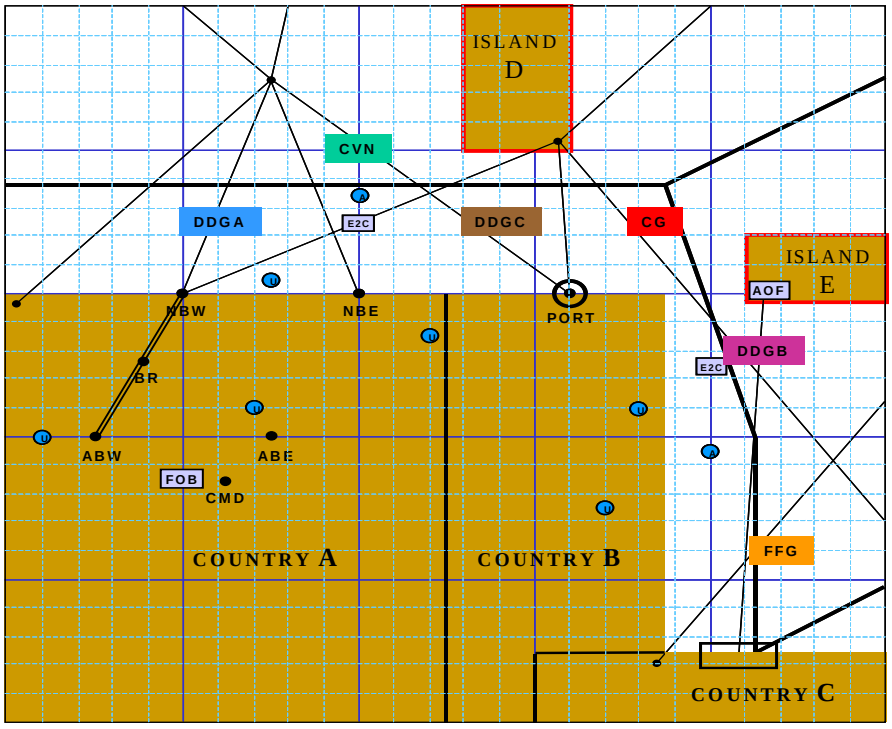

Fig. 1. A Screen-Shot from the A2C2 Experiment 8

organizational design process ([28], [29]) to model dynamic resource allocation and scheduling strategies of an agent. This is the subject of section IV. In section V, we present preliminary results of operationalizing the DDD-based multi-agent-network paradigm. As an illustration, we utilize a simple air defense scenario as well as parts of scenarios used in the A2C2 Experiment 8 [23], wherein two organizational structures, namely the divisional (D) and functional (F) organizations, are to perform two missions d and f. We employ various measures introduced in ([27] and [30]) to measure team performance.

II. DDD-III Environment

A. Overview

THE third generation distributed dynamic decision-making (DDD-III) simulator [24] provides a flexible, controllable research paradigm, which allows researchers to examine the interactions between task (or mission) structure, and the way in which the organization tasked to execute the mission is structured. In essence, DDD-III simulator provides a controllable, multi-player, multi-platform, real-time organizational environment to support laboratory-based empirical experiments in a UNIX/LINUX environment. The DDD-III allows for constraints and manipulations of organizational structures, such as authority, information, communication, resource ownership, task assignment, etc., as well as mission and environmental structures, such as air, sea, and ground environment, a variety of task classes, and controllable platforms with sub-platforms, sensors and weapons (resources).

A typical experiment within the DDD-III framework involves a group of subject matter experts, who act as a team of decision-makers (DMs) – in a hierarchical or networked organization, in an environment that simulates a military operation. Following [28], we define a DM as an entity with information-processing, decision-making, and operational capabilities (via its resource capability, viz., platform ownership) that can control the necessary resources to execute a set of assigned mission tasks, provided that such task execution does not violate the DM's concomitant resource capability thresholds. During the course of an experiment, a DM, through use of platform capabilities, is able to detect, measure, identify, pursue, and eventually process tasks. The DDD scenario requires the players to follow a set of predefined mission

requirements, while simultaneously defending one or more 'penetration zones', and possibly their own assets, against potential land, sea, and air adversaries. Through the scenarios, the designers craft an experiment to uncover key issues in the organization's decision-making and coordination processes. A typical screen-shot of DDD-III experiment is shown in Fig. 1.

B. Structures and Attributes of Mission Tasks and the Organization

During the real-time playout of an experimental run, tasks appear, move (maneuver), and disappear according to a scripted scenario. The experiment designers have the ability to define various dimensions of task structure in order to closely align the mission scenario with a real-life military operation. The characteristics of each task – its class, attributes, precedence constraints, and resource requirements – can be tailored to specify a threat (such as a hostile fighter, minefield, etc.) and to create intra-team conflicts for assets needed in the execution of those tasks.

A mission task T_i (or simply i) is characterized by the following basic features: (i) class or type, i.e., air, surface, or ground; (ii) attributes, $A_i = [a_{i1}, ...]$, that define various characteristics of the task quantitatively; one attribute of interest is the first attribute that represents the task value, $val(i) = a_{i1}$; (iii) estimated processing time t_i; (iv) geographical constraint vector (e.g., the 'location' (x_i, y_i) in a phase space that specifies the concomitant 'distance' d_{ij} to be traveled between tasks i and j); (v) resource requirement vector $[R_{i1}, R_{i2}, ..., R_{iL}]$, were R_{ij} is the number of units of resource l required for successful processing of task i (l =1,. . . ,L, where L is the number of resource types); this feature defines the resources required to successfully process (attack) the task; and (vi) task precedences or prerequisites.

In DDD-III framework, a platform represents a physical asset of an organization that provides resource capabilities used to process tasks. Examples of platforms (or assets) are ships, helicopters, ground units, bases, etc. Each platform P_m (or simply m) $(m = 1, \ldots, K)$ has several features that uniquely define this platform: (i) sub-platforms, i.e., additional assets that reside on-board a parent platform that only become active after being 'launched' from the parent; (ii) ownership, i.e., only owners of the platforms are able to move, carry a pursuit or attack with them, or launch sub-platforms; owners of parent platforms are not necessarily owners of the sub-platforms; (iii) sensors, i.e., specify effectiveness ranges for task detection, measurement, and identification or classification; (iv) geographical location, i.e., (x_m, y_m); (v) maximum velocity v_m, defines how fast a platform can travel; and (vi) resource capability vector $[r_{m1}, r_{m2}, ..., r_{mL}]$, where r_{ml} specifies the number of units of resource type l available on platform m. A platform can be used to attack any task, but the range of the platform's weapons depends on the task type or class. Further details of the DDD-III structures can be found in [24].

C. Real World Challenges

The scope and size of an experiment to simulate and reveal the dynamics of military organizations operating in real world mission scenarios can be enormous. For example, a small-to-moderate size military mission is typically conducted by several hundred individuals. In theory, DDD-III framework allows for several hundred players to conduct an experiment together; in practice, however, the cost and availability of subject matter experts will be prohibitive.

The ability to re-run the same experiment is another important feature for military strategists. Results from a single run of an experiment are less dependable than those of several hundred experimental runs. Since the DDD-III framework accounts for uncertainty within a scenario and the communication environment, the players' strategies are random as well. Consequently, Monte-Carlo runs of the same scenario provide confidence estimates on the experimental results and allow experimenters to extract valuable insights from the experiment.

In addition, human-players may introduce biases into the experimental results. As an example, we draw experience from conducting the A2C2 Experiment 8. The design and conduct of this experiment provided some significant training challenges. The actual experiment was a between-subjects-design on organization, but a within-subjects-design on scenario. Thus, subjects would stay in one organization (either D or F),

but would play both f and d scenarios in a counterbalanced manner [23]. It was important to build a training schedule and training scenarios that would not bias either organization to scenario f or to scenario d. Similarly, it was critical for the experiment to have equal coordinating skill from both organizations. Failure to account for these biases will critically impact the experimental results.

These arguments point to the need for utilization of intelligent agents within the DDD-III experiments to operationalize large-scale (reduced bias) experiments via all agent teams or hybrid human-agent teams.

III. Intelligent Agents and Multi-Agent Networks

A. What Constitutes an Intelligent Agent?

THE general consensus on the term *agent* is the notion of *autonomy*. Maes [32] defines an agent as a computational system with the objective of fulfilling a set of goals in a complex, dynamic environment. More precisely, Wooldridge [47] characterizes an agent as a computer system that is situated in some *environment*, and is capable of *autonomous* actions in this environment in order to meet its design objectives.

The notion of autonomy means that agents are able to act without intervention of humans or other systems, i.e., they have control over their own internal state, and over their behavior. In most domains of reasonable complexity, an agent will not have complete control over its environment. It will have at best partial control in what it can influence. From the point of view of the agent, this means that the same action performed twice in apparently identical circumstances may appear to have entirely different effects, and, in particular, it may fail to have the desired effects. Thus, agents in all, but the most trivial, environments must be prepared for the possibility of failure. This leads to the notion of flexibility, viz., intelligence. For all practical purposes, an agent is *intelligent*, if it is capable of *flexible* autonomous actions in order to meet its design objectives. Following [47], flexibility encompasses three ideas, which are (i) reactivity, i.e., ability to perceive its environment and respond in a timely fashion to changes that occur in it in order to satisfy its design objectives; (ii) pro-activeness, i.e., a display of goal-directed behavior in terms of taking initiatives to fulfill its objectives; and (iii) social ability, i.e., ability to interact with other agents (and possibly humans) within the organization.

