A Multi-Functional Software Environment for Modeling Complex Missions and Devising Adaptive Organizations[*]

Yuri N. Levchuk
Jie Luo
Georgiy M. Levchuk
Krishna R. Pattipati
David L. Kleinman
Dept. of Electrical and Systems Engineering, University of Connecticut, Storrs, CT 06269-3157
e-mail: krishna@sol.uconn.edu

Abstract

This paper presents a software environment that uses our comprehensive modeling and design methodology for representing complex missions and synthesizing the concomitant adaptive organizational structures for different sets of design objectives. The tool box provides a step-by-step visualization of modeling a complex mission and building an "optimal" organization that achieves superior performance, while satisfying organizational constraints. In addition, the software environment allows one to perform a comparative analysis of different organizations for various (user-defined) performance measures, and to quantify the robustness of a given organizational design. The methodology incorporates algorithms for optimizing resource allocation, mission schedule, and information management, for balancing decision-making workload, and for maximizing controllability of the mission processing by the resulting organizational hierarchy. The software environment is illustrated via an example. The methodology and tools presented allow for automation of the modeling and design process. Herein, they constitute valuable instruments for scientific research in the area of organizational decision-making and human team behavior.

1. Introduction

1.1 Motivation

The revolution in information technology is changing the way in which modern organizations, from military establishments to agile manufacturing systems and commercial enterprises, conduct their business. As changing patterns of today's world compel modern organizations to deal with uncertain and unsteady mission environments under time pressure, advanced information systems and capabilities must be tailored to support new decision-making roles and requirements, to facilitate communication, management of resources and information, operational planning, situational awareness, and dynamic distributed decision-making. With every significant commercial or military undertaking requiring a coordinated effort of a large group of geographically distributed individuals controlling a variety of resources, a proper balance among information acquisition, designation of a decision hierarchy, and resource allocation, in short, *a proper organizational design*, is critical to superior organizational performance.

[*] This work was supported by the Office of Naval Research under contract #N00014-93-1-0793.

Over the years, research in organizational decision-making has demonstrated that a strong relationship exists between the specific structure of a task environment (i.e., mission) and the concomitant optimal organizational design ([Reibman and Nolte, 1987], [Papastavrou and Athans, 1992], [Pete *et al.*, 1993], [Tang *et al.*, 1993], [Lin & Carley, 1995]). To utilize this structural dependency, appropriate models of the mission and organization, capturing the actual mission parameters and organizational constraints, must be formulated prior to the design phase. For large missions and organizations, this leads to ever growing demand for automated tools to assist the user in modeling missions and generating optimal organizational architectures. Ideally, such tools must be sufficiently versatile to handle a wide scope of missions and to optimize organizational structures for different (user-defined) criteria. As a by-product, having such an automated design environment would allow one to examine human decision-making and coordination processes under different conditions, to generate new performance measures and design hypotheses, to perform a comparative analysis of different (not necessarily optimal) organizational structures, and to test the robustness of a given organizational design.

1.2 Groundwork for Automating the Modeling and Design Process

Over the past few years, we have developed a comprehensive methodology for devising mathematical and computational models of organizations and their missions, as well as for building optimal and near-optimal organizational structures for different design objectives. Specifically, for the multi-objective problem of designing organizations to complete a complex mission, while minimizing a set of criteria, we presented an iterative procedure to generate an optimal organizational structure and strategy ([Levchuk *et al*., 1997, 1998a, and 1998b]). Also, a multi-attribute cause-effect dependency model for monitoring and failure diagnostics in a complex mission environment has been introduced in [Ying *et al*., 1997]. The potential benefits of a structural match between a mission structure and a concomitant organizational design, predicted by the normative models ([Levchuk *et al*., 1996 and 1997] and [Pete *et al.*, 1998]), have been tested empirically in a computer-mediated team-in-the-loop experiment with human DMs, operationalized in an enhanced Distributed Dynamic Decisionmaking (DDD-III) simulation of Joint C^2 scenarios ([Kemple *et al.*, 1999]).

1.3 Scope and Organization of Paper

In this paper, we present a software environment that uses our comprehensive modeling and design methodology to visualize modeling complex missions and synthesizing the concomitant optimal organizational structures. In section 2, we provide rationale for our user interface composition. We introduce a modified organizational design procedure that incorporates algorithms for optimizing mission schedule, resource allocation, information management and communication, as well as for balancing decision-making workload and maximizing controllability of the mission processing by the resulting organizational hierarchy. In section 3, we illustrate how to use our software environment for mission modeling via a hypothetical example. In section 4, for a resource-to-task allocation phase of our optimization procedure, we present a new task scheduling algorithm that generates a near-optimal mission schedule for a given set of task requirements, resource capabilities, and task transition constraints. In section 5, we use the hypothetical example, introduced in section 3, to illustrate our organizational design process. Finally, we conclude with a summary in section 6.

2. Modeling and Design Process and Software Environment Composition.

2.1 Modeling as a First Step in Designing Human Organizations

Over the years, research in team decision-making has demonstrated that an organization operates best when its structure and processes fit, or match, the corresponding mission environment. To exploit this dependency between the structure of a mission and a concomitant optimal organizational design, the critical information about the mission structure must be captured and quantified to establish a mathematical framework for application of optimization techniques to organizational design process. Thus, developing a model that highlights critical dimensions of the mission structure is one of the keys to a successful organizational design.

The application of systems engineering techniques to modeling and design of organizations allows one to utilize numerical optimization algorithms for optimizing *predicted* human team performance. The systems engineering approach to organizational design is as follows. Once a quantitative model describing the mission and the organizational constraints is built, different criteria, used to judge the optimality of an organization, are combined into a (possibly non-scalar) objective function, and an organizational structure is generated to optimize the objective function.

One of the key features of an organizational structure is its multi-dimensionality, i.e., organizational structure prescribes the attributes that characterize many different processes and relationships among various organizational entities (i.e., information access and allocation of resources to DMs, DM command and control hierarchy, data transfer and communication channel structures, etc.). As a result, an objective function often combines several non-commensurate criteria. When this is the case, the organizational design problem is treated as a multi-objective optimization problem.

In large-scale organizations that involve humans, machines, sensors, computers, and databases, human DMs play a special role. Their shared decision-making and operational functions enable DMs to coordinate their actions in order to achieve their common goal. Since the capabilities of a human are limited, the distribution of *information*, *resources*, and *activities* among DMs in an organization must be set up accordingly, to guarantee that decision-making and operational load on each DM remains below the corresponding thresholds.

In general, decision-makers are provided with limited resources with which to accomplish their objectives. The distribution of these resources among DMs and the assignment of these resources to seek information and to conduct operational activities are key elements in an organization's design.

Thus, a successful organizational design is contingent on completing the following key steps:

(i) elucidating the structure of the mission (i.e., defining a quantitative mission model);

(ii) characterizing the structure of an organization (i.e., defining structural dimensions and delineating organizational constraints);

(iii) specifying performance criteria (and their precedence rank);

(iv) generating an optimal mission schedule;

(v) allocating resources to DMs and designating DMs' functionalities;

(vi) building DM hierarchy and communication structure.

The total decision-making and operational load is generally partitioned among DMs by decomposing a mission into tasks and assigning these tasks to individual decision-makers who are responsible for their planing and execution. When considering a mission, one must, in general, differentiate between the mission objectives (goals) and mission tasks. While mission objectives can generally be independent of organizational constraints, mission decomposition into tasks is contingent upon available resources and DMs' expertise (one cannot accomplish his goals by completing tasks for which he has neither resources nor the required expertise; instead, he should try to find a different way to reach his objectives subject to constraints on resources and expertise).

Although the information about the organization's resources is readily available to a designer, and, thus, it can be used when decomposing a mission into tasks, the description of DMs' expertise for processing a specific task requires the explicit definition of the task. Due to an enormous diversity among all conceivable tasks, we cannot, in general, stipulate DMs' expertise before we develop the mission model and mission task structure. Sometimes, we can define a DM's expertise for a certain group of elementary tasks, and subsequently restrict our attention to missions that are accomplished by completing only those classes of tasks; however, by doing that, we implicitly impose a specific task structure on our mission, and, hence, limit our options of how to reach our mission objectives. With the above reasoning in mind, we will maintain a particular order when developing a joint quantitative model for a complex mission and a concomitant organization to provide an input for our organizational design process (see Fig. 1).

2.2 Organizational Design Procedure

The optimal organizational design problem can be formulated as one of finding both the optimal organizational structure and its optimal feasible strategy that together minimize a multi-variable objective function, with the elements of the organizational environment (i.e., number of DMs, their expertise, available resources, etc.) providing constraints on the organizational design.

In general, all the existing methods of multi-objective optimization are NP hard (optimal algorithms take exponential time). For example, one obvious, although computationally infeasible for large size problems, solution would be to enumerate all feasible mappings of the mission structure onto an organization structure, with subsequent evaluation of feasible strategies for each particular organization architecture. However, this is a rather tedious task, especially if the number of variables (tasks, resources, DMs) is large.

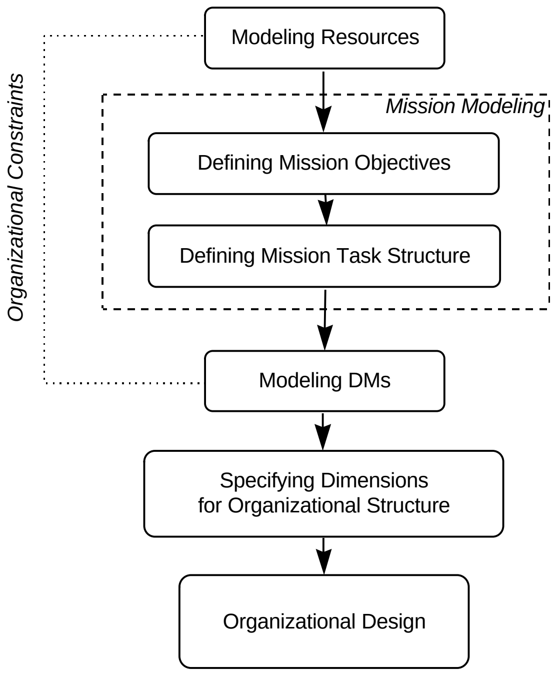

Modeling Resources
Mission Modeling
Defining Mission Objectives
Defining Mission Task Structure
Organizational Constraints
Modeling DMs
Specifying Dimensions for Organizational Structure
Organizational Design

Fig. 1. Modeling a Mission and an Organization.

