1995-03

A computer simulation of logistics networks for wargame umpires

Plunk, Curtis DeWayne

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

A COMPUTER SIMULATION OF LOGISTICS NETWORKS FOR WARGAME UMPIRES

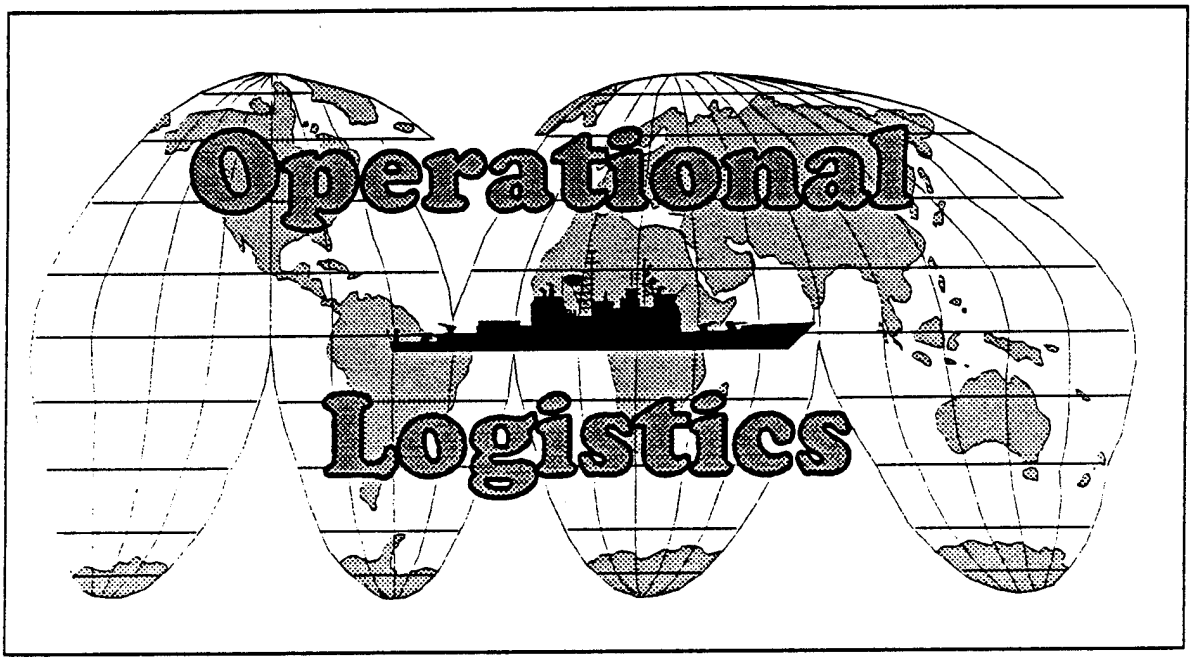
by

Curtis DeWayne Plunk

March, 1995

Thesis Advisor: David A. Schrady

Approved for public release; distribution is unlimited.


Operational
Logistics

REPORT DOCUMENTATION PAGE

Form approved OMB No. 0704-188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1995	Master's Thesis

4. TITLE AND SUBTITLE
A COMPUTER SIMULATION OF LOGISTICS NETWORKS FOR WARGAME UMPIRES

5. FUNDING NUMBERS

6. AUTHOR(S)

Plunk, Curtis D.

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

Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

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

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

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

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

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Logistics Wargaming Simulation (LogWarS) is a computer program designed to facilitate the incorporation of logistics considerations and constraints into wargames played at the Wargaming Department of the Naval War College. LogWarS helps a wargame umpire create a scenario of units and bases, their supply requirements, and the transportation assets available to move supplies. Once the scenario is created, LogWarS steps it forward in time, allowing the umpire to examine the supply status of the units and bases at various points in the future. The umpire uses the supply status to inform the wargame players of their ability to conduct operations. The umpire may also modify elements of the scenario, at any time step, in order to reflect events in the wargame—events such as the destruction of supply depots or the addition of transportation assets. LogWarS adds a graphical interface and new ways to follow the supply status of units and bases to the original version of the Surge and Sustainment Simulation program created by two other Naval Postgraduate School students.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Logistics Models, Computer Simulations, Logistics Networks, Wargames, Surge and Sustainment	83
	16. PRICE CODE

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

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std 239-18

Approved for public release; distribution is unlimited.

A COMPUTER SIMULATION OF LOGISTICS NETWORKS FOR WARGAME UMPIRES

Curtis DeWayne Plunk
Lieutenant, United States Navy
B.S., University of Notre Dame, 1986
B.A., University of Notre Dame, 1986

Submitted in partial fulfillment of the requirements for
the degree of
MASTER OF SCIENCE IN OPERATIONS RESEARCH
from the
NAVAL POSTGRADUATE SCHOOL
March, 1995

Author: [redacted]
Curtis DeWayne Plunk

Approved by: [redacted]
David A. Schrady, Thesis Advisor

[redacted]
Thomas E. Halwachs, Second Reader

[redacted]
Peter Purdue, Chairman, Department of Operations Analysis

ABSTRACT

The Logistics Wargaming Simulation (LogWarS) is a computer program designed to facilitate the incorporation of logistics considerations and constraints into wargames played at the Wargaming Department of the Naval War College.

LogWarS helps a wargame umpire create a scenario of units and bases, their supply requirements, and the transportation assets available to move supplies. Once the scenario is created, LogWarS steps it forward in time, allowing the umpire to examine the supply status of the units and bases at various points in the future. The umpire uses the supply status to inform the wargame players of their ability to conduct operations. The umpire may also modify elements of the scenario, at any time step, in order to reflect events in the wargame—events such as the destruction of supply depots or the addition of transportation assets.

LogWarS adds a graphical interface and new ways to follow the supply status of units and bases to the original version of the Surge and Sustainment Simulation program created by two other Naval Postgraduate School students.

THESIS DISCLAIMER

The reader is cautioned that the computer code developed for this thesis has been completed within a limited period of time and has not been fully tested. Although a strong effort was made to ensure the correctness of the programming, the code still needs to be verified and validated. Any results or conclusions brought forth from the use of this program are done so at the user's risk.

TABLE OF CONTENTS

LIST OF FIGURES

EXECUTIVE SUMMARY

Wargames are increasingly important in both the training of staffs and the examination of the feasibility of operational plans. A major shortcoming of current wargames is the ad–hoc manner in which they treat logistics. The success or failure in a conflict or of an operational plan will be largely determined by an adequate logistics system as much as by a solid combat plan. Historically logistics is largely ignored in wargame play because logistics is difficult to accurately model.

The Logistics Wargaming Simulation (LogWarS) provides the types and quantity of logistic information needed by wargame umpires as they perform their assigned function. There is currently no method for systematically and rigorously introducing logistics factors into wargames—especially in a manner in which time is a variable element. LogWarS provides an automated method for tracking and examining the flow of commodities through the logistics network established for each wargame. LogWarS takes various inputs and presents the resulting supply situation as time unfolds. A critical element of LogWarS is its ability to display the logistics system as time progresses. The wargame umpire is not merely presented with the *solution* to the problem which must either be accepted or rejected. LogWarS presents the umpire with the status of the system at a particular time which the umpire may alter. The program continues to the next specified time—at which point the umpire may again examine the situation and alter it as seems appropriate based upon experience or wargame requirements.

As an example of how LogWarS can be used in a wargame, consider the following scenario. The Blue player has recently deployed the 1st Armored Cavalry Regiment (1ACR) to a position on the Kuwaiti border. Before the deployment of forces is complete, the 1ACR is threatened by a battalion of the Iraqi Republican Guard. The Blue player indicates to the umpire the intention to begin offensive action against the Republican Guard unit. The umpire consults LogWarS and finds that the 1ACR has all its personnel and tracked vehicles but has not yet received its full allotment of ammunition. The ammunition shipment is enroute to the 1ACR by ground transport from the port at Kuwait City and will arrive in 12 hours. The umpire warns the Blue player of this situation and indicates that any immediate combat operations may be hindered by a lack of ammunition. With a full understanding of the status of the 1ACR's supplies, the Blue player elects to wait for the ammunition before commencing combat operations.

LogWarS is intended to be used by wargame umpires as a tool to aid in the realistic play and evaluation of wargames. LogWarS provides a basic level of logistic capability for umpires who may not otherwise be familiar with logistics issues or who do not have the time to support a detailed logistics feasibility model. LogWarS does not require the detailed information of other logistics feasibility programs. To fully support a wargame, LogWarS only requires a small amount of detail about the logistics network used in the wargame. LogWarS scenarios can range in size and complexity from squad level to global. Scenarios can simulate the logistics systems of Blue, Red, coalition, land, sea, or any other combination of forces. LogWarS is a generic tool—once the basic concepts are understood it can be used to simulate any logistics system. LogWarS helps the wargame umpire make assessments of the logistics situation as time elapses and so gives the player more realistic information upon which to base decisions.

LogWarS, an updated version of the Surge and Sustainment Simulation program (S3), uses a graphical interface. This interface allows examination of the logistics situation in more detail than the earlier version of the program. Usage of the S3 model has been limited due to the difficulty of using its menu interface. LogWarS corrects the difficulty and provides a much needed tool for adequately modeling logistics in wargaming.

I. BACKGROUND AND PROBLEM STATEMENT

A. LOGISTICS IN WARGAMES

Wargames have a long history dating back to ancient China. Sun Tzu create a game called *Wei Hai* about five thousand years ago. It is a board game where one player tries to encircle or outflank the other player. *Wei Hai* is a very generalized and simple version of a wargame, just as chess is an abstract wargame. A more recognizable form of wargame was not invented until the mid-seventeenth century. The *Koenigspiel* was invented in the German town of Ulm in 1664 and involved more pieces and a larger board than chess but was played along the same principles. Games of this sort came to be collectively known as *War Chess*. Individual pieces were still used to represent individual people in the game world. The concept of aggregating a large number of people and representing them with one piece did not occur until 1780 at which time the chess–type board was replaced with one depicting various types of terrain. [Ref. 1]

The incorporation of logistics into wargames is another matter entirely. The first game to address logistics was not created until 1797. [Ref. 1, p. 21] The game used a map of real terrain for the board and had pieces representing the logistical support of an eighteenth century army. The rules covered both how to fight the pieces and the effect of proper or improper logistical support on the combat troops. Wargames for the next century concerned themselves with the details of play and the outcomes of attacks and movement; logistics formed only a small part of the wargames.

At the end of the nineteenth century wargames started to take a larger perspective on war—they became more strategic and less tactical. In the years before World War I, wargames were used to test mobilization plans and other operational issues. The original Schlieffen Plan used by Germany in World War I was initially wargamed to determine its logistic feasibility. The logistics problems of the plan were noted and then dismissed. [Ref. 2] Prior to the outbreak of World War II, the Germans once again used wargames to study the role of logistics in warfare. By this time, wargames had become an accepted means of examining both the strategic and tactical issues of warfare. [Ref. 1]

The United States Navy used wargames extensively during the years between the World Wars. The wargames concentrated on naval actions against an aggressive Japan and included serious considerations of logistics requirements and constraints. Over several years of wargames the Navy came to realize the difficulties of war with Japan included "...defending the line of supply across the Pacific with naval forces which would not be

superior to Orange [Japan's] forces. [Ref. 3, p.15]" Wargames became tools to examine various strategies and tactics to defeat Japan. They succeeded to the extent that Admiral Chester Nimitz believed "...nothing that happened in the Pacific was strange or unexpected.... [Ref. 3, p. 7]"

After World War II, wargames were widely used to examine many different issues. The wargames included some devoted entirely to logistics. One example is MISSLOGS, a board game developed by the RAND Corporation for the U.S. Air Force. It allowed an ICBM squadron maintenance officer to wargame the effects of limited budgets and time when assigning priorities for maintenance actions in a missile squadron. The ultimate goal of the wargame was to keep as many missiles as possible ready at any given moment of time. The outbreak of nuclear war was a random event and success in the war was determined by the number of missiles available for launch. [Ref. 4]

In the mid 1970s, hobby wargaming finally took note of logistics and its influence on the course of a war. Logistics constraints were used in wargames about Napoleon's 1812 campaign, the Crusades, and various World War II campaigns. [Ref. 1] The wargames did not concentrate on logistics, but they did not ignore its influence either.

A recent example of a pure logistics wargame using a computer is PROLOG. The wargame is in use at the Naval Postgraduate School as part of the regular course work. In PROLOG the player is in command of a battle-group and must both fight the battle-group and tend to its resupply requirements. The program is tailored specifically for logistics at sea and uses a fixed battle-group in a fixed scenario.

Other recent logistics wargames include the Crisis Action Model (CAMS) produced for the Army War College. It is a computer program run on a desktop computer and provides the player with an understanding of the ability of a logistics network to surge forces for operations overseas. It can be used as a wargame or as a rough check of the feasibility of an OPLAN but because it lacks the ability to control the logistics system as time elapses, CAMS is not completely appropriate as a logistics support tool for a wargame.

B. CONTINUATION OF OTHER WORK

Wargames are increasingly being used to test concepts of operations and to train staffs in how to function in a realistic and stressful environment meant to simulate their wartime mission. Issues raised about the performance of a particular logistic concept of operations can be crucial to the overall concept of operations and the conduct of a wargame.

Logistics considerations, however, are rarely incorporated in wargames or the models used to support them. Logistics systems are often seen as too detailed and time consuming to model or, worse, too minor a factor to warrant true inclusion in a wargame. In a theater level wargame this can lead to highly inaccurate results and game conclusions. "[The players] must have data allowing them to estimate friendly and enemy capabilities [and] levels of supply..." [Ref. 1, p 203] This data forms a basis for the players' decisions in the wargame. The more realistic the data, the more realistic the decisions will be. The result will be a more informative wargame.

As a wargame progresses, the levels of supplies throughout the logistics network should change. The shortage or low supply level of various items adds pressure on the player as well as realism to the wargame. Continually full gas and ammunition stocks will lead players to skewed perceptions of reality. It will lead them to ignore certain critical aspects of the real world. Wargame umpires need tools to determine, even roughly, the supply status of the wargame forces at various points in time during the wargame.

One difficulty incorporating logistics in a wargame is that current logistics models are oriented more toward the analysis of the feasibility of OPLANs. They are not designed to be used as a wargame support tool. One reason is the detailed data inputs required. A feasibility model needs time-phased force deployment data (TPFDD) and while this leads to a good analysis of the feasibility of an OPLAN, gathering this detailed data is much too time–consuming for wargame purposes. Another reason the models are unsuitable as wargame support tools is the all-or-nothing nature of the models. The model's outputs are statistics about the feasibility of the plan. The model cannot be interrupted and examined or changed during execution of the plan. The modeler cannot stop time at a given day and alter the amount of strategic lift and then continue the model. Time is uncontrolled by the modeler, it is not a variable that can be manipulated in any way.

A further difficulty with incorporating logistics into wargames is the general unfamiliarity of the players and umpires with the full scope of a logistics system. They may understand logistics at a particular level but are unlikely to appreciate the full scope of the system. The amount of knowledge required to construct a wargame and the short time in which to do so hinders umpires who do not have either a thorough understanding of logistics or a reliable tool for modeling a logistics system.

Long [Ref. 5] and Halvorson [Ref. 6] began their work on Surge and Sustainment Simulation program (S3) attempting to create a computer program capable of simulating any logistics network in both the surge and sustainment phases of an operation. S3 is

meant to be a tool used in support of wargames, not a wargame in and of itself. They created and refined S3 as a logistics simulation which can produce most of the information a wargame umpire desires. The data requirements are extensive but modest in comparison with other models and it provides the umpires with information about the logistics network which they might not otherwise be able to generate. S3 seeks to extract the essence of a real world logistics system and provide realistic results to the wargame umpire with these minimal inputs. It takes simple data inputs and a simple scenario and combines them to create a convincing simulation of the real world. Once the data and scenario are assembled, the wargame umpire is able to step forward in time at various rates and then stop to examine or alter the status of the logistics network at that time. S3 orders supplies, consumes them, and moves them in a manner resembling the real world. S3 aids umpires in filling a role for which they might not otherwise be fully prepared.

C. THE PROBLEM

The S3 program has several short–comings that hinder its ability to act as a wargaming logistics tool. The largest short–coming is the interface with which umpires interact with the program. The command-line nature of the program makes it cumbersome to operate. It is effective but not efficient in gathering the data required for a given wargame scenario. Once the scenario is executed, S3's presentation of statistics about the logistics system is also cumbersome. It takes several steps to examine the status of a particular unit or base. The actual presentation of the results is also aesthetically unappealing.

The problem at hand is to convert S3 from a capable logistics modeling program into one which the non-analyst will be able to use. Key to this transition is a "user-friendly interface." For this part of the project the basics of the program will remain unaltered; only the outward appearance will change.