The key problem facing an agent is that of deciding which of its actions it should perform to manipulate its environment in order to best satisfy its design objectives. In this vein, agent models are really software architectures for decision making systems that are embedded in an environment. The complexity of the decision-making process is greatly affected by a number of environmental properties. Russell and Norvig suggest the following classification of environment properties [38], which in turn dictate a suitable decision-making process for the agent:

- *Accessible vs. inaccessible*: An accessible environment is one in which the agent can obtain complete, accurate, up-to-date information about the environment's state. The more accessible the environment, the simpler it is to build agents to operate in it.
- *Deterministic vs. non-deterministic*: A deterministic environment is one in which any action has a single guaranteed effect – there is no uncertainty about the state that will result from performing an action. Non-deterministic environments present greater problems for the agent designer.
- *Episodic vs. non-episodic*: In an episodic environment, the performance of an agent is dependent on a number of discrete episodes, with no link between the performance of an agent in different scenarios. Episodic environments are simpler from the agent developer's perspective, because the agent can decide which action to perform based only on the current episode – it need not reason about the interactions between this and future episodes.
- *Static vs. dynamic*: A static environment is one that can be assumed to remain unchanged except by the agent's actions. A dynamic environment is one that has other processes operating on it, and, hence, changes in ways that are beyond the agent's control.
- *Discrete vs. continuous*: An environment is discrete if there are fixed, finite number of actions and observations in it.

Simply said, from this perspective, intelligent agents are merely computer systems that are capable of autonomous action in order to meet their design objectives. An agent will typically sense its environment and will have available repertoire of actions that can be executed to modify the environment, which may appear to respond non-deterministically to the execution of these actions.

B. Intelligent Agent Models

In this subsection, we attempt to formalize agent models. First, we characterize the circumstance of the agent's environment as a set of environment states. At any given instant of time, the environment is assumed to be in one of these states. The capability of an agent is represented by a set of actions. Then, conceptually, an agent can be viewed as a function which maps sequences of environment states to actions.

The basic idea is that an agent decides what action to perform on the basis of its history, i.e., its experiences to date. These experiences are represented as a sequence of environment states, namely, those that the agent has thus far encountered. The (non-deterministic) behavior of the environment can be modeled as a function, which takes the current state of the environment and an action (performed by the agent), that maps them to a set of environment states.

From a computational view point, an agent model involves data structures, the operations that may be performed on these data structures, and the control flow between them. From this abstract view of an agent, we distinguish among three classes of agents ([14], [37], [39]):

- *Purely Reactive Agents*: Reactive agents decide what to do without reference to their history. They base their decision making process entirely on the present, with no reference at all to the past. We will call such agents purely reactive, since they simply respond directly to their environment.
- *Perception*: A more sophisticated design. Here, an agent's decision function is separated into two subsystems: *perception* and *action*. The idea is that the perception captures the agent's ability to observe its environment, whereas the action represents the agent's decision making process.
- *Agents with State*: The main idea of this model is that the agents have some internal data structure, which is typically used to record information about the environment state and history. An agent's decision making process is then based, at least in part, on these internal states.

From the point of view of how an agent's decision making process may be implemented, we consider four classes of agents:

- *logic-based* agents: decision making process is realized through logical deduction;
- *reactive* agents: decision making process is implemented in some form of direct mapping from situation to action;
- *belief-desire-intention* agents: decision making process depends on the manipulation of data structures representing the beliefs, desires, and intentions of the agent; and finally,
- *layered models*: decision making process is realized via various software layers, each of which is more-or-less explicitly reasoning about the environment at different levels of abstraction.

Within the DDD-III framework and from a C2 perspective, BDI models are particularly attractive. These models have their roots in the philosophical tradition of understanding practical reasoning, i.e., the process of deciding, moment by moment, which action to perform in the furtherance of desired goals. Practical reasoning involves two important processes: deciding what goals we want to achieve, and how we are going to achieve these goals. The BDI model essentially balances pro-active (goal-directed) and reactive (event-driven) behaviors.

The process of practical reasoning in a BDI agent is summarized in seven main components:

- a set of current beliefs, representing information the agent has about its current environment;
- a belief revision function, which takes a perceptual input and the agent's current beliefs, and on the basis of these, determines a new set of beliefs;
- an option generation function, (options), which determines the options available to the agent (its desires), on the basis of its current beliefs about its environment and its current intentions;

- a set of current options, representing possible courses of action available to the agent;
- a filter function (filter), which represents the agent's deliberation process, and which determines the agent's intentions on the basis of its current beliefs, desires, and intentions;
- a set of current intentions, representing the agent's current focus, i.e., those states of affair that it is committed to trying to bring about; and
- an action selection function (execute), which determines an action to perform on the basis of current intentions.

Belief-desire-intention models originated in the work of the Rational Agency project at Stanford Research Institute in the mid 1980s. The origins of the model lie in the theory of human practical reasoning developed by the philosopher Michael Bratman [2], which focuses particularly on the role of intentions in practical reasoning. The conceptual framework of the BDI model is described in [3]. One of the interesting aspects of the BDI model is that it has been used in one of the most successful agent models to date, e.g. ([12], [13], and [16]).

In the early 1980s, a similar framework has been developed within the C2 community. The model is the stimulus–hypothesis–option–response (SHOR) framework of Wohl ([43], [44], [45]). Similar to BDI, the SHOR model involves several stages:

- *Stimulus* (S) – A received stimulus initiates the decision-making process.
- *Hypothesis* (H) – Based upon the stimulus and other information available at that time, various hypotheses are generated concerning the actual situation or the actual state-of-the-world faced by the DMs.
- *Option* (O) – Depending on the possible situations (the hypotheses), the DM generates a set of options or possible actions.
- *Response* (R) – The effects or outcomes of the options are evaluated in view of the uncertain nature of the situation, and an appropriate action or response is selected. The response then interacts with the external system generating additional stimuli, which may lead to additional iterations of the process.

Other assessment-response type of structures for various applications have been developed by other investigators within the C2 community, e.g., ([5], [17], [25], [31], [34]).

C. Why Multi-Agent Networks?

A single agent is limited by its knowledge, its perspective, and its computational resources. As the domain becomes larger and more complex, open and distributed, as in DDD-III, a set of cooperating agents is needed to address the distributed decision making processes more effectively. A multi-agent network (MAN) in this situation is attractive because it offers robustness, efficiency, and inter-operation of existing legacy systems. Although centralized solutions are generally more efficient, when the problem being solved is itself distributed, distributed decision making becomes a more natural and efficient approach to consider. In addition, there are many instances where a centralized approach is intractable, as, for example, when the systems and data belong to independent organizations, who want to keep their information private and secure for competitive reasons.

Multi-agent network (MAN) can be defined as a loosely coupled network of problem solvers, who work together to solve problems that are beyond the individual capabilities or knowledge of each problem solver [20]. MAN may be comprised of homogeneous or heterogeneous agents. An agent in the system is considered a locus of problem solving activity; it operates synchronously or asynchronously with respect to other agents, and it has certain level of autonomy [26]. In this context, autonomy refers to an agent's ability to make its own decisions about what to do, when to do it, what information to communicate with others, and how to interpret the information received.

Jennings et. al. [20] note that the main characteristics of MAN include: (1) each agent has incomplete information or capabilities for solving the problem, i.e. each agent has a limited view point; (2) there is no global system control; (3) data is decentralized; and (4) computation is asynchronous. Similarly, Huhns and Stephens [19] note that information involved in MAN is distributed, and typically resides

in information systems that are large and complex in the following ways: (1) geographical sense, (2) component wise, and (3) scope and conceptual sense, both in the number of concepts and in the amount of data about each concept.

The challenges in MAN lie in how to formulate, describe, decompose, and allocate problems and synthesize results among a group of intelligent agents; how to enable agents to communicate and interact; how to ensure coherent coordination and cooperation; how to facilitate situation awareness, namely, agents' reasons, perception, and knowledge about other agents; how to resolve conflicts of interest, etc. That is, the locus of MAN design is to facilitate effective operation and productive interaction among the agents involved, which are assumed to have incomplete and uncertain knowledge about the domain and incomplete, and possibly uncertain, observations. The design involves a computational infrastructure for such interactions to take place. The infrastructure will include protocols for agents to communicate and interact with. These protocols enable agents to exchange and understand messages, and facilitate cooperation in solving the global problem.

D. Inter-Agent Interaction

The rationale for interconnecting agents is to enable them to cooperate in solving problems, to share expertise, to work in parallel on common problems, to be developed and implemented modularly, to be fault-tolerant through some form of redundancy, to represent multiple viewpoints and the knowledge of multiple experts, and to be reusable [19]. Agents in MAN may be characterized by whether they are benevolent or self-interested. Benevolent agents work together toward achieving common goals, whereas self-interested agents have distinct goals, but may still interact to advance their own goals. In our work, it is assumed that the agents are designed to work together or that the payoffs to self-interested agents are only accrued through collective efforts. Thus, problem solving, i.e., decision making, in MAN is a process of constructing a sequence of actions, which involves an agent's own goals, capabilities, its view of environmental constraints, and a fair degree of coherence about additional constraints from other agents' activities, commitments to other agents, and unpredictable nature of their interaction due to their limited view of others.

