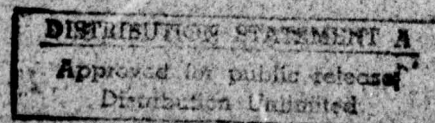AD-E 500017

IDA PAPER P-1221

IDAHEX

A MANEUVER-ORIENTED MODEL OF
CONVENTIONAL LAND WARFARE

VERSION 1.0

Volume 3: Player's Manual

Paul Olsen

November 1976

INSTITUTE FOR DEFENSE ANALYSES
PROGRAM ANALYSIS DIVISION

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1 REPORT NUMBER P-1221	2. GOVT ACCESSION NO.	3 RECIPIENT'S CATALOG NUMBER
4 TITLE (and Subtitle) IDAHEX: A Maneuver-Oriented Model of Conventional Land Warfare. Version 1.0, Volume 3: Player's Manual.		5 TYPE OF REPORT & PERIOD COVERED Final rept. 6 PERFORMING ORG. REPORT NUMBER P-1221-VOL-3
7 AUTHOR(s) Paul Olsen		8 CONTRACT OR GRANT NUMBER(s) IDA Independent Research Program
9 PERFORMING ORGANIZATION NAME AND ADDRESS Institute for Defense Analyses Program Analysis Division 400 Army-Navy Drive, Arlington, VA 22202		10 PROGRAM ELEMENT PROJECT TASK AREA & WORK UNIT NUMBERS N/A
11. CONTROLLING OFFICE NAME AND ADDRESS		12 REPORT DATE November 1976 13 NUMBER OF PAGES 61 p.
14 MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) IDA/HQ SBIE 77-19001, AD-E500 017		15 SECURITY CLASS (of this report) Unclassified 15a DECLASSIFICATION DOWNGRADING SCHEDULE N/A

16 DISTRIBUTION STATEMENT (of this Report)

This document is unclassified and suitable for public release.

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

DDC
RECEIVED
MAR 3 1978
A

18 SUPPLEMENTARY NOTES

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)
Land Warfare, Ground Combat, Simulation, Interactive Model, War Game, Computer-Assisted War Game, Ground Forces, Amphibious Landings, Airborne Operations, Maneuver

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

IDAHEX is an interactive computer model of two-sided conventional land warfare. It keeps the players informed of the situation and accepts their instructions to their forces. Units can move by land, sea, or air. A unit's movement rate is variable, depending upon its posture, the conditions of its movement, and the adequacy of transport. Attrition in engagements is assessed by a heterogeneous Lanchester square process. Air support can be played. Supplies consumption

403 219   JOB

20. (continued)

can be assessed, and logistics can be played. The model recognizes severed lines of retreat and lines of supply and imposes appropriate consequences. The documentation consists of three volumes: (1) A Guide for Potential Users; (2) Game Designer's Manual; (3) Player's Manual

IDA PAPER P-1221

IDAHEX

A MANEUVER-ORIENTED MODEL OF CONVENTIONAL LAND WARFARE

VERSION 1.0

Volume 3: Player's Manual

Paul Olsen

November 1976

PREFACE

IDAHEX is a computerized model of land warfare at the theater level. Its documentation consists of:

Volume 1: *A Guide for Potential Users*
Volume 2: *Game Designer's Manual*
Volume 3: *Player's Manual*

Volume 1 outlines the model's fundamental characteristics. Volume 2 comprehensively describes the model and its data base. Volume 3 (this manual) contains enough information for someone with a rudimentary knowledge of land warfare to play an IDAHEX game under the general supervision of the game designer.

Comments and inquiries are welcomed. They should be directed to the author (telephone: 703-558-1874).

CONTENTS

FIGURES

TABLES

1. INTRODUCTION

For Complete abstract, see VOL. 1 AD-A050675.

IDAHEX is a model of warfare implemented as a computer program. Usually, no distinction is drawn between the model and the program; this manual refers to them both as "IDAHEX". IDAHEX can be regarded as a system that is used by the "game designer" to design a war game and then by the players to play it. There are two sides in the war, "Red" and "Blue", inducing a "Red player" role and a "Blue player" role. Although both roles may be assumed by a single person, IDAHEX is careful to distinguish them and to prevent one player from giving commands to the other's forces. Although IDAHEX can be played in batch processing mode, interactive use is far more convenient. IDAHEX can be played interactively with each player on a separate terminal or with both players sharing a single terminal.

Sections 2, 3, 5, 6, and 7 form a self-contained outline of the model's structure; to understand the model fully, the player must read the *Game Designer's Manual* (Volume 2). Sections 4 and 6 explain how the players communicate with IDAHEX. Although this manual is directed toward the players, it sometimes identifies game design variables by name, in italics, so that the game designer can recognize them and answer players' questions about the values he has assigned to them.

2. THE ELEMENTS OF PLAY

This section explains how IDAHEX structures the area of war, the forces, and maneuver.

2.1 THE AREA OF WAR

The game board is termed the "area of war", the area in which the forces exist. It is partitioned into congruent, regular hexagons, as Figure 2.1 illustrates. The hexagons are termed "cells". The depth of any cell, the distance from one side to the side directly opposite it, is fixed by the game design variable *depth*. The cells are numbered for identification. A cell may be "inactive": in effect, it is excluded from the area of war, and no forces may enter it.

A cell's "environment" is the complex of physical conditions in the cell that affect ground combat or vulnerability to air strikes. Examples: clear, hilly, muddy, urban. The environment is treated as though it is uniform throughout the cell. Between two adjacent cells there may be a barrier. Barriers include rivers, ridges, and, in general, any obstacles that significantly affect movement or attack between the cells. The type of route between two adjacent cells represents all physical conditions affecting (unopposed) movement between them except barriers. Because there may be many geographical paths leading from one cell to an adjacent cell, the route type should be interpreted as a general characterization of trafficability, of how hard it is to get from one cell to the other, and not as a characterization of a particular track, road, or railroad.

The game designer must give the players a map of the area of war that identifies the cells by their numbers and indicates each cell's environment, the type of route between each pair of adjacent cells, and the type of barrier between each pair of adjacent cells. The cells' identification numbers never change, but environment, route, and barrier types may change during the game. IDAHEX informs the players when that happens.
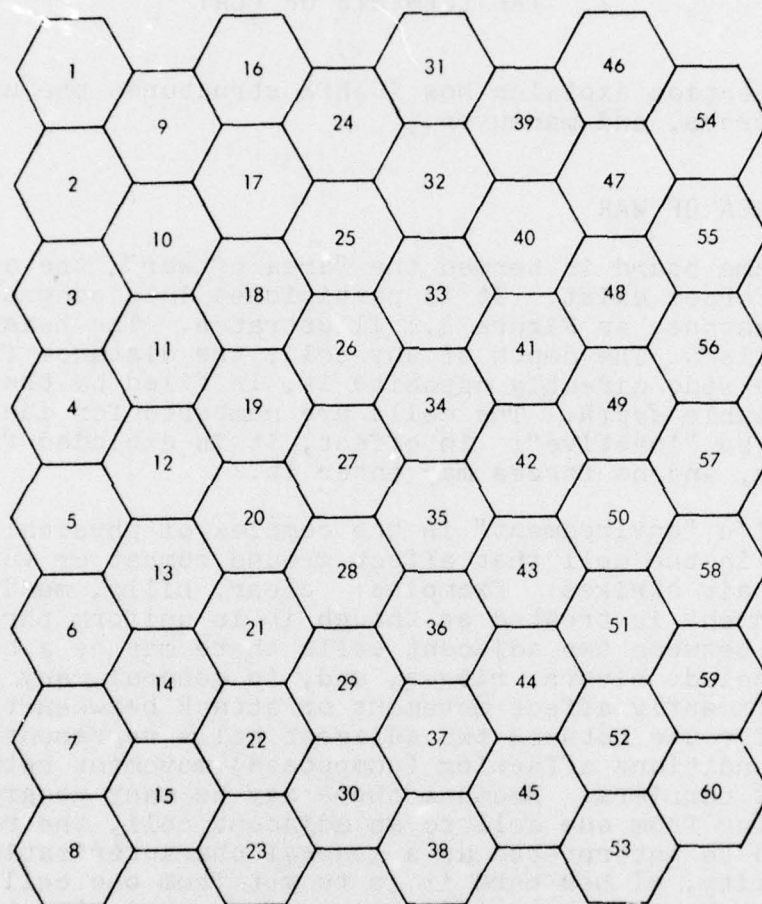
Figure 2.1.   EXAMPLE OF AREA OF WAR

2.2   THE FORCES

There are two forces, Red and Blue. Each consists of indivisible "battle units", often called simply "units". The game designer assigns each unit a unique identification number, a positive integer. The identification number of every Red unit must be less than the identification number of every Blue unit. A unit's "type" affects its mobility and determines the types of resources it is permitted to carry. A game's list of unit types might be:

1. Red motorized rifle division
2. Blue armored division
3. Red tank battalion
4. Red tank division
5. Red road transport unit
6. Blue rail transport unit
7. Blue infantry division
8. Blue road transport unit
9. Red rail transport unit

IDAHEX identifies unit types by numbers; therefore, the game designer should give the players a list like the one above.

2.2.1   Battle Unit Status

A battle unit's "status" is described by its location, posture, and objective.

Each battle unit is located in exactly one cell. The unit's location can not be fixed more precisely: there is no pretense of knowing, for example, that it is 3 km northeast of the cell's center. Its location *is* the cell. Several units may have the same location, even if they belong to opposite sides.

At any moment of the game, each battle unit is in one of six "posture classes":

-1. destroyed
0. inactive
1. hold
2. disengagement
3. movement
4. attack.