The goals of this version of S3—now called LogWarS—include:

- To create a graphical version of the current S3 program. It is thought wargame players cannot and will not use a simulation that is command-line driven. A window and menu display is a required element of the final program. This interface should include the ability to view logistic system performance statistics in graphical vice tabular format. The interface should be intuitive—leading the first–time user through scenario construction with minimal outside assistance or reference.

- To create an initial data base. The program should come with an initial data base to speed the development of scenarios. This master data base should be easily modified by the user as the need arises.
- To alter the current data structure to utilize master data base files which are read at the beginning of the scenario building process. As the scenario is created and altered, the program should save the changes to separate data files in different storage areas. This approach allows the user to create multiple, scenario-specific instances of units, bases, and transporters from a single common data file.
- To make individual unit and combined unit closure information readily available to the user of the program. The user should be able to view closure information at several levels of aggregation.
- To fully incorporate subunits as a tool for rapidly constructing units from generic components. The program should be able to add and subtract subunits from the units they form. Subunits should be available for the construction of any scenario.

LogWarS is not the first program to tackle the issues of a logistical tool for the support of a wargame. It is in fact a new version of the S3 program. Like S3, LogWarS addresses the need to rapidly create a logistics network simulating the movement of supplies from a location in CONUS—or elsewhere in the world—to the soldier at a deployed position. Once the scenario is created, LogWarS shifts its emphasis to presenting the wargame umpire with information about how the status of the logistics system changes over time. The goal is to assist the umpire in providing logistics information to the players while requiring the least amount of effort creating a model of the logistics network.

LogWarS achieves this goal by addressing several features of S3 that must be altered to make the program useful to a wargame umpire. The key alteration is the addition of a graphical user interface to the basic program. The interface provides easier access to all LogWarS functions and allows examination of the logistic situation in more meaningful ways than the earlier S3 version of the program.

Every effort has been made to ensure ease of data entry into the LogWarS data base. It is important that this not be taken to mean that no data entry is required. The user has several decisions to make when constructing a scenario and must enter new data or modify master data files to implement the scenario design decisions. If the master data section of the data base is inadequate, the user must take the time to find the necessary information and then input it into the data base.

II. PROGRAM DESCRIPTION

LogWarS is written in the MODSIM [Ref. 7] object-oriented programming language and uses graphics created by SIMGRAPHICS [Ref. 8]. The object-oriented nature of the language strongly influences the approach LogWarS takes in modeling a logistics system. The limitations of SIMDRAW result in dialog boxes which are functional but not as aesthetically pleasing as desired. Both MODSIM and SIMGRAPHICS combine to allow LogWarS to perform its two basic functions—aiding the user constructing a scenario and executing the scenario while allowing the user to monitor and influence events.

Before any discussion of the uses of LogWarS can take place, a general description of how LogWarS functions is required. The concept of a scenario and its basic components must also be discussed. Both Long and Halvorson have discussed some of the underlying structure of S3 and also provided excellent discussions of scenarios. The immediate goal is not to duplicate their efforts, but to provide a basic understanding of LogWarS for the first time user of the program. The emphasis is on an individual who has little analytical or computer science background. A basic understanding of logistics is helpful.

A scenario is a collection of logistic system elements—commodities, units, bases, and transporters—organized to represent a specific logistics network. A scenario contains all the information required to instruct LogWarS in how to run the simulation. A scenario is the data required to set-up LogWarS to support a wargame. Scenarios contain information about the types and quantities of commodities, the types and number of transporters, and the locations and characteristics of both bases and units.

To aid in the construction of scenarios, LogWarS incorporates a custom logistics data base which is not derived from any other Department of Defense computer data base. The data is manually entered by the user and reflects information from a variety of sources. The current LogWarS data base uses information derived from sources as varied as the Army *Staff Officers' Field Manual* [Ref. 9] and *The Aircraft and Ships of the U.S. Fleet* [Ref. 10].

The data base stores specific information on elements of the scenarios—*i.e.*, the speed and cargo capacity of transporters—and is divided into two sections—the master data files and scenario-specific data files. The two sections of the data base contain the same types of elements stored as data files. The difference between the sections is in how the sections are used by LogWarS to carry-out its basic functions. The master section of the

data base forms a largely permanent collection of elements, available for use in any scenario, which are used to initially build scenarios. Elements may be added to or created for a specific scenario and the user has the option of saving the elements to either or both sections of the data base. The scenario data files are stored in a separate location from the master data files so any changes made to the scenario elements will only effect the execution of that scenario—or any other scenario stored in that location.

As shown in Figure 1, data files in the two sections may represent the same element. The 10th Mountain Division data file in the Master Subdirectory is a master data element. It is part of the reference data base and can be used to construct any scenario. As a master data file, it contains a great deal of information about the 10th Mountain Division. This information may or may not be appropriate to a specific scenario being constructed. The 10th Mountain Division data file in the Scenario Subdirectory stores the particular information for the 10th Mountain Division which is required for the scenario. For example, the 10th Mountain Division in the Master Subdirectory might include many different types of ammunition in its inventory. When it is altered for use in the scenario, only those types of ammunition which are of interest in the scenario are stored in the 10th Mountain Division data file in the Scenario Subdirectory. In addition to the 10th Mountain, the scenario stored in the Scenario Subdirectory in Figure 1 requires Fort Drum as one of its bases but not Pope AFB or Rota, Spain.

Data files are generally stored with names which come from the first eight characters of the name of the item being stored in the data file. Care must be taken to ensure each element of the scenario section or the master section of the data base has a unique sequence for the first eight characters of its name.

A. THE ELEMENTS OF A SCENARIO

LogWarS attempts to model the flow of supplies through a logistics network during both the surge and sustainment phases of a conflict. The scenario created using LogWarS is not intended to completely represent the real world—such an undertaking would result in a massive computer program that would still miss many of the details of the real system. A LogWarS scenario is a *model* of the real world—a simplification—which has many artificialities but still adequately captures the essence of the real logistics network. It

In a LogWarS scenario the logistics network has four basic elements—bases, units, transporters, and commodities. These four elements become part of an elaborate web which provides a manageable way to examine the performance of a real logistical network.

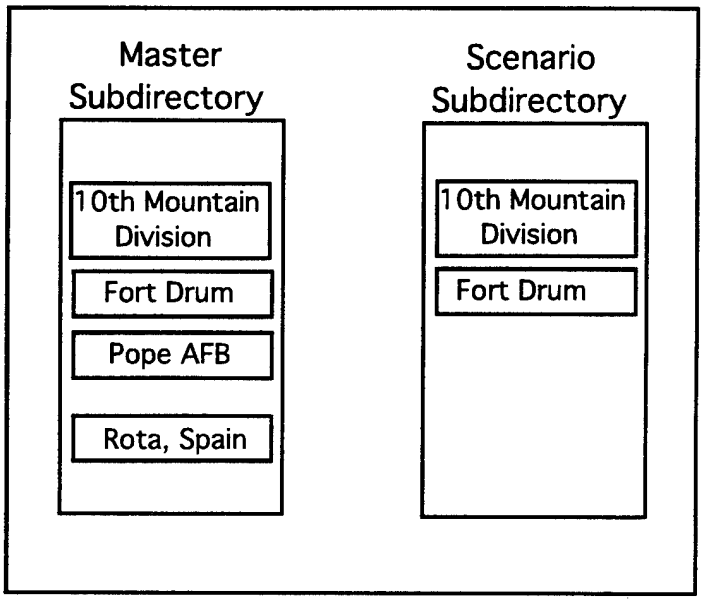


Figure 1. The LogWarS Directory Structure.

Figure 2 depicts a generic logistics network for a scenario in which the circles represent bases, the squares represent units, and the lines represent the air, sea, road, and rail links between the bases and units. Transporters, which represent transportation assets, will move along the links to move commodities from one location to another. Commodities represent supply items in the scenario which are consumed by units and are reordered by them to maintain a stock level. The network structure of the scenario is created in LogWarS and when the scenario is executed the units consume their stock of commodities on hand and place orders for replacements with the nearest base. The order is either filled at the nearest base or it is sent to another base further up the logistics network until the order is filled. Transporters are assigned to move the commodities from the bases filling orders to the units which ordered the commodities.

All the consumption, ordering, and transportation of commodities takes place in simulated time. There is not an instant response to commodity orders, the ordering unit must wait for the transporter to move the commodity from a base to the unit before the commodity is restocked. For example, the simulated time delay between ordering a commodity and its resupply means that a unit might consume all of a commodity on hand before it receives a new supply. The user of LogWarS moves forward in simulated time and can observe the unit running-out of its stock of the commodity. The user may accept

this situation or change the stock of commodity on hand at the unit and then continue the scenario to the next desired point in simulated time.

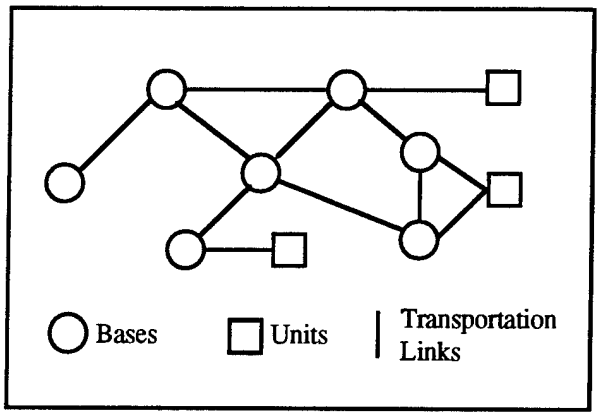


Figure 2. A Basic Logistics Network.

With the basic structure of a LogWarS scenario in mind, it is now possible to discuss each of the four elements of a scenario in greater detail.

1. Commodities

At the heart of a LogWarS scenario are commodities. A commodity is some item of interest in the wargame that is consumed by units as time elapses. The wargame planner must determine what elements of the supply system should be tracked. These become the commodities in LogWarS. For example, a large global wargame might be concerned with the flow of ammunition in general and not with each specific type of ammunition. In this case a commodity called ammunition would be appropriate. In that same wargame it might also be determined that JP–5 and personnel are also important items. These become commodities assigned to the units which need the commodity. At the unit level a consumption rate is established (*i.e.*, rounds of ammunitions per day) and then the simulation is executed. As time elapses, the commodities in the units are consumed depending on the consumption rate. When the quantity of each commodity on hand at the unit reaches its reorder point, the unit places an order for the commodity. LogWarS searches the bases in the scenario and attempts to fill the order. In the example above, the result is that the wargame umpire will be able to track an order of ammunition to a specific unit and determine where the wargame may have bottlenecks in the supply system. The umpire will be able to state, "There are 100 laser guided bombs at Advance Base Echo but none have yet arrived at Air Field Charlie."

It is important to note that virtually anything can be a commodity. The key features of a commodity are a unique name and a set of consumption rates. Thus, individual personnel can become commodities by creating a commodity with the name *personnel* and then assigning a set of consumption rates. Figure 3 illustrates the commodity screen for a *personnel* commodity. Once the simulation is started, the units with *personnel* as one of their commodities will begin consuming them and will reorder *personnel* when the reorder point is reached.

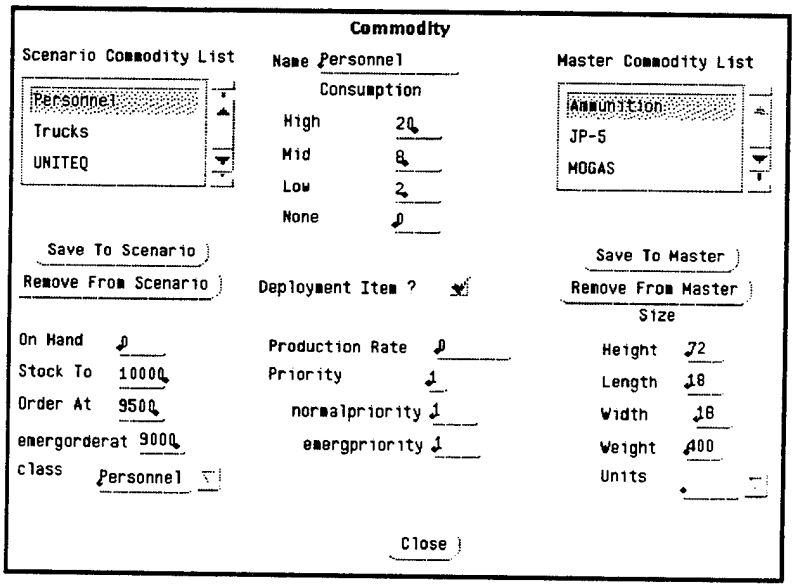


Figure 3. A Commodity Dialog Box with *Personnel* as the Commodity.

In the normal sense of the word, people are not commodities. However, their role in a logistics network is sometimes best simulated in LogWarS as a commodity. The significant factors of personnel which make them a good commodity are their demand by units and their need for transportation to units. A consumption rate for a *personnel* commodity is the way of simulating casualties in combat in LogWarS. The use of terms such as "consumption rate," "stock to," and "on hand," which are easily applied to material things, are less easily applied to a construct that represents real people. It is necessary, however, if personnel constraints on wargame operations are to be modeled.

In the same manner that individual personnel can be a commodity, platoons or companies might be a commodity in a new scenario for a global scale wargame. The only difference is the scale and fidelity of the game. The choice of the scale and fidelity is one of the essential choices in a wargame and is discussed further on page 17. When the new

scenario is executed companies will be consumed by those units which have them as commodities.

2. Units

Units refer to any grouping of military troops or equipment the user feels is important. The important distinguishing feature of a unit is that it has a geographic location, an inventory of commodities and their usage rates, and some connection to the other bases or units in the scenario. The unit acts as a location to which commodities must be transported and at which they are consumed and resupply orders are generated.

In the actual program code a unit *is* a geographic location. When the simulation begins, units appear at their geographic locations at times which depend on their activation delay. The activation delay is used to simulate the time it takes the lead elements of a unit to reach their deployment site. When initially deployed, the units have little or none of their commodities on hand at the location. The unit is effectively "a name in the sand"—a place to which the commodities in the unit's inventory will be sent. Once the unit is deployed to its location, it begins to order commodities to meet its stocking requirements. At this point, commodities begin to move from the base from which the unit originated to the deployed location of the unit. At some point in time the unit has all its deployment commodities and begins to consume all of its commodities at a rate depending on the current combat intensity. As the commodities on hand at the unit fall below their reorder level, the unit orders replacement commodities. These orders are transmitted back through the logistics network until they are filled by any base in the scenario. The replacement commodities are then loaded on whatever appropriate transporter is available for movement to the requesting unit.

Figure 4 depicts the data associated with a typical unit. The unit shown, the 24th Marine Expeditionary Unit (MEU), consists of 2,500 personnel and is originating from the U.S.S. Wasp. The 24th MEU is connected to the rest of the logistics network by a road and a rail port.

3. Bases

Bases, like units, are essentially geographic points in the supply system. They act as transshipment points for commodities moving to units and holding areas for stockpiles of commodities awaiting a need for them at the unit level. Like units, bases also have supply levels for the specific commodities they stock. If the any of the commodities in the base's stockpile falls below the reorder level—because of filling other unit or base

requests—the base generates an order which is transmitted through the logistics network and is handled exactly as an order from a unit is handled. Bases are connected to each other and to various units through the transportation network. The ports of a base allow capacity restriction to be placed on how many transporters can move to and through each base. The important distinction between bases and units is that bases do not consume any commodities as the simulation proceeds.

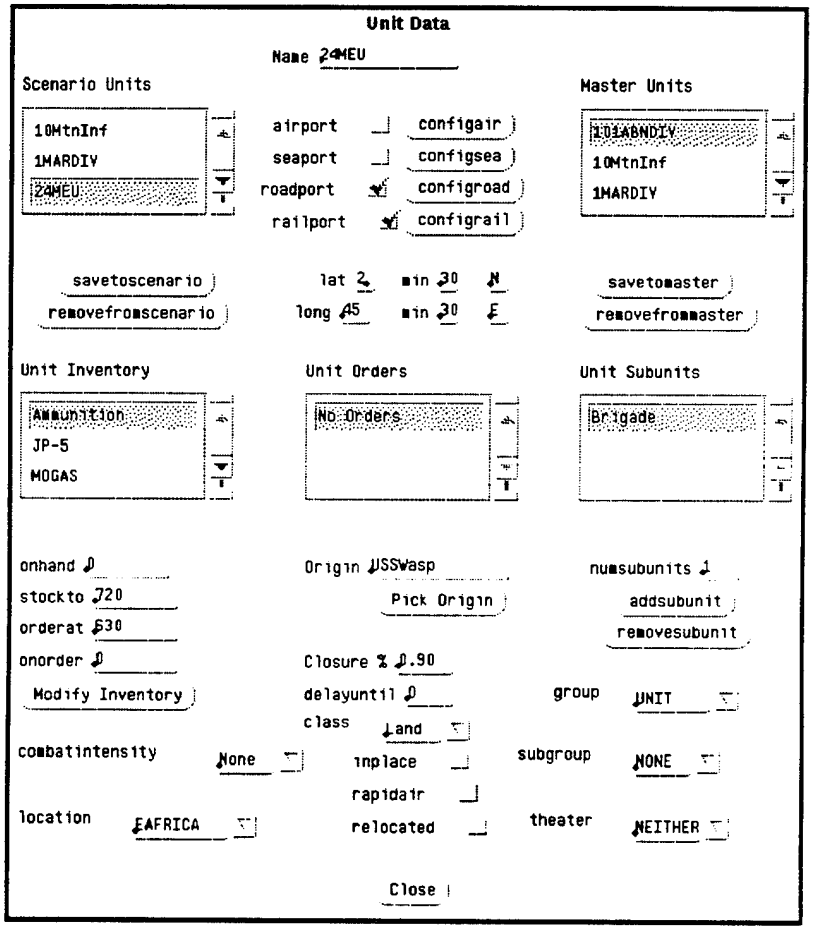


Figure 4. A Unit Data Dialog Box.