Coherence is the notion of how well a system behaves as a unit. Agents own their private goals and share some common goals. The agents' private goals are not necessarily known to other agents. One of the challenges for a multi-agent network is how it can maintain *global coherence without explicit global control*. In this case, agents must be able to determine goals they share with other agents, determine common tasks, avoid unnecessary conflicts, and pool knowledge and evidence. In this light, agents communicate in an effort to enable them to coordinate their actions and behaviors, resulting in systems that are more coherent.

The power of MAN lies in the existence of sophisticated patterns of interactions among the problem solvers. The common types of interactions include *coordination* (organizing problem solving activity so that harmful interactions are avoided and beneficial interactions are exploited), *cooperation* (coordination among non-antagonistic agents who work together towards a common objective), and negotiation (co-ordination among competing or simply self-interested agents, who are coming to an agreement that is acceptable to all the parties involved).

The need for interaction in MAN occurs because agents solve sub-problems that are interdependent through overlaps in the sub-problems, the situation in which the sub-problems are parts of a larger problem whose solution requires that certain constraints exist among the solutions of the sub-problems, or through distributed locations of information, expertise, processing, and communication resources. In cooperation, wherein coordinating agents are non-antagonistic and see themselves as working together towards a common objective, agents' behaviors are guided by cooperative strategies meant to improve their collective performance.

To produce coherent behaviors, the early models emphasized planning to resolve the interdependence among sub-problems by utilizing a synchronizer agent that recognizes and resolves sub-goal interactions.

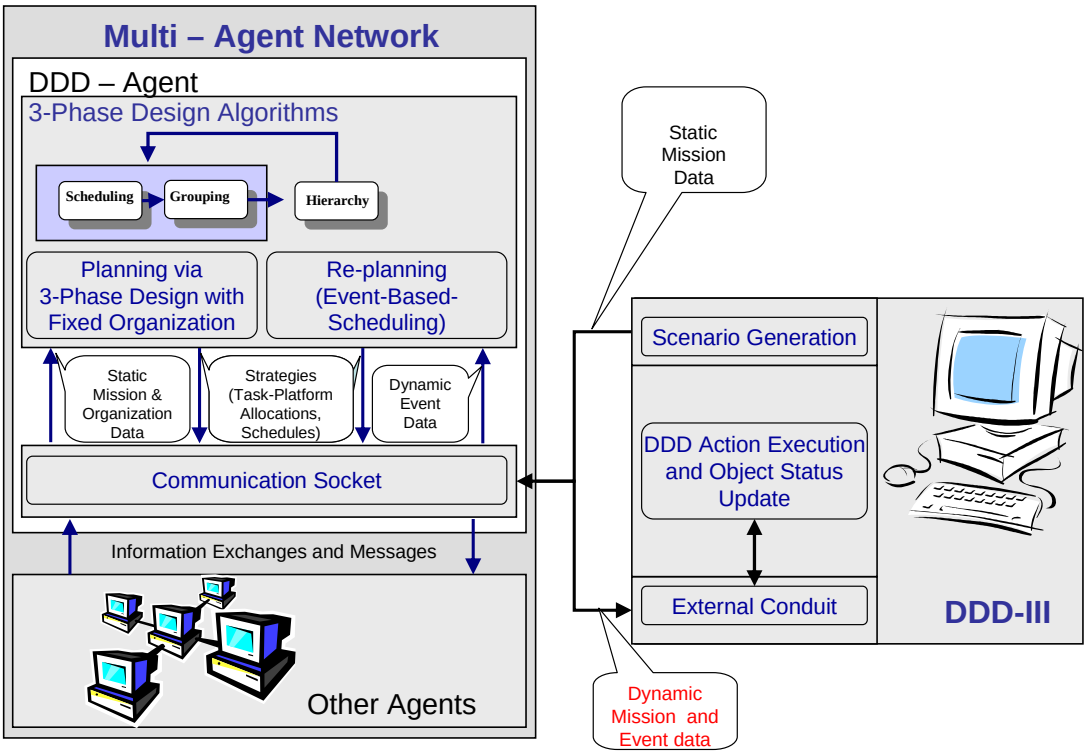

Fig. 2. The Overall Architecture of the DDD-Based Multi-Agent Network

A more flexible approach in later models, where agents are allowed to dynamically coordinate without particular assumptions on the distribution of sub-problems, expertise or other resources. Coordination is operationalized in the form of communicating plans and goals at some levels of abstraction. Communication among agents facilitates mutual expectation among the cooperating agents and, in turn, improves coherence. Work on these approaches can be found, e.g. ([10], [11]).

Current approaches of cooperation favor explicit models of teamwork, wherein teams monitor their performance and adjust their responses based on the current situation. Explicit models of teamwork are particularly important in dynamic environments, wherein various unexpected events (such as agents' failures or arrival of new tasks) may occur. These approaches are natural extensions of the BDI single agent model. Examples of work done in this category include ([21], [22]); related work can be found in ([6], [15]).

IV. DDD-III Based Multi-Agent Network

SIMILAR to OMAR agent design ([7] and [8]), we will illustrate our modeling paradigm of a collaborative synthetic agent via two examples, an air defense scenario and parts of A2C2 Experiment 8 [23], operationalized in the distributed dynamic decision-making (DDD-III) team-in-the-loop real-time simulator. The overall system architecture within this framework is as shown in Fig. 2. Details of the DDD-III components, described in Fig. 2, can be found in ([8] and [24]) and in section II of this paper.

A. DDD Agent Design

We have adopted the concept of an intelligent synthetic agent as a computational system that is situated in an environment, and is capable of flexible autonomous actions in this environment in order to meet its design objectives ([32] and [47]). Within the DDD-III paradigm, a set of cooperative decision-makers,

who share a common *objective* of successfully completing a set of assigned tasks under resource and time constraints, work together to achieve this common goal.

The synthetic agent acts as a decision-maker (DM). The DDD agent is characterized by its ability to recognize its own capability and that of others in its team. It successfully schedules its assigned tasks to not only its own resources (platforms) but also coordinates with other agents so as to minimize the overall mission completion time.

Taking an abstract view of the agent model as a map of its data structures, potential manipulations of the data structures, and control flow between them, we classify our DDD agent as a *perceptive agent with states* ([37], [39]). That is, the DDD agent's decision function is separated into its *perception*,i.e., its ability to observe its environment, and *action*, i.e., its decision making process, which is based in part on its internal states. The DDD agent's states represent its internal data structure, which is used to record information about the environment state and history.

From a concrete perspective of implementing the agent's decision making process, it is natural within our problem context to adopt the *belief*- (viz., perception) *desire*- (viz., objectives) *intention* (viz., probable actions), i.e., *BDI* paradigm ([2], [12]). In this framework, an agent's decision-making process depends on the manipulation of data structures representing the beliefs, desires, and intentions of the agent, as discussed in section III. Naturally, we are employing the SHOR paradigm of Wohl [43]: the *stimulus*- corresponds to task detection, *hypothesis*- represents task identification, *option*- models deliberation of possible actions based on the agent's hypotheses about the adversary and *response*- corresponds to pursue, attack, coordinate, etc. In this vein, our modeling paradigm is also similar to the design of multi-stage interacting DMs [17]. That is, the state-based DDD agent employs the following processing stages (see Fig. 3):

1) *Environment sensing* (*ES*): defines the agent's perception of existing tasks and of other agents. In the DDD-III context, this step consists of task identification, which is operationalized through the agent's platform sensing capabilities and direct communication with other agents in the organization.
2) *Information processing* (*IP*): identifies the agent's active mapping between the types (friendly, hostile, or neutral) and requirements (precedence constraints and resource requirements) of active tasks in the system, its own capability, and perceived capabilities of other agents in the organization.
3) *Action Selection* (*AS*): specifies the agent's strategy to achieve its objectives. In the DDD-III context, the agents choose to wait, obtain additional information, or pursue and attack a task alone or with other agents.

Finally, in accordance with the bounded rationality concept [33], the agents should have *limited cognitive resources* with which they can accomplish their objectives either in information processing or in task processing. The optimal decision strategy must distribute the information and activities among agents so that the decision-making and operational load of each agent remains below the corresponding thresholds.

B. DDD-III based Multi Agent Network Design

The DDD mission scenarios can be modeled to allow each DM to have access to only a portion of the information available to the team. The total information set maybe incomplete and inaccurate due to lack of updating, missed detection, or communication channel errors. Therefore, the *team information processing* is characterized by a significant degree of uncertainty with partial overlap among DMs. In this context, the critical issue to be addressed in the multi-agent information processing is 'who should communicate what, with whom, and when'. Furthermore, the DDD mission scenarios afford a reasonable degree of task processing overlap among DMs. That is, with regard to resource constraints, two or more agents must share responsibility for a given task. The salient issues in *team task processing* is 'what should be done, who should do what, when and with which resources'.