A unit in posture class -1 or 0 is said to be "inactive". (Inversely, a unit in a positive posture class is said to be active".) An inactive unit does not exist from the perspectives of other units. It can not move; it can not attack, nor can it be attacked. A unit in posture class -1 is a special kind of inactive unit: it was de-activated to represent its destruction,

usually as a result of suffering intolerably high losses. A
unit in posture class 0 is ordinarily a reinforcement or a
package of replacements. It may become active (enter a positive
posture class) later in the war. Its location is the cell where
it is expected to enter the area of war if it becomes active,
but while it remains inactive, it has no effect on enemy units
passing through its location.

A unit in posture class 2 is trying to break contact with
any enemy units it may be fighting, as the first step in changing
location. Its "objective" is the cell toward which it is dis-
engaging. A unit in posture class 3 is moving from its location
to another cell, its objective. Ordinarily, a unit in posture
class 4 is trying to enter a new location, which may or may not
contain enemy units, but in some cases it is trying to revert
from posture class 2, 3, or 4 to posture class 1 without changing
location. In the former instance, its objective is the cell it
seeks to enter; in the latter, its objective is just its present
location.

Posture class 1 embraces all remaining activities as well
as simple idleness. In particular, a unit in posture class 1
is not in the process of changing location. It may or may not
be engaged. Its objective is, by convention, its location.

Each positive posture class consists of from 1 to 10 postures.
Posture class -1 consists of a single posture, numbered -10.
The postures in posture class 0 are numbered 0 through 9, but
IDAHEX does not distinguish one posture in posture class 0
from another. The postures in posture classes 1 through 4 are
numbered as follows:

                10-19 hold
                20-29 disengagement
                30-39 movement
                40-49 attack.

These numbers are used to identify the postures in communications
between the players and IDAHEX. Table 2.1 presents alternative
ways of describing a unit's posture. The number of postures in a
posture class and their interpretations depend upon the game
design data. Therefore, the game designer must explain the
postures to the players, at least giving them a list like Table
2.2. There might, for example, be three movement postures,
representing tactical march, administrative march, and air move-
ment. There might be several attack postures, representing
different degrees of organization and willingness to trade casu-
alties for space.

2-4

Table 2.1. EQUIVALENT DESCRIPTIONS OF POSTURE CLASS

in posture class 1; in a hold posture;   holding

in posture class 2; in a disengagement posture; disengaging

in posture class 3; in a movement posture;   moving

in posture class 4; in an attack posture;   attacking

Table 2.2. ILLUSTRATIVE LIST OF POSTURES

Hold Postures

10. standard hold
11. halt
12. delay
13. transfer
14. hasty defense

Disengagement Postures

20. standard disengagement

Movement Postures

30. tactical march
31. administrative march
32. air movement

Attack Postures

40. standard attack
41. attack from administrative march
42. hasty attack

2.2.2 Resources

The game designer should give the players lists of the sides' resources. For example, the list of Red resource types might be:

1. tanks
2. small arms and APCs
3. artillery
4. SAMs and AAA
5. trucks
6. ammunition
7. fuel and other consumables
8. tank crewmen
9. other personnel

The Blue list might be:

1. small arms and APCs
2. artillery
3. tanks
4. trucks
5. supplies
6. personnel

There is no correspondence between Red resource types and Blue resource types: in the example, Red type 3 resources are artillery while Blue type 3 resources are tanks, and Red has SAMs and AAA while Blue has none. The resource types must be listed in the following order:

ground-to-ground weapons
ground-to-air weapons
transport
supplies
personnel

These five resource categories are combined to form larger categories:

materiel
{
ground-to-ground weapons
ground-to-air weapons } weapons
transport
} equipment
supplies
personnel
} support

A category may include one or more resource types or, with the exception of ground-to-ground weapons, it may be empty.

A player can learn what types of resources each side has and how the resource types are numbered by giving the command

2-6

"display units" (described in Section 4). Since diverse resources may be aggregated into a single type, the game designer probably should explain the method of aggregation--perhaps identifying a specific resource as the typical resource, or *numeraire*, of each type.

A battle unit's type determines what types of resources it can have, in accordance with the game design variables *nrst* and *iars*. A unit can never receive enemy resources, but in addition IDAHEX prevents it from receiving friendly resources of a type it is prohibited from having. The game designer should tell the players which types of resources each type of unit is allowed to have. He should also tell them $toe(i,j)$--the planned effective quantity of type j resources in a type i battle unit--for each unit type i and resource type j.

Each cell is "owned" by either Red or Blue. The ownership of every cell at the start of the game is declared by the game design data. Suppose a given cell is owned by Red; ownership changes to Blue when a Blue battle unit assumes a hold posture in the cell without opposition, or when Red units holding the cell are defeated in an engagement and the victorious Blue attackers are allowed to occupy it. Ownership changes from Blue to Red according to an analogous rule.

3. MANEUVER

A task force is a collection of one or more battle units
--"task force elements"--that have the same status and will
continue to have the same status as long as they remain in the
task force. The elements of a task force must all belong to
the same side. Each task force is identified by a positive
integer, which bears no relation to its elements' identifi-
cation numbers.

3.1 EVENT SEQUENCING

A task force's change of status is always caused and
directed by an "order". Sometimes, orders are generated by
IDAHEX; usually, they are input by the players. An order has
two components: the desired objective and the desired posture.
Associated with an order there may be a "start time", the earliest
time at which execution of it should begin. A task force cannot
always achieve the desired posture and desired objective
directly. For example, it could not attack units in an adjacent
cell without first moving there. Execution of an order is a
process that may span time and may take the task force through
a sequence of statuses. The time required to go from one
status to another may be 0, but the task force still enters
every status in the sequence. For fixed values of the game
design variables *pmapup* and *pmapdn*, a task force's current
status and the order it is executing uniquely determine its
next status. Figure 3.1 of Volume 2 (page 3-2) shows precisely
how. This manual only states the essentials.

Referring to Figure 3.1 of this volume, suppose a task
force is located in cell 6 in posture class 1, and suppose its
order declares cell 9 to be the desired objective and a certain
hold posture to be the desired posture. In essence, the task
force is under orders to change location to cell 9 and assume a
hold posture there. The task force will enter statuses in the
following sequence:

6-28-76-8

Figure 3.1. AREA OF WAR WITH BATTLE UNITS

location	posture class	objective
6	2	9
6	3	9
6	4	9
9	1	9

That is, it will first disengage from any combat in its present
location, then move to the objective, then attack any enemy
units in the objective, and finally, enter the objective. It
will enter posture class 2 (disengagement) even if it is not
engaged, in which case the disengagement is simply a formality
of model logic. It will enter posture class 4 even if cell 10
contains no enemy units at the time, in which case the attack
is simply a formality of model logic, involving no combat. In
order to change location, the task force must enter the posture
classes disengagement, movement, attacking, and holding in that
order. Completion of the sequence is not assured: the time
needed to get from one status to the next status in the sequence
may exceed the length of the war. In particular, the task force
may take forever to get from posture class 3 to posture class
4--to complete its movement--if it lacks essential supplies or
transport. It may never succeed in getting from posture class
4 to posture class 1 in cell 9 if, as can happen, it must defeat
enemy units defending the cell before it can occupy the cell.

The variables *pmapup* and *pmapdn* are set by the game designer
subject to the following restrictions:

$$20 \leq pmapup(\text{pp}) \leq 29 \quad \text{if} \quad 10 \leq \text{pp} \leq 19$$
$$30 \leq pmapup(\text{pp}) \leq 39 \quad \text{if} \quad 20 \leq \text{pp} \leq 29$$
$$40 \leq pmapup(\text{pp}) \leq 49 \quad \text{if} \quad 30 \leq \text{pp} \leq 39$$
$$10 \leq pmapup(\text{pp}) \leq 19 \quad \text{if} \quad 40 \leq \text{pp} \leq 49$$

$$-10 \leq pmapdn(\text{pp}) \leq -1 \quad \text{if} \quad 10 \leq \text{pp} \leq 19$$
$$40 \leq pmapdn(\text{pp}) \leq 49 \quad \text{if} \quad 20 \leq \text{pp} \leq 49$$

The positive posture classes form a cyclic group, and *pmapup*(pp)
is the posture a task force enters when it transitions to the
next higher posture class: from posture class 1 it goes to 2,
from 2 to 3, from 3 to 4, and from 4 to 1. The variable *pmapdn*
is not used to take a task force from its present posture to the
next lower posture class--that is generally illegal. Rather, it
tells what posture a disengaging, moving, or attacking task force
enters when it aborts tne disengagement, movement, or attack and
attempts to revert to a hold posture in its present location.

The order that a task force is executing implies whether or not the task force should transition to another posture class. If not--if the order implies it should assume another posture in the same posture class--then it will enter this new posture directly (but possibly after some delay).

To get specific examples of status sequencing, suppose there are four hold postures, one disengagement posture, two movement postures, and three attack postures, and *pmapup* and *pmapdn* are defined as follows:

pp	*pmapup*(pp)	*pmapdn*(pp)
10	20	-10
11	20	-10
12	20	-10
13	20	-10
20	30	42
30	40	42
31	41	42
40	10	42
41	11	42
42	13	42

The preceding assignments are motivated by the following interpretations of the postures:

- 10 standard defense
- 11 halted
- 12 prepared for transferring resources to other units (*itrfp* = 12)
- 13 hasty, disorganized defense
- 20 disengaging
- 30 tactical march
- 31 administrative march
- 40 standard attack
- 41 attack from administrative march
- 42 hasty, disorganized attack

Presumably, the ground combat attrition data make a unit less effective on defense in posture 13 than posture 11, and less effective in posture 11 than posture 10. Likewise, an attacker should be less effective in posture 41 than posture 40. Based on the above values of *pmapup* and *pmapdn* and the area of war in Figure 3.1, Table 3.1 shows the sequences of statuses induced by various orders. The last example in the table depicts a task force aborting an attack and reverting to a hold posture at its present location.