4. Transporters

Transporters are the means by which commodities are moved from one location to another. They fall into several general categories—aircraft, ships, trucks, and rail equipment. They can only traverse a link between two bases or a base and unit which is appropriate to their type, *i.e.*, trucks can only move on road links. If a base or unit does not

have a particular kind of link—determined by whether it possesses an appropriate port—transporters that require that link cannot go to that base or unit. The 24th Marine Expeditionary Unit in Figure 4 can receive rail or truck class transporters because it has a rail port and a road port. Because it has no seaport, the unit cannot receive any ship class transporters.

a. Prepositioned Ships

Prepositioned ships (prepos) are a special kind of transporter that are preloaded with commodities at the beginning of the simulation. The prepos are then given an initial geographic location and an initial route to follow. Their cargo is given a destination and an amount aboard the ship. When the simulation begins, the ships move along their route and will unload their cargo of commodities when they reach their destination. At that point the cargo will move by other transporters until it reaches its ultimate destination. Once a prepo is off–loaded it becomes another transporter that is used by the system to move supplies.

5. Summary

A LogWarS scenario is a simulation of a network. Commodities flow through the network on transporters. These commodities are produced and stored at bases. They are ultimately consumed by units. It is the demand of units for commodities that determines production and distribution of the commodities. This production and distribution is automated by LogWarS. The user determines unit location in the network and consumption/demand levels for commodities. Once these factors are set, LogWarS proceeds to meet the demands as best as possible.

B. ANALYTICAL BASIS OF LOGWARS

LogWarS takes as its inputs a force structure and location. The output is a list of closure and supply levels at all locations in the logistics network at a specific time. By varying the input data, a wargame umpire can arrive at a conclusion about the robustness of the network. By stepping forward through time, the wargame umpire can determine the status of the logistics network as it changes during the wargame.

The use of expected values for travel times, on-load/off-load, consumption, and other factors is far from ideal. By giving only one number for these times, the model is using a point estimate or expected value. It does not allow for the possibility that the numbers might be more or less according to some probability distribution. A truly

analytical simulation would use distributions for each of these factors and after multiple replications the *answers* would be fit with confidence intervals.

While this is the desired approach when using simulations to solve problems, it is the wrong approach for LogWarS for several reasons. First, LogWarS does not seek or provide an *answer*. One model run may be used by many very different people to answer very different questions. Part of the model output might be used by a warfighter to check on the supply level of a certain unit, while a force planner might observe the relative level of supply used for two different platforms engaged in the same mission. Each of these individuals is not trying to determine the actual outcome in the real world—LogWarS does not have enough data or sophistication to do this. They are trying to determine the influence of these logistics factors on the decisions of the wargame players. They are also trying to determine which questions to ask about the logistics system and how it is run. LogWarS is not designed to produce true *answers*—as a model of reality, any answers must be approximations of the real world—but rather to provide guidance, good guesses at the likely outcome, and to raise questions in the minds of the wargame players. Secondly, no two runs of the simulation will have exactly the same inputs. As a wargaming tool, the inputs to LogWarS will depend upon the player decisions in the wargame. The results from LogWarS will in turn influence the course of the wargame in a dynamic manner which is unsuitable for traditional simulation problem solving methods (*i.e.*, multiple replications of the simulation to achieve an answer).

The stochastic elements of a logistics network used in LogWarS—the transit times for the transporters and the consumption rates for the commodities—are not expected to vary greatly from their expected values. Using deterministic inputs avoids the needless complication of the model which adding a stochastic element would create. Use of stochastic elements greatly increases the complexity of the model and also vastly increases the data input requirements. The umpire would be required to input distributions for each stochastic element. There is little high quality data for these required inputs. Gathering such data, while a valuable undertaking, is well beyond the scope of this thesis and far more work than an umpire can be expected to make. The deterministic model used by LogWarS provides a good first approximation of the performance of a logistic system. It will embody the essence of the real system without being overly detailed or data intensive. More detail does not imply more accuracy or fidelity but does require much more data.

III. HOW TO USE THE PROGRAM

A. THE PROPER ROLE FOR LOGWARS

LogWarS is a support tool for umpires of wargames—an aid for the play of a wargame and not a replacement for umpire decisions or judgment. LogWarS provides essential data base and modeling support for a wargame by acting as a logistics data base, showing the supply and support status of all forces at any given time as the wargame progresses. It provides the capability to track and display the supply status of the entire logistics network from CONUS bases to deployed units.

LogWarS takes the logistics data input by the players and players' choices and uses these inputs to determine the results at subsequent times during play of the wargame. The umpire can decide to advance the game one hour or one year—the only difference in LogWarS is the amount of time the program will take to execute. LogWarS essentially does the bookkeeping of a wargame and provides the logistics information necessary to inform and influence the players' decisions in the wargame. This allows the players and umpires to concentrate on playing of wargame, not the details of administering the wargame. In this manner, LogWarS will not be a drag on the pace of a wargame but can actually speed wargame play and, hence, enhance the time pressure experience for the players.

LogWarS scenarios range from global to theater in scope and support the strategic to operational level of detail. The choice is purely one of wargame design—LogWarS does not impose any restrictions. LogWarS will handle any wargame scenario from World War III to action by a squad of soldiers in the Middle East and can be used to support seminar-style or one and two sided wargames. By running two separate LogWarS scenarios, both Blue and Red logistics systems can be modeled, providing logistics constraints for both Blue and Red forces.

B. FACTORS TO CONSIDER WHEN CREATING A SCENARIO

1. Level Of Detail and Performance Tradeoffs

Because LogWarS is a general simulation program, the choice of level of detail is fundamental to designing a scenario for the wargame. A wargame must "...accurately reflect the factors appropriate for the players' decision levels [Ref. 1, p. 215]." A global wargame with Corps commanders acting as the players should not require the player to decide the employment of individual companies. In this same wargame, it would be inappropriate to use individual companies as the basis of the logistics data base in

LogWarS. The decision about the scale of the wargame should drive the choice of commodities, consumption rates, and unit size. The scenario should use the minimum detail required to provide the desired information because using excessive detail slows the performance of LogWarS and dramatically increases its memory requirements. Excessive detail also dramatically increases the required data input and the difficulty of tracking unit supply status as the scenario is executed.

The choice of scenario detail does not need to be consistent for the entire scenario. For example, a global wargame may be concerned with the overall results of supporting a Marine Expeditionary Brigade (MEB) in Korea but also want to know how changes in the logistics system effect individual battalions of the 82nd Airborne Division. This is readily accomplished through the proper selection and definition of commodities and units.

2. What Must Be Modeled

LogWarS is designed to simulate the flow of supplies from some stockpile or production sight to units that require the supplies. Modeling the network from factory to supply depot is not necessary although it is easily done with LogWarS. The specific supply items that are of interest during the wargame become the LogWarS scenario's commodities and are the most important items to be simulated. Because the commodities move on transporters and are temporarily stored at bases, bases and transportation assets are two other critical elements of a logistics network to model. Finally, units must be modeled because they are the only part of the simulation that use commodities. Units remove commodities from their stock as time progresses and place orders for replacement for a commodity when the amount of the commodity on hand falls below a reorder level. The connections between bases and units must be specified for the commodities to be routed to their destination. These four elements—commodities, transporters, bases, and units—are required for a fully functioning logistics scenario.

C. CREATING A SCENARIO

In its current form, LogWarS is neither robust nor tolerant of user errors. It is a capable logistics modeling tool but is very sensitive to user inputs. Deviations from the proper sequence of scenario construction may lead to unpredictable results or, in the worst case, program termination. LogWarS requires more work before it can be used as anything more than a demonstration of a concept.

Perhaps the best way in which to demonstrate the abilities and limitations of LogWarS is to create a wargame scenario from scratch. For this purpose, the following

sections will give the background of a wargame scenario and then demonstrate how to implement it in LogWarS. It is important to remember that this scenario is *not* intended to be completely realistic. It is meant to demonstrate the ways in which LogWarS can be used for wargaming support and ways in which it can be *tricked* into performing certain desired tasks. Before detailing the construction of the LogWarS demonstration scenario, it is important to outline the general steps in constructing any scenario.

As an aid in understanding the construction of a scenario with S3, Halvorson [Ref. 6] introduced the concept of a scenario construction pyramid. Figure 5 extends that concept to constructing a scenario with LogWarS.

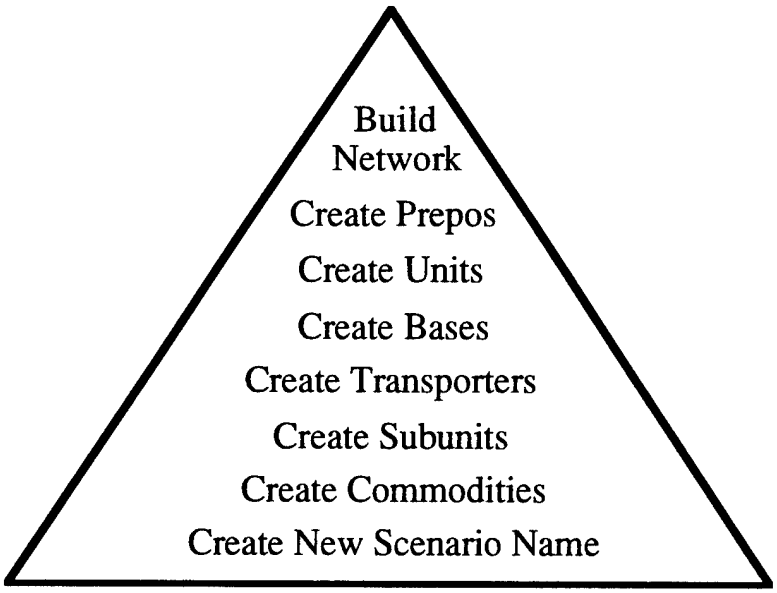


Figure 5. The Scenario Construction Pyramid After Ref. [6].

The scenario for this demonstration involves humanitarian relief operations in Somalia. The players of the wargame have determined a tentative operations plan in which Marine units will go ashore first with Navy support. Once airfields become available, the Air Force will begin flying in relief supplies from the United States and from excess warstocks in Europe. As soon as feasible Army units will move into the country to consolidate control, expand the area of influence, and continue relief efforts. The players have determined the commodities of interest during the play of the game and the transportation assets available. The details of the elements of the scenario are presented in Figures 6 through 9.

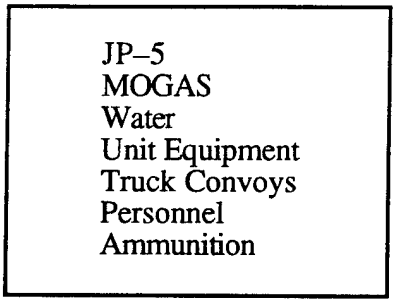


Figure 6. The Commodities in the LogWarS Demonstration Scenario.

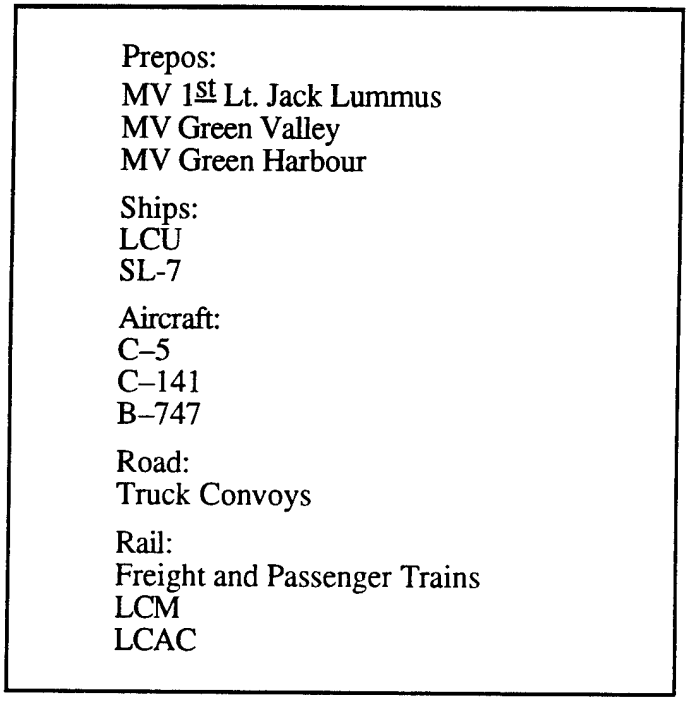


Figure 7. The Transportation Assets in the LogWarS Demonstration Scenario.

1. Operating LogWarS

This thesis will refer to LogWarS menu items using the format **Menu Name>Menu Item**. The names of buttons in dialog boxes will be denoted by **Boldface** type while names of text and value fields will be capitalized. For example, **File>Quit** means select the quit menu item after selecting the file menu in the LogWarS main window, Figure 10. **Cancel** means select the cancel button of a dialog box and Speed means the speed field in a dialog box.

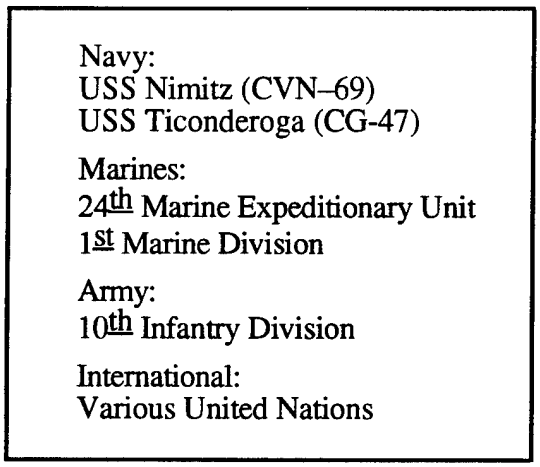


Figure 8. The Units in the LogWarS Demonstration Scenario.

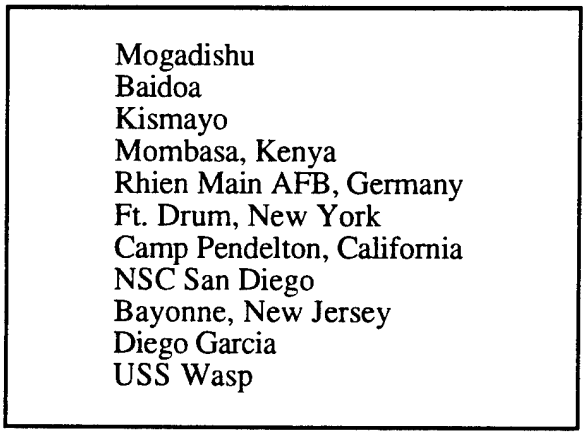


Figure 9. The Bases in the LogWarS Demonstration Scenario.

The main window of LogWarS, seen in Figure 10, consists of a menu bar at the top and an otherwise empty window.