Recall that the critical shortcoming of a single agent is limited knowledge, perspective, and computational resources. The nature of the DDD-III framework suggests that a *multi-agent-network* (MAN),

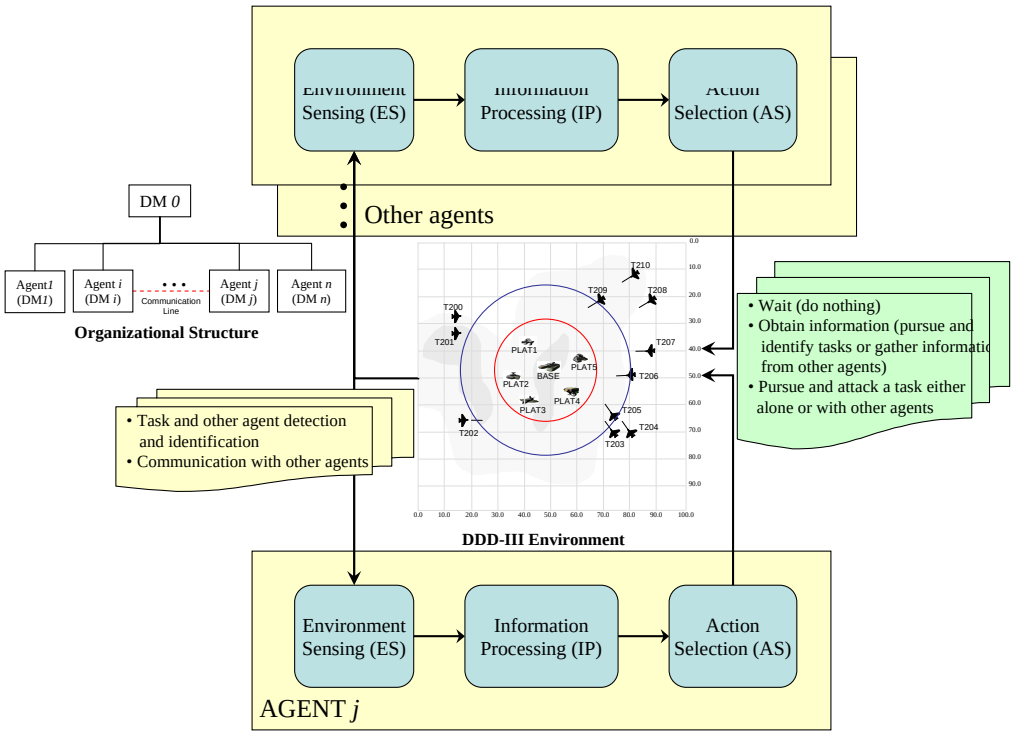

Fig. 3. The DDD Agents within the DDD-based Multi-Agent Network

wherein a loosely coupled network of cooperating agents (problem-solvers) work together to solve problems that are beyond the individual capabilities or knowledge of each agent [20], is more suitable. Within the DDD-III framework, the MAN is comprised of *heterogeneous, communicating, and cooperative* agents.

Recall that the locus of MAN design is to facilitate effective operation and productive interaction among the agents involved, which are assumed to have incomplete and uncertain knowledge about the domain and incomplete and, possibly uncertain, observations. Within the DDD-III framework, the challenges in MAN lie in how to facilitate situation awareness, namely an agent's perception and knowledge about other agents; how to enable agents to communicate and interact; and how to ensure coherent coordination and cooperation. Within the three stage decision-making process, these issues were resolved as follows.

1) *Environment Sensing (ES)*:
In the DDD-III paradigm, each DM is allowed a set of platforms, viz., resource capability, and at the same time is assigned a set of tasks (individually or as a team). As human players are able to see the current tasks and available platforms, it is assumed that all DMs in the organization share the same common knowledge about these objects. The knowledge limitation lies in the fact that, based on its resource capability, i.e. identification range, identities of existing tasks may not be available to an agent unless the tasks are within its identification range. Such knowledge has to be communicated among agents within a team (or possibly only among potential coordinating partners). In the current implementation, all new task identification regarding its type and requirements is communicated to all agents (all players broadcast their new findings). The communication between agents is conducted in accordance with the DDD-III communication protocol ([24], [8]). This approach, in essence, partially alleviates the issue of 'who should communicate what, with whom, and when'. Communication channel (viz., socket) errors can still occur; that is, the total information set may still be incomplete and inaccurate due to communication errors.

2) *Information processing (IP)*:
Recall that due to task processing overlaps among DMs, MAN needs to address the issues of 'what should be done, who should do what, when and with which resources'. That is, in MAN, IP stage, which emphasizes 'what should be done', requires interaction among team members.

In the DDD-III context, the nature of interaction is one of cooperation, wherein the coordinating agents are cooperatively working together towards a common objective. Consequently, behavior of agents is guided by cooperative strategies meant to improve their collective performance. As the task information is communicated to all team members, the question of 'what should be done', viz., which task should the team consider at the present time, can safely be addressed as a team, i.e., centralized approach. In this vein, we need to formalize the overall team scheduling problem.

Conceptually, the *centralized* scheduling problem is formalized as follows. A set of currently existing tasks *ready* to be processed with 'known' type, resource requirements, locations, precedence relations, and specified processing times must be assigned and executed concurrently by a set of platforms with given resource capabilities, ranges of operation, and velocities. Tasks are allocated to groups of platforms in such a way that for each such platform package to task assignment, the vector of task's resource requirement is component-wise less than or equal to the aggregated resource capability of the platform group. The task processing can only begin only when the processing of all of its predecessors is completed and all platforms from the group assigned to this task have arrived at the appropriate location. Furthermore, task processing by all assigned platforms must be executed in a limited time window. It is assumed that a resource can only process one task at a time. Available platforms are to be re-routed among the tasks, so as to minimize the overall mission completion time. A detailed mathematical formulation of the centralized scheduling problem can be found in [28].

Aside from the fact that the scheduling problem is NP-hard, one can argue that no human team will behave optimally in a similar situation. Therefore, the DDD-based MAN paradigm focuses on heuristic scheduling algorithms with good performance. Details on various heuristic scheduling algorithms can be found in [28].

The DDD-based MAN employs the multi-dimensional dynamic list scheduling method (MDLS), wherein it finds the platform-task allocation and mission schedule by sequentially assigning tasks to platforms until the task set is exhausted. MDLS heuristic has two main steps: (1) select the task to be processed; and (2) select the group of platforms to be assigned to it for processing. This will fit nicely within our three stage decision-making paradigm.

In the information processing stage, we address the first step of selecting a task to be processed from the ready tasks (a task becomes ready when all its predecessors have been completed). The selection is determined by the current assignment information and precedence structure [28]. The priority value to select task i, $P(i)$, is operationalized by the following criteria: (1) opportunity window to process task i, $win(i)$; (2) value of task i, $val(i)$; and possibly other criteria as introduced in [28]. Formally, the priority value is calculated as follows:

$$P(i) = \frac{val(i)}{1 + win(i)} \tag{1}$$

Accordingly, we have the task selection procedure shown in Fig. 4.

3) *Action Selection (AS)*:

The AS stage of the decision-making process addresses the second step of the MDLS heuristic. A group of platforms is selected for processing the selected task. A task is assigned to groups of platforms in such a way that the vector of task's resource requirements is component-wise less than or equal to the aggregated resource capability vector of the group of platforms assigned to it:

$$\sum_{m \in GROUP} r_{ml} \geq R_{il}, \forall l = 1, ..., L \tag{2}$$

The salient issue is how to distribute the processing of a task under resource requirement constraints among available platforms to achieve minimal execution time for the overall mission. Similar to [28], we obtain a trade-off between minimizing a task's completion time and minimizing the

Find the set:

$$READY1 = \left\{ i \in READY | \sum_{m \in FREE} r_{ml} \geq R_{il}, \forall l = 1, ..., L \right\}$$

DO UNTIL $READY1 = \emptyset$

Task selection:

$$i^* = \arg \max_{i \in READY1} P(i)$$

$$READY1 \leftarrow READY1 \backslash \{i^*\}$$

Platform group selection for $i = i^*$:

Find the set:

$$FREE1 = \left\{ m \in FREE | \sum_{l=1}^{L} \min(r_{ml}, R_{il}) \neq 0 \right\}$$

$$v_{im} = w_{im}, \forall m \in FREE1$$

$$GROUP = \emptyset$$

DO UNTIL $\sum_{m \in GROUP} r_{ml} \geq R_{il}, \forall l = 1, ..., L$

$$m^* = \arg \max_{m \in FREE1} v_{im}$$

$$FREE1 \leftarrow FREE1 \backslash \{m^*\}$$

$$GROUP \leftarrow GROUP \cup \{m^*\}$$

END DO

END DO

Fig. 4. Centralized Task and Platform Group Selection Procedure

allocated resources, which may be needed by other tasks. We operationalize the first consideration by minimizing the travel time of platform m to task i, $t_{im} = \frac{d_{im}}{v_m}$, which depends on the distance d_{im} and the platform's maximum velocity v_m. The second consideration is operationalized by maximizing the ratio of accuracy when assigning platform m to task i compared to assigning it to any other task $j \neq i$. Accuracy of task i, when it is assigned to platform i, depends on task-resource requirements, platform-resource capability, and task value. It is computed as follows:

$$Acc(i) = \frac{100 \times val(i)}{nR_i} \sum_{l=1}^{L} \left(\frac{\min(r_{ml}, R_{il})}{R_{il}} \right)^2 \tag{3}$$

Note that nR_i represents the number of resource types a task requires. Accordingly, we have the platform group selection procedure shown in Fig. 4.