Table 3.1. EXAMPLES OF STATUS SEQUENCES

present location	present posture	present objective	desired posture	desired objective	Sequence of Statuses location	Sequence of Statuses posture	Sequence of Statuses objective
6	10	6	10	9	6	20	9
					6	30	9
					6	40	9
					9	10	9
6	11	6	31	9	6	20	9
					6	30	9
					6	31	9
6	31	9	10	9	6	41	9
					9	11	9
					9	10	9
6	31	9	40	9	6	41	9
					6	40	9
6	0	6	10	6	6	10	6
6	40	9	10	6	6	42	6
					6	13	6
					6	10	6

The preceding configuration of postures can be simplified at the risk of oversimplifying: let there be just two hold postures, one disengagement posture, one movement posture, and one attack posture, and define *pmapup* and *pmapdn* as follows:

$$pmapup(\text{pp}) = \begin{cases} 20; & 10 \le \text{pp} \le 19 \\ 30; & 20 \le \text{pp} \le 29 \\ 40; & 30 \le \text{pp} \le 39 \\ 10; & 40 \le \text{pp} \le 49 \end{cases}$$

$$pmapdn(\text{pp}) = \begin{cases} -10; & 10 \le \text{pp} \le 19 \\ 40; & 20 \le \text{pp} \le 49 \end{cases}$$

Suppose the game designer has designated posture 11 as the transfer posture (by setting *itrfp* = 11). Only a unit in the transfer posture can transfer resources to another unit. Refer again to Figure 3.1. A task force in a hold posture in cell 6 whose desired posture is 11 and desired objective is 9 would go through the following sequence of statuses:

location	posture	objective
6	20	9
6	30	9
6	40	9
9	10	9
9	11	9

When the unit achieves posture 11, it will be ready and able to transfer resources to other units located in cell 9. A task force whose

        location = 6,
        posture   = 40,
        objective = 9,

and whose

        desired posture   = 10,
        desired objective = 6,

would go through the following sequence:

location	posture	objective
6	40	6
6	10	6

Thus, the task force aborts an attack and goes directly into the standard hold posture, in contrast to the last example in Table 3.1; there is no "disorganized defense" posture in which to put it.

Because that can happen whenever the game designer selects a skeleton configuration of postures, IDAHEX provides another way of reducing a task force's defensive capability in that situation. A unit's defensive capability in a hold posture--specifically, its resources' vulnerability to fire from enemy battle units--depends on how long it has had to prepare its defense. This "defense preparation time" is computed as the current time minus the virtual time at which it entered posture class 1 in its location. On output, the virtual time at which a unit entered its present posture class is labeled "t entry", "tentry", or "time of entry". The time is virtual, rather than actual, because IDAHEX may penalize a unit's hasty reversion to a hold posture by setting the virtual time of entry later than the time it actually entered posture class 1, with the result that its defense preparation time may be negative.

In every example of task force movement thus far, the objective has been a cell adjacent to the task force's location, but IDAHEX's event sequencing logic does not require that. A task force may receive an order stating a desired objective that is not adjacent to its present location. The task force will be able to execute the order only if (1) it can fly and (2) the order causes it to enter an air movement posture. The game designer must tell the players which, if any, types of units are capable of transporting themselves through the air.

Every task force has a "transport mode", coded as a non-negative integer. A single-element task force's transport mode is necessarily 0; a multi-element task force's transport mode is selected by the player of the side to which it belongs. If the transport mode is positive, the task force is said to be stacked. (If not, it is unstacked.) In a stacked task force, the burden of transporting the task force's resources falls entirely on a subset of its elements termed the carriers. The carriers are those elements whose "transport class" agrees with the task force's transport mode; the game design datum $trptcl(i)$ defines the transport class of a type i battle unit. Every element of a stacked task force that is not a carrier is a "passenger". One may think of the passengers as being physically loaded on the carriers, thereby enjoying the carriers' mobility. Stacking is particularly useful as a way of moving units faster than they can move by themselves. For example, a player might create a stacked task force consisting of wheeled vehicle transport units as carriers and straight-leg

infantry units as passengers. Stacking can also be used to move units by air. The game design datum *mrair*(i) fixes the rate at which a type i unit can fly, assuming its transport capacity meets its requirement. This is ordinarily 0 for most types: an armored division cannot fly, and neither can an airborne infantry brigade unless it has organic aircraft. But a player might create a task force consisting of infantry brigades as passengers and units that can fly (units whose principal resources are aircraft) as carriers.

3.2 EVENT SCHEDULING

As Section 3.1 explains, a task force's execution of an order involves going through a sequence of one or more statuses. Associated with any change of status is a delay time. The task force stays in its present status (present location, present posture, present objective) a length of time equal to the delay and then enters the next status (next location, next posture, next objective) in the sequence. This subsection only mentions the factors that affect the delay; Section 3.2 of Volume 2 explains fully.

3.2.1 Changing Posture within Positive Posture Class

Suppose that the task force's present posture class is positive, and that its next posture class is the same as its present posture class and its next objective is the same as its present objective--i.e., it is going to a different posture in the same class. The delay is given by the game design variable *ptran*: $ptran(i,j,k)$ is the time required to go from the j-th posture to the k-th posture within posture class i.

3.2.2 From Hold Posture to Disengagement Posture

In this case the present posture is a hold posture and the next posture is a disengagement posture. The delay is 0.

3.2.3 Disengagement

In this case, the present posture is a disengagement posture and the next is a movement posture, and the next objective is the same as the present objective. The delay is most simply interpreted as the time required to break contact with the enemy, but in reality a force being pursued by the enemy might never break contact completely. The delay is better interpreted as the amount by which contact with the enemy increases

the time needed for the task force to relocate from its present location to its objective. If the task force is not engaged, the delay is, of course, 0. Suppose it is engaged. If the next posture is an air movement posture, the delay is infinite: in effect, an engaged task force can not disengage and go directly to an air movement posture.

Suppose the task force is engaged and its next posture is not an air movement posture. Two situations must be distinguished: (1) there are friendly units in hold postures in the task force's present location; (2) there are none. In the first situation, the friendly units are assumed to prevent the enemy units with which the task force is engaged from pursuing it, and therefore the delay does not depend on the time required for the task force to move to its objective; the delay represents only the time required to break contact with the enemy in the present location. Of course, some types of units are more adept at breaking contact than others; because the elements of the task force operate together, the task force can only disengage as fast as its slowest element. In the second situation, the absence of friendly units that could act as a rearguard for the task force permits the enemy units engaged with it to pursue it. Instead of describing the task force's movement and its continuing engagement as contemporaneous processes, IDAHEX extends the task force's disengagement delay (before it moves to its objective). That is, its delay equals the delay it would have experienced in the first situation plus an additional term corresponding to fighting that might in reality occur along the route of march. Some types of units may be more adept than others at escaping from a pursuing enemy. The task force as a whole does not complete its disengagement delay until enough time has elapsed to permit every element to escape. Hence, the additional term is the maximum of several numbers--one for each element if the task force is not stacked, or one for each carrier if it is. Each of these numbers may be interpreted as the amount of time during which the unit has interrupted its movement to the objective in order to deal with enemy pursuit; this time is proportional to the unit's anticipated movement delay.

3.2.4 Movement

In this case, the present posture is a movement posture, the next posture is an attack posture, and the next objective is the same as the present objective. Recall from Section 3.1 that before a task force changes location it enters an attack posture with the new location as the objective, even if the objective contains no enemy units (in which case the "attack"

is purely a logical formality). The "movement delay" is the length of time the task force remains in the movement posture before entering the attack posture. It represents the time required to move (unopposed) from the present location to the objective. If the objective contains enemy units in hold postures, the task force must attack and defeat them in order for its location to become that cell, but before it can attack there it must move there. Two kinds of movement are possible--surface and air. Every movement posture implies either surface movement or air movement: the game design variable *airmove* reveals which.

Suppose the task force is in a surface movement posture. Unless its present objective is adjacent to its present location, its movement delay is infinite. If the objective is adjacent to the present location, the movement delay is determined as follows:

(1) If supplies are included among the resource types of the task force's side, the supplies the task force needs in order to move are compared with the supplies it has. If the quantity of any type of supplies is deficient, the movement delay is infinite--in effect, the task force cannot move.

(2) If the task force is not stacked, this step is skipped. The movement delay of a stacked task force is infinite unless the carriers have sufficient load capacity to accommodate the passengers. Only those types of the carriers' resources whose load class (given by the game design variable *loadcl*) agrees with the task force's transport mode contribute load capacity. The passengers' resources burden this capacity. If the burden exceeds the capacity, the movement delay is infinite.

(3) Assume that the task force's supplies are adequate. If it is stacked, assume that carriers have enough load capacity to accommodate the passengers. Its transport capacity is compared with its demand for transport. Each type of resource may have transport capacity, and each type may demand transport, but generally speaking, resources that normally move themselves, such as tanks and trucks, have no demand for transport and may contribute transport capacity, while resources that are ordinarily transported on other resources--such as small arms, supplies, and personnel--demand transport and have little or no transport capacity. If the task force is stacked, resources that have already been allocated in Step 2 to carry passengers' resources do not contribute transport capacity, nor do any of the passengers' resources.

(4) The maximum rate at which a given unit in the task force can move depends upon the ratio of transport capacity to demand found in Step 3, the unit's type, its posture (which coincides with the task force's), and the type of route between the present location and the objective. The task force's movement rate is the minimum of the rates at which its elements can move, if it is not stacked, or the rates at which it carriers can move, if it is. (The task force moves as fast as its slowest element if not stacked, or as fast as its slowest carrier if stacked.)