One way of simplifying the search for the optimal organizational design is to exploit the connection between multi-dimensionality of organizational structure and the concerted composition of a multi-variable objective function. In general, the objective function combines variables representing both mission objectives and design parameters (e.g., decision-making workload, resource utilization, cost of the design, etc.). Each dimension of the organizational structure stipulates a corresponding portion of the design parameters (e.g., DM-resource allocation and mission schedule define the operations workload of a DM, while information access structure, allocation of decision variables, and communication structure stipulate a decision-making workload of a DM).

The relative weights of the optimization criteria that determine organizational performance can be represented via weighting coefficients assigned to each variable in the objective function. Therefore, in theory, we can build an organizational structure by iteratively optimizing different structural dimensions, beginning with those dimensions that delineate the heaviest portion of the objective function (e.g., an organizational strategy determines the mission processing schedule as well as the individual operational DM's workload, so it generally specifies a large portion of parameters in the multi-variable objective function). Each following dimension is optimized subject to a fixed structure on those dimensions that have been optimized already. The iterative application of the algorithm allows one to simultaneously optimize all performance criteria, subject to the acceptable trade-off among design objectives specified by equi-cost surfaces of the objective function.

The above logic allows for integration (into our organizational design procedure) of various algorithms that optimize mission schedule, resource allocation, information management and communication, coordination delays, decision-making workload, and so on. This approach is utilized next in our five-phase iterative optimal organizational design algorithm (this algorithm is an extension of a three-phase algorithm that was introduced in [Levchuk *et al*., 1997, 1998]). For a given mission structure, an organization is designed via the following five phases (see Fig.2):

Phase I. The first phase of the algorithm determines the task-resource allocation and task sequencing that optimize mission objectives, taking into account task precedence constraints and synchronization delays, task resource requirements, resource capabilities, as well as geographical and other task transition constraints. The complete version of this algorithm is presented in section 4 of this paper. The generated task-resource allocation specifies the workload per unit resource. In addition, for every mission task, the first phase of the algorithm determines a set of non-redundant resource packages capable of jointly processing a task. This information is later used for iterative refinement of the design, and, if necessary, for on-line strategy adjustments.

Phase II. The second phase of the algorithm combines resources into non-intersecting groups, to match the operational expertise and workload threshold constraints on available DMs, and assigns each group to an individual DM to define the DM-resource allocation. Thus, the second phase delineates the DM-task-resource allocation schedule and, consequently, the individual operational workload of each DM.

Phase III. The third phase of the algorithm designates information processing and decision-making functionalities among DMs by allocating appropriate information task variables to DMs. In this phase, the decision-making workload of each DM is balanced to match the corresponding expertise and workload threshold constraints, and minimal required inter-DM coordination is estimated.

Phase IV. This phase of the algorithm defines the information access structure and communication structure among DMs by allocating sensor displays and data links to DMs according to the information requirements at each DM node (established in Phase III), information access constraints and displays and data links availability. It optimizes the information load of each DM, as well as communication among DMs.

Phase V. Finally, Phase V of the algorithm completes the organizational structure by specifying a DM command hierarchy to optimize the responsibility distribution and inter-DM control coordination, as well as to balance the control workload among DMs according to expertise constraints on DMs.

Each phase of the algorithm provides, if necessary, a feedback to the previous stages to iteratively modify the task-resource and DM-resource allocation, as well as the information access and communication structures among DMs.

On-line Adaptation Phase. In addition to generating an optimal organizational structure and strategy for a specified mission model, our software tool box provides an instrument for a quick and efficient search for adaptation options in the event of resource or DM failure (see [Levchuk *et al*., 1998] for details). After a faulty resource (or a group of resources, in the case of a DM node failure) has been identified, the organizational constraints are readjusted accordingly, and the *new task-resource allocation strategy* is generated in Phase I of the algorithm for the remaining portion of the mission, after which Phases II through V are used to determine the necessary changes in the organizational structure. In this case, Phases II through V are completed in an evolutionary mode (e.g., platform clusters in Phase II are obtained via regrouping of the old

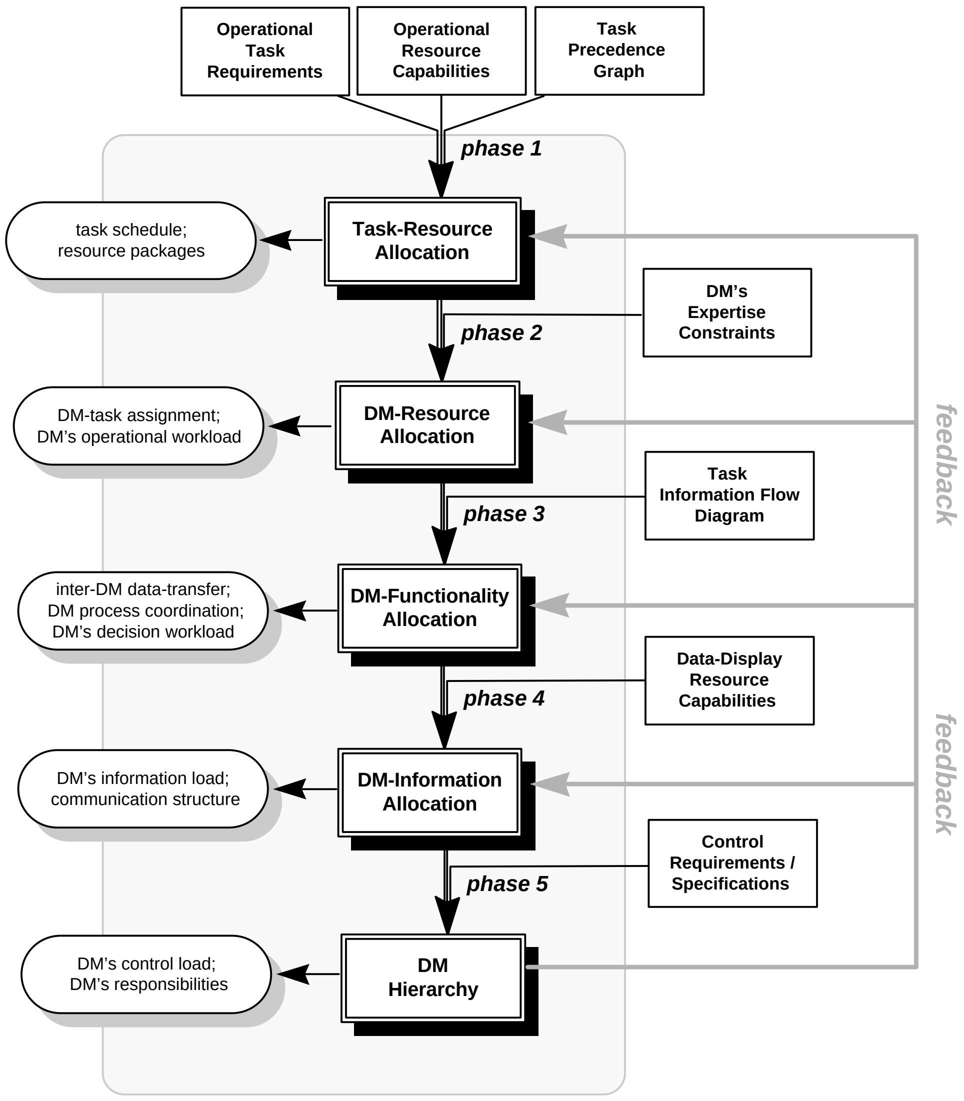

Operational Task Requirements
Operational Resource Capabilities
Task Precedence Graph
phase 1
Task-Resource Allocation
task schedule; resource packages
DM's Expertise Constraints
phase 2
DM-Resource Allocation
DM-task assignment; DM's operational workload
Task Information Flow Diagram
phase 3
DM-Functionality Allocation
inter-DM data-transfer; DM process coordination; DM's decision workload
Data-Display Resource Capabilities
phase 4
DM-Information Allocation
DM's information load; communication structure
Control Requirements / Specifications
phase 5
DM Hierarchy
DM's control load; DM's responsibilities
feedback
feedback

Fig. 2. Organizational Design Process

platform groups, rather then generating new platform groups from scratch). Finally, if the process of generating new organizational strategy and/or structure fails ([Levchuk *et al.*, 1998]), the mission must be *aborted*.

2.3 Component Architecture and Capabilities of Our Modeling and Design Tool Box

Our software environment, designed to assist the user in modeling complex missions and building concomitant optimal organizations, consists of the following seven components (Fig.3):

(1) *Resources Delineator* (R);

(2) *Mission Structure Outliner* (M);

(3) *DM Structure Profiler* (D);

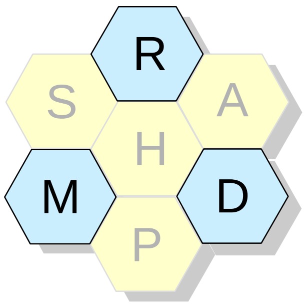
(4) *Performance Criteria Architect* (P);

(5) *Schedule Generator* (S);

(6) *Allocation Definer* (A);

(7) *Hierarchy Designator* (H).