Within the DDD-III paradigm, a task is executed concurrently (within a small time window) by all assigned platforms. Since the travel times of the assigned platforms differ, the closer or faster platforms should wait for others before attacking the task or, alternately, all assigned platforms should synchronize their departure times so as to arrive at the task location concurrently, viz., begin task execution together. An assignment is considered whenever a task (or a group of tasks) is completed. At that time, all the platforms processing the completed task become free.

The platform to task allocation procedure can be conducted either as a centralized or as a distributed process. The centralized allocation procedure is as shown in Fig. 4, wherein all players are being told (by a central processor) of which platforms are needed to process the current tasks and when they should start the task processing sequence. Our approach to the distributed platform to task allocation employs the auction algorithm, wherein a set of ready tasks bid for available platforms in the organization.The platform owners, viz., the decision makers, adjust the 'prices', $\{p_m\}$, so as to achieve their common objective of minimizing the overall mission completion time. Unlike the centralized approach, the tasks and the DMs (viz., the platforms) find their matches through bidding and price adjustments.

Find the set:

$$READY1 = \left\{ i \in READY | \sum_{m \in FREE} r_{ml} \geq R_{il}, \forall l = 1, ..., L \right\}$$

$READY2 = \emptyset$

DO UNTIL $READY1 = \emptyset$

Task selection:

$i^* = \arg \max_{i \in READY1} P(i)$

$READY1 \leftarrow READY1 \backslash \{i^*\}$

$READY2 \leftarrow READY2 \cup \{i^*\}$

$FREE2 = \emptyset$

Platform group selection for $i = i^*$:

Find the set:

$$A(i) = FREE1 = \left\{ m \in FREE | \sum_{l=1}^{L} \min(r_{ml}, R_{il}) \neq 0 \right\}$$

$v_{im} = w_{im} - p_m, \forall m \in FREE1$

$GROUP(i) = \emptyset$

DO UNTIL $\sum_{m \in FREE1} r_{ml} \geq R_{il}, \forall l = 1, ..., L$

$m^* = \arg \max_{m \in FREE1} v_{im^*}, \pi_i = \max_{m \in FREE1} v_{im}$

IF $FREE1 = \{m^*\} \cup \emptyset$

$\phi_i = -\infty$

ELSE

$\phi_i = \max_{m \in FREE1 \backslash \{m^*\}} v_{im}$

END IF

$\delta = \min \{\pi_i - \lambda, \pi_i - \phi_i + \epsilon, \lambda - p_{m^*}\}$

$b_{im^*} = p_{m^*} + \delta$

$FREE1 \leftarrow FREE1 \backslash \{m^*\}$

$GROUP(i) \leftarrow GROUP(i) \cup \{m^*\}$

$FREE2 \leftarrow FREE2 \cup \{m^*\}$

END DO

END DO

Bidding process for $\forall\ i \in READY2$:

$B(m) = \emptyset, \forall\ m \in FREE2$

DO UNTIL $READY2 = \emptyset$

Select i (breadth first)

Bid at b_{im} on $\forall\ m \in GROUP(i) \Rightarrow B(m) \leftarrow B(m) \cup \{i\}$

$READY2 \leftarrow READY2 \backslash \{i\}$

END DO

Assignment process $\forall m \in FREE2$:

DO UNTIL $FREE2 = \emptyset$

Select m (breadth first)

IF $B(m) \neq \emptyset$

$p_m = \max_{i \in B(m)} b_{im}$, announce new price p_m to all $B(m)$

$i^* = \arg \max_{i \in B(m)} b_{im}$, assign task i^* to platform $m : x_{i^*m} = 1$

Deassign previous assignment i' to $m : x_{i'm} = 0$

END IF

$FREE2 \leftarrow FREE2 \backslash \{m\}$

END DO

Fig. 5. Decentralized Task and Platform Group Selection Procedure

In adopting the auction method, we view the platform-to-task allocation as an asymmetric-multi-assignment problem, wherein 'ready' tasks are to be assigned to some (possibly not all) of 'free' platforms. The problem can be formulated as follows:

$$\begin{aligned} \max \quad & \sum_{(i,m)\in E} w_{im}x_{im} \\ s.t \quad & \sum_{m\in A(i)} x_{im} \geq 1, \forall i = 1, ..., N \\ & \sum_{i\in B(m)} x_{im} \leq 1, \forall m = 1, ..., K \\ & x_{im} \geq 0, \forall (i,m) \in E \end{aligned} \tag{4}$$

where E denotes a set of tasks i and platforms m that can be matched to form pairs $\{(i,m)\}$. For each task i, we denote by $A(i)$, the set of platforms that can be matched with i such that $A(i) = \{m|\,(i,m) \in E\}$ and for each platform, we define by $B(m)$, the set of tasks that can be matched with m such that $B(m) = \{i|\,(i,m) \in E\}$. The cost of assigning platform m to task i, w_{im}, which depends on the platform travel time and the ratio of accuracy (see Eq. (3)) when platform m is assigned to task i, compared to assigning it to some other task $j \neq i$, is computed as follows:

$$w_{im} = \frac{v_m}{d_{im}} \frac{(1+val(i))\,\frac{100}{nR_i}\sum_{l=1}^{L}\left(\frac{\min(r_{ml},R_{il})}{R_{il}}\right)^2}{\sum_{j\in READY1\setminus\{i\}}(1+val(j))\,\frac{100}{nR_j}\sum_{l=1}^{L}\left(\frac{\min\left(r_{ml},R_{jl}\right)}{R_{jl}}\right)^2} \tag{5}$$

The assignment problem as described in Eq. (4) can be easily converted to an equivalent assignment problem [4] as follows:

$$\begin{aligned} \max \quad & \sum_{(i,m)\in E} w_{im}x_{im} \\ s.t \quad & \sum_{m\in A(i)} x_{im} - x_{si} = 1, \forall i = 1, ..., N \\ & \sum_{i\in B(m)} x_{im} + x_{ms} = 1, \forall m = 1, ..., K \\ & \sum_{i=1}^{N} x_{si} - \sum_{i=1}^{K} x_{ms} = K - N \\ & x_{im} \geq 0, \forall (i,m) \in E \\ & x_{si} \geq 0, \forall i = 1, ..., N \\ & x_{ms} \geq 0, \forall m = 1, ..., K \end{aligned} \tag{6}$$

The dual of the asymmetric assignment problem is given by:

$$\begin{aligned} \min \quad & \sum_{i=1}^{N}\pi_i + \sum_{m=1}^{K} p_m - (K-N)\,\lambda \\ s.t \quad & \pi_i + p_m \geq w_{im}, \forall (i,m) \in E \\ & \lambda \leq p_m, \forall m = 1, ..., K \\ & \lambda \geq \pi_i, \forall i = 1, ..., N \end{aligned} \tag{7}$$

An auction algorithm is developed from the dual optimization problem and consists of two phases: (i) bidding phase, wherein each ready task i computes the current value of platforms m (i.e., potential profit if i is assigned to a set of platforms $\{m\}$) and bids for the platforms of its

Platform Group Prunning:

$m^* = \arg \min_{m \in A(i)} v_{im^*}$
$A1 = A(i)$
WHILE $A1 \neq \emptyset$
  $m^* = \arg \min_{m \in A1} v_{im^*}$
  $A1 \leftarrow A1 \backslash \{m^*\}$
  IF $\sum_{m \in A(i) \backslash \{m^*\}} r_{ml} \geq R_{il}, \forall l = 1, ..., L$
    $A(i) \leftarrow A(i) \backslash \{m^*\}$
  END IF
END WHILE

Fig. 6. Platform Group Prunning Procedure

choice; (ii) assignment phase, wherein each platform (temporarily) offers itself to the highest bidder. Accordingly, we have the platform-to-task-allocation procedure as shown in Fig. 5.

Note that, we have not accounted for the resource requirement constraint shown in Eq. (2) in the multi-assignment problem, in Eq. (4). Neglecting this, quite possibly, leads the auction algorithm to a situation, wherein all platforms are allocated to only one preferred task while ignoring other tasks. We address this problem by introducing platform group pruning after the assignment process is completed. The platform group pruning guarantees that, no more than the needed platforms, are allocated to a task. The platform pruning procedure is shown in Fig. 6.

V. Preliminary Results

A Methodology for quantifying similarity, in terms of a fit (match) between a mission scenario and an organizational strategy, is presented in [27]. It is suggested that structural fit can be characterized in terms of workload balance, communication requirements, and DM-to-DM dependency. It is argued that balancing the necessary coordination to maintain good performance and minimizing excessive DM-DM interaction are keys to successful organizational strategy.

In the following, preliminary results from the DDD-III-based intelligent agent network are presented. To illustrate the basic structures of the agent design, a simple coordination-free scenario is presented. This example emphasizes the inner-agent model, viz., the three-stage decision-making process, and its performance within the DDD-III environment. The second example is derived from A2C2 Experiment 8, wherein only the first eight to nine minutes of the experiment is considered. The latter example illustrates the agent coordination capability in performing complex tasks, which require inter-platform coordination, and points to the potential of utilizing the agent framework within C2 experiments.