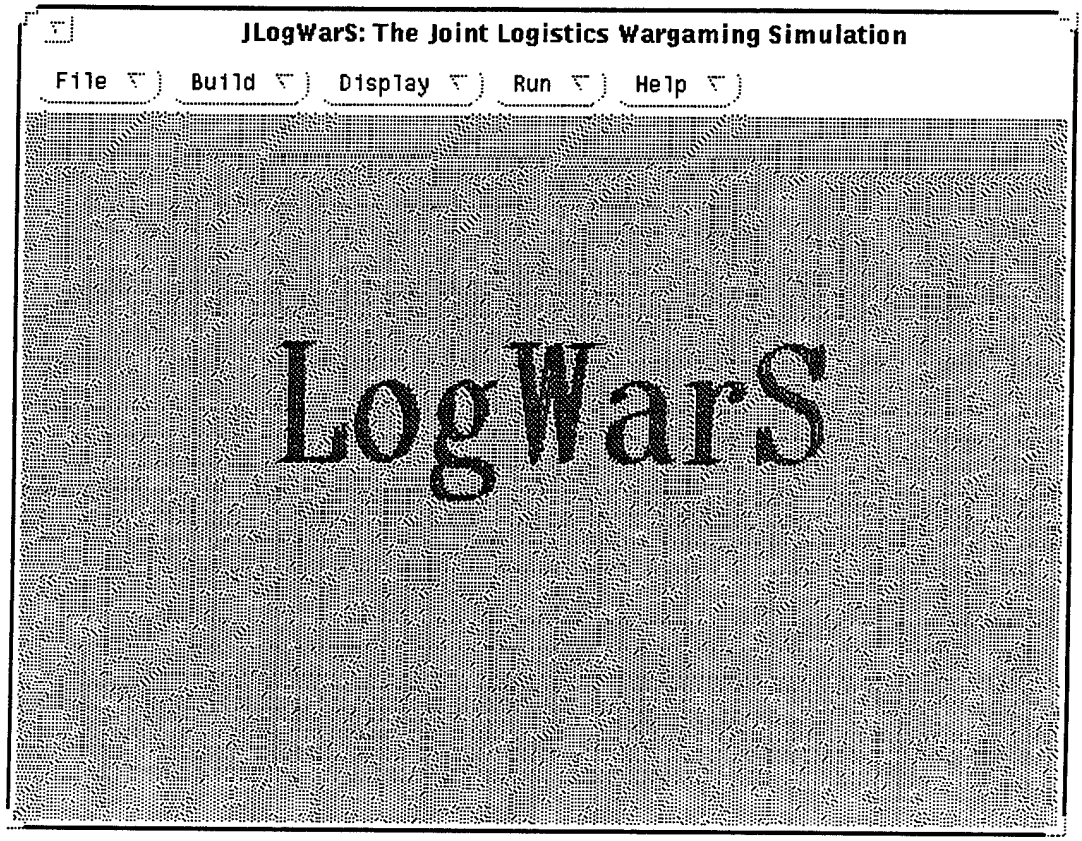


Figure 10. The Main Window of LogWarS.

In addition to the main window, LogWarS has customized dialog boxes to speed the construction of a scenario. Most dialog boxes share a core set of features in addition to the specific feature each requires.

In the upper right and left corners of the dialog box shown are features called list boxes. A list box is a collection of names, a list, contained in a box with up and down arrow buttons which allow the user to scroll through the list. In the case of the Commodity dialog box in Figure 11, each of the list boxes contains a list of commodities. The left box contains a list of commodities used in the scenario under construction and the right box contains the master list of commodities for LogWarS. This structure appears in most LogWarS dialog boxes; the scenario elements are in the left list box and the master elements are located in the right. To select a particular item to be modified, scroll through the appropriate list box and select the name of the item. The name of the commodity currently available for modification appears on the Name line (top row, center) and data associated with the commodity appears in the rest of the dialog box.

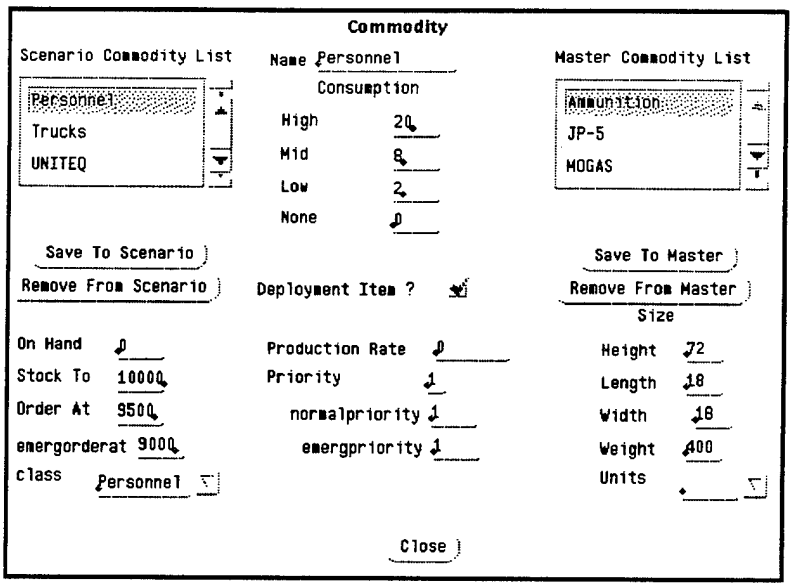


Figure 11. A Commodity Dialog Box.

Common to most dialog boxes are the **Save To...** and **Remove From...** buttons. Pressing the **Save To Scenario** button in the Commodity dialog box will save the data for the Personnel commodity from the dialog box to a data file in the scenario section of the data base. The **Save To Master** button will save the commodity to the master section of the data base. To create a new commodity for the scenario, select a commodity from the master list and then press the **Save To Scenario** button. After selecting the new commodity from the scenario list, modify it as appropriate and then press the **Save To Scenario** button again. The process is similar for constructing a new commodity for the master section of the data base. It is important to note that the **Save...** buttons generally cause files to be written to the hard drive. The new files will overwrite files with the same name without warning. The **Remove...** buttons generally delete reference to the element in either the scenario or master sections of the data base. All modified items should be saved before selecting another item from a list box or pressing the **Close** button.

2. Building Scenarios

To create a new scenario, from the main LogWarS window pull down the **Build** menu and choose **Scenario**, **Build>Scenario**. This will bring up the Scenario Construction dialog box, an example of which is shown in Figure 12. Enter a new name on the Name line and press the **New** button. To modify an existing scenario select its name

from the scenario name list box of the Scenario Construction dialog box and then press the **Load** button. Selecting a scenario name and then pressing the **Delete Scenario** button will remove all the scenario master files from the hard drive but will not remove any commodity, subunit, unit, base, or transporter data files associated with the scenario. Figure 12 depicts a Scenario Construction dialog box with a Somalia scenario being constructed.

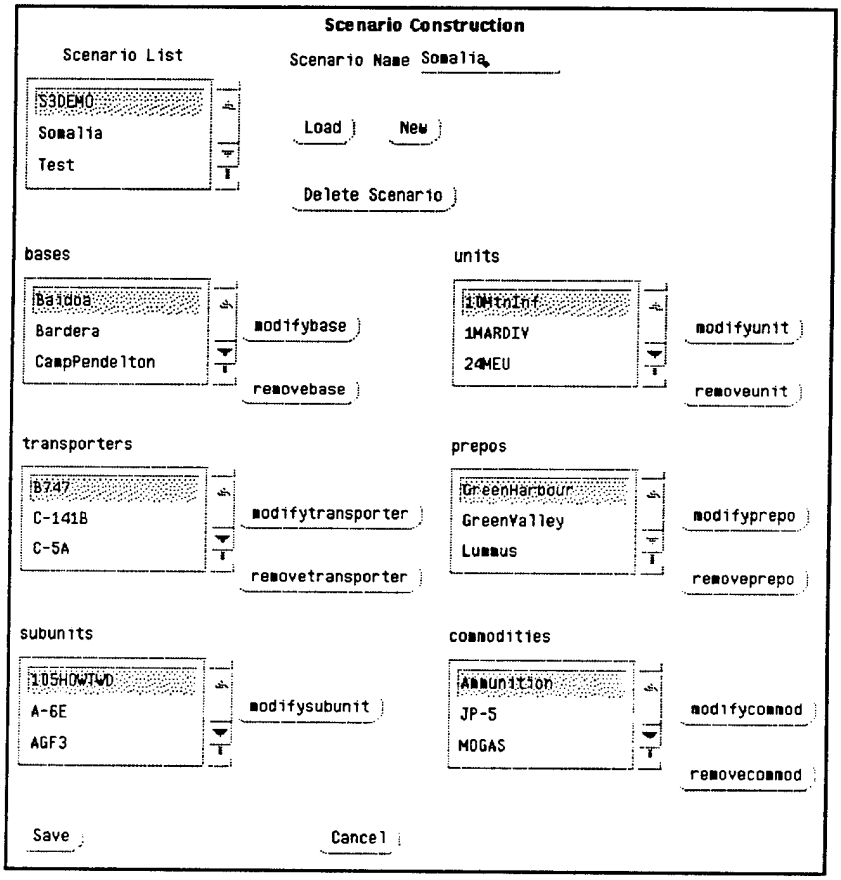


Figure 12. A Scenario Construction Dialog Box.

For the demonstration scenario, there is no currently built scenario that is suitable for modification. Therefore, type the new name—Somalia—on the Name line and press the **New** button. LogWarS automatically creates all the appropriate scenario master files and loads them. The data files are created in the data files directory listed in the Preferences dialog box. The user must manually create all directories used for LogWarS data files using the appropriate commands for the computer's operating system.

After pressing the **New** button, all the list boxes in the Scenario Builder dialog box, except the Subunits, should be empty because there are no entries in the categories.

Pressing the **Save** button will store the data in the various list boxes to the appropriate data files for the current scenario. The **Save** button is equivalent to using **File>Save Scenario**.

At this point there is a new scenario named Somalia which is ready to be filled with the required scenario elements. There are two ways to create the required commodities, subunits, transporters, bases, units, and prepos. The first involves selecting the appropriate **Modify...** button in the Scenario Construction dialog box. The second involves using the appropriate menu item in the **Builder** menu. Both methods are equivalent—the choice of method is one of user preference.

3. Building Commodities

After determining which commodities will be of interest in the scenario, determine their default consumption levels, if any, and their physical characteristics. A commodity may be created or modified from the **Commodity** dialog box, Figure 11. To reach the dialog box, from the LogWarS main window click the **Build** menu and then select the **Commodity** menu item, **Build>Commodity**. Alternatively, from the Scenario Construction dialog box, Figure 12, select the **Modify Commod** button. Using the Commodity dialog box, either construct completely new commodities for the scenario or choose specific commodities from the Master Commodities list. Changes will only effect the commodity used by the scenario, not the master commodity. (To be sure of this, always select the commodity name from the Scenario Commodities list box before altering the data.) The commodity size information must be provided when first constructing commodities for a scenario; the consumption and stocking level data of a commodity are specific to each unit or base and should be entered later during unit and base construction.

If the commodity is going to be used to determine unit closure the Deployment Item box should be checked. LogWarS considers a unit closed when a certain, user-determined, percentage of specific commodities are at the unit's deployment location. The Deployment Item check box is used to indicate which commodities should be used to determine closure. Once all the data for the commodity is correct, press the **Save To Scenario** button once again.

In the case of the LogWarS demonstration scenario, one of the commodities of interest is a truck. The truck commodity, which does not exist in the Master Commodity list, will be used to track the arrival of transportation assets in theater. In the demonstration scenario, trucks will be a deployment item because the umpire needs to know when they

arrive in theater so that the umpire may create transporters to represent convoys of these trucks to move supplies in Somalia. The truck commodity is not a transportation asset; it is a means of tracking the arrival of trucks to the theater so real transportation assets can be deployed at the proper time.

The process of selecting an existing commodity or creating a new one continues until all the desired commodities have been added to the Scenario Commodity list. When all the changes to the scenario commodities have been made, press the **Close** button in the Commodity dialog box. It is a good idea to save the scenario at this point, using either the **Save** button in the Scenario Construction dialog box or clicking **File** in the main LogWarS window and then selecting the **Save Scenario** menu item, **File>Save Scenario.**

4. Building Subunits

Once the commodities required by the scenario are created, it is time to create any new subunits needed to speed the creation of the scenario units later. Recall that subunits are simply collections of commodities used to speed construction of units. It is important to ensure the commodities used in creating the new subunits are already in the scenario's commodity list. If they are not, LogWarS will automatically add the commodities to the scenario but the user will not receive any warning. Any unexpected commodities in the scenario could drastically alter the performance of the model—the consequences are unknown and, therefore, unexpected commodities should be avoided.

The Subunit Data dialog box is displayed by selecting the **Build** menu and the **Subunit** menu item from the LogWarS main window, **Build>Subunit,** or by pressing the **Modify Subunit** button in the Scenario Construction dialog box. Using either method, the Subunit Data dialog box, Figure 13, appears showing all the subunits available in LogWarS. This is the only case where it is not possible to create scenario-specific elements. The subunits displayed will be available in every scenario created for LogWarS and, what is more important, any changes made to a subunit will effect every other scenario which uses that subunit.

The Subunit Data dialog box displays subunits in three different classes—Air, Sea, and Land. The classes are used to ease the creation and organization of subunits. It is important to include a class for each subunit in order for LogWarS to properly function. The class does not effect subunit use in any other part of the program but trying to create or save a subunit without a class will result in unpredictable errors and possible program termination.

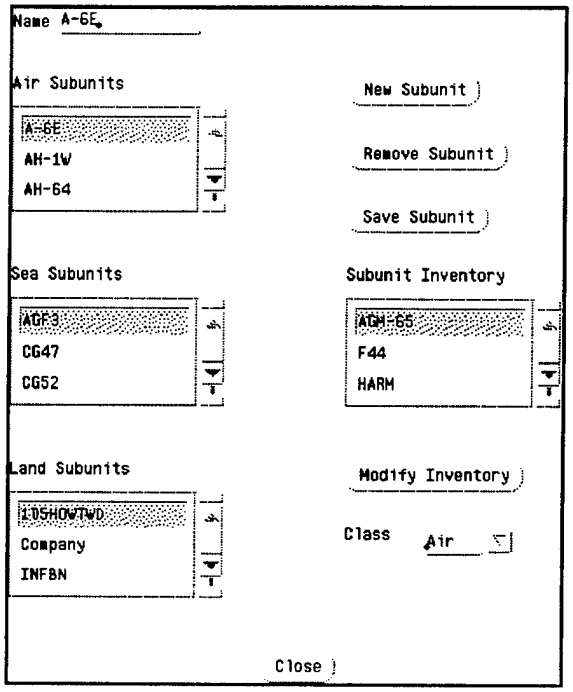


Figure 13. The Subunit Data Dialog Box.

In addition to presenting ordered lists of subunits available for use in the scenario, the Subunit Data dialog box displays a list of the commodities in the currently selected subunit's inventory in the Subunit Inventory list box. To select a subunit for examination or modification, scroll through the appropriate list box and select the subunit's name. The Subunit Inventory list box will now contain all the commodities associated with that subunit. To modify the inventory of the subunit or view it in greater detail, press the **Modify** button. The already familiar Commodity dialog box, Figure 11, will appear. The inventory of the subunit appears in the Scenario Commodity list of the Commodity dialog box while the Master Commodities list contains the master list of commodities for LogWarS. The Commodities in the subunit can be altered in the same manner as discussed on page 25.

Pressing the **Save** button in the Subunit Data dialog box stores the currently selected subunit—either an existing subunit or a newly created one—along with its inventory, in a list in the LogWarS program. The **Save** button does not store the data onto the hard drive—it only stores the altered subunit to the list in LogWarS. To make the change permanent, close the Subunit Data dialog box and any other open dialog boxes using their **Close** buttons and select the **File** menu and then the **Save Master** menu item from the main LogWarS window, **File>Save Master**. The **Remove** button in the

Subunit Data dialog box deletes the currently selected subunit from the list of subunits available in LogWarS. Again, **File>Save Master** must be used to make any permanent changes to the subunit list.

The demonstration scenario requires two unique subunits. The first will be a collection of JP-5 called an Airplane. This Airplane is not a transporter. It will be used to ensure the U.S.S. Nimitz unit has the proper JP-5 commodity requirements. To create this new subunit, open the Subunit Data dialog box, type "Airplane" on the Name line, select Air from the Class list, and press the **New** button. Press the **Modify** button and then select the JP–5 commodity from the Master Commodity list in the commodity box which appears. Set the consumption rate to 70 for None and Low, 140 for Mid and High. Set the On Hand and Stock To values as desired and then press the **Save To Scenario and Close** buttons in the Commodity dialog box. The Airplane subunit will now simulate one airplane using 20,000 pounds (70 barrels) of JP–5 each day in little or no combat. Press the **Save Subunit** button in the Subunit dialog box to save the subunit for later storage on the hard drive.

The second subunit, simulating a group of 1,000 ground personnel and 157 tactical ground vehicles, will represent a generic infantry Brigade. The compliment of 1,000 personnel was chosen to ease the calculation of commodities required for the subunit. Since the Brigade will represent a generic collection of 1,000 troops, add Ammunition, MRE, MOGAS, Personnel, UNITEQ, and Water commodities to its inventory from the Master Commodity list. As the commodities are being added to the inventory, alter the stocking levels and the consumption levels as needed to simulate a brigade of 1,000 troops. For example, the Water commodity will be consumed at a rate of ten gallons per man per day for low or no combat but will increase for higher combat intensities. To reflect this in the Brigade subunit, 10,000 is entered on the None and Low fields of the Commodity box and 12,000 and 16,000 are entered on the Mid and High fields respectively. The Personnel commodity will not normally have any consumption rate—consumption for the Personnel commodity represents a casualty rate—because of the humanitarian nature of the operations but will have consumption rate for higher combat intensity. Since the Brigade represents 1,000 troops there will be 1000 Personnel on the Stock To line. The Brigade subunit now represents a collection of six commodities, all consumed and stocked to scale for the 1000 personnel and 157 vehicles represented by the subunit.

Remember to save the subunits using **File>Save Master** after all the modifications are complete.