The first three components (*Resources Delineator*, *Mission Structure Outliner*, and *DM Profiler*) assist the user in developing analytic quantitative models for the mission and organization that serve as an input for our organizational design algorithm. The *Performance Criteria Architect* component is used to stipulate performance measures for the prospective organizational design and to define a performance cost function that aggregates mission objectives and design parameters. The last three components (*Schedule Generator*, *Allocation Definer*, and *Hierarchy Designator*) provide a step-by-step visualization of the optimal organizational design process (e.g., *Schedule Generator* component produces the task-resource allocation that corresponds to Phase I of our organizational design algorithm, while *Allocation Definer* and *Hierarchy Designator* allow the user to complete Phases II through V of the design process).



Tool Box Components:

R: *Resources Delineator*;

M: *Mission Structure Outliner*;

D: *DM Structure Profiler*

P: *Performance Criteria Architect*;

S: *Schedule Generator*;

A: *Allocation Definer*;

H: *Hierarchy Designator*.

Fig. 3. Component Architecture for Our Software Environment

The modular structure of our software environment (illustrated in detail throughout the following sections) allows one to apply different optimization algorithms at different stages of the design process to handle the complexity of a specific problem at hand. Our tool allows the user to explore various levels of model complexity, delineate (and visualize) different dimensions of the resulting mission structure, and establish the functional interdependencies among the mission elements that define the dynamics of the mission environment. In addition, our tool adds the capability of user-defined design modifications at various stages of the process and displays the metrics of organizational performance, the attainment of mission objectives, and the workload distribution across the organization. Hence, our tool provides a complete *visualization* of the organizational design process, enabling the user to conduct a comprehensive study of the mission and the concomitant organizational architectures (prior to actual mission processing by the organization).

By specifying the task-resource allocation preference, as well as resource packages to process each mission task, our tool generates a pool of organization strategies (rather then a single strategy). Thus, in the event of structural failures, the adaptation mode of our tool allows the rapid *on-line* search for adaptation options performed among preprocessed data.

In summary, our multi-functional software environment provides the user with the following features:

- creating and updating a mission model;
- for a given mission and organizational constraints, generating the optimal organizational design;
- on-line structural and strategy adaptation to resource and DM failures;
- assessing the robustness of a given organization;
- for a given mission and organization, evaluating various organizational strategies;
- for a given mission, generating the performance predictions for various user-defined organizations;
- for a given organizational strategy, evaluating different designs capable of employing the strategy;
- for a given mission and organization, devising and visualizing the optimal organization strategy.

The internal architecture and functionalities of each component, as well as the application of our software environment to modeling a mission and designing an optimal organization, are illustrated in the following sections.

3. Step-by-step Modeling of a Complex Mission.

To illustrate how to use our proposed software environment package for modeling a mission and designing a concomitant organization, we develop a simplified hypothetical war-game mission scenario example. To highlight our tool's capabilities in modeling new, unfamiliar objects, our example introduces fictitious mission and resource makeup.

Example. Our hypothetical mission scenario is as follows. In a bellicose interstellar realm of the Third Millennium, a Joint Task Force Group (JTFG) is given a set of resources and is assigned to conduct a multi-faceted operation aimed at attacking and destroying three strategic enemy objects: a Geothermal Power-Plant, a Radar Tower, and a Sonar Station. From intelligence sources, it is known that the enemy units used to protect the above objects include missile towers (named Pulverisers), stationary plasma batteries (Punishers), and Light Laser batteries. It is also known that, due to a significant energy consumption while in their active mode, the Punisher and Light Laser battery units (the exact number of which is unknown) are not activated until they receive an alarm message from appropriate enemy intelligence units. Until their activation, the Punisher and Light Laser battery units are invisible to energy sensors (which is another reason for keeping them inactive until the battle commences). It is estimated that it takes the enemy one minute to activate each additional unit. On the other hand, the Pulverisers remain in constant readiness and can fire at any given time. In addition to the above information, friendly intelligence reports that the enemy is using five hovercraft air scouts (termed Peepers) for land recognizance. While Peepers' main function is to spot any land warfare advancing toward the Power-Plant, Radar Tower, and Sonar Station, and to send the alarm message to all enemy units tasked to protect these objects, Peepers can also carry various air-to-ground missiles and can be used to destroy the land warfare units. The composition of friendly assets (with versatile capabilities) available for the operation is as follows (see Fig.4): (i) an air-fighter (Avenger); (ii) a stealth air-fighter (Hawk); (iii) a fast attack vehicle (Jeffy); (iv) a mobile rocket launcher (Merl); (v) a very heavy assault tank (named Goliath); (vi) an amphibious tank (Triton); (vii) a mobile radar (Informer); and (viii) a heavy assault tank (Reaper). The friendly forces are initially located in the middle bottom portion of the map (see Fig.5). The enemy objects are located in the upper section of the map in a dormant volcanic

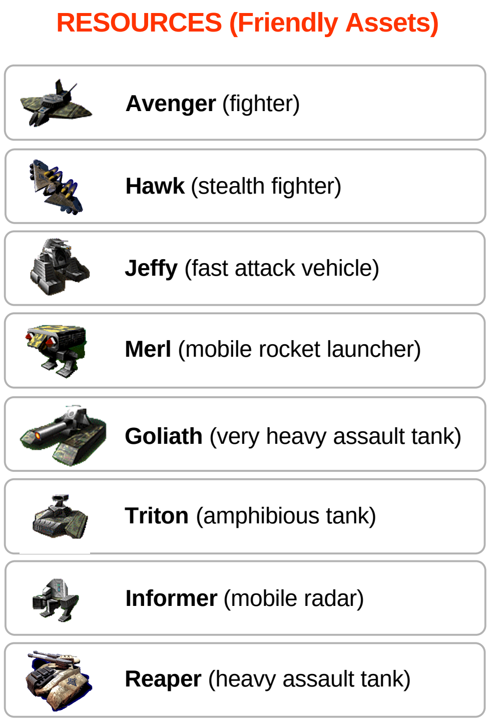

RESOURCES (Friendly Assets)
Avenger (fighter)
Hawk (stealth fighter)
Jeffy (fast attack vehicle)
Merl (mobile rocket launcher)
Goliath (very heavy assault tank)
Triton (amphibious tank)
Informer (mobile radar)
Reaper (heavy assault tank)

Fig. 4. Organization’s resource makeup.

area, and Peeper units are scattered across the map, hovering over the no-flight zone that separates the friendly forces from the enemy objects (Fig.5).

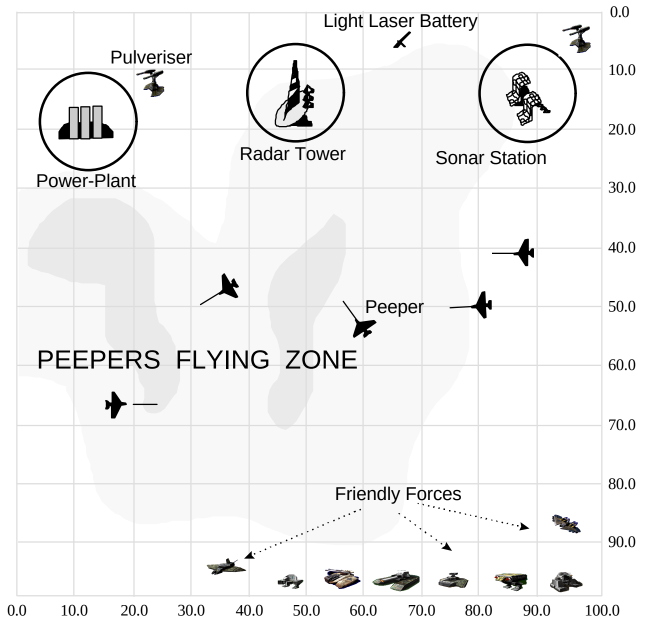


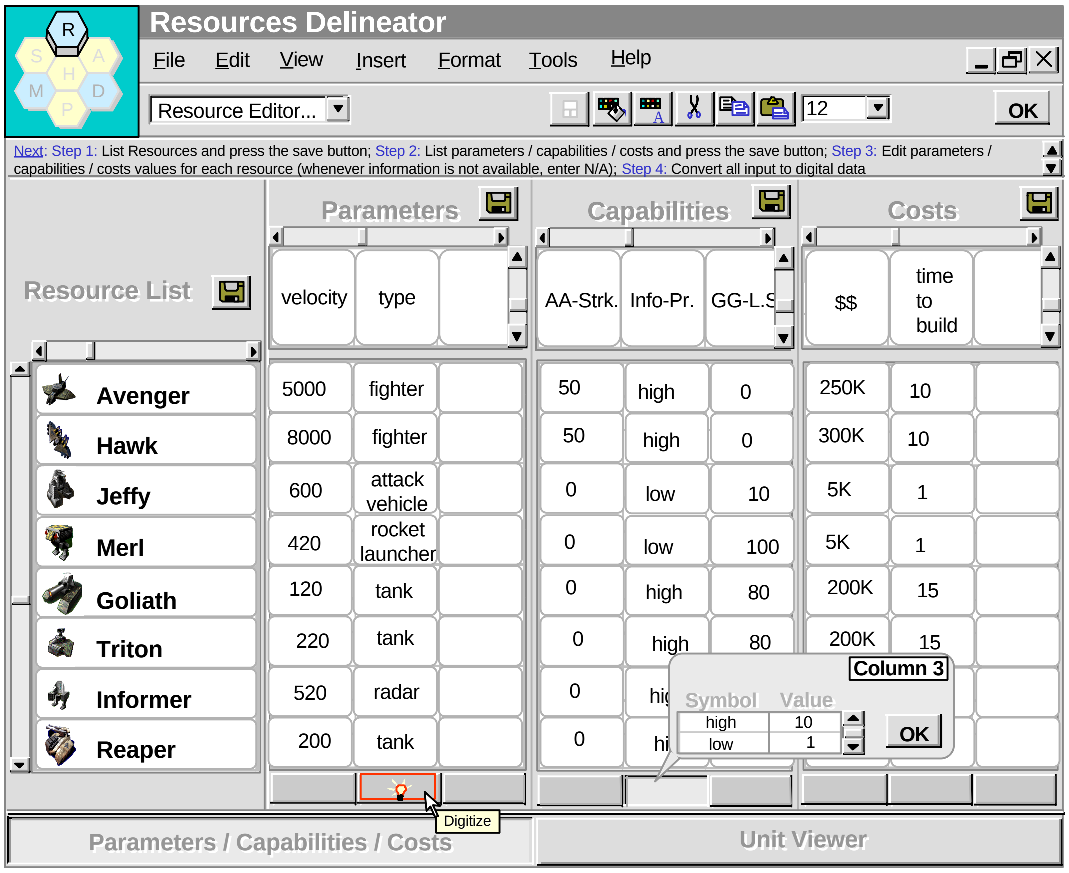
Fig. 5. Terrain Map.