(5) If there is a barrier between the present location and the objective, the time a unit requires to cross it depends upon the unit's type, the unit's posture, and the type of barrier. The time required for the task force to cross is the maximum time required by any element, if it is not stacked, or by any carrier, if it is.

(6) The surface movement delay equals the sum of two numbers. The first equals the distance travelled--which is estimated as the distance from the center of the present location (a cell) to the center of the objective (an adjacent cell)--divided by the task force's movement rate. The second number equals 0 if there is no barrier, or the number found in Step 5 if there is.

Alternatively, suppose the task force is in an air movement posture. The objective need not be adjacent to the present location: a task force can fly directly to a nonadjacent cell. The first three steps of the procedure for determining the task force's movement delay are the same as in the case of surface movement. Step 4 becomes:

(4′) The maximum rate at which a given unit in the task force can move depends upon the ratio of transport capacity to demand found in Step 3 and the unit's type; the game design datum $mrair(i)$ fixes the rate at which a type i unit can transport itself by air if the ratio of transport capacity to demand is 1. The latter rate should be 0 for any type of unit whose organic air transport capability is insufficient to transport the unit. The task force's movement rate is the minimum of the rates at which its elements can move, if it is not stacked, or the rates at which its carriers can move, if it is.

Step 5 of the procedure for determining the surface movement delay does not apply. Step 6 becomes:

(6′) The air movement delay equals the distance travelled-- which is estimated as the straight-line distance from the center of the present location (a cell) to the center of the objective (a cell)--divided by the task force's movement rate.

A particular game may not include aircraft among a side's
resource types but may still assign some types of its battle
units a positive air movement rate (*mrair*). IDAHEX must then
infer the load capacity and (air) transport capacity of a
flying task force from the resources it has.

3.2.5 Attack

Suppose the present posture is an attack posture, the
next posture is a hold posture, and the objective is not the
same as the present location. If the task force's objective
contains no enemy units in hold postures, the delay is 0.
Otherwise, the delay is indefinite: it depends on the course
of combat.

3.2.6 Entering Hold Posture Class in Present Location

Three cases must be distinguished. In the first case,
the task force's present posture class is -1 or 0, its next
posture class is 1, and its next location is its present
location. This case arises when a task force consisting of
battle units that are not active is given an order to activate
them, an order stating a positive desired posture class. The
delay is infinite if the present posture class is -1. The
delay is 0 if the present posture class is 0.

In the second case, a task force in a disengagement,
movement, or attack posture whose present objective is not its
present location is trying to revert to a hold posture in its
present location (perhaps only momentarily, before moving in
a different direction). Then its next posture class is 4,
and its next objective is its present location. The delay is
0. And then the third case arises.

In the third case, the task force is in an attack posture,
but its objective is its present location. Its next posture
class is 1, and its next objective and next location are its
present location. The delay is 0.

3.2.7 Entering Nonpositive Posture Class

In this case the next posture class is -1 or 0. The delay
is 0 unless the task force's next posture class is 0 and its
next location is owned by the enemy or contains active enemy
units; then the delay is infinite.

A player can order a task force in posture class 0 to a different location, which it can attain instantaneously, and then activate its elements by ordering it to posture class 1. This capability allows reinforcements to arrive in the area of war at an appropriate place and time. To avoid unrealistic events the player must follow instructions from the game designer stating where and when inactive units may be activated.

A player can also order a task force in posture class -1 to posture class 0 (and then to posture class 1), but since there is normally no reason for a unit to enter posture class 0 from another posture class, IDAHEX warns the game designer if that happens.

3.3 TACTICAL SITUATIONS

Maneuver of opposing battle units into proximity may precipitate tactical situations in which it is advisable or essential that IDAHEX issue new orders to existing task forces or create new task forces and issue them orders. In some cases the sole purpose of the modifications is to respond immediately to a local situation that one of the players probably failed to foresee. Often, the modifications have the additional purpose of resolving opposing forces' conflicting aims.

The players should watch for one tactical situation with potentially severe consequences. If a task force in an attack posture is actually engaged and its location is occupied by the enemy, then it is destroyed--IDAHEX orders it into posture class -1. The penalty is severe, but is the only acceptable way of resolving the situation: the task force controls neither its location nor its objective and therefore has no place to exist. IDAHEX takes extensive precautions to avert this situation. If a task force is threatened with destruction because an enemy task force is about to occupy its location, IDAHEX orders the task force to revert to holding its location. If the task force does that, it still incurs a penalty: its defensive capability may be degraded because the hold posture it enters may represent a hasty, disorganized defense and because its defense preparation time may be zero or even negative.

The defense preparation time of a battle unit in a hold posture is a measure of how prepared the unit is to defend its location. It is defined as the current time minus the unit's virtual time of entry into posture class 1. Ordinarily, when a unit enters posture class 1, its virtual time of entry is set equal to the actual time. But when a unit in a disengagement, movement, or attack posture reverts to a hold posture in its location while an enemy unit directly threatens to seize its

location from the flank or rear (not from the cell that was the
objective of its disengagement, movement, or attack), then the
unit's virtual time of entry into posture class 1 is set ahead
of the actual time by an amount that measures how far out of
position it was before it reverted to posture class 1.

3-14

4. THE COMMANDS

Periodically, the Red player and the Blue player may input commands to IDAHEX. A command is an instruction to battle units or a request for information. IDAHEX prevents a player from issuing instructions to enemy units or obtaining the enemy player's instructions to his units. IDAHEX signals the player when it is ready to receive commands. Since both players may be sharing one terminal, it names the side for which it will accept commands. The Red player may go before the Blue player, but that is purely a formality: both players' units begin executing their instructions simultaneously. In no way do the forces take turns.

IDAHEX signifies that is ready to receive a command by writing "Enter command." on the player's terminal. The player replies by entering a character string enclosed in quotation marks. IDAHEX examines the reply to identify the command. If complete specification of the command requires more inputs from the player, IDAHEX writes prompting sentences on the terminal. ("Enter command." is the initial prompting sentence.) The player must reply to a prompting before IDAHEX will proceed. Each reply is a character string enclosed in quotation marks.[1] It may occupy more than one line of input. A reply may contain no items: it consists only of blanks, or it is empty. Examples:

```
"     "
""
```

Or a reply may contain exactly one item. Examples:

```
"oneword"
" oneword"
"29.2"
" 29.2 "
"17"
```

Or it may contain several items. Examples:

---

[1] It may be permissible to omit the quotation marks if the reply contains exactly one item.

```
               "firstword secondword"
               "firstword,secondword"
               "29.2 17, 12.1 , 15"
               "word,17,12.1"
```

In general, two consecutive items can be separated by one or
more blanks, by a comma, or by a comma together with one or
more blanks. Hence, each of the following replies contains
exactly two items:

```
                              "7 45"
                              "7,45"
                              "7, 45"
                              "7 , 45"
                              " , "
                              ","
```

In the last two replies the two items are both null--i.e., the
items contain no non-blank characters. IDAHEX interprets a null
item as a character string of blanks if it expects the item to
be a character string, and as the number 0 if it expects a
number. As one might infer, IDAHEX maps a reply into a "target
list" of items it expects to receive. A target list is actually
a list of variables to which the reply implicitly assigns values.

   Each non-null item in a reply--each item containing a non-
blank character--is classified as either a number or a word.
A non-null item is a number if and only if its first non-blank
character is a decimal digit or a minus sign (-); a non-null
item is a word if and only if it is not a number. A null item
is both a number and a word. *All the words in a reply must
precede all the numbers.*

   Each variable in the target list is classified as either a
character-string variable or an arithmetic variable. It is a
character-string variable if and only if it is intended to
receive a word; it is an arithmetic variable if and only if it
is intended to receive a number.

   Each item in a reply is assigned to the corresponding
variable in the target list, its "target variable". When a
null item is assigned to a variable, the variable's value
becomes a character string of four blanks if the variable is
a character-string variable, and its value becomes 0 if it is
an arithmetic variable. If, in comparing the reply with the
target list, IDAHEX discovers that reply items are missing,
it implicitly inserts a null item to take the place of each
missing item. Hence, the null reply ("") gives every character-
string variable in the target list the value "   " and every
arithmetic variable in the target list the value 0.

                              4-2

Actually, only the first four characters of a word are assigned to its target variable. Therefore, a player may truncate any word to its first four characters. In particular, the command names "mission", "deliver", "transfer", "cancel", "abandon", and "display" may be abbreviated as "miss", "deli", "tran", "canc", "aban", and "disp".

The sequel generally shows the correct format of a reply in a line labeled "Reply" or "Initial reply". *Any item not enclosed between a "less-than" character (<) and a "greater-than" character (>) must appear exactly as shown, except that a word may be truncated to four characters. An item enclosed between a "less-than" character and a "greater-than" character is an argument, to be replaced by an appropriate word or number without the "less-than" and "greater-than" characters. It may be omitted, but then every item following it in the reply must also be omitted.* Omitting it has precisely the same effect as including it in the reply as a null item (which, if it should be a number, has the same effect as including it in the reply as the number 0).

The "initial reply in the command sequence" is the player's reply to the prompting sentence "Enter command." A line labeled "Initial reply" shows its format.

IDAHEX's command repertoire is occasionally expanded. The program release that the players are using may support commands in addition to those described in this section. The current command repertoire can be learned by invoking the help command.

4.1 THE MISSION COMMAND

As Section 3.1 explains, a task force's change of status is always caused and directed by an order. A mission is a sequence of orders. Every task force has a mission, and every mission is assigned to exactly one task force (but two task forces may have identical missions). A task force and its mission are both identified by the same positive integer, which is assigned by IDAHEX or the player when the task force is created. The orders comprising a mission are stored in a pop-up stack and are executed in sequence, from the top to the bottom. The order at the top of the stack is termed the active order. When execution of it is completed, it is removed from the stack, and the next order, if any, pops to the top. If a start time is associated with the active order, execution of the order does not begin until the game time equals or exceeds the start time.