5. Building Transporters

The Transporter Data dialog box, Figure 14, is displayed by selecting the **Build** menu and the **Transporter** menu item from the LogWarS main window, **Build>Transporter**, or by pressing the **Modify Transporter** button in the Scenario Construction dialog box. The Transporter Data dialog box is used to construct the various types of transportation assets which will be needed in the scenario. An example of a type of transporter is a C–5A. There may be many specific C–5A aircraft in the scenario but there is only one basic description of its characteristics—one type. Only in rare instances would the user create one-of-a-kind transporters using the Transporter Data dialog box.

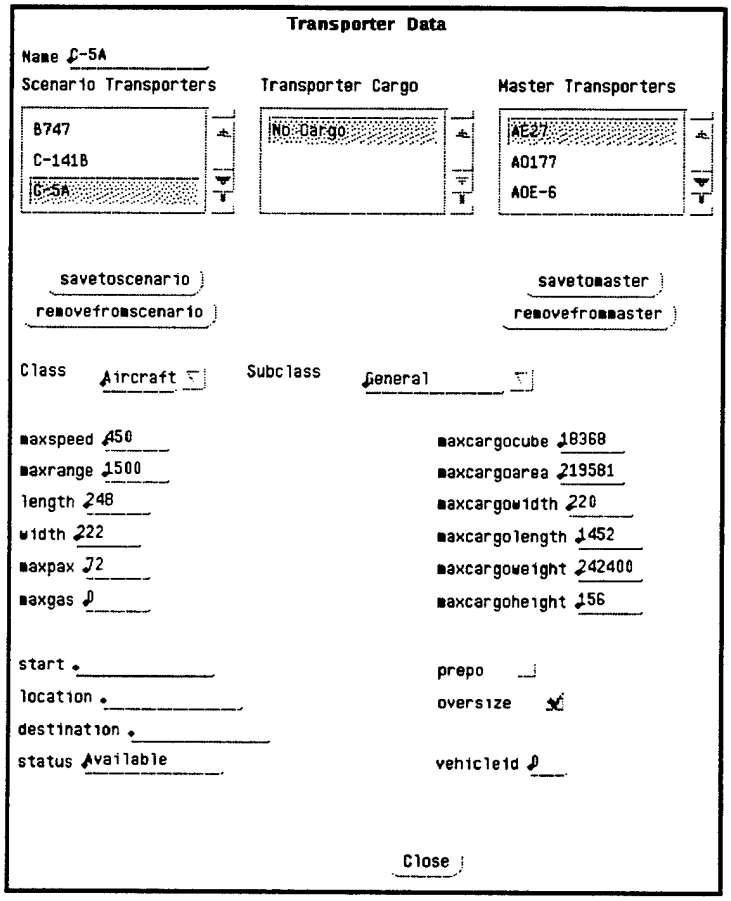


Figure 14. The Transporter Data Dialog Box.

The top section of the Transporter Data dialog box contains user modifiable values and fields. The Class of a transporter is used to determine which port the transporter will use at bases or units, *i.e.*, transporters with an Air class will only use airports. The Subclass is used to determine the general cargo hauling capabilities of the transporter class.

Long [Ref. 5, p. 44] provides an excellent description of transporter cargo lifting capabilities as modeled in S3 and LogWarS.

The Maxspeed is used to determine travel time between ports. The travel time uses the great circle distance between the two ports in miles divided by the speed in miles per hour. LogWarS assumes the transporter travels at its maximum speed from the moment it leaves port until the moment it enters the next port. The are no range or fuel constraints placed upon transporters.

The cargo capacity values, from the number of passengers to the maximum cargo height, are used by LogWarS to determine feasible cargo loads for the transporter. Long [Ref. 5, p. 43] thoroughly discusses these details of transporter construction.

The lower section of the dialog box is for transporter status information which is useful later as the scenario is being executed. The Start, Location, and Destination fields are the names of the bases which describe the current route of the transporter. The Status indicates the current actions of the transporter. A transporter may be Available, Loading, Unloading, or Enroute. The Oversize check box indicates whether the transporter can be used to carry over-size cargo. Over-size is related to whether the transporter can carry large cargo items such as M–1A1 tanks. The Vehicle Id is a unique number for each transporter within a transporter type. For example, a scenario might have ten C–5As. They would each have a unique vehicle identification from one through ten.

6. Building Bases

Before building the scenario bases, determine which bases will be part of the logistics network and the supply levels at those bases. Most details of the bases are not required at this point—the base may be created now and configured later—however, the types of ports the base will have must be defined now.

To create a new base or modify an existing one, from the main LogWarS window pull down the **Build** menu and choose **Base**, **Build>Base**. Alternatively, choose the **ModifyBase** button from the Scenario Construction dialog box. Either method will display the Base Data dialog box, an example of which is shown in Figure 15. If any base in the new scenario is already named in the Master Base list, construction time for the new scenario can be reduced. Like constructing a new base from scratch, a base copied from the Master Base list must be carefully examined to ensure it matches the requirements of the scenario.

To configure a base, examine each of the fields in the Base Data dialog box. If the base will have an airport, select the check box next to Airport and then press the **configair** button to configure the base's airport. Section 9 on page 38 discusses configuring a port in detail.

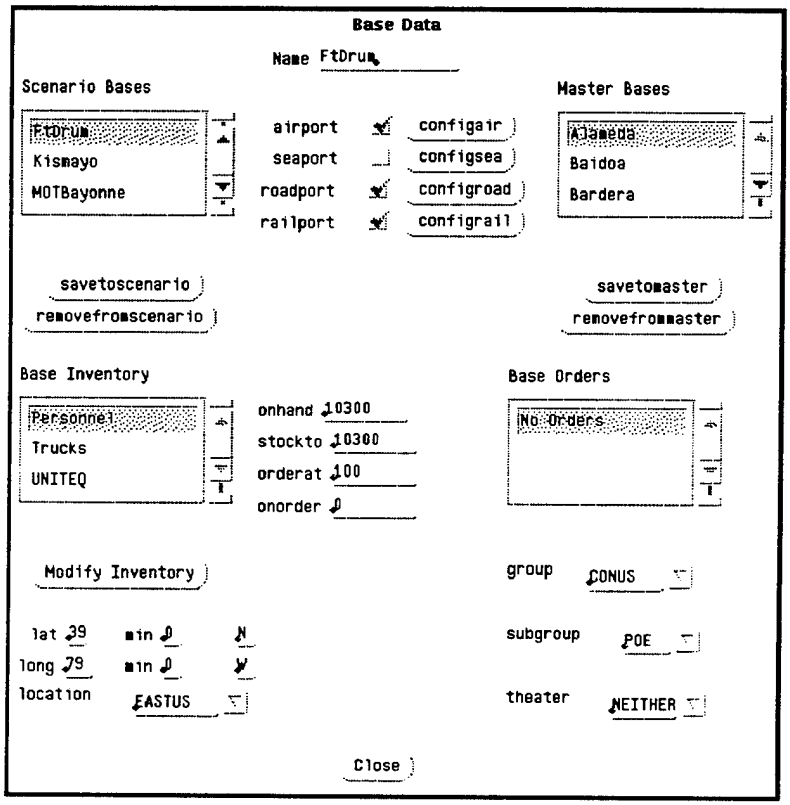


Figure 15. The Base Data Dialog Box.

The Group field in the Base Data dialog box is either CONUS, intermediate location (ILOC), THEATER, or UNIT. The group may be used by LogWarS to aid in searching for bases to provide commodities ordered by units. LogWarS ignores the group and orders the commodities directly from the origin base in the surge phase of operations, before a unit has *closed*, when most of a unit's inventory should be coming from its base of origin. When a unit issues an order for a commodity in the sustainment phase of the scenario, the bases in the THEATER group are examined first to see if they have the required commodities to fill the order. If the THEATER bases cannot fill the order, because of lack of supplies or because they are not connected to the ordering unit, the bases in the ILOC group are checked next. If these bases also fail to provide the commodity, the CONUS

bases are searched last. Bases in the UNIT group are never searched to provide commodities to fill an order thus ensuring units do not supply other units.

The Subgroup field is either port of embarkation (POE), port of supply (POS), port of debarkation (POD), or NONE. The subgroup is used by LogWarS to properly create the routing for shipments of orders. In the surge phase commodities are shipped from a unit's origin to the nearest suitable POE then to a suitable POD near the unit and finally to the unit. The POS is only used during the sustainment phase when it acts as an intermediate point to help route shipments to their destinations.

The Theater field is either regional contingency 1 (RC1), regional contingency 2 (RC2), or NEITHER. The theater is used by a portion of LogWarS which tracks closure of unit equipment into theater and the movement of unit equipment between theaters.

The Location gives a rough geographical position for the base (*i.e.*, EASTUS, WESTUS, NEUROPE, SEUROPE). The location is used by LogWarS to help properly route ship transporters around land masses and does not take the place of the latitude and longitude inputs. LogWarS will accept an input latitude and longitude for North America and a location of AUSTRALIA but the results when the scenario is executed are unpredictable.

When a commodity from the Base Inventory list is selected, the Base Data dialog box is updated to show the status of that commodity in the On Hand, Stock To, Order At, and On Order fields. To modify any of the base's commodities, press the **Modify Inventory** button. The Commodity dialog box will appear with the base's commodities in the Scenario Commodities list box. The commodities in a base inventory are used as supply dumps. The on hand, stock to, order at, and on order information is critical because this determines both the location of commodities throughout the scenario supply network and also how quickly the supplies are reordered by the bases when commodities are removed from stock to fill unit orders.

The Orders list box contains all the orders generated by the base to fill its inventory requirements. Selecting an order from the Orders box brings up the Order Details dialog box, Figure 16. The elements of the Order Details dialog box are for information only—any alteration will not effect the program.

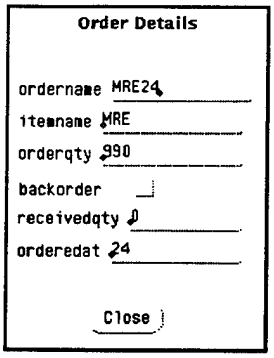


Figure 16. An Order Detail Dialog Box.

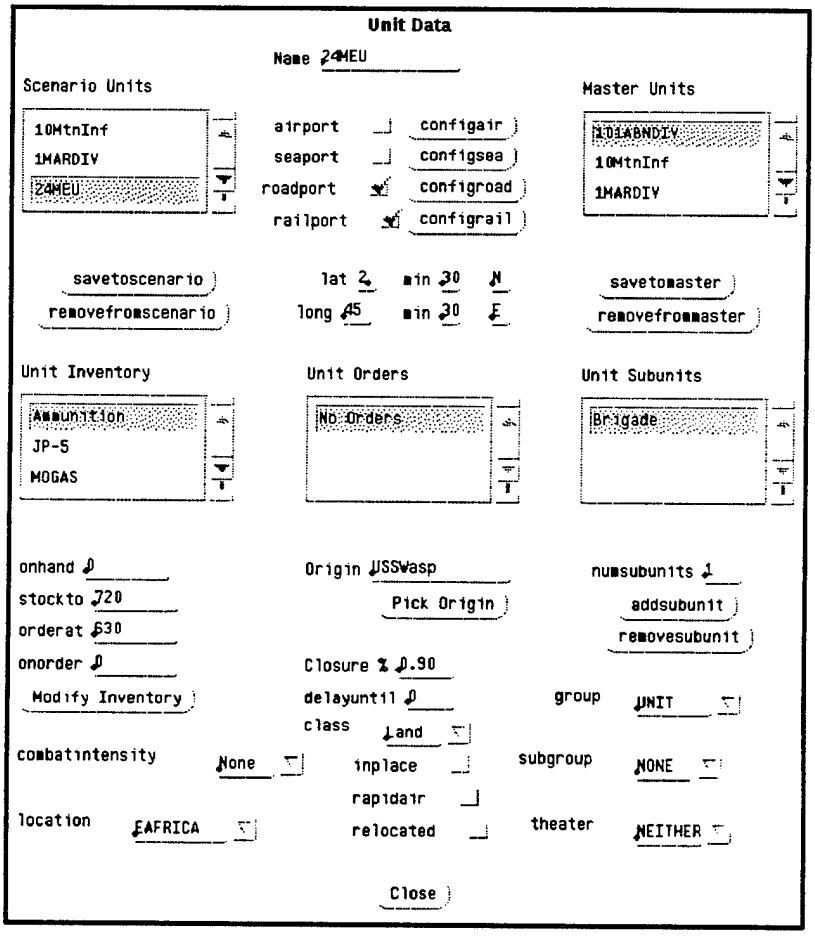


Figure 17. The Unit Data Dialog Box.

7. Building Units

The Unit Data dialog box is displayed by selecting the **Build** menu and **Unit** menu item from the main LogWarS window, **Build>Unit**, or by pressing the **ModifyUnit**

button in the Scenario Construction dialog box. Figure 17 shows the Unit Data dialog box which appears. It is similar to the Base Data dialog box so this section will only discuss the new fields and features of the Unit Data dialog box.

Similar to the Base Data dialog box, the Unit Data dialog box has a Unit Inventory list box. The commodities listed in this box represent the commodities required for this unit to function. The commodities all have On Hand, Stock To, Order At, and On Order values. Unlike a base inventory, the commodities in a unit's inventory also make use of each commodity's consumption rates. Selecting the **Modify Inventory** button brings-up the standard Commodity dialog box. The consumption levels listed for each commodity will be used to determine when the commodities need to be reordered. Consumption takes place on a daily basis. Every 24 hours the amounts of the unit's commodities are reduced by the consumption rate for each commodity in use at the end of that time period. When the commodity On Hand value reaches the Order At value an order is placed to bring the unit up to the commodity's Stock To value.

The most significant new feature of the Unit Data dialog box is the Unit Subunits list box. This list box displays the names of all the subunits that are used to construct the currently selected unit. To add subunits to a new or existing unit, first select a unit from either the Scenario Units list box or the Master Units list box. When the **Add Subunit** button is pressed, a list of all subunits in LogWarS appears in a dialog box similar to Figure 18. Either select a subunit from the list or press the **Cancel** button. If a subunit is selected a second dialog box will appear asking for the number of the subunits to add to the unit. LogWarS will accept any number but will only use the nearest integer amount.

Adding a subunit to a unit has the effect of increasing the On Hand, Stock To, On Order, and Order At levels of the commodities in the unit's inventory for all the commodities in the subunit. The new subunits will also increase the commodity consumption levels depending on the characteristics of the subunit. For example, adding 60 Airplane subunits (which have JP-5 as their only commodity) to the U.S.S. Nimitz unit will add JP-5 to the list of commodities in the inventory and all the information about JP-5 will be scaled to reflect 60 Airplanes. If the U.S.S. Nimitz already had JP-5 as a commodity, the Airplane subunits would only alter the stockage and consumption data for the JP-5 commodity to reflect the added requirements of 60 Airplanes.

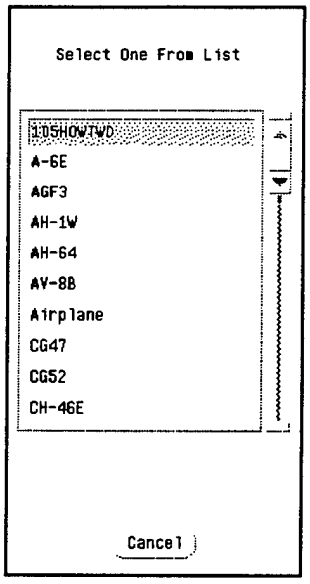


Figure 18. A Selection Dialog Box to Chose a Subunit.

Selecting a subunit name from the Unit Subunits list in the Unit Data dialog box will cause the Numsubunits field to indicate the quantity of that subunit in the unit. Pressing the **Remove Subunit** button will delete all of that particular subunit from the unit by altering the commodities of the unit's inventory to reflect the removal of all the subunits. In the example of the Airplanes in the U.S.S. Nimitz, the unit inventory would appear as if no Airplanes were part of the U.S.S. Nimitz.

The Unit Data dialog box has several new fields compared to the Base Data dialog box. The new fields deal with the movement of units to deployment locations and their actions at their deployment locations. The Delay Until field displays how many days the unit should wait from the start of the scenario until it begins to move to its deployed location. The Delay Until value may be changed by the user to control the simulation and reflect the conduct of the wargame. The Rapid Air box indicates if the personnel assigned to a unit should time their movement from the origin base to the deployed location in order to arrive at approximately the same time as the unit's unit equipment. If the box is checked LogWarS will not attempt to time the arrival and the personnel may arrive at the unit location well ahead of their unit equipment.

The Closure % value is used by LogWarS to determine when a unit has closed and should start consuming commodities at the unit's current consumption rate. A value of 1.00 represents one hundred per cent and indicates that all commodities marked deployment in the unit's inventory must be at or above their Stock To value before a unit will become

closed. A Closure % value of 0.75 indicates that only seventy-five per cent of the unit's deployment commodities must be On Hand before the unit is considered closed. The Inplace box is used during scenario execution to indicate if the unit has closed.