Equipped with the above information, the commander (CJTFG) sets out to devise a plan for the mission that will specify all the tasks to be completed, as well as analyze the decision-making involved and stipulate who completes what task, which resources are used to complete each specific task, and how JTFG will coordinate in order to guarantee the best performance. The CJTFG's initial strategy is to use the available resources to suppress the enemy air-defense (consisting of Peeper hovercraft air scouts), and than to advance the joint friendly land warfare forces across the no-flight zone towards the enemy objects, while friendly air units position themselves to observe the "activation" of enemy defense (Punisher and Light Laser battery units). Once in the vicinity of enemy objects, the target allocation among friendly platforms is performed to maximize the inflicted damage (if not to totally wipe out the enemy) or the mission is aborted if no such allocation is possible. The retrieval of all friendly units back into its initial position zone ends the mission.

3.1 *Resource Delineator*: Capturing Resource Constraints

The aggregated capabilities of all JTFG's resources define the constraints on feasible strategies available to JTFG to achieve its mission objectives. Thus, the analysis of these capabilities is a prerequisite to deciding on a specific task makeup of the mission that would ensure the fulfillment of JTFG's goals.

The *Resources Delineator* component of our software tool box allows the user to specify a set of the organization's resources (platforms, assets) together with their attributes (such as capabilities, velocities, sensors and weapons ranges, initial geographical locations, costs per unit, etc.). The (digitized) resource attributes are used throughout the organizational design procedure to evaluate the constraints on the resource-to-task allocations (thus., they provide the input for other components of our software environment). For our example, the following parameters have been chosen to quantify the resource capabilities for processing the above mission: (1) Air-Detection capability; (2) Air-to-Air-Strike capability; (3) Communication capability; (4) Info-Processing capability; (5) Data Analysis capability; (5) Ground-Detection capability; (6) Air-to-Ground-Strike capability; (7) Ground-to-Ground-Light-Strike capability; (8) Ground-to-Ground-Heavy-Strike capability; (9) Fast-Movement capability; (10) Slow-Movement capability. Fig.6 illustrates editing resources and their parameters. Note that we also allow one to edit the various costs for each resource unit.



Next: Step 1: List Resources and press the save button; Step 2: List parameters / capabilities / costs and press the save button; Step 3: Edit parameters / capabilities / costs values for each resource (whenever information is not available, enter N/A); Step 4: Convert all input to digital data

Resource List	velocity	type		AA-Strk.	Info-Pr.	GG-L.S	$$	time to build	
Avenger	5000	fighter		50	high	0	250K	10	
Hawk	8000	fighter		50	high	0	300K	10	
Jeffy	600	attack vehicle		0	low	10	5K	1	
Merl	420	rocket launcher		0	low	100	5K	1	
Goliath	120	tank		0	high	80	200K	15	
Triton	220	tank		0	high	80	200K	15	
Informer	520	radar		0	hig				
Reaper	200	tank		0	hi				

Symbol	Value
high	10
low	1

Fig. 6. Editing Organization’s resource constraints.

3.2 *Mission Structure Outliner*: Modeling a Mission

The *Mission Structure Outliner* component of our software tool box allows the user to delineate (and visualize) different dimensions of the resulting mission structure, and establish the functional interdependencies among the mission elements that define the dynamics of the mission environment. The subcomponent architecture of *Mission Structure Outliner* is presented in Fig. 7.

3.2.1 Defining Mission Task Makeup

The CJTFG must now decide on a set of the specific tasks to accomplish the mission. He can achieve this by devising a *mission decomposition diagram* (mission decomposition tree) that stipulates the task makeup of the mission (represented by the "leaves" of the mission tree).

The *Mission Decomposition Builder* subcomponent allows the user to decompose the mission into tasks. For our example scenario, Fig. 8 illustrates the decomposition of the mission into tasks that follows the CJTFG's general outline for the mission (his initial strategy), but expands the number of the tasks to adequately represent the information about the mission environment and organization's resources (e.g., the constraints on the platforms' motion, sensors, weapons, etc.).

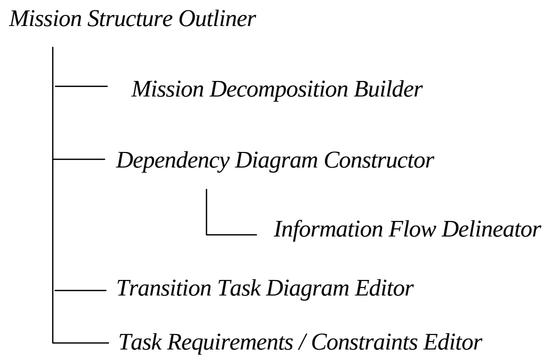
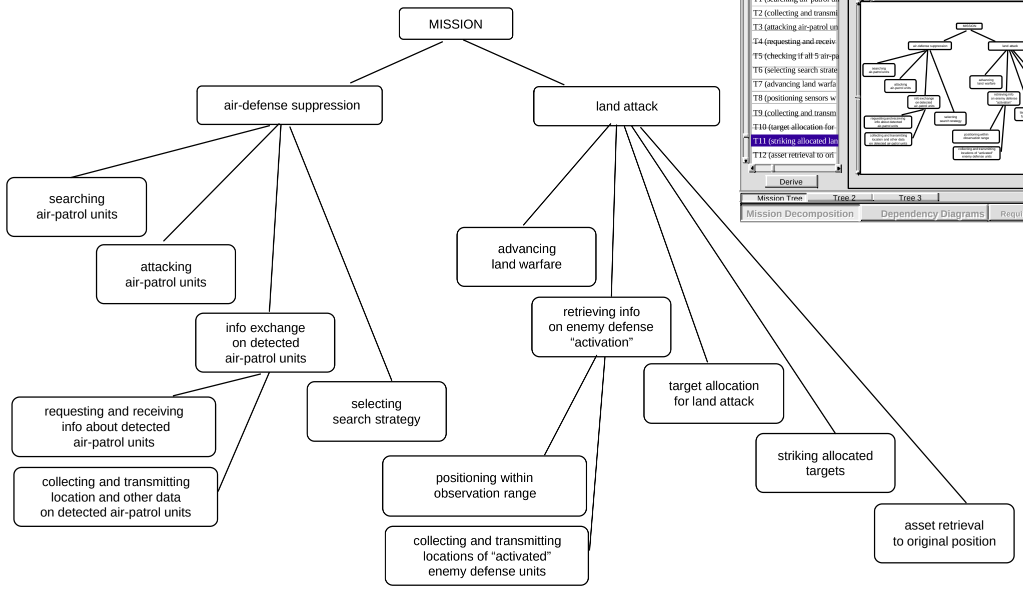
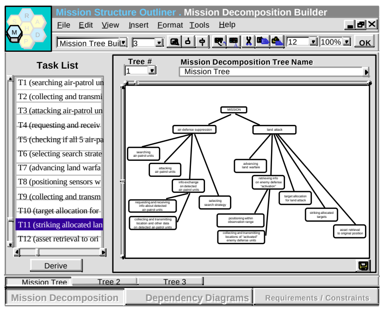

Mission Structure Outliner
Mission Decomposition Builder
Dependency Diagram Constructor
Information Flow Delineator
Transition Task Diagram Editor
Task Requirements / Constraints Editor

Fig. 7. Composition of *Mission Structure Outliner* component.


MISSION
air-defense suppression
land attack
searching air-patrol units
attacking air-patrol units
info exchange on detected air-patrol units
selecting search strategy
requesting and receiving info about detected air-patrol units
collecting and transmitting location and other data on detected air-patrol units
advancing land warfare
retrieving info on enemy defense "activation"
target allocation for land attack
striking allocated targets
asset retrieval to original position
positioning within observation range
collecting and transmitting locations of "activated" enemy defense units


Mission Structure Outliner . Mission Decomposition Builder
File Edit View Insert Format Tools Help
OK
Derive
Mission Tree
Tree 2
Tree 3
Mission Decomposition
Dependency Diagrams
Requirements / Constraints

Mission Tree Buil | 3 | 12 | 100%

Task List

T1 (searching air-patrol un
T2 (collecting and transmi
T3 (attacking air-patrol un
T4 (requesting and receiv
T5 (checking if all 5 air-pa
T6 (selecting search strate
T7 (advancing land warfa
T8 (positioning sensors w
T9 (collecting and transm
T10 (target allocation for
T11 (striking allocated lan
T12 (asset retrieval to ori

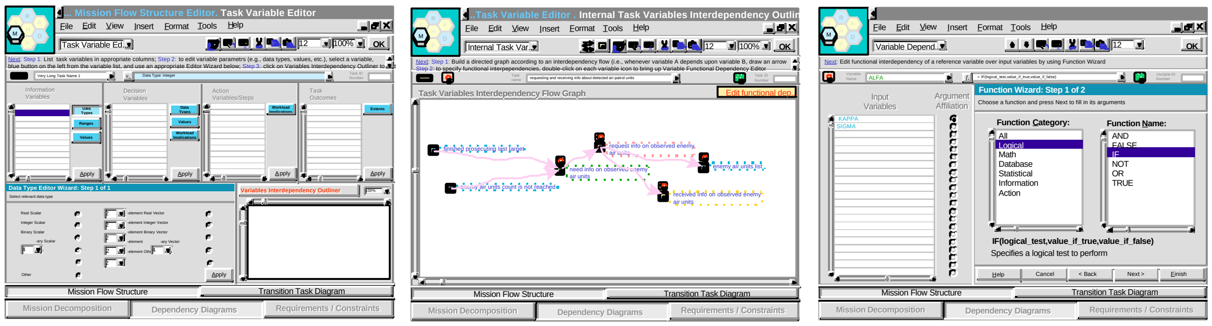
Tree #: 1
Mission Decomposition Tree Name: Mission Tree

Fig. 8. Mission Decomposition Diagram.