A mission is created or modified by the mission command:

Initial reply: "mission <number1>"

Suppose number1 happens to be the identification number of an existing task force (which must be positive). The player is then modifying the task force's mission. Its present mission is erased, and the player constructs a new one by entering a sequence of orders.

Prompting: Enter orders.

Reply: "<dobj> <dpost> <strt>"

Actually, the player enters a sequence of replies. The first reply defines the first order in the mission, the second reply defines the second order, and so on. Recall that an order has two components: a desired objective and a desired posture. The item dobj defines the desired objective (a cell number), and dpost defines the desired posture. There is an IDAHEX variable named nstart.[1] The player may specify a start time for each of the first nstart orders in a mission, the argument strt is ignored if it appears in a reply after the first nstart replies. Of course, if strt is omitted from one of the first nstart replies, the corresponding order's start time is taken to be 0.[2]

The player signals IDAHEX that he has input the final order in the mission by entering a reply that defines the desired objective as a nonpositive number (as the null reply ("") does). The dummy order defined by this reply does not become part of the mission. More than one reply to the prompting sentence "Enter orders." may occur on the same line, but if so, at least one blank should separate them.

If the player is modifying an existing mission, those are his only inputs. Alternatively, suppose he is creating a new task force and corresponding mission--which is true if and only if number1 is not the identification number of an existing task force (or it is omitted from the initial reply). The identification number assigned to the task force and mission is number1

---

[1]The value of nstart is set in the IDAHEX main program, cgcm. It equals 1 in Version 1.0.

[2]In that case execution of the order will begin as soon as execution of the preceding order, if any, is completed-- assuming, as should be the case, that the game time is non-negative when the game begins.

if number1 > 0 and is not too large; IDAHEX chooses a number otherwise.[1] IDAHEX requests the player to "Enter orders.", which he does just as before. After he has entered the final order, he must list the elements of the task force, in any sequence:

    Prompting: List task force.

    Reply: "$\langle$unit1$\rangle$, $\langle$unit2$\rangle$,...,$\langle$unitn$\rangle$"

Each reply item identifies a task force element by its battle unit number. Any unit listed that already belongs to a task force is automatically detached from it. Finally, if the task force contains more than one element, IDAHEX requests the player to select a transport mode:

    Prompting: Enter transport mode.

    Reply: $\langle$mode$\rangle$

The item mode must be a nonnegative integer. (See Section 3.1.)

    The examples below are based upon the area of war in Figure 3.3 and the configuration of postures assumed by Table 3.1, namely:

pp	$pmapup$(pp)	$pmapdn$(pp)
10	20	-10
11	20	-10
12	20	-10
13	20	-10
20	30	42
30	40	42
31	41	42
40	10	42
41	11	42
42	13	42

The examples assume nstart = 1 and therefore specify a start time only for the first order in a mission.

    Example 1. Suppose units 4 and 9 are in identical hold postures in cell 17. In the following communications with IDAHEX, the Red player constitutes them as a task force and assigns a mission.

---

[1] A mission identification number may not exceed nmnmax, whose value is fixed in cgcm.

```
                         Enter command.
                         "mission"
                         Enter orders.
                         "16 31"
                         "16 11"
                         "12 10"
                         ""
                         List task force.
                         "9 4"
                         Enter transport mode.
                         "0"
```

Each line after the prompting sentence "Enter orders." states
an order: the first number is the desired objective (i.e.,
the identification number of the cell that is the desired
objective), and the second number is the desired posture.
Because nstart = 1, the reply "16 31" is mapped into a list of
three items, the third being the start time. Since the reply
contains only two items, the start time is defined to be 0.
The mission implies the following sequence of statuses for the
task force:

location	posture	objective	
17	20 (disengaging)	16	
17	30 (tactical march)	16	
17	31 (admin. march)	16	Order 1 finished.
17	41 (attack from 31)	16	
16	11 (halted)	16	Order 2 finished.
16	20 (disengaging)	12	
16	30 (tactical march)	12	
16	40 (standard attack)	12	
12	10 (standard hold)	12	Mission finished.

   Example 2. Suppose task force 38, consisting of unit 27,
is moving from cell 6 to cell 5 in accordance with a mission
whose active order is "desired objective = 5, desired posture
= 10", and whose next order is "desired objective = 1, desired
posture = 10". The player directs that the movement to cell 5
continue, but thereafter the task force should go to cell 2
instead of cell 1:

```
                         Enter command.
                         "miss 38"
                         Enter orders.
                         "5 10"
                         "2 10"
                         ""
```

The modified mission implies the following sequence of statuses for task force 38:

location	posture	objective	
6	40 (standard attack)	5	
5	10 (standard hold)	5	Order 1 finished.
5	20 (disengaging)	2	
5	30 (tactical march)	2	
5	40 (standard attack)	2	
2	10 (standard hold)	2	Mission finished.

The original mission implied next location = 6, next posture = 40, next objective = 5. *Since the modified mission implies the same next status as the original one, the time at which the task force will achieve its next status is not rescheduled.* The player could have achieved the same sequence of statuses for unit 27 by first canceling mission 38, using the cancel command explained below, and then entering the following command sequence:

```
Enter command.
"miss 38"
Enter orders.
"5 10"
"2 10"
""
List task force.
"27"
```

There would be only one difference in the results: because the cancel command destroys all record of the original mission, and a new task force has been created (albeit with the same iden- tification number and same composition), the movement delay is re-evaluated, and the time at which unit 27 completes its movement may be rescheduled. In general, *it is wise to modify the mission of a disengaging or moving task force, through the mission command, rather than canceling it and recreating the task force with a new mission.*

Example 3. Suppose task force 39 is moving from cell 6 to cell 5. In the following communications, the player directs it to move instead to cell 9:

```
Enter command.
"miss 39"
Enter orders.
"9 10"
""
```

The new mission implies the following sequence of statuses for task force 39:

location	posture	objective	
6	42 (hasty attack)	6	
6	13 (hasty defense)	6	
6	20 (disengaging)	9	
6	30 (tactical march)	9	
6	40 (standard attack)	9	
9	10 (standard hold)	9	Mission finished.

Example 4. Suppose unit 21 is in posture class 0 in cell 6. The player assigns a mission to the task force consisting of unit 21:

```
                    Enter command.
                    "miss"
                    Enter orders.
                    "6,12,1.51"
                    "9,12"
                    ""
                    List task force.
                    "21"
```

The mission implies the following sequence of statuses for unit 21:

location	posture	objective	
6	10 (standard hold)	6	
6	12 (transfer)	6	Order 1 finished.
6	20 (disengaging)	9	
6	30 (tactical march)	9	
6	40 (standard attack)	9	
9	10 (standard hold)	9	
9	12 (transfer)	9	Mission finished.

The first change of status will not occur until time 1.51. The example illustrates how resupply and replacement can be accomplished: if new resources should enter the area of war in cell 1 at time x, the game design data should incorporate them into a dummy unit whose initial location is 1 and initial posture class is 0; the player whose side should receive the resources can issue a mission command to activate the unit at time x.

Example 5. Assume unit 21 is in posture class 0. In the following communications with IDAHEX, the Blue player activates unit 21 in cell 8 instead of its present location, cell 6.

```
                              Enter command.
                              "miss"
                              Enter orders.
                              "8 0"
                              "8 10"
                              ""
                              List task force.
                              "21"
```

The mission implies the following sequence of statuses for
unit 21:

location	posture	objective	
8	0 (inactive)	8	Order 1 finished.
8	10 (standard hold)	8	Mission finished.

The first change of status is scheduled to occur immediately,
assuming that the current time is nonnegative.

        Thus, a player can activate one of his units in a cell
different from its initial location; to do so, he must first
change its location while it remains in posture class 0. This
capability is necessary since the location where a package of
supplies and replacements should become available might depend
on the course of the game. Indeed, IDAHEX prohibits activation
of a unit in a cell owned by the enemy or containing enemy
units. And it might be convenient to design the game so that
supplies and replacements originate in corps, army, or front
depots, which relocate to keep up with the combat forces, rather
than fixed theater depots. A player could use the capability
to change inactive units' locations in order to cheat, activating
units wherever he pleased. Therefore, IDAHEX places an advisory
message in a file intended for the game designer whenever an
inactive unit changes location.

4.2 THE DETACH COMMAND

        The detach command detaches a battle unit from a task force.
The unit then has no mission. There is no effect on the mission
of the remaining task force or the time at which the task force
is scheduled to enter its next status.

        Initial reply: "detach ⟨unitnbr⟩"

The argument unitnbr is the number of the battle unit to be
detached.

Example. The player commands IDAHEX to detach unit 4
from the task force to which it belongs, if any:

> Enter command.
> "deta 4"

It is not necessary to give a detach command to detach a
unit from a task force in preparation for including it in a new
one: naming a unit as an element of a new task force in a
mission command automatically causes it to be detached from
any task force to which it already belongs.

4.3   THE ATTACH COMMAND

The attach command attaches a battle unit to an existing
task force. If the unit belongs to another task force, it is
detached before being attached to the designated task force.
It must have the same status as the designated task force and
must, of course, belong to the same side.

> Initial reply: "attach <unitnbr> <tfnbr>"

The argument unitnbr is the number of the battle unit to be
attached, and tfnbr is the number of the task force to which
it should be attached.