A. Defending a Friendly Air-base Scenario

A team of seven homogeneous agents is assigned to defend a friendly air-base from an adversary. Each team member, who shares similar traits, is able to individually process a single incoming task without external workload coordination. That is, the resource capability of each agent, shown in TABLE I, matches or exceeds the resource requirements of each mission task. The tasks arrive randomly and impose no precedence constraints. See Fig. 7. The scenario is scripted within the DDD-III framework, and is executed fully by a network of intelligent agents, wherein each agent utilizes the three-stage decision making process described in section IV.

Accrued task gain is used to measure the task processing efficiency of the team as a function of time. It is argued that an efficient team achieves high accuracy and timeliness. Details of accrued task gain calculation can be found in [27]. Another efficiency metric considered is the workload distribution among DMs within a team. Balanced workload over all team members is desired, since higher workload and

DM_0				
8 F15	2 A12	2 A7	2 A9	2 GNS
4 F14	2 A12	2 A7	2 A9	2 GNS
1 BAS	1 F14	1 F15		

DM_i				
8 F15	2 A12	2 A7	2 A9	2 GNS
4 F14	2 A12	2 A7	2 A9	2 GNS

TABLE I

AGENT RESOURCE CAPABILITY FOR THE AIR DEFENSE SCENARIO

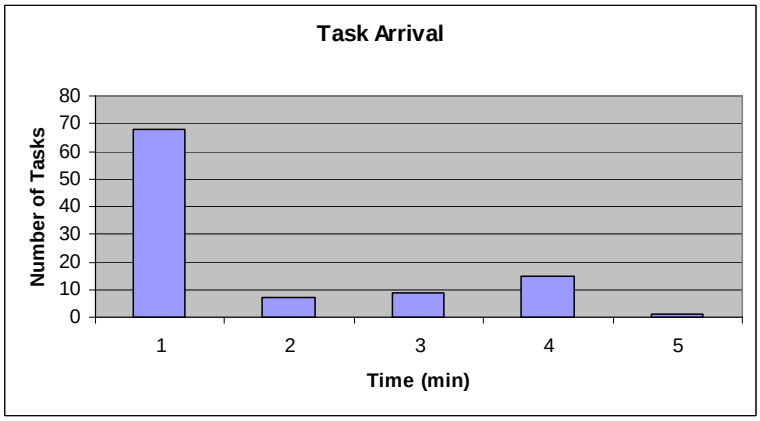

Fig. 7. Arrival Time of 100 Tasks in an Air-base Defense Scenario

increased workload imbalance lead to degraded organizational performance. Due to the nature of the mission scenario, only internal DM workload is relevant in the first example. The internal workload of decision-maker j is calculated as follows:

$$I(j) = \sum_{m=1}^{K} \left(\frac{1}{v_m} \sum_{i=1}^{N} x_{im} \right) y_{jm} \tag{8}$$

where $x_{im} = 1$ if platform m is allocated to task i and $y_{jm} = 1$ if DM j owns the utilized platform m. The internal workload depends on the maximum velocity of the assigned platform to account for the fact that a faster platform will be unduly penalized for processing more tasks.

The basic strategy, adopted in the action selection (*AS*) stage, takes into account the knowledge that all DMs have identical capability to undertake the incoming tasks. Hence, individual DM strategy favors task

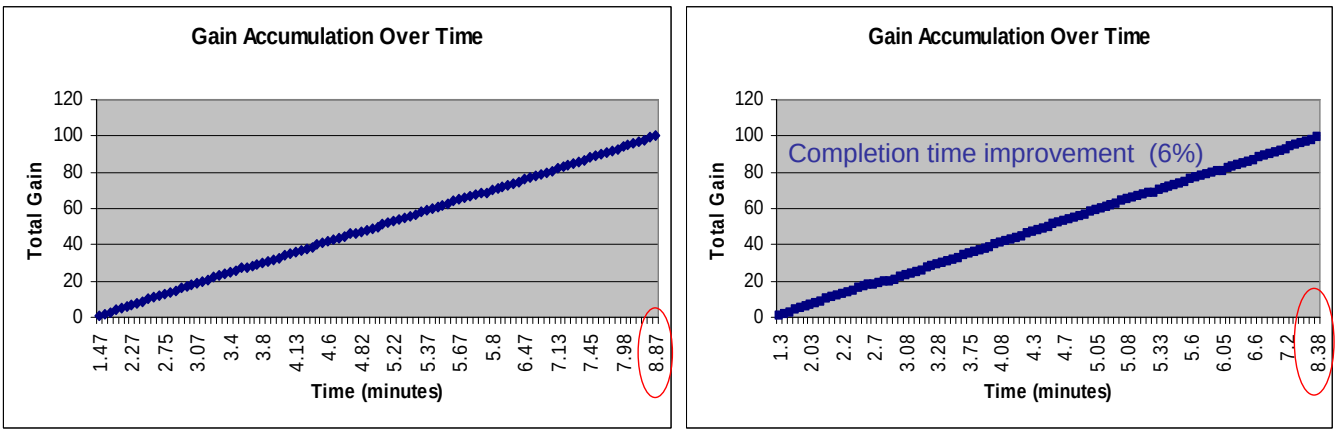

Fig. 8. Accrued Task Gain Over Time

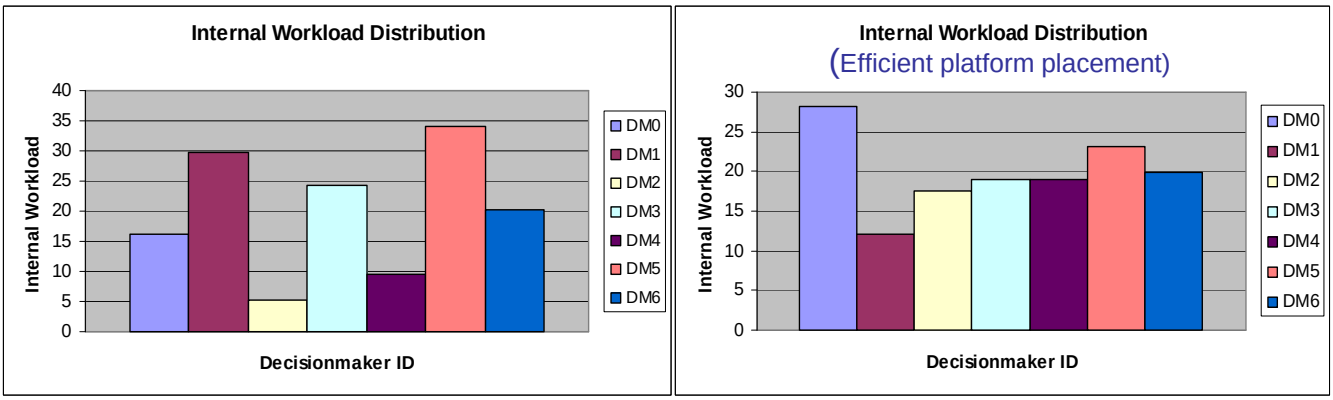

Fig. 9. Team's Internal Workload Distribution

processing with minimal effort, viz., platform fuel efficiency, or, equivalently, minimum platform-to-task distance. That is, any agent who is geographically closer to any incoming task should be assigned to the task.

Simulation results, in the first plot of Fig. 8, show that, as expected, the team of agents achieves high accrued gain over the entire mission duration. The internal workload distribution, shown in the first graph of Fig. 9, however, indicates that the team is inefficient in its processes. The basic strategy fails to recognize poor placement of platforms. Poor platform placement leads to uneven platform-to-task distances, which in turn, leads to uneven team workload. By appropriate placement of platforms belonging to different DMs, the scenario leads to a balanced team workload distribution, as shown in the second plot of Fig. 8, and, in turn, to more efficient task processing, as shown in the second graph of Fig. 9. Efficient platform placement leads to 6% improvement in the timeliness measure (mission completion time).

B. Congruent-Incongruent Scenarios Derived from Early Tasks of A2C2 Experiment 8

The second example utilizes organizational architectures and a set of tasks (excluding most of the hostile mosquito tasks, which have very short windows of opportunity to be executed, e.g. only 1.0, 3.0, or 5.0 seconds) used in the A2C2 Experiment 8. It should be noted, however, that the purpose of this example is only to illustrate the agent coordination capability in performing complex tasks, which require inter-platform coordination, and is not to replicate the results of experiment itself.

A2C2 Experiment 8, from which the architectures and missions are derived, considers two organizational structures, functional (F) and divisional (D) [23]. A functional organization is a team of decision-makers with non-overlapping resource capabilities, whereas a divisional organization is a geographically-organized team (with a reasonable degree of overlapping resource capabilities). The architectures represent two extreme cases of organizational structures, and, therefore, are thought of as suitable test cases for organization-to-mission congruence study. Two scenarios, termed functional (f, requiring minimum overlaps in the task resource requirements) and divisional (d, requiring overlaps in the task resource requirements), were designed to create the matched situations for F on f (functional structure - functional scenario) and D on d (divisional organization and divisional scenario) cases, and to create mismatches for F on d and D on f cases.