According to Fig.8, the task makeup of the mission has been defined as follows: T1 (searching air-patrol units); T2 (collecting and transmitting location and other data on detected air-patrol units); T3 (attacking air-patrol units); T4 (requesting and receiving info about detected air-patrol units); T5 (checking if all 5 air-patrol units are eliminated); T6 (selecting search strategy); T7 (advancing land warfare); T8 (positioning sensors within an observation range); T9 (collecting and transmitting locations of “activated” enemy defense units); T10 (target allocation for land attack); T11 (striking allocated land targets); T12 (asset retrieval to original position).

3.2.2 Quantifying Information Flow

Defining the task makeup of the mission allows a designer to partition the operational load of processing the mission among human DMs (by assigning different tasks to individual DMs). To assess the decision-making involved in a task execution, we associate with each task four basic types of variables: (i) *information variables*; (ii) *decision variables*; (iii) *action variables* (otherwise called *operations variables*); and (iv) *outcome variables*. The composition of these task variables delineates the structure of each task, as well as the associated decision-making and operational workload. In addition, the functional interdependencies among the task variables specify the *cause-effect information flow* among the mission tasks, stipulating the dynamics of the mission processing.


Mission Flow Structure Editor. Task Variable Editor
Task Variable Editor . Internal Task Variables Interdependency Outline
Task Variables Interdependency Flow Graph
Function Wizard: Step 1 of 2
Function Category:
Function Name:
Mission Flow Structure
Transition Task Diagram
Mission Decomposition
Dependency Diagrams
Requirements / Constraints

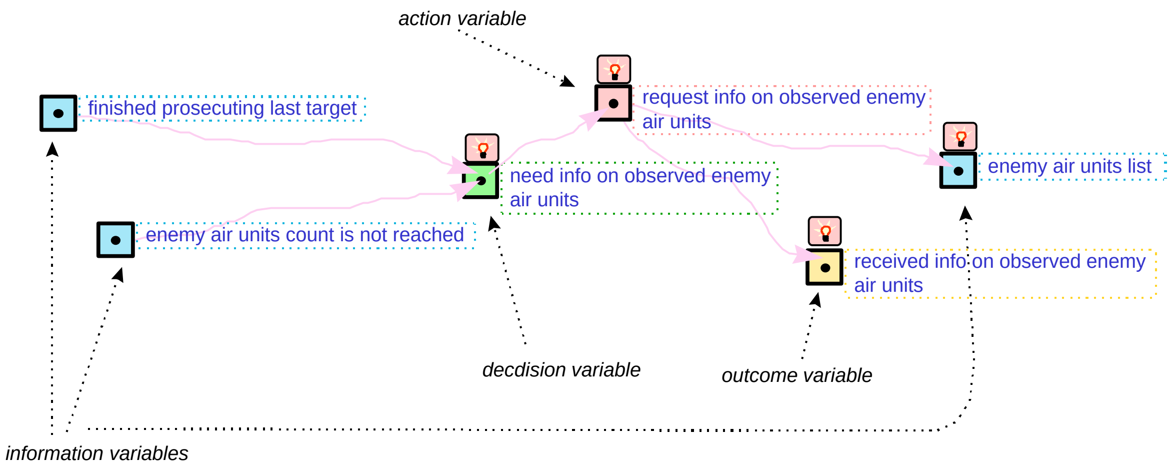

action variable
finished prosecuting last target
request info on observed enemy air units
need info on observed enemy air units
enemy air units list
enemy air units count is not reached
received info on observed enemy air units
decdision variable
outcome variable
information variables

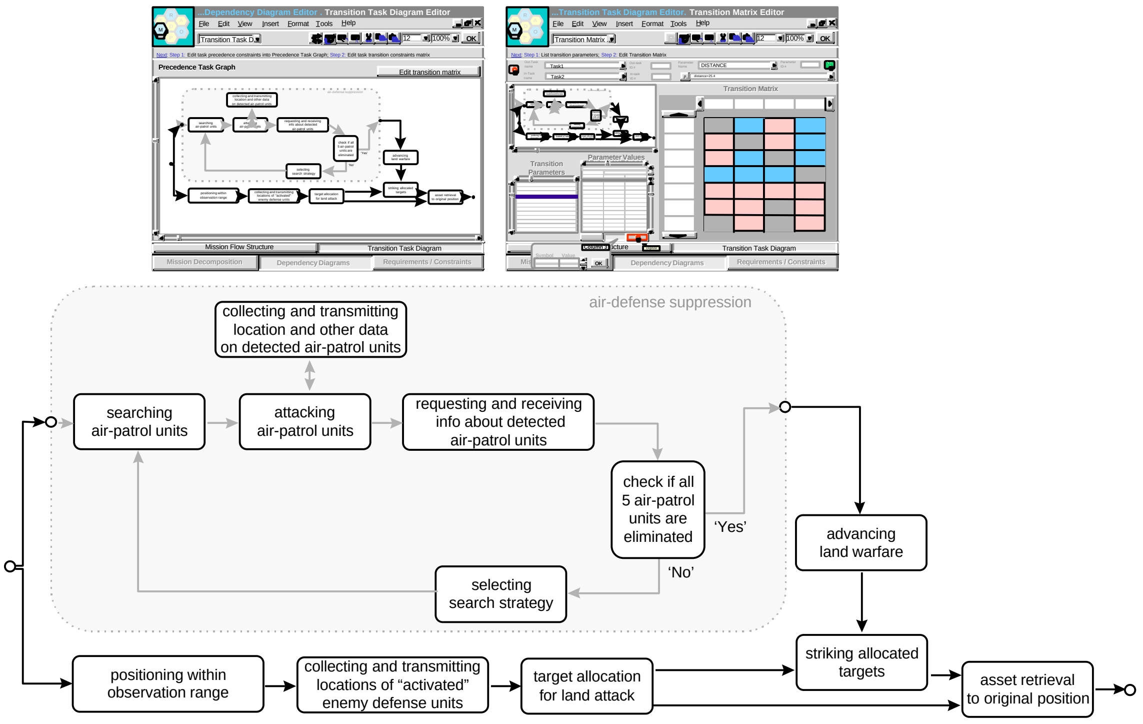
Fig. 9. Task Variables Interdependency for Task T4.

The *Dependency Diagram Constructor* subcomponent allows the user to define, for each task, the appropriate information, decision, action, and outcome variables, as well as to delimit, for each variable, its data content (delineated in variable's data types and its values/ranges, edited via appropriate *Editor Wizards*). For our example scenario, Fig. 9 illustrates the composition of the task variables for the task T4 (requesting and receiving info about detected air-patrol units).

The *Information Flow Delineator* subcomponent allows the user to depict the cause-effect information flow among tasks by specifying interdependencies among variables associated with different tasks (in a similar way as in Fig. 9).

3.2.3 Precedence Task Graph and Task Transition Parameters

Our tool box allows the user to add additional precedence constraints on task processing to those prescribed by information flow among the mission tasks. E.g., the constraints that are justified by operational planning but are not explicit in cause-effect structure among the mission tasks.



air-defense suppression
collecting and transmitting location and other data on detected air-patrol units
searching air-patrol units
attacking air-patrol units
requesting and receiving info about detected air-patrol units
check if all 5 air-patrol units are eliminated
'Yes'
'No'
selecting search strategy
advancing land warfare
striking allocated targets
positioning within observation range
collecting and transmitting locations of "activated" enemy defense units
target allocation for land attack
asset retrieval to original position

Fig. 10. Mission Precedence Task Graph.

The *Transition Task Diagram Editor* subcomponent allows the user to add the additional precedence constraints to those prescribed by cause-effect task structure, as well as to edit the corresponding task transition parameters. For our example, Fig. 10 depicts the mission task graph specified by *Transition Task Diagram Editor* (also used to state the distances among the mission tasks).

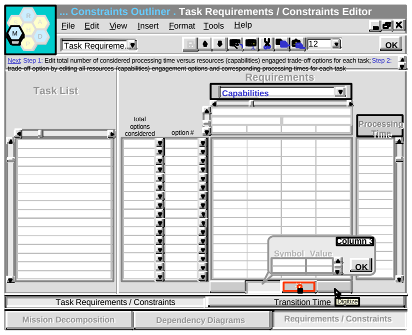
3.2.4 Task Requirements

While every successful task completion requires that a certain minimum amount of resources/capabilities/expertise be allocated to this task, the allocation of a larger amount of resources/capabilities/expertise generally facilitates the task processing and, thus, it may affect the task outcome (e.g., allocation of a large number of platforms for searching air-patrol units accelerates the completion of this task). I.e., a certain trade-offs exists, in general, between the amount of resources/capabilities/expertise allocated and the values of the task outcome parameters/variables.

The *Task Requirements / Constraints Editor* subcomponent allows the user to specify, for each task, the impacts of allocating different amount of resources/capabilities/expertise in terms of the corresponding trade-offs. For our example scenario, Fig. 11 illustrates editing the above trade-offs between the amount of the resources allocated and the completion time for the task T1 (searching air-patrol units).