Attaching a unit to a task force may push back the time
at which the enlarged task force is scheduled to enter its
next status. Let t1 be the time at which the original task
force is scheduled to enter its next status. Let t2 be the
time at which the enlarged task force's transition from its
present status to its next status would be scheduled if it
were just beginning the transition at the present time. The
enlarged task force is scheduled to enter its next status at
time t1 or t2, whichever is greater.

Example. The player commands IDAHEX to attach unit 4
to task force 17:

> Enter command.
> "atta 4,17"

4.4   TRANSFERRING RESOURCES

The game permits transfers of resources if and only if one
hold posture is reserved as the transfer posture. (Its number
is given by the game design datum *itrfp*.) One set of active
units, called the givers--can transfer resources to another

set, called the takers, subject to these restrictions: the
givers and the takers must all belong to the same side, they
must all have the same location, and every giver must be in
the transfer posture. A taker may be in any of the postures
from 10 through 49, including the transfer posture. A unit may
be both a giver and a taker. A taker can accept resources in
excess of the planned effective quantities for a unit of its
type, even if the planned effective quantities are 0, but it
cannot accept resources it is prohibited from having. (See
Section 2.2.2.)

4.4.1 The Delivery Command

The delivery command permits the players to arrange for a
transfer of resources that will occur automatically, at the
earliest possible moment. The command creates a "delivery
order" (not to be confused with the orders in a mission). A
delivery order has four components: (1) the task force desig-
nated to deliver the resources; (2) the delivery destination;
(3) the relative size of the delivery; (4) the intended
recipients of the delivery. The delivery destination is the
cell where the delivery is to occur. The relative size is a
number between 0.0 and 1.0, inclusive, that indicates how much
should be transferred. Once created, a delivery order continues
to exist until it is executed or it is canceled by the player.

The "delivery task force", the one designated to make the
delivery, must exist when the delivery order is created and when
it is executed. Two or more delivery orders may name the same
delivery task force, but their delivery destinations should
differ. If two delivery orders designate the same delivery
task force and the same delivery destination, IDAHEX arbitrarily
selects one of them. The list of intended recipients may be
empty.

        Initial reply: "deliver ⟨tfnbr⟩ ⟨cellnbr⟩"

The argument tfnbr is the delivery task force's number, and
cellnbr is the delivery destination's cell number. Next, the
player must indicate the size of the delivery.

        Prompting: Enter relative size of delivery.

        Reply: "⟨dlsize⟩"

The item dlsize should be a fixed decimal constant between 0
and 1. Finally, the player must list the delivery's intended
recipients. (The list's purpose is explained below.)

Prompting: List recipients.

Reply: "⟨id1⟩, ⟨id2⟩,...,⟨idn⟩"

A reply containing no items defines the empty list. Each item--
id1,...,idn--must identify either a friendly battle unit or a
friendly task force. An item identifying a unit is simply the
unit's number; an item identifying a task force is simply the
sum of the task force's number and 10000.

Suppose task force m has just entered the transfer posture
in cell i. IDAHEX must decide whether the transfer of resources
will be governed by a transfer command (see Section 4.4.2) that
the player will issue later or by a delivery order. *IDAHEX
infers that the player intends to issue a transfer command, and
therefore makes no delivery of resources at this time, if either
of the following conditions holds:*

(1) with this change of status, the task force has
    accomplished its mission;

(2) with this change of status, the task force has
    completed execution of the active order in its
    mission, and its new active order has a start
    time that exceeds the current time.

*If neither condition holds, IDAHEX searches for a delivery order
--one whose delivery task force is m and delivery destination is
i. If none is found, a delivery order is generated: it desig-
nates task force m as the delivery task force and cell i as the
delivery destination, fixes the delivery's relative size at 1.0,
and leaves the list of intended recipients empty.* A generated
delivery order is treated in the same way as a delivery order
created by the delivery command. Consequently, there is no reason
for a player to invoke the delivery command except to make the
relative size less than 1.0 or to name intended recipients.

When IDAHEX executes a delivery order, it identifies the
givers and the takers. The givers are the elements of the
delivery task force. To find the takers, the list of intended
recipients is expanded by replacing each task force mentioned
in it with the task force's elements. If the resulting list is
not empty, the takers consist of every active, friendly unit in
the list whose location is the delivery destination; if the list
is empty, the takers consist of every active, friendly unit
whose location is the delivery destination and whose posture is
not the transfer posture.

In a resource transfer governed by a delivery order, a giver can only give resources in excess of its planned effective quantities. To be precise, for any i, the quantity of type i resources that a particular giver can transfer to the takers is limited to the amount by which its actual stock of type i resources exceeds its planned effective stock.[1] Each taker demands resources of a given type to the extent that its stock falls short of its planned stock. The total quantity of resources of a given type transferred from the givers equals $\min\{f*S, D\}$, where f is the delivery's relative size, S is the maximum amount that the givers can give, and D is the takers' total demand. The quantities taken from the various givers and the quantities given to the various takers are determined so as to balance the units' stocks as much as possible. Basically, a unit's stocks are perfectly balanced if its stock of each type of resources coincides with its planned effective stock. Section 4.2 of Volume 2 explains fully.

<u>Example</u>. The player arranges for a delivery of resources by task force 4 to unit 2, unit 6, and task force 30 in cell 10.

> Enter command.
> "deli 4 10"
> Enter relative size of delivery.
> ".65"
> List recipients.
> "2 6 10030"

4.4.2 The Transfer Command

The transfer command causes an immediate transfer of resources from the givers to the takers.

Initial reply: "transfer <cellnbr>"

The argument cellnbr is the number of the cell that is the transfer location--the cell where the transfer is to occur. Next, the player must identify units and task forces from which resources should be transferred.

_____

[1] Recall that the planned effective stock of type i resources in a type j battle unit is $toe(j,i)$.

4-13

Prompting: List resource givers.

Reply: "⟨id1⟩, ⟨id2⟩,...,⟨idn⟩"

A reply containing no items defines the empty list, whose
implications are described below. Each item in the reply
identifies either a friendly battle unit or a friendly task
force. An item identifying a battle unit is simply the unit's
number; an item identifying a task force is the sum of the
task force's number and 10000. Next, the player must list
units and task forces to which resources should be transferred.

Prompting: List resource takers.

Reply: "⟨id1⟩, ⟨id2⟩,...,⟨idn⟩"

A reply containing no items defines the empty list. Each item
identifies either a friendly battle unit or a friendly task
force as before. Finally, the player must indicate the amounts
of resources to be transferred from the givers to the takers.

Prompting: Enter resource index and amount
             to be transferred, line by line.

Reply: "⟨rsnbr⟩ ⟨amt⟩"

Actually, the player enters one or more replies. The first
item of a reply, rsnbr, is a resource type, identified by
number, and the second item, amt, is the total quantity of
type rsnbr resources to be transferred, which may be non-
integral. The player's last reply must define the resource
type to be nonpositive (as the null reply does), signaling
that he is finished. He may decline to specify the amount to
be transferred for any type of resources, or every type.

If the player's list of givers is empty, the givers
consist by default of every active, friendly unit whose location
is the transfer location and whose posture is the transfer
posture. If his list of takers is empty, the takers consist
by default of every active, friendly unit whose location is
the transfer location and whose posture is not the transfer
posture.

If the set of givers is identical to the set of takers,
then regardless of any transfer amounts the player has specified,
the units' resources are redistributed among them to balance
their stocks as much as possible; basically, a unit's stocks
are perfectly balanced if they coincide with its planned
effective stocks.

4-14

Alternatively, assume the set of givers is not identical to the set of takers. Suppose, for a given i, that the player has specified the amount of type i resources to be transferred. If at least one of the takers is permitted to have this type of resources, then the amount actually transferred is the amount the player specified or the amount the givers have, whichever is less. On the other hand, suppose the player has not specified the amount of type i resources to be transferred. The amount transferred equals min{S, D}, where S and D are defined as in Section 4.4.1. In either case, the amount transferred is taken from the givers and distributed among the takers so that the units' stocks are as balanced as possible.

Example. The player commands a transfer of resources in cell 10 from unit 55 and task force 29 to the active, friendly units in cell 10 that are not in the transfer posture.

> Enter command.
> "transfer 10"
> List resource givers.
> "10029 55"
> List resource takers.
> ""
> Enter resource index and amount
> to be transferred, line by line.
> "3 27.45"
> "8 0"
> "2 12"
> ""

The amount of type 8 resources transferred will be 0.

4.5 THE CANCEL COMMAND

The cancel command cancels a mission and dissolves the corresponding task force or cancels a delivery order.

Initial reply: "cancel <cd> <number1> <number2>"

The argument cd must be the word "mission" (or "miss") to cancel a mission, and "delivery" (or "deli") to cancel a delivery order. If a mission is being canceled, number1 is its number, and the argument number2 is not used. If a delivery order is being canceled, number1 is the identification number of the delivery order's delivery task force, and number2 is the cell number of the delivery destination.

Example. The player cancels mission 29:

Enter command.
"canc miss 29"

Example. The player cancels the delivery order for a delivery by task force 4 to cell 10:

Enter command.
"canc deli 4 10"

4.6 THE ABANDON COMMAND

Recall that the resources held by the passengers in a stacked task force burden the carriers' carrying capacity, and the movement rate of a unit not being carried depends on the ratio of transport capacity to transport requirement. Therefore, a player may want to lighten one of his battle units. The abandon command causes the abandonment and destruction of specified quantities of a battle unit's resources.

Initial reply: "abandon ⟨unitnbr⟩"

The argument unitnbr is the number of the battle unit that should abandon resources. IDAHEX requests the player to list the quantities to be abandoned.

Prompting: List integral quantities to be
             abandoned, in order.