The Relocated box indicates if the unit has completed its movement from a deployed location in one theater to a new deployed location in another theater. This is useful information during the simulation of a multiple regional contingency scenario. Once the unit is Inplace or Relocated it begins consuming the commodities in its inventory depending on the consumption level specified by the Combat Intensity value. The Combat Intensity value can be changed by the user as the scenario is executed to simulate changing missions and varied levels of combat intensity.

8. Building Prepos

The Prepo Data dialog box, Figure 19, is used to create new scenario or master prepositioned transporters or to modify existing prepositioned transporters. The dialog box is reached from the LogWarS main window by clicking the **Build** menu and then selecting the **Prepo Ship** menu item, **Build>Prepo Ship**, or by pressing the **ModifyPrepo** button in the Scenario Construction dialog box.

When creating a prepo transporter the user is assuming the task of creating a fully loaded transporter. In so doing, most of the automatic error checking of LogWarS is bypassed. As a result, it is extremely easy to create prepo transporters that have errors. Strict attention to detail is required. Any error in creating a prepo is likely to cause program termination, either during scenario construction or execution. It is especially important to ensure the cargo and route for the prepo are suitable for the scenario. Often master prepo transporters will have commodities in their cargoes, or bases in their routes, which do not exist in the scenario. If the scenario is executed with these errors, the results will be unpredictable.

The demonstration scenario requires several prepositioned ships, one of which is the 1st Lt. Jack Lummus. Select a prepo from the Master Prepo list, save it to the Scenario Prepo list, and then select it from the Scenario Prepo list again. Begin modifying the prepo by giving it a new name. To indicate its transporter type—which provides LogWarS with the cargo capabilities of the Lummus—press the **SelectTrans** button. The dialog box which appears contains all the transporter types currently in the scenario. Select the appropriate transporter from the list. If the appropriate transporter is not in the list, any transporter can be selected. Then after the new prepo is saved, the Prepo Data dialog box

can be closed and the Transporter Data dialog box can be used to create the appropriate transporter for the scenario. It is then a simple matter to return to the Lummus and change the transporter type.

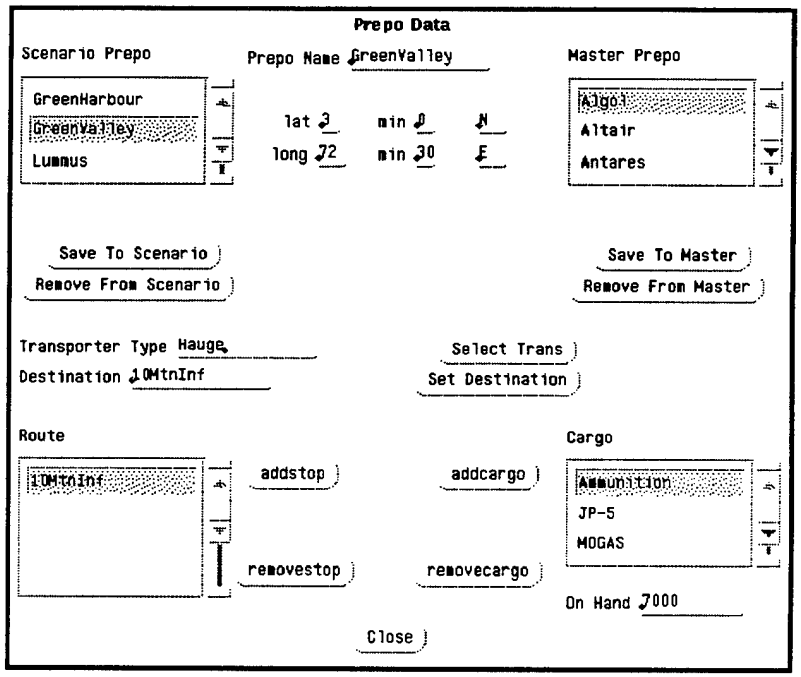


Figure 19. The Prepo Data Dialog Box.

To select the destination for the cargo aboard the Lummus, press the **Set Destination** button. A dialog box appears which contains a list of all the bases or units in the current scenario. Select the correct name from the list and that name will appear both on the Destination line and as part of the Route list if it was not already part of the route. Note, the destination does not necessarily have the proper port to handle the prepo transporter. The user is required to ensure the destination is feasible.

The user must also determine if the stops on the route of the prepo transporter are feasible. To add stops to the original route of the prepo, press the **AddStop** button. A dialog box appears which contains a list of all the bases and units in the scenario. Select the stop for the prepo transporter on its way to the destination. When one of the items is selected, it will appear in the Route list box of the Prepo Data dialog box. Repeat this process until the desired route is created. Finally, ensure the destination is the bottom entry in the Route list box. If it is not, remove the incorrect destination by selecting it in the Route list box and pressing the **RemoveStop** button. Then use the **Set Destination** button to redesignate the correct destination. When the route building process is complete,

the Route list box will contain a route in top-to-bottom order; the first stop will be at the top of the list and the destination will be at bottom of the list.

Finally, one commodity must be added to the Lummus before there is enough information for LogWarS to properly save it as a prepo transporter. Press the **AddCargo** button in the Prepo Data dialog box. A dialog box containing all the commodities available in the scenario will appear. After selecting one of the commodities, a different dialog box will appear asking for the amount of the commodity aboard the prepo. Any value greater that zero is acceptable. The new commodity appears in the Cargo list in the Prepo Data dialog box. If commodity is selected, the amount of the commodity loaded onto the Lummus appears on the On Hand line. LogWarS does not check the original cargo of prepositioned transporters to ensure the capacity of the transporter is not exceeded. It is left to the user to determine the reasonableness of the load-out of the prepo. To remove cargo from the Cargo list, select the cargo item and press the **RemoveCargo** button.

It is important that every prepo have complete latitude and longitude, transporter type, and destination information before either **Save To...** button is pressed. Failure to follow this caution will result in program termination.

When the construction of prepos is complete, save the scenario and master data by selecting the **File** menu and **Save Scenario** menu item, **File>Save Scenario**, and also the **File** menu and **Save Master** menu item, **File>Save Master,** from LogWarS main window.

9. Configuring the Ports at a Unit or Base

The Port Data dialog box appears after pressing a **Configure...** button from either a Unit Data or Base Data dialog box. It is important to remember that regardless of the configuration of a unit's or base's ports, they will not be available to LogWarS unless the ports' check boxes in the Unit or Base Dialog boxes are selected. This is one way the umpire can control the supply network's ability to move commodities. Figure 20 shows a typical Port Data dialog box.

The Maxcapacity of the port refers to the number of transporters which may be in the port at any given time. For example, a capacity of 20 for an airport means that 20 aircraft can be on the ground loading or unloading at any one time. For a railyard, a capacity of 20 would allow 20 trains of any size to be in the port at any one time. The size of the transporters does not matter in this version of LogWarS.

The Base or Unit Name field and Port Class field in the dialog box are for information purposes only. While they may be altered, doing so will not effect LogWarS in any way.

The Network list box displays the names of all the bases and units which can be reached directly from this port. Travel to the ports in the Network box is essentially a one segment trip, no stops need be made in order to reach any of the ports. To modify the network of bases or units the port is connected to, press the **Define Network** button in the Port Data dialog box. The Connected Bases and Units dialog box, Figure 21, will appear. Only the names of the bases and units that have the proper port type will appear in the Bases and Units list boxes. To add connections to the current base or unit simply select a base or unit from one of the lists and press the **Add To Network** button. To remove bases or units from the network for the current unit or base, select the base or unit in the Current Network list box and press the **Remove Link** button. When the modifications are complete, press the **Close** button. To save these changes to the port's network, press the **Save** button in the Port Data dialog box. Due to the way LogWarS models movement from one airport or seaport to another, no air or sea network is required. Accordingly, for airports and seaports, the **Define Network** button in the Port Data dialog box will be inactive.

Defining the network for each base and unit defines the transportation network for the entire scenario. To reduce wasted effort it is best to follow the scenario construction sequence in Figure 5 and wait until all units and bases have been constructed before returning to each base and unit to define its network.

Continuing the examination of the Port Data dialog box in Figure 20, the Arrivals list box indicates which transporters are waiting to unload supplies at the port. The Berths list box contains transporters currently unloading cargo at the port. The Parked list box contains all the transporters that are at the port awaiting further instructions. In the case of Camp Pendelton, truck convoys 10 through 13 are waiting for assignments. A large number of parked transporters during scenario execution is an indication of excess transportation capacity.

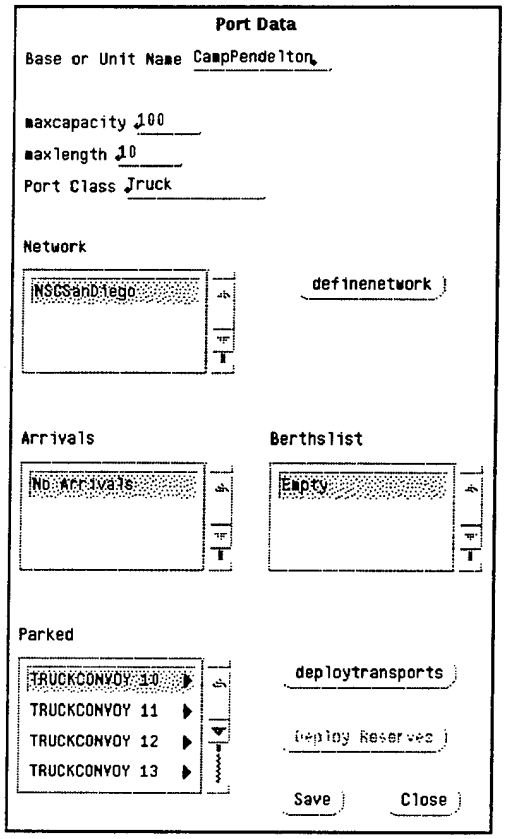


Figure 20. A Port Data Dialog Box.

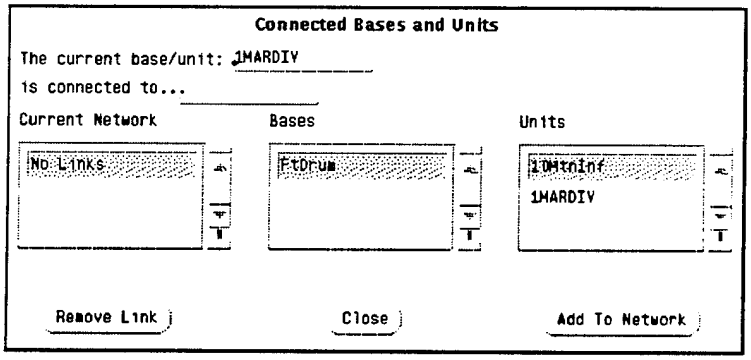


Figure 21. The Connected Bases and Units Dialog Box.

To initially place transporters at the port, press the **Deploy Transports** button in the Port Data dialog box. A selection dialog box similar to Figure 18 appears, filled with transporter types in the scenario which are able to use the port. Either select a name from the list or press the **Cancel** button. If a name is chosen, a new dialog box appears which asks for the number of the transports to place at the port. LogWarS will not allow the

maximum capacity of the port to be exceeded when the user is deploying transporters. After deploying transporters, they will appear in the Parked list box. The entry will consist of the name of the transporter, its vehicle ID number, and its ROS class combined into one word with no spaces.

In the case of air and sea ports, reserve transporters may be deployed to the port at the beginning of the scenario and given a time delay before they become available. For an airport or seaport the **Deploy Reserves** button in the Port Data dialog box will be active. Pressing the button brings up the same dialog boxes as when deploying normal transports, with the addition of a dialog box which asks for the reserve class of the transporter.

To modify any transporter list in the Port Data dialog box, simply select the desired list. The Transporter Data dialog box will be displayed with the contents of the selected list loaded into the Scenario Transporters list box. Modifications can be made to any of the transporters but changes might not take effect until after the scenario has been saved and reloaded.

D. RUNNING A LOGWARS SCENARIO

To execute a scenario, select the **Run** menu and choose the **Execute** menu item, **Run>Execute**, from the main LogWarS window after either loading the scenario (**File>Load Scenario** or press the **Load** button from the Scenario Construction dialog box). LogWarS will take a few minutes to create the necessary logistics network and to begin ordering commodities. Once the first 24 hours of simulated time have elapsed, a Time Step dialog box, Figure 22, will appear. While the Time Step box is visible, simulation time is not progressing. The user is free to examine the logistics network using choices under the **Display** menu but all selections under the **Build** menu are disabled during scenario execution.

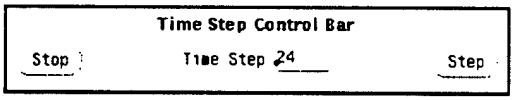


Figure 22. The Time Step Dialog Box.

To continue the scenario, press the **Step** button in the Time Step dialog box and LogWarS will advance the scenario by the number of hours indicated in the Time Step field. To halt the current scenario the user must wait for the Time Step dialog box to become visible then press the **Stop** button.

1. Supply Status Display

To display the supply status of a base or unit at the current simulation time, from the main LogWarS window pull down the **Display** menu and choose **Statistics**, **Display>Statistics**. This will fill the Supply Status Information window, an example of which is shown in Figure 23. The window consists of eight bar graphs each representing one of the eight classes of supplies used by LogWarS. Each bar depicts the percentage of commodities in a class of supply on hand compared to the commodities' stock to requirements—the supply status. In the example shown in Figure 23, at 75 days into the scenario the 1st Marine Division has all its personnel but few MREs—part of the FF&V class—and little ammunition. This status indicates something amiss in the scenario—insufficient transporters, insufficient stocks of FF&V and ammunition, or, possibly, an inadequately connected logistics network.

The status of single bases and units or combinations of bases and units may be displayed. The Supply Status Window shows a snapshot of a base's or unit's supply level at a specific time and is useful because it can be displayed rapidly.

2. Closure Display

To display the closure and historical supply status of a base or unit, from the main LogWarS window pull down the **Display** menu and choose **Closure**, **Display>Closure**. This will fill the Closure Status Information windows, one of which is shown in Figure 24. The Closure Status Information consists of two windows—each graphically depicts four of the eight classes of supply used in LogWarS. Figure 24 shows one of the windows filled with information from the 1st Marine Division.

Each graph in the Closure Display shows the supply status for a single supply class (on the vertical axis) versus time (on the horizontal axis). Once a user-defined amount of deployment commodities are on hand, a unit is considered closed and it begins to consume the commodities in its inventory. (A discussion of deployment commodities and unit closure levels can be found on page 25.)

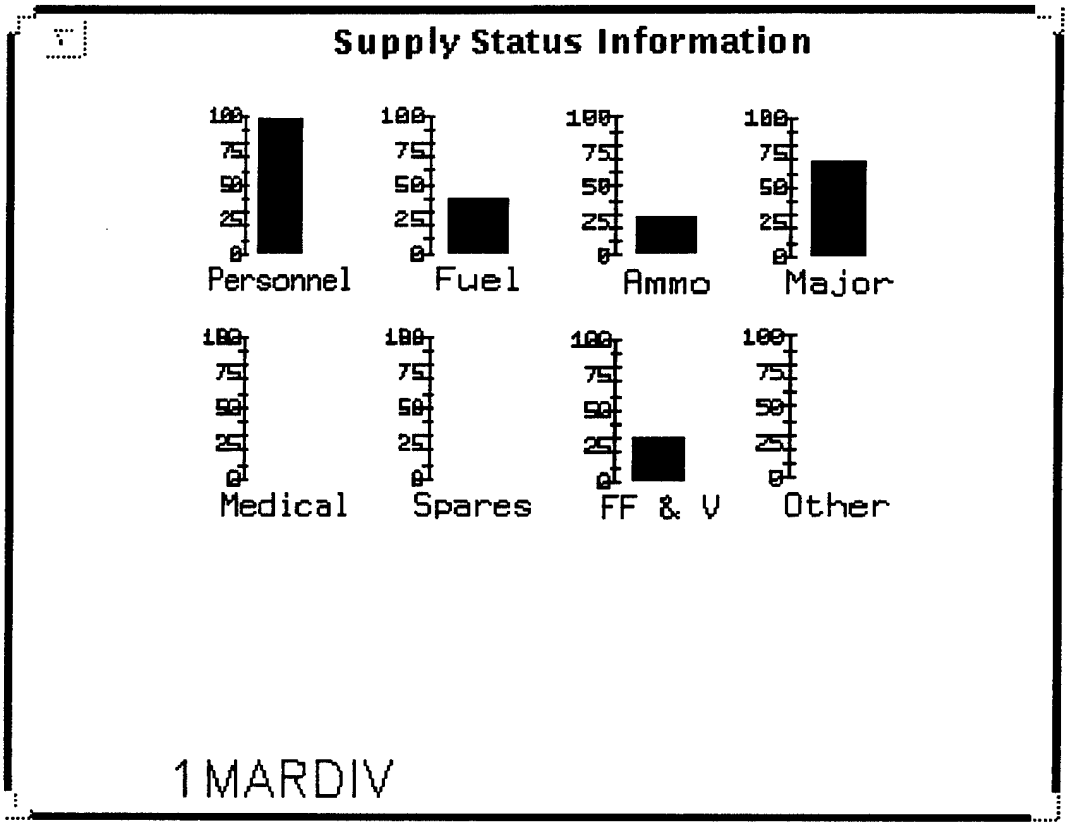


Figure 23. The Supply Status Window.