The example supposes that a network of intelligent agents, acting as organizations F and D, carry out real-time agent-in-the-loop decisions on the early parts of f and d scenarios. In order to capture the overall scenario within a short simulation time, the precedence constraints of the mission tasks are deliberately ignored. The accrued task gain and workload metrics are used to evaluate the simulation results. The results for accrued task gain, with confidence intervals, based on 10 simulation runs, for

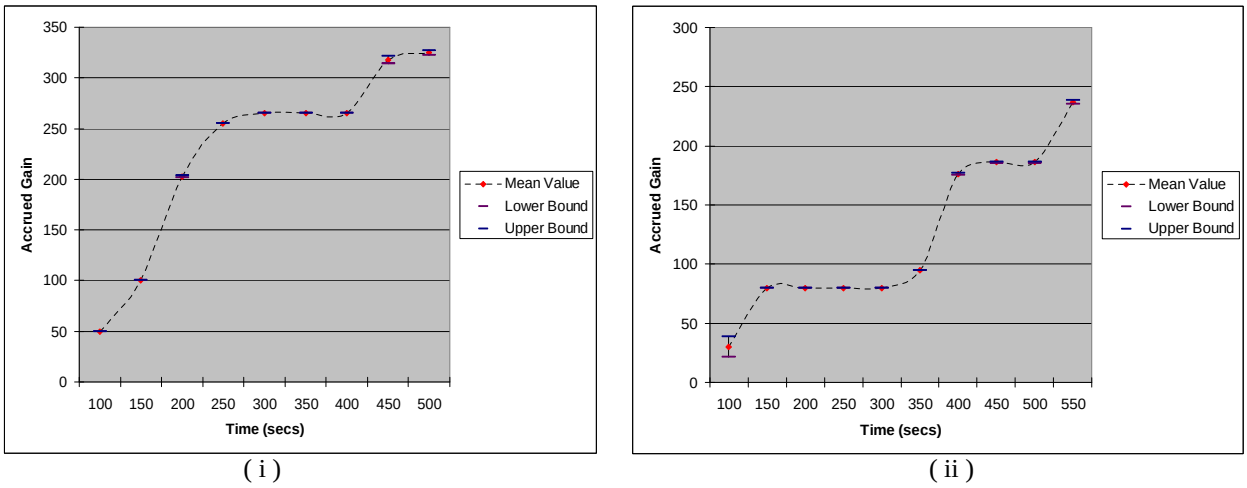

Fig. 10. Accrued Task Gain (Based on 10 Simulation Runs) for Organization F While Executing Early Parts of Missions (i) f and (ii) d

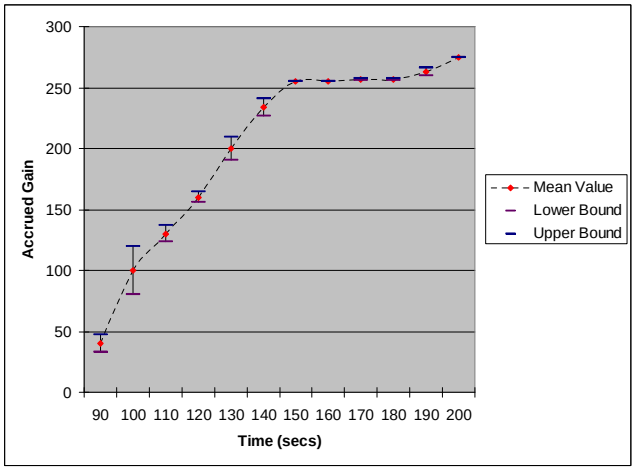

Fig. 11. Accrued Task Gain (Based on 10 Simulation Runs) for Organization D While Executing Early Part of Mission d

organization F performing early parts of mission f are shown in the first graph of Fig. 10, whereas the same results on mission d are displayed on the second plot. Recall that scenario f was designed to be congruent with organization F and incongruent (misfit) with organization D, and similarly for scenario d. The preliminary simulation results indicate that, in the congruent situation, viz., F on f, organization F performs better than in the incongruent situation, viz., F on d. The results are consistent with the hypothesis of congruence theory and human-in-the-loop experimental results in [27]. Results for the D organization performing d mission are presented in Fig. 11.

Note that, for both organizations, the mission tasks with high payoffs are selected first. This is consistent with the DDD agent's *AS* stage, which favors high-valued tasks. Since the team of DDD agents follows a prescribed team strategy as described in section IV, the results from 10 simulation runs indicate similar task selections and platform-to-task allocations. The differences are attributed to the uncertainty in the communication channel (between the DDD-III and the agents) that lead to varying delays affecting the DDD agent's *IP* stage, rather than a changing strategy. It should also be noted that the results only indicate the tasks completed by the team at the indicated time duration and not all of the tasks that are being processed (i.e., pursue or get close to the task). That is, the organizations are doing more than that

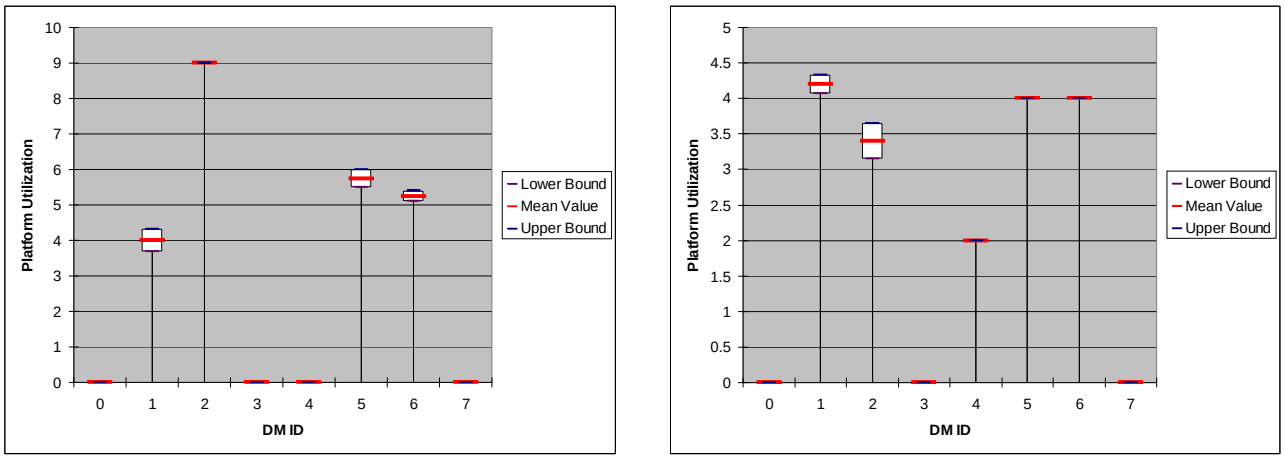

Fig. 12. Internal DM Workload Distribution (in Term of Platform Utilization, Based on 10 Simulation Runs) for Organization F While Executing Early Parts of Missions (i) f and (ii) d

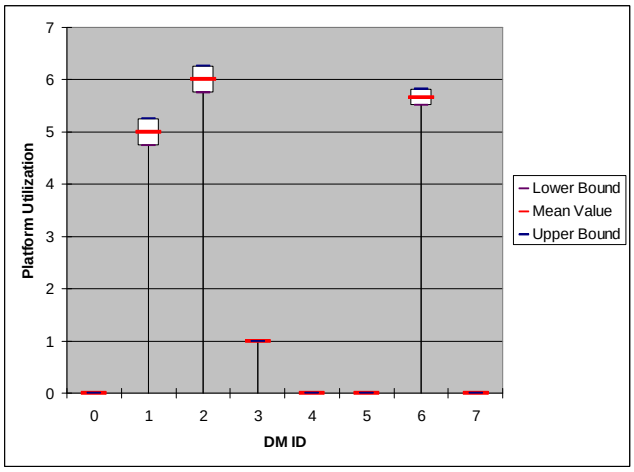

Fig. 13. Internal DM Workload Distribution (in Term of Platform Utilization, Based on 10 Simulation Runs) for Organization D While Executing Early Part of Mission d

indicated by the number of completed tasks.

The internal workload metric, which, in this case, is calculated as the number of platforms of each DM utilized to process tasks, and is presented in Figs. 12 and 13 for organizations F and D, respectively. As only a small number of the overall mission tasks was considered, the interpretation of the results is limited. It can be observed that for organization F performing mission f, only DM_2, DM_3, and DM_5, DM_6 are busy, whereas for F performing mission d, DM_4 is also busy. For organization D performing mission d, DM_1, DM_2, DM_3 and DM_6 are in charged of the tasks.

The team decision-making process was implemented via a centralized approach. The choices of the completed tasks indicate that most completed tasks are multi-platform-high-valued tasks requiring inter-DM coordination. Future improvements in agent models will implement distributed agent-based strategies and will include human cognitive biases and limitations.

VI. Conclusion

THE need for a network of intelligent agents within C2 experimental settings was presented. A brief overview of modeling techniques for the design of a network of collaborating agents, followed by techniques for modeling the decision-making processes of synthetic agents in task selection and resource allocation settings within the DDD-III [24] simulation, were discussed. In the proposed framework, the decision-making process of a network of intelligent agents were addressed via limited look-ahead, auction-based scheduling and resource allocation algorithms from the phase I of the three-phase organizational design process ([28], [29]).