Reply: "⟨q1⟩, ⟨q2⟩,...,⟨qn⟩"

The item q1 is the quantity of the unit's type 1 resources to be abandoned, q2 is the quantity of type 2 resources to be abandoned, and so on. Each item should be a nonnegative integer. The abandoned resources are immediately deducted from the unit's stocks; they cease to exist.

Example. Suppose the player's side has six types of resources. The player commands the destruction of 10 type 2 resources and 12 type 5 resources in battle unit 4:

Enter command.
"abandon 4"
List integral quantities to be
abandoned, in order.
"0 10 0 0 12"

A player cannot use the abandon command to destroy enemy resources: IDAHEX prevents a player from issuing commands to enemy units.

4-16

4.7   THE END COMMAND

The end command signifies that the player is finished entering commands at this time but may want to enter more commands before the game proceeds.

Initial reply: "end"

4.8   THE GO COMMAND

The go command signifies that the player is finished entering commands and is ready for game action to continue. When both players have issued the go command, the game proceeds.

Initial reply: "go"

4.9   THE STOP COMMAND

The stop command requests IDAHEX to stop the game by exiting from the IDAHEX computer program.

Initial reply: "stop"

Example:

> Enter command
> "stop"
> Do you wish to stop the war?
> (All work not saved will be lost.)
> yes

The word yes is entered by the player in response to IDAHEX's question. Note that it is not enclosed in quotation marks. Any response other than yes causes the stop command to be ignored.

The game situation can be saved prior to issuing the stop command by issuing the save command.

4.10   THE DISPLAY COMMAND

The display command is a request for information to be written on the player's terminal. The capabilities of IDAHEX's information display facility may exceed those described in this subsection; a player can learn the capabilities of the version he is using by entering the command "help display".

4-17

Initial reply: "display 〈keywd〉 〈number1〉 〈number2〉"

The argument keyword is a word indicating the kind of infor-
mation to be displayed. It is chosen from one of the follow
ing: "missions", "deliveries", "status", "unit", "balance",
"who". The following subsections describe the information
obtainable under the various values of keyword.

4.10.1 Missions

Initial reply: "display missions 〈number1〉 〈number2〉"

If number1 > 0 and number2 = 0 (or, equivalently, the item
number2 is omitted), the command causes display of information
on mission number1 (and therefore task force number1) provided
the task force belongs to the player's side. If number2 ≥
number1 > 0, the command causes display of information on each
of the player's missions numbered between number1 and number2.
If number1 = number2 = 0, the command causes display of infor-
mation on all of the player's missions.

Example. The player wants information on the composition
and mission of task force 29, which belongs to his side.

Enter command.
"disp miss 29"

Example. The player wants information on all his task
forces and their missions.

Enter command.
"disp miss"

4.10.2 Delivery Orders

Initial reply: "display deliveries 〈number1〉 〈number2〉"

If number1 = 0 and number2 = 0, the command causes display
of information on all the player's outstanding delivery orders.
If number1 > 0, delivery orders whose delivery task force is
not task force number1 are excluded. If number2 > 0, delivery
orders whose delivery destination is not cell number2 are
excluded.

Example. The player requests information on all his
delivery orders:

Enter command.
"disp deli"

4-18

Example. The player requests information on all delivery orders designating task force 29 as the delivery task force:

Enter command.
"display delivery 29"

No information will be displayed unless task force 29 belongs to his side.

Example. The player requests information on all of his delivery orders for cell 10:

Enter command
"disp deli,,10"

4.10.3 Battle Unit Status

Initial reply: "display status ⟨unitnbr1⟩ ⟨unitnbr2⟩"

If unitnbr2 ≥ unitnbr1 > 0, the command causes display of the status of every battle unit numbered between unitnbr1 and unitnbr2. If unitnbr2 = 0 (or is omitted) and unitnbr1 > 0, the status of battle unit unitnbr1 is displayed. If unitnbr2 = unitnbr1 = 0, the status of every battle unit is displayed.

Example. The player wishes to learn the status of battle unit 4:

Enter command.
"disp stat 4"

Example. The player wishes to learn the status of all battle units:

Enter command.
"disp stat"

4.10.4 Battle Unit Status and Resources

Initial reply: "display unit ⟨unitnbr⟩"

If unitnbr > 0, the command causes display of information on the status and resources of battle unit unitnbr. If unitnbr = 0, the command causes display of information on the status and resources of every battle unit, including enemy units.

4-19

Example. The player requests information on battle unit 4:

    Enter command.
    "disp unit 4"

Example. The player requests information on all battle units:

    Enter command.
    "display units"

4.10.5 Battle Unit Resources Available for Combat

IDAHEX groups engaged battle units into combat forces. Two units belong to the same combat force if and only if they are from the same side, participating in the same engagement, and located in the same cell. (A unit can never participate in more than one engagement at the same time.) The members of a combat force share their support resources and use their weapons in concert. The quantity of the force's weapons of a given type that can participate in combat may be reduced if the force fails to have enough supplies and personnel of various types to meet its resources' requirements for support; also, certain types of weapons can not participate in combat without the protection of other weapons. The "fractional involvement" of a given type of equipment is the fraction of the force's equipment of that type that can participate in combat; the fractional involvement of a given type of support resources is the fraction of the force's support resources of that type that are available and needed to support other resources.

    Initial reply: "display balance ⟨unitnbr⟩"

If unitnbr > 0, IDAHEX interprets it as the identification number of an engaged battle unit, finds the combat force to which the unit belongs, and displays fractional involvement data for the force. Alternatively, if unitnbr = 0, IDAHEX requests the players to list a group of one or more units, which will be regarded as though it were a combat force in finding the fractional involvement of its resources. (The only restriction is that these units must all belong to the same side.)

Example. Suppose units 11, 13, and 69 comprise the set of Red units participating in a given engagement and located in a given cell. The player wants to learn their resources' fractional involvement.

4-20

Enter command.
"display balance 13"

The preceding command is equivalent to each of the following
two commands:

"disp bala 11"
"disp bala 69"

Example. The player wants to learn what the fractional
involvement of the resources in units 58, 60, and 69 would be
if these units presented constituted a combat force:

Enter command.
"disp bala"
List units.
"58, 60, 69"

IDAHEX displays the fractional involvement of each type of
resources (in the indicated combat force or the specified group
of units) on a line labeled "f.i.". On the next line IDAHEX
displays the "delta number" of each type of resources. The
delta number of a given type of equipment, say type i, is
defined as $P - Q$; P is the total quantity of type i equipment
that could be protected by the weapons of the combat force or
group of units, assuming the fractional involvement of every
type of weapon were 1; Q is the total quantity of type i equip-
ment in the combat force or group of units. If type i equip-
ment can protect itself, $P - Q = +\infty$, which is written as a
string of asterisks. Unprotected equipment can not participate
in combat. The delta number of a given type of support resources,
say type k, is defined as $S - D$; D is the aggregate demand for
type k support by the resources in the combat force or group of
units; S is the total quantity of type k support resources in
the combat force or group of units. Thus, a negative delta
number for a type of weapons may indicate that their participa-
tion in combat is limited by a shortage of weapons capable of
protecting them; a negative delta number for a type of support
resources may indicate that participation of weapons in combat
is limited by a shortage of this type of support.

4.10.6 Battle Units by Location

Initial reply: "display who ⟨cellnbr⟩"

The command causes display of the identification number of
every battle unit whose posture class is nonnegative and whose
location is cell cellnbr.

4-21

Example. The player wants a list of the non-destroyed
battle units--enemy as well as friendly--in cell 9:

> Enter command.
> "disp who 9"

4.11 SAVING AND RESTORING GAME SITUATIONS

The save command causes IDAHEX to save a complete descrip-
tion of the present game situation together with all the game
design data except *envmap*, *rtemap*, and *barmap* in an unformatted
file.

Initial reply: "save"

IDAHEX requests the player to identify by number the file in
which the data should be stored.

Example. The player saves the current situation in file
91:

> Enter command.
> "save"
> Enter number (65-99) of file
> in which game situation should
> be saved.
> "91"

IDAHEX requests a file number greater than 60 to discourage the
player from giving the number of a file such as file 50 or file
60 that should not be overwritten.

The restore command causes IDAHEX to read game design data
and a complete description of a game situation from an unfor-
matted file created by a previous save.

Initial reply: "restore"

Example. The player restores the game situation, includ-
ing all missions and delivery orders, exactly as it was at the
time of the last save command (by either player) designating
file 91:

> Enter command.
> "rest"
> Enter number of file from which
> game should be restored.
> "91"

4-22

The players are given an opportunity almost immediately after IDAHEX is invoked to restore a game situation that was saved by a save command given in an earlier invokation of IDAHEX. They do this not by giving a restore command but by responding to a specific prompting on the Red player's terminal. This permits IDAHEX to acquire all the game design data except *envmap*, *rtemap*, and *barmap* from the unformatted file in which the game situation was saved rather than the formatted file 50; *envmap*, *rtemap*, and *barmap* are still acquired from file 60.

4.12 THE HELP COMMAND

In its simplest form, the command is invoked as follows:

    Enter command.
    "help"

In response IDAHEX lists on the player's terminal all the commands recognized by the IDAHEX release he is using. IDAHEX also explains how the help command can be used to obtain information on the other commands.