The 1st Marine Division, in Figure 24, attained closure—100 per cent of the required amount of its deployment commodities on hand in this case—approximately 500 hours into the scenario. From 500 hours onward, the graphs depict the 1st Marine Division's supply status over time and show the unit rapidly consuming and reordering commodities. The commodities in both the Fuel and the Ammunition classes of supply have been reordered several times—note the increase in supply level at several points—while the commodities in the Personnel and Major classes have not reached their reorder points yet.

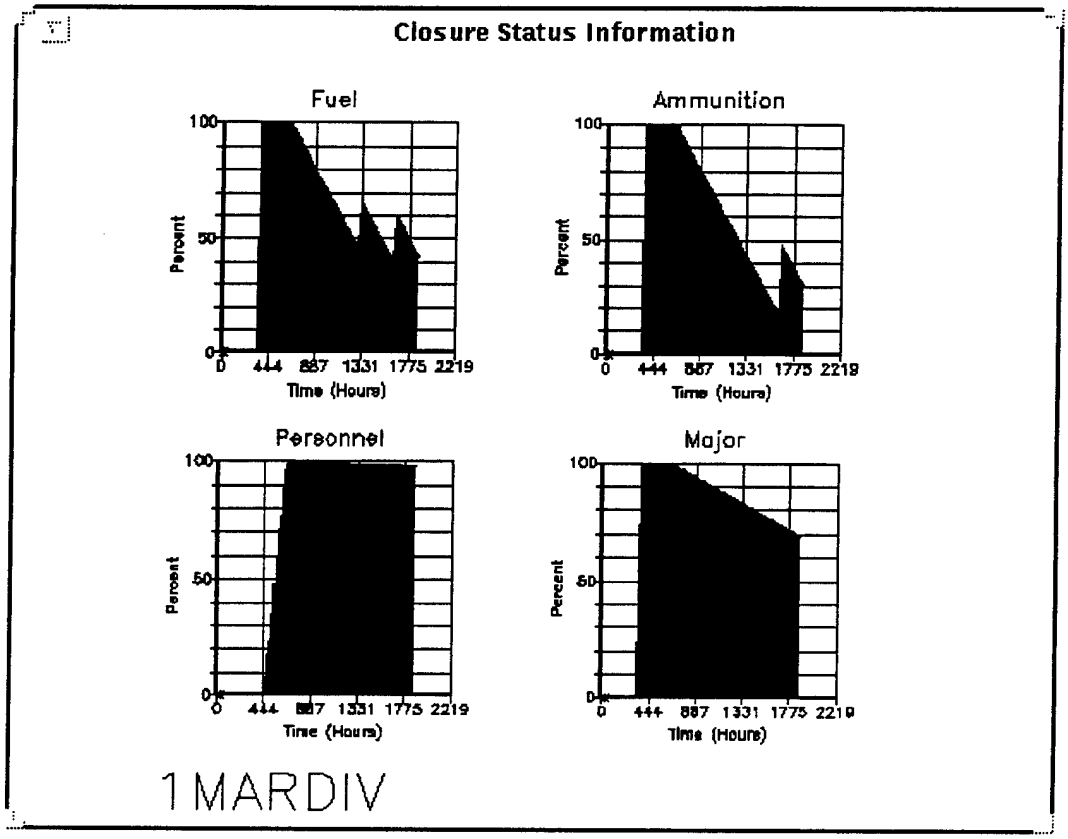


Figure 24. A Closure Display Window.

The inability of the logistics network to keep the 1st Marine Division supplied with fuel and ammunition indicates the need for either more transporters to carry these commodities or greater amounts of the commodities stocked at bases near the 1st Marine Division. The Closure Display can show the historical supply status for individual units or bases in the scenario as well as for all units or bases combined. It provides a great deal of information but requires more time to update its display than the Supply Status window.

3. Port Performance Display

To display the performance of a particular base's ports, from the main LogWarS window pull down the **Display** menu and choose **Closure**, **Display>Port Performance**. This will display the Base Port Performance dialog box, an example of which is shown in Figure 25.

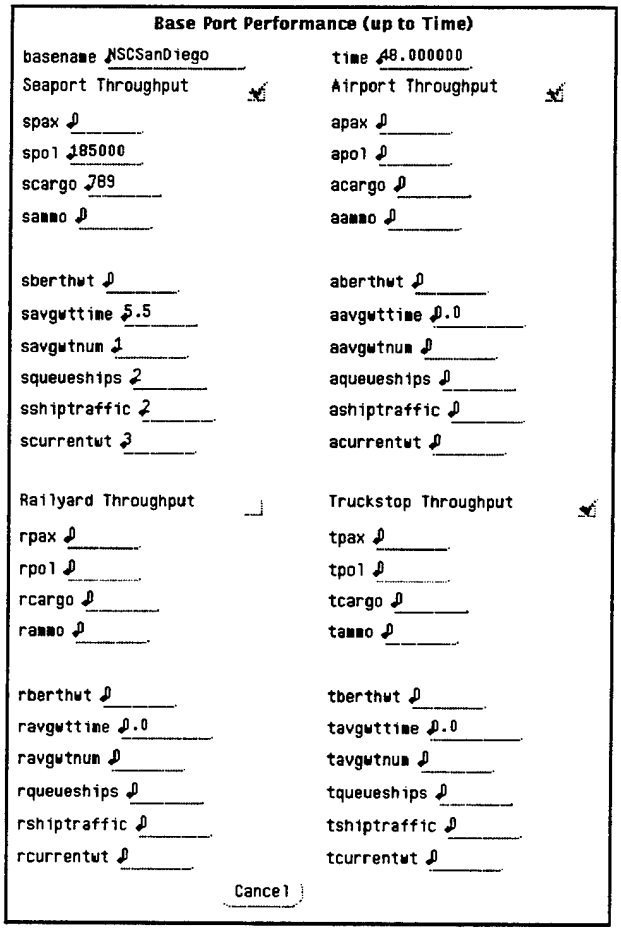


Figure 25. The Port Performance Dialog Box.

The performance of NSCSanDiego, in Figure 25, indicates that none the 1st Marine Division's 10,000 personnel moved through during the first 48 hours but most of its POL and unit equipment did. The ship transporters waited an average 5.5 hours for berth space at San Diego and there are three ships currently waiting. This sort of wait for berthing space indicates the port likely needs to have its maximum capacity increased. Halvorson [Ref. 6, p. 80] thoroughly discusses the limitations of the performance statistics and should be consulted to understand the uses of the statistics because some of them can be misleading.

4. Base, Unit, and Transporter Displays

While the simulation is paused at a particular time, the user can select any of the items from the **Display** menu. Each of the items—Base, Unit, or Transporter—brings up the appropriate Data dialog box and provides a way to examine or modify the details of the

elements of the scenario. Figure 26 shows an example of a Unit Data dialog box displayed by selecting the **Display** menu and the **Unit** menu item from the LogWarS main window, **Display>Unit.**

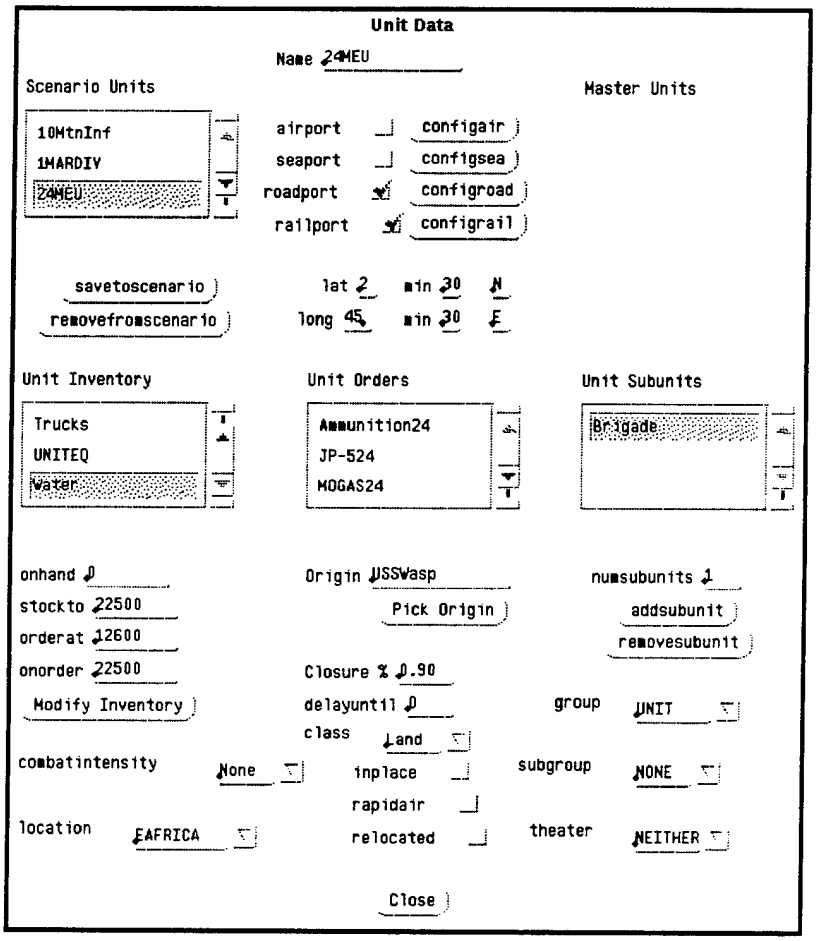


Figure 26. The Unit Data Dialog Box from the **Display>Unit** Menu.

The Unit Orders list box for the 24th MEU, Figure 26, indicates the unit ordered Ammunition, JP–5, and MOGAS 24 hours into the simulation. At the time this dialog box was displayed, the 24th MEU has no Water on hand (Water is selected in the Unit Inventory list box and there is a zero on the On Hand line). To verify that Water was ordered by the 24th MEU, scroll down the Unit Orders list box to find a Water entry.

The dialog boxes which appear during scenario execution are similar to the dialog boxes used to construct scenario elements. The only difference is the lack of Master list boxes in the upper right corner of the dialog boxes. The user should exercise caution modifying a scenario while it is being executed. For example, deploying additional

transporters to a port should cause no difficulty but creating a new type of transporter and then deploying it to a port may result in program errors. While the scenario is running it is best to limit changes to modifications of the elements which currently exist.

IV. CONCLUSION

LogWarS is an evolution of the S3 logistics wargaming tool which improves the user's ability to both enter and interpret data. LogWarS' graphical interface is functional and a great improvement over the command-line interface of S3. LogWarS' graphical display of results greatly increases comprehension of logistic network performance. The alterations to S3's data structure now allow the user to store multiple scenarios with little concern for interference between them. The user is also able to make use of a master data base of logistic elements which may be reused to aid in the construction of many different scenarios.

The alterations to S3 resulting in LogWarS have increased the useability of the program and shown several areas which require further attention. The most important future change to LogWarS involves its network structure. LogWarS currently forces the player to know exactly how supplies should flow before the game is built. A better system would allow the player to input the bases or units that exist and the connection from them to a world-wide network of geographic points. Once this network is established, LogWarS could decide the correct routing. The goal should be to provide supplies to theater via a logistics network, not to properly connect the supply bases of that network. This new network architecture would have other benefits. With the free connection of bases it would be easier to model the shifting of units from one MRC to another. Further, the new structure would allow analysis of the network for feasibility and collection of information about the expected performance of the network. Statistics such as the maximum capacity and bottlenecks in the system could be easily gathered and presented to the user before the scenario is executed.

While LogWarS is easier to use than S3, it still has far to go before it can be described as easy to use. Its sensitivity to the correctness of user inputs greatly hinders its usefulness for supporting real wargames and makes it difficult for the average user to operate. In computer terms, the current version of LogWarS is in the development phase and not ready for public release.

LogWarS is designed to bridge the gap between a rough feasibility-checking program and a full-scale logistics planning tool using TPFDD level detail inputs. The strength of LogWarS is its ability to fill the middle ground—to provide a high level of fidelity while at the same time not requiring a tremendous amount of data input. In this in-between area, where time and information may both be in short supply, LogWarS shows

how to fill the gap and indicates a promising direction for the development of future wargame logistics support tools.

APPENDIX A. ASSUMPTIONS AND LIMITATIONS OF LOGWARS

Documenting the underlying assumptions and limitations of LogWarS is essential to its proper use. The following are some of its major underlying assumptions.

The simulation does not include any stochastic elements. If LogWarS is run one or more times without changing the input data and with the same game inputs, the exact same final results will be attained. This means no real analysis can be done with the data that comes from LogWarS. All solutions will be point solutions with no indication of how much confidence can be placed on the results. Because of the lack of a stochastic element, this model should not be used to predict real world results.

Each Base has only one port of each type—*i.e.*, only one seaport. This requires the modeler to aggregate the capacities of the port type if more than one should exist. This is likely to be a problem with truckstops since many can exist at a given base but only one can be modeled.

The only constraint on transporters using ports is the number of berths or parking spots. Once all the berths are occupied no new transporter can enter the port until one leaves the port. While this is a realistic constraint, the ability to further limit the types of transporters which can use a port is desirable. For example, any aircraft transporter in LogWarS can land at any base or unit with an Airport. LogWarS will send strategic lift aircraft to deployed unit locations whether or not this is reasonable.

Transporters are not constrained by range or fuel requirements. Any transporter can travel as far as the network will allow it. Ships and aircraft transporters are able to transit around the world without having to stop for fuel.

If all the bases of the simulation are connected by rail or road links, a train or convoy can start at one end of the rail or road network and continue all the way to a destination on the other end of the network.

Supply routes take travel distances and land masses into account in only a rudimentary way. The distance between points is calculated using great circle distances. For bases or units close together in open terrain, or for air routes, this is not an unreasonable method. The great circle distances are not as reasonable for distances between locations in rough terrain or for strategic sealift. LogWarS does use a few preset waypoints to make the strategic sealift distances more realistic than using great circle distances. This prevents most ships from steaming through a land mass but is still not completely accurate.

There is no facility for manually routing shipments of commodities. Regardless of a player's desires, LogWarS will ship supplies to their destinations by a route it deems best. At this point in the development of LogWarS, the shipment routing logic does not attempt to produce an optimal route.

The logic for allocating transporters is crude. Transporters are assigned to cargo shipments. Once a shipment has a transporter assigned, this assignment is not altered. If no transporters are immediately available for a cargo at a port, one could be requested from the other side of the world. This is true even if an appropriate transporter in the port will be available shortly after the request is made. The result is cargo waiting at a port for a transporter while an appropriate transporter is waiting in the port. This myopic assignment logic can produce wildly inaccurate times to transport supplies from origin to destination.

Transporters moved to a base or unit as commodities will not automatically become transporters once they reach their destination. The only way to model the movement of transportation assets to a location in LogWarS is to create a commodity with the transportation asset's name and, when the commodity arrives at the base or unit, manually creating transporters. In this manner, LogWarS can simulate the movement of trucks in an Army truck company from their home base to a deployed location.

The model assumes that the subset of all commodities used for the simulation will effect the transportation network in the same manner as the real world would be effected by all required shipments for the scenario being simulated. In the real world, the vast number and types of supplies can overwhelm a logistics network. LogWarS assumes the few commodities that are used in the scenario will effect the transportation network in the same way. There are two ways to stress the transportation network in LogWarS to make it more accurate. One way is to artificially constrain the transportation assets available, using trial and error to set the number of transporters to a level which does not allow unrestricted movement of commodities. The other way to stress the system is to add a dummy commodity which occupies a large physical space and is ordered by all units.

One of the goals of using LogWarS is to learn some of the important logistic considerations in a conflict. An important lesson for umpires and players alike is the importance of usage rates for various supplies. Without good usage rates there can be no prediction of the need for resupply; the system would be mere guess–work. Therefore, an important learning objective is to get the player to think about how much of a commodity a unit uses. The most thorough source for usage rates is FM-101-10-1/2, the Army's *Staff Officers' Field Manual: Organizational, Technical, and Logistical Data Planning Factors.*

[Ref. 9] Data for unit sizes, unit lift requirements, and transporter capabilities can be found in other unclassified literature such as the Military Traffic Management Command's *Deployment Planning Guide* [Ref. 11] and the U.S. Army Command and General Staff College's *G1/G4 Battle Book* [Ref. 12]. Thinking about commodity usage also forces the player to think about whether the usage measurement is appropriate, *i.e.*, using x number of surface–to–air missiles per day as a measure as opposed to using x number of surface–to–air missiles per raid of aircraft.