3.3 *DM Structure Profiler*: Modeling Human Organization

We model large-scale organizations involving humans, machines, sensors, and computers, as hierarchical multi-channel decision networks that allow human decision-makers to coordinate their actions in order to achieve their common goal(s). The feasible allocations of the *information*, *resources*, and *activities* among DMs in such an organization are constrained by the DM's *capabilities* and/or DM's *expertise* makeup (although these terms are sometimes used interchangeably, our model regards each of them as a separate entity, as it reserves the term *expertise* to describe one's ability to perform information processing, decision-making, and operational functions, while it defines the DM's *capabilities* as a function of his resources). The *DM Structure Profiler* component of our software tool box assists the user in modeling such hierarchical networks of human DMs by specifying the constraints on a decision-making structure of the organization.



Next: Step 1: Edit total number of considered processing time versus resources (capabilities) engaged trade-off options for each task; Step 2: trade-off option by editing all resources (capabilities) engagement options and corresponding processing times for each task

		Air-Det.	AA-Strk.	Com	Info-Pr.	D.Anal.	Gr.-Det.	AG-Strk.	GG-L.Strk.	GG-H.Strk.	Fst.-Mov.	Sl.-Mov.	completion time
T1 (searching air-patrol units)	option 1	1	0	1	1	1	0	0	0	0	1	0	10 min
	option2	2	0	1	1	1	0	0	0	0	2	0	6 min

Fig. 11. Trade-off between task requirements and task completion time for Task T1.

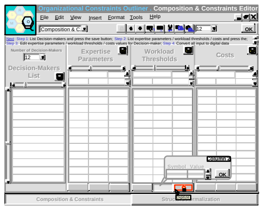
Example (continued). In our hypothetical mission scenario, we assume that there are three commanders (including CJTFG) that share the responsibility to direct the friendly platforms (each of the friendly platforms has an auto-pilot that can operate the platform according to the commands he receives from a human DM). The DMs have different expertise in information processing and managing the platforms.

3.3.1 Modeling a Human Decision-Maker

For each human DM in our organization, we must guarantee that decision-making and operational load on each DM remains below the corresponding thresholds. The *DM Profiler* subcomponent allows the user to define the constraints on the number of decision-makers, their expertise, and their workload thresholds. For our example scenario, Fig. 12 illustrates some of the above constraints.

3.3.2 Information Management

The key assumption behind the distributed information processing and decision-making is the ability of DM network to share and exchange data. This ability depends on two key dimensions of organizational structure: (i) the *information access structure* (that specifies the allocation of sensor displays, as well as the amount of displayed data); (ii) the *communication structure* (that specifies the allocation of different channels and data links, as well as their capacity for transferring data).



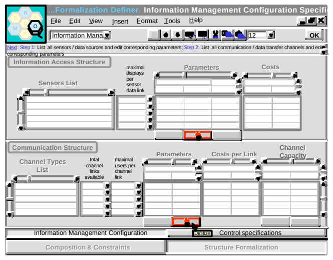
	Expertise Parameters		Workload Thresholds	
	Land Warfare	Aircraft	Info Processing	Operations Workload
CJTFG	7	7	10/min	5/min
DM2	6	2	10/min	6/min
DM3	3	7	12/min	8/min

Fig. 12. Expertise and threshold trade-offs constraints on DMs.

The *Information Management Configuration* subcomponent allows the user to specify the constraints on the information access structure and the communication structure. For our example scenario, Fig. 13 illustrates these constraints.

3.3.3 Authority Structure and Control

In addition to his operational and decision-making workload from executing mission tasks, DMs play another important role in a hierarchical decision networks. A specific positioning of a DM in a decision hierarchy of an organization assigns to such a DM an authority and an additional responsibility to control his subordinates. Different organizations can exercise different degrees as to the authority and responsibility of their DMs (ranging from a near-anarchy in some parts of the scientific community and soft coordination hierarchy of business enterprises to a disciplined command and control structures for military organizations).


Formalization Definer: Information Management Configuration Specifi
File Edit View Insert Format Tools Help
Information Access Structure
Sensors List
Parameters
Costs
Communication Structure
Channel Types List
Costs per Link
Channel Capacity
Information Management Configuration
Control specifications
Composition & Constraints
Structure Formalization

Information Access Structure

Sensors List	maximal displays per sensor data link	Parameters	
		range	resolution
Mobile radar	1	20	200
Avenger's radar	1	10	500
Hawk's radar	2	15	400

Communication Structure

Channels	channel links	max users per channel link	Channel Capacity
"Blue"	2	2	2000/sec
"Red"	1	2	5000/sec

Fig. 13. Information Access and Communication Constraints.

The *Control Delineator* component allows the user to identify the control variables for the mission tasks, as well as to integrate the control functionality of the DM hierarchy into the overall mission makeup (by stating the interdependencies among previously defined task variables and new control variables and by correspondingly updating the information flow structure). Oftentimes, some of the previously defined information and task outcome variables can serve as the control variables, thus allowing to minimize changes in the mission structure due to added control responsibilities in an organization. For our example, we assume that such is the case (i.e., all control variables are either information and task outcome variables). The control functions have been already included into the mission action task variables (e.g., those are decisions to either continue or abort the mission).

3.4 Performance Measures and Objection Function

In general, more than a single criterion is considered to judge organizational performance. Furthermore, the notion of optimality is subjective (since different people may regard different performance criteria as dominant in judging the design optimality). The *exact specification* of the lexical ordering and/or the relative weights of prospective optimization criteria allow one to reflect the *advocated bias* toward assessing the organizational performance. The performance criteria are expressed via a multi-variable objective function to build an organization whose structural and process parameters optimize the objective function.

For our hypothetical example, we use the *Performance Criteria Architect* component of our software environment to define (quantify) the following parameters:

- mission processing time;
- DM's decision-making workload;
- DM's operational workload;
- DM's information load;
- DM's control workload;
- inter-DM communication;
- inter-DM coordination.

4. Multi-Objective Scheduling Algorithm.

The problem of finding the optimal resource-to-task allocation arises in many different areas of science. E.g., it is one of the key problems for multiprocessor system design, transportation engineering, management science, manufacturing and production systems engineering, and so on. While scheduling is primarily concerned with allocating limited resources to tasks over time, different objective functions can be considered as the criteria for the schedule optimality.

Generally, the order of tasks processing is restricted by the precedence relations among the mission tasks. The constraints on the task precedence are modeled via a directed acyclic graph (DAG), and the time requirements for each task can be represented by assigning the corresponding weights to DAG's nodes.

In general, a platform needs some extra time when switching from processing of one mission task to another, so the appropriate adjustments (termed time-transitions or, simply, transitions) must be taken into account when assigning platforms to tasks. Transitions can be modeled by specifying a transition matrix (e.g., in a multiple travelling salesmen problem, MTSP, where tasks represent cities and where transitions are functions of the distances between the cities, as well as of the “speed” of each salesman, transitions are modeled via the inter-task distance matrix that is "scaled" differently for each platform).

Next, we present a heuristic algorithm that generates the near-optimal platform-to-task allocation (PTA). While our primary objective is to minimize the makespan, i.e., to minimize the total completion time of the mission, other criteria can be considered when generating the task platform schedule. Based on the adopted performance criteria, a *performance coefficient* is evaluated throughout the algorithm to rank the platform-to-task assignments. The algorithm belongs to a class of the priority list scheduling algorithms. One of its favorable features is that it can be used dynamically (i.e., it can generate the allocation schedule on-line, and, thus, it can be used for schedule adaptation; as is the case, for example, with the travelling salesmen – travelling cities problem, described in [Levchuk *et al*., 1998]).

4.1 Problem formulation

4.1.1 Notations and definitions for input variables.

N – number of tasks;

K – number of platforms;

t(i) – time to complete task T_i;

trans(i,j) – transition time between tasks T_i and T_j;

t_out(i) – list of immediate predecessors of a task T_i in DAG;

t_in(i) – list of immediate successors of a task T_i in DAG;

4.1.2 Intermediate variables used in the algorithm.

start(i) – time to start processing task T_i;

path(i) – the length of the maximal path from task T_i to the end task in DAG (to the finish of the mission) - critical path (CP);

x(i,j,k) = 1 if platform P_k is scheduled to complete tasks T_i and T_j consecutively;

free(k) – time at which platform P_k is freed to process another task (is updated); it is equal to the total time needed for platform P_k to finish all tasks assigned to it up to current time - a dynamically updated variable;

ready_tasks – a set of tasks that can be scheduled at current time; i.e., they include those tasks all predecessors of which are already completed;

last_task(k) – the last task that was processed by platform P_k (=0 if no such task); - a dynamically updated variable;

n_out(i), n_in(i) – number of successors and predecessors of task T_i in DAG;

best_task, best_platforms – task and platform (group of platforms) chosen for assignment at each step;

insert_task – task chosen for insertion.

The algorithm explores the *nearest-neighbor structure* among the tasks (i.e., the "localized" task precedence relations analysis) to evaluate the corresponding task transitions and to generate an allocation of tasks to platforms that optimizes the corresponding criteria (e.g., the one that minimizes the total span of time, termed "makespan", needed for platforms to complete all tasks).

4.2 PTA Algorithm.

The algorithm assigns tasks to platforms (groups of platforms) in a step-by-step manner. At each step *performance coefficients* are computed to evaluate the allocation of task to a platform and to generate a priority list. The allocation with *largest coefficient* is given the *highest priority* when allocating platforms to tasks (and the corresponding task is assigned to the corresponding platform, or group of platforms).

PTA algorithm.

Input: N tasks with processing times, as well as transitions and precedence constraints (DAG, t_out, t_in, t, N, K). For each node – a list of platform groups capable of completing this task.

Output: A platform-task schedule, together with the corresponding *allocation matrix* and *Gannt_chart*, indicating the starting times to process each task and the total mission completion time.