Preliminary results of operationalizing the DDD-based multi-agent-network paradigm were presented via two different mission scenarios. A coordination-free scenario illustrated the basic structure, in terms of the three-stage decision-making processes, of the DDD-III agent. The second example, which is derived from A2C2 Experiment 8, highlighted the potential of utilizing the agent framework in C2 experiments. It is argued that as the agent models matured, they can be used in large-scale experiments involving hybrid human-agent teams.

References

[1] Barringer, H., M. Fisher, D. Gabbay, G. Gough, and R. Owens. (1989). "METATEM: A framework for programming in temporal logic," *REX Workshop on Stepwise Refinement of Distributed Systems: Models, Formalisms, Correctness* (LNCS Volume 430), pages 94-129. Springer-Verlag: Berlin, Germany.

[2] Bratman, M. E. (1987). *Intentions, Plans, and Practical Reason*. Harvard University Press: Cambridge, MA.

[3] Bratman, M. E., D. J. Israel, and M. E. Pollack. (1988). "Plans and resource-bounded practical reasoning," *Computational Intelligence*, 4:349-355.

[4] Bertsekas, D. P., D. A. Castanon, and H. Tsaknakis. (1993). "Reverse Auction and the Solution of Inequality Constrained Assignment Problems," *SIAM J. on Optimization*, 3:268-299.

[5] Kevin L. Boettcher and Alexander H. Levis, "Modeling the Interacting Decision Maker with Bounded Rationality," IEEE Trans. Syst., Man, Cybern., Vol. SMC-12, No. 3, pp. 334-344, May/June 1982.

[6] K. Decker and V. Lesser. (1995). "Designing a family of coordination algorithms," In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 7380, San Francisco, CA.

[7] Deutsch, S.E. (1998). "Interdisciplinary foundations for multiple-task human performance modeling in OMAR," *Proceedings of the Twentieth Annual Meeting of the Cognitive Science Society*, Madison, WI.

[8] Deutsch, S.E. & Adams, M.J. (1995). "The operator model architecture and its psychological framework," *Proceedings of the 6th IFAC Symposium on Man-Machine Systems*, Cambridge, MA.

[9] Durfee, Edmund H. (1989). "Planning for Problem Solving: Experiments in acquiring and sharing knowledge to improve decision making," *Proceedings of the Workshop on Knowledge, Perception, and Planning* , Detroit, MI.

[10] E. H. Durfee. (1988). *Coordination of Distributed Problem Solvers*. Kluwer Academic Publishers: Boston, MA.

[11] M. P. Georgeff. (1983). "Communication and interaction in multi-agent planning," In *Proceedings of the Third National Conference on Artificial Intelligence (AAAI-83)*, Washington, D.C.

[12] Georgeff, M. P. and A. L. Lansky. (1987). "Reactive reasoning and planning," *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 677-682, Seattle, WA.

[13] Georgeff, M. P. and A. S. Rao. (1996). "A profile of the Australian AI Institute," *IEEE Expert*, 11(6):89-92.

[14] Genesereth, M. R. and N. Nilsson. (1987). *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers: San Mateo, CA.

[15] Barbara Grosz and Sarit Kraus. (1996). "Collaborative plans for complex group actions," *Artificial Intelligence*, 86(2):2693571, 1996.

[16] Haddadi, A. (1996). *Communication and Cooperation in Agent Systems (LNAI Volume 1056)*. Springer-Verlag: Berlin, Germany.

[17] Handley H.A.H. and Levis A.H. (2000). "On organizational adaptation via dynamic process selection," *Proceeding of 7th International Command and Control Research and Technology Symposium*, Monterey, CA, Track 1 - 23 pages.

[18] Hayes, C. C. (1999). "Agents in a nutshell - a very brief introduction," *IEEE Trans. on Knowledge and Data Engineering*, 11(1):127 - 132.

[19] Huhns, M. N. and L. M. Stephens. (1999). "Multiagent Systems and Societies of Agents," in *Multiagent Systems: A modern Approach to Distributed Artificial Intelligence, Weiss, G. ed.*, MIT Press: Cambridge, MA.

[20] Jennings, N. R., K. Sycara, and M. Wooldridge. (1998). "A Roadmap of Agent Research and Development," in *Autonomous Agents and Multi-Agent Systems*, pg. 275-306. Kluwer Academic Publishers: Boston, MA.

[21] N. R. Jennings. Specification and implementation of a belief desire joint-intention architecture for collaborative problem solving. Journal of Intelligent and Cooperative Information Systems, 2(3):289318, 1993. 79.

[22] N. R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. Artificial Intelligence, 74(2), 1995.

[23] Kleinman, D. L., G. M. Levchuk, S. G. Hutchins, and W. G. Kemple. (2003). "Scenario Design for Empirical Testing of Organizational Congruence," to appear in *Proceedings of the 2003 Command and Control Research and Technology Symposium*.

[24] Kleinman, D. L., P. Young, and G. S. Higgins. (1996). "The DDD-III: A Tool For Empirical research in Adaptive Organizations," *Proceedings of the 1996 Command and Control Research and Technology Symposium*, Monterey, CA.

[25] Joel S. Lawson, Jr., ”Naval Tactical C3 Architecture 1985-1995,” Signal, Vol. 33, No. 10, August 1979, pp. 71-76.
[26] Lesser, V. R. (1999). ”Cooperative multiagent systems: A personal view of the state of the art,” *IEEE Trans. on Knowledge and Data Engineering*, 11(1): 133 - 140.
[27] Levchuk, G. M., D. L. Kleinman, S. Ruan, and Krisna R. Pattipati. (2003). ”Congruence of Human Organizations and Missions: Theory versus Data,” to appear in textitProceedings of the 2003 Command and Control Research and Technology Symposium.
[28] Levchuk G.M., Levchuk, Y.N., Luo Jie, Pattipati K.R., and Kleinman D.L. (2002a). ”Normative design of organizations-part I: mission planning,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 32(3), 346-359.
[29] Levchuk G.M., Levchuk, Y.N., Luo Jie, Pattipati K.R., and Kleinman D.L. (2002b). ”Normative design of organizations - part II: organizational structure,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 32(3), 360-375.
[30] Levchuk, G.M., C. Meirina, Y.N. Levchuk, K.R. Pattipati, and D.L. Kleinman (2001), ”Design and Analysis of Robust and Adaptive Organizations,” *Proceedings of the 2001 Command and Control Research and Technology Symposium*.
[31] Alexander H. Levis and Kevin L. Boettcher, ”Decisionmaking Organizations with Acyclical Information Structures,” IEEE Trans. Syst., Man., Cybern., Vol. SMC-13, No. 3, May/June 1983, pp. 384-391.
[32] Maes, Pattie (1995), ”Artificial Life Meets Entertainment: Life like Autonomous Agents,” *Communications of the ACM*, 38, 11, 108-114.
[33] March, J.G. and H.A. Simon. (1958). *Organizations*, John Wiley and Sons, NY, NY.
[34] Krishna R. Pattipati. (1980). *Dynamic decision-making in multi-task environments: theory and experimental results*, Univ. of Connecticut, Ph.D. Thesis.
[35] Rosenschein, S. J. and L. P. Kaelbling. (1996). ”A situated view of representation and control,” *Computational Theories of Interaction and Agency (P. E. Agre and S. J. Rosenschein, editors)*, pages 515-540. The MIT Press: Cambridge, MA.
[36] J. S. Rosenschein and G. Zlotkin. Rules of Encounter: Designing Conventions for Automated Negotiation Among Computers. MIT Press, Boston, MA, 1994.
[37] Russell, S. J. and E. Wefald. (1991). *Do the Right Thing Studies in Limited Rationality*. The MIT Press: Cambridge, MA.
[38] Russell, S. and P. Norvig. (1995). *Artificial Intelligence: A Modern Approach*. Prentice-Hall.
[39] Russell, S. J. and D. Subramanian. (1995). ”Provably bounded-optimal agents,” *Journal of AI Research*, 2:575-609.
[40] Stone, P. and M. Veloso. ”Multiagent systems: A survey from the machine learning perspective,” *Autonomous Robotics*, 8(3).
[41] Sycara, K. P. (1990). ”Negotiation planning: An AI Approach,” *European Journal of Operational Research*, 46: 216 - 234.
[42] Sycara, K. P. (1998). ”Bayesian learning in negotiation,” *International Journal of Human-Computer Studies*, 48.
[43] Joseph G. Wohl, ”Force Management Decision Requirements for Air Force Tactical Command and Control,” IEEE Trans. Syst., Man, Cybern., Vol. SMC-11, No. 9, pp. 618-639, September 1981.
[44] Joseph G. Wohl, E. E. Entin, M. G. Alexandridis, J. S. Eterno, Toward a Unified Approach to Combat System Analysis, Alphatech, Inc. Technical Report TR-151, January 1983, ADA124570.
[45] Joseph G. Wohl, E. E. Entin, J. S. Eterno, Modeling Human Decision Processes in Command and Control, Alphatech, Inc. Technical Report TR-137, January 1983, AD A125218.
[46] Wooldridge, M. (1997). ”Agent-based software engineering,” *IEE Transactions on Software Engineering*, 144(1):26-37.
[47] Wooldridge, M. and N. R. Jennings. (1995) ”Intelligent agents: Theory and practice,” *The Knowledge Engineering Review*, 10(2):115-152.