The consumption of commodities occurs at a user defined rate— a specified amount of commodity per day. This concept is acceptable for commodities like food which can conceptually have their consumption evenly spread over a 24 hour period. For certain commodities, such as smart ordnance, the consumption should be based upon actual threats encountered. For example, a Patriot missile battery may respond to one air attack in a week and use seven missiles in that attack. For the Patriot battery this equates to a consumption rate of one missile per day. While the logistics network may be able to support a demand for one missile each day it may be completely unable to support the demand for seven missiles in one day. The difference between the two concepts is one of continuous versus discrete consumption. LogWarS uses continuous consumption automatically and relies upon the umpire to manually enter changes in a unit's inventory to reflect discrete consumption such as the Patriot missile battery responding to attack.

LogWarS generally models a pull-type supply system. Halvorson [Ref. 6, p. 20] showed it can be used to crudely model a push-type logistics system by manually prepositioning supplies. There is no other method to send supplies to a particular unit before it identifies a need for those supplies and generates orders for them.

There is no way to automatically predict when a unit will attain closure because it is not possible to predict the travel time for shipments for more than one travel segment at a time. When a shipment must move by several different modes of transportation, LogWarS can indicate the travel time for the current transporter but not for the amount of time required to wait for and load on the next transporter.

The point at which a unit is considered closed is problematic for LogWarS as well as in reality. There is no generally agreed upon standard to define the point at which a unit is closed. Closure depends upon the individual unit, the relation between the amount of commodities requested and those on hand, the commodity in question, the individual commander's desires and aggressiveness, and the requirements of the operations plan for the scenario. An operating definition of closure for use in LogWarS is that closure occurs

when a unit first receives a user-defined percentage of its commodities that are marked as deployment items. Once a unit is closed, it begins to consume its commodities at the consumption rate appropriate to the level of combat activities. From this point in time until the unit moves again, the unit is concerned with sustainment—maintaining the stocking levels of its commodities.

Finally, the structure of the program does not allow stopping the game in mid–play. The way the program is currently designed there is no way of saving the data to a file and then retrieving it again later to restart the program where play stopped. The problem involves the inability to save the list of pending activities. This list contains information about the various parts of the program that are scheduled to occur at some future time. It is not created by LogWarS but is part of the MODSIM programming language that was used to create the program. Because of this, interrupting the simulation causes all future-scheduled actions to be lost. The practical significance for the wargame umpire is that during a long wargame the computer should not be shutdown until LogWarS is no longer needed. When LogWarS is running in the UNIX environment, this is not a problem since the computer is always running.

APPENDIX B. DESCRIPTION OF FILES

The LogWarS executable file and its associated preference file (pref.dat) must be located in the same directory on the hard drive. The master data file subdirectory and the scenario data file subdirectories are also maintained in the executable file's directory.

LogWarS uses an element's name to distinguish it from other elements of the same type. When storing an element for later reuse, LogWarS will only use the first eight characters of the name as a data file name. Thus if a base, a unit, and a transporter all used the same first eight characters for their name, only one data file will be created. Which data file is created is unpredictable. It is imperative to use a unique sequence for the first eight characters of all the elements' names. Because of differences in operating systems, it is also best not to rely on upper-case and lower-case differences between names.

1. .MST FILES

These are the files that contain the master data base for the program. They are located in the master data file subdirectory and are identical to the scenario files. The only unique master data file is the Scenes.mst file, Figure 27, which contains the names of all the scenarios in LogWarS and the directory they are loaded in.

```
NumberOfScenarios: 3
S3DEMO ../halvorson/
Somalia ../halvorson/somalia/
Test ../halvorson/test/
```

Figure 27. The Scenes.mst Data File.

2. .SCN FILE

These files are located in each scenario data file subdirectory. Each file, depicted in Figure 28, contains a list of the names of the other main files for that particular scenario. LogWarS uses this file to determine which other files to open during scenario construction.

3. .CMD FILE

All the commodities for the Master Commodity List are stored in the Commods.mst file in the master data file subdirectory. The scenario-specific commodities which appear in the Scenario Commodity List are stored in the *ScenarioName*.cmd file in the scenario data

file directory. Both files store the same information in the same format. Figure 29 displays two commodities stored in the Commods.mst file.

```
Somalia.cmd
Somalia.trn
Somalia.bse
Somalia.unt
Waypoint.mst
Somalia.blk
Somalia.rlk
Somalia.tlk
Somalia.ppo
```

Figure 28. The Contents of the Somalia.scn File.

```
NumberOfCommodities: 77

Name: Mk-82 Class: Ammo ProduceAt: 250.00
Length: 72.00 Width: 24.00 Height: 24.00
Weight: 500.00 Priority: 4 EmerPriority: 1
OutSize: FALSE

Name: Water Class: Fuel ProduceAt: 1000000.00
Length: 0.00 Width: 0.00 Height: 0.00
Weight: 300.00 Priority: 5 EmerPriority: 4
OutSize: FALSE
```

Figure 29. A Commods.mst Data File.

Note that the file, like all LogWarS data files, the Commods.mst file begins with a number to indicate how many records are in the data file.

Once commodities are added to units or bases, they are also stored as part of the unit or base data file. Figure 30 displays part of the commodity data for CVN–68.

Note that size and priority data are stored in the scenario commodity master file (Somalia.cmd) while stocking level and consumption data is stored in the unit or base data file. Thus, units and bases in a given scenario can have different requirements for commodities but the commodity will always have the same size and priority throughout all the bases and units in a scenario.

```
Commodities: 16

F44
StockTo 60119.00 Deployment: FALSE
HighRate: 0.000 MedRate 0.000 LowRate 0.000 NoneRate 0.000

20MM
StockTo 12.00 Deployment: FALSE
HighRate: 0.000 MedRate 0.000 LowRate 0.000 NoneRate 0.000
```

Figure 30. Part of the CVN–68 Data File.

4. .TRN FILE

Scenario-specific lists of transporter types are stored in the *Scenario.trn* data file in the scenario data file subdirectory. A master list of transporter types available to LogWarS is stored in the *Transports.mst* data file in the master data file subdirectory. Both files contain information in the same format and are similar to Figure 31.

```
NumberOfFiles: 8

C-141B.dat
C-5A.dat
C-17.dat
FREIGHTT.dat
PAXTRAIN.dat
SL-7.dat
TRUCKCON.dat
B747.dat
```

Figure 31. A Scenario.trn Data File.

Each individual transporter data file is similar to Figure 32.

5. .BSE FILE

The names of all the bases used in a given scenario are stored in the *Scenario.bse* file. The names for all the master bases are stored in the Base.mst data file in the master files subdirectory. Figure 33 depicts the contents of an early base data file for the Somalia scenario. The individual bases have separate data files. Figure 34 depicts the data for Fort Drum stored in the FtDrum.dat file.

```
Type: C-141B
Class: Aircraft
SubClass: General
Length: 168.00 Width: 160.00
MaxSpeed: 425.00 MaxRange: 1500.00

MaxCargoArea: 878.00 MaxCargoCube: 7024.00 MaxCargoWeight: 90200.00
MaxCargoLength: 1120.00 MaxCargoWidth: 123.00 MaxCargoHeight: 109.00
 MaxPax: 0.00 MaxGas: 0.00
OverSize: FALSE
```

Figure 32. The C–141B Data File.

```
NumBases: 2

FtDrum
MOTBayonne
```

Figure 33. The Somalia.bse Data File.

```
FtDrum

Group:  CONUS SubGroup:  NONE  Location:  EASTERNUS Theater: NEITHER

Latitude:  39   0 N
Longitude:  79  0 W

HasAirport: TRUE  MaxCapacity: 10 MaxSize: 1500.00
HasSeaport: FALSE  MaxCapacity: 0 MaxSize: 0.00
HasRail: TRUE  MaxCapacity: 20 MaxSize: 40.00
HasTruckStop: TRUE  MaxCapacity: 20 MaxSize: 20.00

TransporterTypes: 2

B747 3
UNITTRAIN 10

Commodities: 3
DRAGON
OnHand: 2500.00 StockTo 2500.00
OrderAt: 2200.00 EmerOrderAt 2000.00

Personnel
OnHand: 10300.00 StockTo 10300.00
OrderAt: 10000.00 EmerOrderAt 2575.00

TOW
OnHand: 4500.00 StockTo 4500.00
OrderAt: 4200.00 EmerOrderAt 4000.00

 NumberReserveTypes:  0
```

Figure 34. The FtDrum.dat Data File.

The contents of the base data file are largely self-explanatory. Fort Drum is part of the CONUS group of bases and is not a member of a subgroup. It is located approximately in EASTERNUS—the Eastern United States—with the latitude and longitude shown. Fort Drum has an airport with a capacity of 10 aircraft. It does not have a seaport but does have both a railyard and a truck stop. The data file indicates Fort Drum has two types of transporters located at the base when the scenario begins. There are three Boeing 747 aircraft and 10 unit equipment trains. The Fort Drum data file contains several commodities that are not part of the Somalia scenario. Because of these commodities, Fort Drum must be modified in LogWarS before the Somalia scenario can be executed. The final entry, NumberReserveTypes, indicates there are no reserve ships or aircraft at Fort Drum when the scenario begins.

6. .UNT FILE

The names of all the units in a given scenario are stored in the *Scenario.unt* data file. The names for all the master units are stored in the Units.mst data file in the master files subdirectory. Typical contents for either of these files appears in Figure 35.

```
NumberOfUnits: 4

82ABNDIV
226TFW
AF-Blue
VP-1
```

Figure 35. A Scenario.unt Data File.

The contents of a typical unit's data file are depicted in Figure 36. The unit data file elements not part of the base data file include: Class, DelayUntil, Inplace, ActiveAt, InitialCombatIntensity, RapidAir, and Subunits.

7. .BLK, .RLK, AND .TLK FILES

These files are unique to each scenario. The base link (.blk) file, Figure 37, contains the information to connect units to their base of origin, while the rail link (.rlk) and truck link (.tlk) files, Figure 38, contain the rail and road network for the scenario.

The base link file consists entirely of unit and origin pairs for every unit in the scenario. The rail and truck link files consist of base and unit names followed by all the

other units or bases which are connected base. The .rlk files are identical in form to the .tlk data files.

```
82ABNDIV
Class:  Land
  Location:  MIDDLEEAST
 Theater:  RC1

Latitude:  25   0 n
Longitude:  67  0 e

DelayUntil: 100.00
InPlace: FALSE
ActiveAt: 0.90
InitialCombatIntensity: None

HasAirport: FALSE  MaxCapacity: 0 MaxSize: 0.00
HasSeaport: FALSE  MaxCapacity: 0 MaxSize: 0.00
HasRail: FALSE  MaxCapacity: 0 MaxSize: 0.00
HasTruckStop: TRUE  MaxCapacity: 20 MaxSize: 20.00

TransporterTypes: 2

TRUCKCONVOY 10
TANKTRUCKS 5

Commodities: 2

Personnel
HighRate: 301.00 MedRate 172.00 LowRate 117.00 NoneRate 3.00
OnHand: 0.00 StockTo 11800.00
OrderAt: 11574.00 EmerOrderAt 6000.00
Deployment: TRUE

AH-64
HighRate: 0.00 MedRate 0.00 LowRate 0.00 NoneRate 0.00
OnHand: 0.00 StockTo 33.00
OrderAt: 32.00 EmerOrderAt 25.00
Deployment: TRUE

Not RapidAir
 NumberReserveTypes:  0

Subunits: 0
```

Figure 36. The 82nd Airborne Division Data File.

The .tlk data file in Figure 38 shows Baidoa and Mogadishu are connected by a road and that Mogadishu is also connected to both the 1st Marine Division and the 24th Marine Expeditionary Unit.

```
NumUnits:  9

Unit:  BaidoaPeople

Origin:  Baidoa

Unit:  KismayoPeople

Origin:  Kismayo

Unit:  MogadCivilians

Origin:  Mogadishu
```

Figure 37. A Portion of a .blk Data File.

```
NumBases:  22

Base:  Baidoa
NumNodes:  3

Mogadishu
10MtnInf
BaidoaPeople

Base:  Mogadishu
NumNodes:  5

Baidoa
Bardera
1MARDIV
24MEU
MogadCivilians
```

Figure 38. A Portion of a .tlk Data File.

8. .PPO FILE

All scenario prepo names are stored in a *Scenario.ppo* data file in the scenario data file subdirectory. The master prepo names are stored in the Prepo.mst file in the master files subdirectory. Figure 39 depicts a typical Prepo.mst file. Information about each of the specific prepo transporters is contained in a *name.dat* file either in the data file subdirectory or master data file subdirectory. Figure 40 displays the contents of the Lummus.dat file.

```
NumberOfFiles: 13

Bonneyman
Altair
Algol
Antares
Bellatrix
Capella
Denebola
Pollux
Regulus
PVTHauge
Baugh
Philips
Lummus
```

Figure 39. The Contents of a Typical Prepo.mst Data File.

```
Lummus

Transporter:  Hauge
Latitude:    7   0  S
Longitude:  72  30  E

Cargo Destination:  10MtnInf
Cargo Routing:  1
10MtnInf

Cargo:  1
Ammunition  10.000000
```

Figure 40. The Contents of the Lummus.dat Prepo Data File.

At this point the Lummus has only its destination for a route and one type of cargo—ten units of Ammunition.

9. SUBUNIT FILES

All the subunits for the Subunit Data dialog box are stored in the SubUnits.mst file in the master data files subdirectory. Figure 41 displays several subunits stored in the SubUnits.mst file. Each of the listed subunits is stored as a separate data file with the name indicated—*i.e.*, FA-18E.dat is the name of the file which contains the specific information for the FA–18E subunit. Figure 42 displays the contents of the FA–18E.dat file and shows that an FA–18E is a collection of both AIM–9 and JP–5 with no stocking requirements but with consumption rates. The size and priority data for the two commodities are stored in the scenario commodity master file.

```
NumberOfUnits: 5

FA-18E.dat
F-14D.dat
A-6E.dat
EA-6B.dat
E-2C.dat
```

Figure 41. A SubUnit.mst Data File.

```
FA-18E
Class:  Air

Commodities: 2

AIM-9
StockTo 0.00 Deployment: FALSE
HighRate: 4.000 MedRate 2.000 LowRate 1.000 NoneRate 0.000

JP-5
StockTo 0.00 Deployment: TRUE
HighRate: 1000.000 MedRate 800.000 LowRate 400.000 NoneRate 400.000
```

Figure 42. The FA–18E Data File.

LIST OF REFERENCES

1. Perla, P. E., *The Art of Wargaming*, Naval Institute Press, 1990.

2. Van Creveld, M., Supplying War: Logistics from Wallenstein to Patton, Cambridge, 1977.

3. Vlahos, M., *Wargaming, an Enforcer of Strategic Realism: 1919–1942*, Naval War College Review, pp. 7–22, Vol. 39, March 1986.

4. Voosen, B.J. and Corona, D., *MISSLOGS: A Game of Missile Logistics*, RAND Corporation, 1959.

5. Long, J. A., *Modeling Theater Level Logistics for Wargames*, Master's Thesis, Naval Postgraduate School, Monterey, CA, 1993.

6. Halvorson, T. G., *Improvement to a Surge and Sustainment Model for Wargaming*, Master's Thesis, Naval Postgraduate School, Monterey, CA, 1994.

7. CACI Products Company, MODSIM II: The Language of Object-Oriented Programming: Reference Manual, 1993.

8. CACI Products Company, SIMGRAPHICS II: User's Manual for MODSIM II, 1992.

9. Department of the Army, *Staff Officers' Field Manual: Organizational, Technical, and Logistical Data Planning Factors*, FM 101-10-1/2, 1987.

10. Polmar, N., *The Aircraft and Ships of the U.S. Fleet*, Naval Institute Press, 1987.

11. Military Traffic Management Command, *Deployment Planning Guide*, Newport News, VA, 1991.

12. U.S. Army Command and General Staff College, *G1/G4 Battle Book*, Fort Levenworth, KS, 1992.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5101	2
3.	Defense Logistics Studies Information Exchange U.S. Army Logistics Management Center Fort Lee, VA 23801-6043	1
4.	PROF David. A. Schrady, Code OR/So Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	2
5.	PROF Thomas. E. Halwachs, Code OR/Ha Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
6.	LCDR Terry Redman, Code OR/Rt Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
7.	Deputy Chief of Naval Operations, N424D ATTN: CDR Kevin Cheezum, USN Department of the Navy Washington, DC 20350-2000	1
8.	President Naval War College ATTN: WGD 334L 686 Cushing Road Newport, RI 02841-1207	1
9.	Mr. Jerry Goldstein, N814 Office of the Chief of Naval Operations 2000 Navy Pentagon Washington, DC 20350-2000	1
10.	Captain Neal T. Lovell TRAC-Monterey P.O. Box 8692 Monterey, CA 93943-0692	2

11. LT Curtis D. Plunk
c/o Mrs. Karen Nelson
832 South Irving Street
Arlington, VA 22204 1