Initialization:

ready_task={i, if n_in(i)=0}; start(i)=0; x(i,j,k)=0; path(i)=0;

compute *n_in(i)*=|*t_in(i)*| and *n_out(i)*=|*t_out(i)*| for each task i;

Critical paths calculation:

For each i calculate *path(i) = length of the critical path* (minimal path to the end of DAG).

Coefficients calculation and assignment selection:

Compute coefficients *coeff(group,i)* for each task i and each group of platforms that can process it.

A task and a group of platforms corresponding to the maximal coefficient – **best_coeff** - are chosen for assignment. The task is denoted as **best_task** and a group of platforms - **best_platforms**.

Start time update:

start(best_task)=path(best_task)-best_coeff;

"Insertion":

Using insertion procedure described below, check if insertion of a task is possible (*state='insertion'*) or not (*state="no insertion"*);

if state="insertion', choose the largest coefficient (**insert_coeff**) and corresponding task (**insert_task**) and P_group (**insert_group**). Go to step 7.. Else go to step 8;

Insertion Assignment:

output variables update: for each m∈insert_group,

x(last_task(m),insert_task, m)=1; x(insert_task,best_task, m)=1;

start(insert_task)=path(insert(task)-insert_coeff; free(m)=start(best_task)+t(best_task);

ready_tasks update:

for each j∈t_out(insert_task)∪y_out(best_task) n_in(j)=n_in(j)-1;

if n_in(j)=o, then add *j* to *ready_tasks*.

No-insertion Assignment:

output variables update: for each m∈best_tasks,

x(last_task(best_platform),best_task,m)=1; free(m)=start(best_task)+t(best_task);

ready_tasks update:

for each *j Î t_out(best_task) n_in(j)=n_in(j)-1*;

if *n_in(j)=0*, then add *j* to *ready_tasks*.

If *ready_tasks=Æ*, then END and report assignment matrix x(, ,) and start times s( );

Otherwise go to Step 3.

Critical paths calculation:

initialize a set Task_List={i, if n_out(i)=0}; for each i∈Task_List, path(i)=t(i);

pick j∈Task_List and remove it from this set;

for each l∈t_in(j), n_out(l)=n_out(l)-1; path(l)=max(path(l),t(l)+path(j))

if n_out(l)=0, then add l to Task_List;

if Task_List=∅, END and report path(i) i=1,..,N; otherwise repeat step b);

(Performance) Coefficients calculation:

For each task i∈*ready_tasks* set and each platform k=1,..,K find the coefficients **coeff(k,i)=path(i)-max(start(i),free(k)+trans(last(k),i))**

For each group of platforms that can process the task the coefficient *coeff(group,task)* is found as a maximum of the coefficients corresponding to the platforms from that group.

Insertion:

initialize T_feasible=ready_tasks∩{i: i can be completed by best_platforms}; state='no_insertion'; before_time=max{free(m), m∈best_platforms}; after_time=start(best_task);

for each i∈T_feasible repeat step c);

for each P_group⊂best_platforms such that P_group can process task T_i repeat step d)

if max{max(trans(last_task(k),i)+free(k),start(i))+t(i)+trans(i,best_task) | k∈P_group}$\leq$ start(best_task)

then compute coefficient corresponding to assigning i to P_group as in Step 3. State='insertion'.

5. Organizational Architecture.

After the *Performance Criteria Architect* component is used to stipulate performance measures for the prospective organizational design and to define a performance cost function, the last three components of our software environment (*Schedule Generator*, *Allocation Definer*, and *Hierarchy Designator*) allow the user to perform a step-by-step design of the organizational structure, while implementing, if desired, the user-defined design modifications at various stages of the design process to adjust the displayed metrics of organizational performance, attainment of mission objectives, and the workload distribution across the organization.

The *Schedule Generator* component produces the task-resource allocation that corresponds to Phase I of our organizational design algorithm, while *Allocation Definer* and *Hierarchy Designator* allow the user to iteratively complete Phases II through V of the design process.

An output organizational structure prescribes the relationships among the organizational entities by specifying:

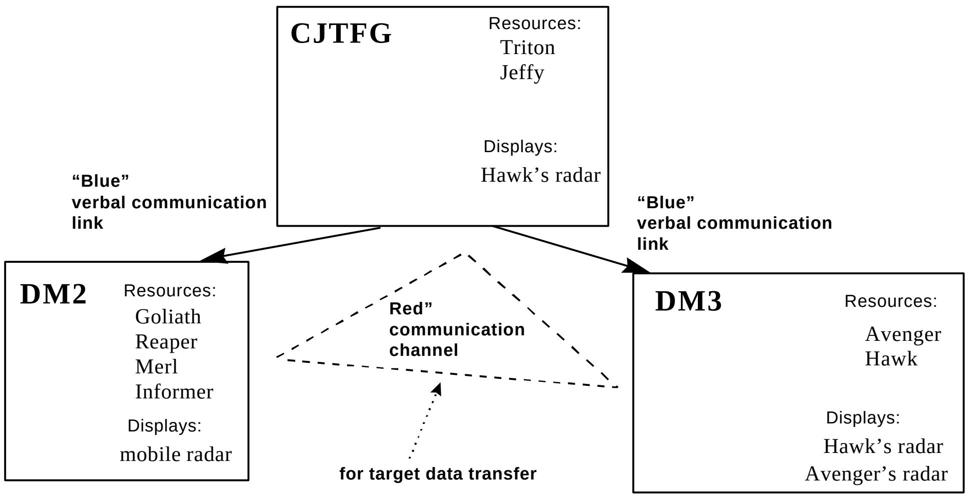


Fig. 13. Organizational Structure.

- DM-resource access/allocation;
- DM command hierarchy;
- inter-DM communication network.

It defines each individual DM's *capabilities* (by assigning each DM a share of the information and resources) and specifies the rules that regulate inter-DM *coordination*. The organizational structure, together with a set of thresholds constraining the DM workload, determines the boundaries of a feasible strategy space (e.g., all feasible DM-task-resource assignments), from which the organization can choose a particular strategy. The feasible strategy space delimits the strategy adjustments that an organization can undertake without structural reconfiguration. For our example, Fig. 13 depicts the organizational structure generated, while Fig. 14 stipulates the DM-to-task allocation.

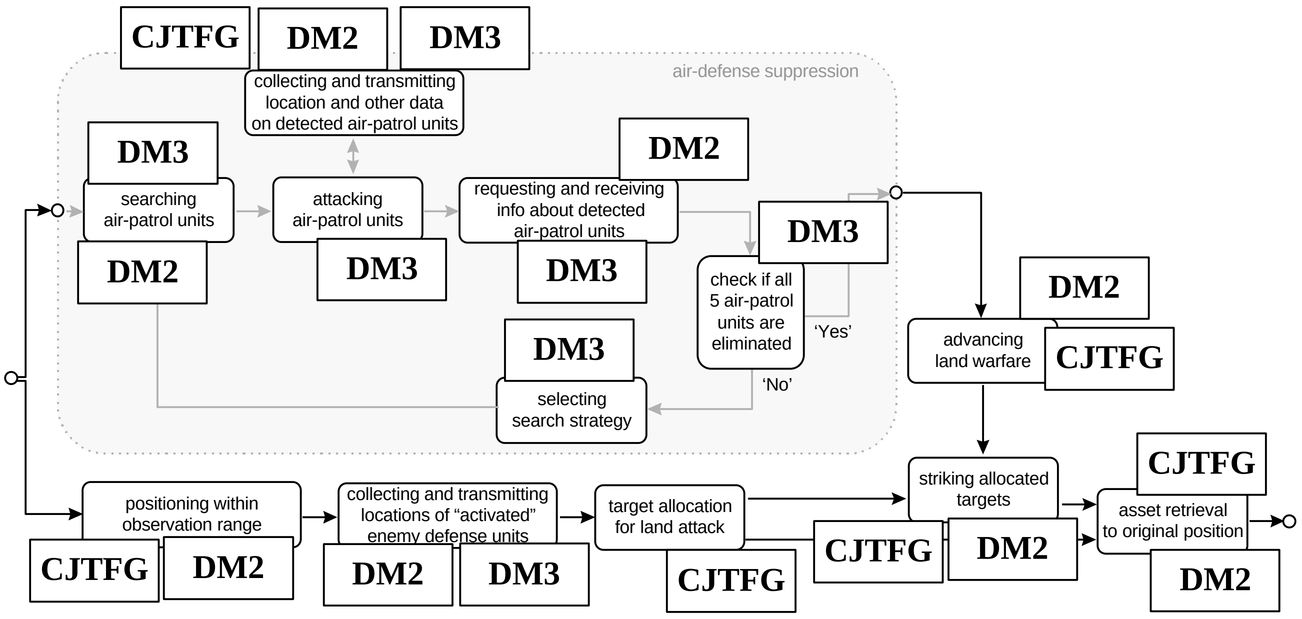

CJTFG
DM2
DM3
air-defense suppression
collecting and transmitting location and other data on detected air-patrol units
DM3
searching air-patrol units
DM2
attacking air-patrol units
DM3
requesting and receiving info about detected air-patrol units
DM2
DM3
DM3
check if all 5 air-patrol units are eliminated
'Yes'
'No'
DM3
selecting search strategy
DM2
advancing land warfare
CJTFG
positioning within observation range
CJTFG
DM2
collecting and transmitting locations of "activated" enemy defense units
DM2
DM3
target allocation for land attack
CJTFG
striking allocated targets
CJTFG
DM2
CJTFG
asset retrieval to original position
DM2

Fig. 14. Allocation of tasks to DMs.

6. Summary.

Our tool provides a *visualization* of the step-by-step organizational design process, enabling the user to conduct a comprehensive study of the mission and the concomitant organizational architectures.

7. References

[Burns, W.J., *et al.* 1993] Burns, W.J. and R.T. Clemen, "Covariance structure models and influence diagrams," *Manag. Sci.*, vol. 39, pp. 816-833, 1993.

[Carley and Lin , 1995] Carley, K.M. and Z. Lin,"Organizational designs suited to high performance under stress," *IEEE Trans. SMC.*, vol. 25, pp.221-231, 1995

[Carley, 1991] Carley, K.M., Designing Organizational Structures to Cope with Communication Breakdowns: A Simulation Model. *Industrial Crisis Quarterly*, Volume 5, 1991, pp.19-57.

[Chankong and Haimes, 1983] Chankong, V. and Y. Haimes, "Multiobjective Decision Making : Theory and Methodology," North-Holland, NY, 1983

[Cohon, 1978] Cohon, J.L. , "Multiobjective programming and planning", *Academic press*, NY, 1978

[Curry *et al.*, 1997] Curry M.L., K.R. Pattipati, and D.L. Kleinman, "Mission Modeling as a Driver for the Design and Analysis of Organizations," *Proceedings of the 1997 Command and Control Research and Technology Symposium*, Monterey, CA, June 1997.

[Deb et al. 1997] S. Deb, Pattipati, K.R., and R. Shrestha,"QSI integrated diagnostic tool set", IEEE, Autotest, CA, Sept., pp 408-421, 1997.

[Desrochers *et al*., 1992] Desrochers, M., J. Desrochers, and M. Solomon, *A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows,* Operns. Res. 40, 342-354, 1992.

[El-Rewini *et al*., 1994] El-Rewini, H., T.G. Lewis and H.H. Ali., *Task Scheduling in Parallel and Distributed Systems,* Prentice Hall, Englewood Cliffs, NJ, 1994.

[Hoffman and Padberg, 1993] Hoffman, K.L., and M. Padberg, *Solving Airline Crew Scheduling problems for Brunch-and-Cut,* Mgmt. Sci., vol. 39, pp. 657-682.

[Kemple *et al*., 1996a] Kemple, W.G., D.L. Kleinman and M.C. Berigan, "A2C2 Experiment: Adaptation of the Joint Scenario and Formalization," *Proc. 1996 Command and Control Research and Technology Symposium*, Monterey, CA, 1996.

[Kemple *et al*., 1996b] Kemple, W.G., S.G. Hutchins, D.L. Kleinman, K. Sengupta, M.C. Berigan and N.A. Smith, "Early Experiences with Experimentation on Dynamic Organizational Structures," *Proc. 1996 Command and Control Research and Technology Symposium*, Monterey, CA, 1996.

[Kemple *et al*., 1997] Kemple, W.G., Drake J., D.L. Kleinman, E.E. Entin , D. Serfaty, “Experimental Evaluation of Alternative and Adaptive Architectures in Command and Control”, *Proceedings of the 1997 Command and Control Research and Technology Symposium*, Washington, DC, June 1997.

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

[Levchuk *et al*., 1996] Levchuk, Y.N., K.R. Pattipati, M.L. Curry and M. Shakeri, “Design of congruent organizational structures : Theory and algorithms,” *Proceedings of the 1996 Command and Control Research and Technology Symposium*, Monterey, CA, June 1996.

[Levchuk *et al*., 1997] Levchuk, Y.N., K.R. Pattipati, and M.L. Curry, “Normative Design of Organizations to Solve a Complex mossion : Theory and Algorithms,” *Proceedings of the 1997 Command and Control Research and Technology Symposium*, Washington, DC, June 1997.

[Lin & Carley, 1995] K.M. Carley and Z. Lin, “Organizational design suited to high performance under stress”, IEEE Transactions on Systems, Man, and Cybernetics, Volume 25, 1995, pp.221-231.

[Papastavrou and Athans 1992] J.D. Papastavrou and M. Athans, “On Optimal Distributed Detection Architectures in a Hypothesis Testing Environment’, IEEE Transactions on Automatic Control, Volume 37, 1992, pp.1154-1169.

[Pattipati *et al*., 1994] Pattipati, K.R., V. Rogovan, M. Shakery, S. Deb, and R. Shrestha ,” TEAMS: Testability Engineering and Maintenance System”, Invited Paper at 1994 American Control Conference, Baltimore, MD, May 1994, pp. 1989-1996.

[Pattipati *et al*., 1996] Pattipati, K.R., A. Pete, Y.N. Levchuk and D. L. Kleinman, "Decision networks and organizations," Invited chapter in *the Encyclopedia of Life Support Systems*, 1996.

[Pete *et al*. 1993] A. Pete, K. R. Pattipati, and D. L. Kleinman,”*Distributed detection in teams with partial information: A normative-descriptive model”, IEEE Trans. Syst., Man, Cybern.*, 23:1626-1648, 1993.

[Pete *et al*. 1994] Pete, A. , D. L. Kleinman, and K. R. Pattipati. Structural congruence of tasks and organizations. *Proceedings of the 1994 Symp. on Command and Control Research and Decision Aids,* pages 168-175, NPS, Monterey, CA, 1994.

[Pete *et al*., 1995a] Pete, A., D. L. Kleinman, and K.R. Pattipati, "Designing organizations with congruent structures," *Proceedings of the 1995 Symposium on Command and Control Research and Technology*, Washington, DC, 1995.

[Pete *et al*., 1995b] Pete, A., K. R. Pattipati, and D. L. Kleinman, "Congruent organizational structures and subtask assignment," Submitted for publication to: *IEEE Trans. Syst., Man, Cybern.*, 1995.

[Pete *et al*., 1995c] Pete, A., K. R. Pattipati, and D. L. Kleinman, "Structural reconfiguration and informal coordination in administrative organizations," Computational and Mathematical Organization Theory, Netherlands, 1995.

[Pete *et al*., 1996] Pete, A., K. R. Pattipati, and D. L. Kleinman, "Optimization of decision networks in structured task environments," *IEEE Trans. Syst., Man, Cybern.,* Nov. 1996.

[Pete *et al*., 1995d] Pete, A., D. L. Kleinman, and P. W. Young, "Organizational performance of human teams in a structured task environment," *Proceedings of the 1995 Symposium on Command and Control Research and Technology,* Washington, DC, 1995.

[Pete *et al.*, 1998] Pete, A., K. R. Pattipati, D. L. Kleinman, and Y.N. Levchuk, "An Overview of Decision Networks and Organizations", *IEEE Trans. Syst., Man, Cybern.,* May. 1998.

[Perdu *et al*., 1997] Perdu D. M. and A.H. Levis, A methodology for Adaptive Command and Control Teams Design and Evaluation, *Proceedings of the 1997 Command and Control Research and Technology Symposium*, Washington, DC, June 1997.

[Reibman and Nolte 1987] A. Reibman and L.W. Nolte, "Design and performance comparison of distributed detection networks," IEEE Trans. Aerosp. and Electr. Syst., vol. 23, pp. 789-79, November, 1987.

[Shachter, 1986] Shachter, R. D. , "Evaluating influence diagrams," *Oper. Res*., vol. 34, pp. 871-882, 1986.

[Shachter, 1988] Shachter, R. D. , "Probabilistic inference and influence diagrams," *Oper. Res.*, vol. 36, pp. 589-604, 1988.

[Shachter and Kenley, 1989] Shachter, R. D. and C.R. Kenley, "Gaussian influence diagrams," *Manag. Sci.,* vol. 35, pp. 527-550, 1989.

[Tang *et al*., 1993] Z.B. Tang, Pattipati, K. R. and

Kleinman, D.L.,"*Optimization of Distributed Detection Networks: Part II. Generalized Tree Structures",* IEEE Transactions on Systems, Man and Cybernetics , vol. 23, pp. 211-221, 1993.

[Wierzbicki, 1980] Wierzbicki, A. P.,"The use of reference objectives in multiobjective optimization," In Fandel and Gal, editors, *Multiple Criteria Decision Making, Theory and Applications*. Springer Verlag, New York, NY, 1980

[Ying *et al*., 1997] Ying Jie, Y.N. Levchuk, M.L. Curry, K.R. Pattipati, and D. L. Kleinman, Multi-Functional Flow Graphs: A New Approach to Mission Monitoring, *Proceedings of the 1997 Command and Control Research and Technology Symposium*, Washington, DC, June 1997.

[Zeleny, 1984] Zeleny, M. ,*MCDM - Past decade and future trends*, JAI Press, Greenwich, CT, 1984.

Marius M. Solomon. *Time window constraints routing and scheduling problems.* Transportation science, Vol. 22, No. 1, 1988.

T.C.E. Cheng, C.C.S. Sin. *A state-of-the-art review of parallel-machine scheduling research.* European J. of Op. Res. 47, 271-292, 1990.

Behrooz Shirazi, Mingfang Wang, Girish Pathak. *Analysis and evaluation of Heuristic methods for static task scheduling.* J. of parallel and distributed computing 10, 222-232, 1990.

Lap Mui Ann Chan, Ana Muriel, David Simchi-Levi. *Parallel machine scheduling, linear programming, and parameter list scheduling heuristics.* Op. Res. 46, No. 5, Sept-Oct 1998

Sanjoy K. Baruah. *The multiprocessor scheduling of precedence-constrained task systems in the presence of interprocessor communication delays.* Op.Res. 46, No. 1, Jan-Feb 1998.

Marshall L. Fisher, Kurt O. Jornsten, Oli B.G. Madsen. *Vehicle routing with time windows: two optimization algorithms.* Op. Res., Vol. 45, No. 3, May-June 1997

M.L. Fisher. *Optimal solution of VRPs using minimum K-trees.* Op. Res. 42, 626-642, 1994.

S.L. Van de Velde. Duality-based algorithms foir scheduling unrelated parallel machines. ORSA J. on Comp., Vol. 5, No. 2, spring, 1993.

Marco Spuri, John A. Stankovic. How to integrate precedence constraints and shared resources in real-time scheduling. IEEE trans. on comp., vol. 43, No. 12, december 1994.