4.13 SUMMARY OF COMMANDS

Initial reply:	"abandon ⟨unit_id⟩"
Prompting:	List integral quantities to be abandoned, in order.
Reply:	"⟨quantity1⟩ ⟨quantity2⟩···⟨quantityn⟩"
Initial reply:	"attach ⟨unit_id⟩ ⟨task_force_id⟩"
Initial reply:	"cancel delivery ⟨task_force_id⟩ ⟨cell_id⟩"
Initial reply:	"cancel mission ⟨task_force_id⟩"
Initial reply:	"deliver ⟨task_force_id⟩ ⟨cell_id⟩"
Prompting:	Enter relative size of delivery.
Reply:	"⟨real_number⟩"
Prompting:	List recipients.
Reply:	"⟨id1⟩ ⟨id2⟩···⟨idn⟩"
Initial reply:	"display balance ⟨unit_id⟩"
Prompting[1]:	List units.
Reply[1]:	"⟨unit_id1⟩ ⟨unit_id2⟩···⟨unit_idn⟩"
Initial reply:	"display deliveries ⟨task_force_id⟩ ⟨cell_id⟩"

---

[1]Occurs if and only if unit_id ≤ 0.

```
Initial reply:      "display missions <mission_id1> <mission_id2>"

Initial reply:      "display status <unit_id1> <unit_id2>"

Initial reply:      "display unit <unit_id>"

Initial reply:      "display who <cell_id>"

Initial reply:      "detach <unit_id>"

Initial reply:      "end"

Initial reply:      "go"

Initial reply:      "help <command_name>"

Initial reply¹:     "mission <mission_id>"
Prompting:          Enter orders.
Replies:            "<objective1> <posture1> <time1>"

                        ⋮        ⋮        ⋮

                    "<objectiven> <posturen> <timen>"
Prompting:          List task force.
Reply:              "<unit_id1> <unit_id2>···<unit_idn>"
Prompting:          Enter transport mode.
Reply:              "<integer>"

Initial reply:      "restore"
Prompting:          Enter number of file from which
                    game should be restored.
Reply:              "<file_number>"

Initial reply:      "save"
Prompting:          Enter number (65-99) of file in which
                    game situation should be saved.
Reply:              "<file_number>"

Initial reply:      "stop"
```

---

¹The argument mission_id should be omitted unless the goal is
to modify an existing mission, whose number equals mission_id.

```
Initial reply:    "transfer <cell_id>"
Prompting:        List resource givers.
Reply:            "<id1> <id2>...<idn>"
Prompting:        List resource takers.
Reply:            "<id1> <id2>...<idn>"
Prompting:        Enter resource index and amount to
                  be transferred, line by line.
Replies:          "<resource_number1> <amount1>"
                        .              .
                        .              .
                        .              .
                  "<resource_numbern> <amountn>"
```

5. COMBAT

An engagement arises when a battle unit attempts to occupy a cell containing enemy units in hold or disengagement postures. To be precise, an engagement arises when a task force in an attack posture attempts to enter a hold posture in its objective while its objective contains one or more enemy units in hold or disengagement postures. Its objective then becomes the "engagement location". The task force constitutes the engagement "attackers". Other units from the task force's side may join the engagement subsequently, possibly attacking from different directions; they, too, become "attackers". The enemy units whose location is the attackers' objective and whose postures are hold or disengagement constitute the engagement "defenders". Thus, at the outset of the engagement, one side is the attacker and the other side is the defender. These roles remain fixed throughout the engagement. That does not mean that defenders cannot counterattack : a successful counterattack precipitates a new engagement in which some units that were defenders become attackers and some that were attackers become defenders.

An engagement ends when all its attackers have left or all its defenders have left. If an attacker's location is not the engagement location, it leaves the engagement when its objective becomes a cell other than the engagement location. If an attacker's location is the engagement location, it leaves the engagement when it enters a posture class other than 1 or 2. A defender leaves the engagement when it enters a posture class other than 1 or 2.

A defender may leave its engagement by being destroyed, but ordinarily one leaves by entering a movement posture. While it is in a movement posture, the enemy cannot engage it. That is one reason for the disengagement delay, and specifically, for making one term of the delay proportional to the anticipated movement delay. Loosely speaking, if the tactical situation implies that the unit is vulnerable to pursuit by engagement attackers, its disengagement delay (hence, the interval during which it is engaged) is extended to account for the combat that its rearguard might actually have with pursuing units.

Each engagement has a stylized FEBA that measures the attackers' progress. The variable feba expresses the FEBA position as a fraction of the cell depth. At the start of a given engagement, feba = 0. At that point all the attackers are in attack postures oriented toward the engagement location. If the attackers are sufficiently strong relative to the defenders, the FEBA advances--feba increases, to a maximum of 1. One might imagine that when feba is increasing, the attackers are beating back the defenders; a more general, more contemporary, interpretation is that the attackers are penetrating the defenders' formation. The game design datum *febad* is the criterion for deciding when the attackers have penetrated sufficiently to be allowed to occupy the engagement location. As soon as feba ≥ *febad*, the attackers are allowed to enter hold postures in the engagement location, ownership of it passes to their side, and the defenders must disengage and move out or be destroyed.

An engagement's FEBA is independent of other engagements' FEBAs and the general disposition of forces in the area of war. It may be interpreted as a measure of the attackers' penetration of the engagement location, but essentially it is just an abstraction used to determine how long the engagement lasts before the attackers defeat the defenders.

5.1 ATTRITION

This subsection outlines how losses are determined in a given engagement. Section 5.1 of Volume 2 explains fully.

Only ground-to-ground weapons can destroy enemy resources in ground combat. The game design data fix the basic rate at which each type of Red ground-to-ground weapon can destroy each type of Blue materiel and the basic rate at which each type of Blue ground-to-ground weapon can destroy each type of Red materiel. Ground-to-ground weapons do not kill personnel directly; rather, destruction of materiel of a given type by an enemy weapon of a given type implies certain personnel losses. The basic rate at which a given type of weapon can kill a given type of materiel must be adjusted for: (1) the weapon's allocation of fire, which depends on the composition of the enemy force in the engagement; (2) the posture of the battle unit to which the weapon belongs; (3) the posture of the battle unit to which the materiel belongs; (4) the environment in the engagement location. If the weapon belongs to an attacker, further adjustments may be needed for: (5) any barrier between the attacker's location and the engagement location; (6) the defense preparation time of the unit to which the materiel belongs. After all these adjustments are made, the result is a potential kill rate for every combination

of shooting ground-to-ground weapon type, target materiel type, shooting unit, and target unit: define $K(i,j,m,n)$ as the potential rate at which a type i weapon belonging to unit m destroys type j materiel belonging to unit n, where unit m and unit n are on opposite sides in the engagement.

Let unit m be a participant in the engagement. Let i be a type of ground-to-ground weapon belonging to it. The quantity of type i weapons in unit m that can participate in combat is 0 if unit m is a passenger in a stacked task force. If not, the quantity is limited by two factors: support and protection. First, assuming that the support resources category of the unit's side is nonempty, insufficient quantities of support resources may limit the fraction of type i weapons that can participate in combat. Second, certain types of weapons may not participate in combat unless certain other types are participating in sufficient quantities to protect them. The first factor means that a force cannot employ its weapons unless they are supplied and manned. The second enforces a crude combined arms concept; typically, it means that a force must have enough small arms participating in combat to protect its participating tanks, and enough participating small arms and tanks to protect its participating artillery. The force to which unit m belongs, for the purpose of determining its weapons' participation in combat, is the set of every unit that is from the same side, participating in the same engagement, and located in the same cell as unit m, excluding any unit that is a passenger in a stacked task force. The units in the force share their support and employ their weapons in concert so that each has the same fraction of weapons of a given type participating in combat.[1]

Once the quantity of participating type i weapons in unit m is found, it is adjusted to get the effective quantity, which may differ because a very low or very high density of friendly forces in the cell where unit m is located may degrade its weapons' effectiveness. The quantity $ERS(m,i)$ is defined as the effective quantity of type i weapons in unit m that can participate in combat.

If unit m is an engagement attacker, let unit n be any defender; if unit m is a defender, let unit n be any attacker. The potential loss rate of type j materiel in unit n due to fire from the type i ground-to-ground weapons in unit m is computed as

$$K(i,j,m,n) * ERS(m,i).$$

---

[1]The quantity of a unit's type i weapons participating in combat, divided by the quantity it has, defines the "fractional involvement" of its type i resources. This number can be found through the command "display balance".

5-3

This rate implies potential loss rates of the various types of personnel in unit n. Let nw be the number of types of ground-to-ground weapons for the side to which unit m belongs. The total potential loss rate of type j materiel in unit n due to all fire from unit m is

$$\sum_{i=1}^{nw} K(i,j,m,n) * ERS(m,i).$$

Summing over all the friends of unit m in the engagement would yield the total potential loss rate of type j materiel in unit n. Two additional adjustment factors are applied to the potential loss rate to find the actual loss rate.

The first adjustment, which the game designer may choose to omit, scales loss rates according to the loss rates that would be predicted from the engagement's ground force ratio. Because the loss rates computed above obey the Lanchester square law, it is appropriate to determine the values of the weapons in the engagement by the antipotential potential method. (See Section 5.1.2 of Volume 2.) The method, as IDAHEX uses it, generates a value for each type of the attackers' ground-to-ground weapons and each type of the defenders' ground-to-ground weapons. A weapon's value reflects its ability to destroy enemy weapons under the specific conditions of the engagement; the value of a given type of Red or Blue weapon usually differs from engagement to engagement and changes over the course of any one engagement. Let $v(i)$ be the value of a type i ground-to-ground weapon held by the attackers in this engagement at this time. Let unit m_1, unit $m_2, \ldots,$ unit m_k be the attackers. Their total value is

$$\sum_{j=1}^{k} \sum_{i=1}^{nw} ERS(m_j,i) * v(i),$$

where nw is the number of types of the attackers' ground-to-ground weapons. The defenders' total value is computed analogously. The ground force ratio is defined as the ratio of the attackers' total value to the defenders'. If the game designer wants losses scaled according to the ground force ratio, the game design data include curves that predict the fraction of their total value that the attackers lose and the fraction of their total value that the defenders lose given the ground force ratio, the attackers' posture, and the defenders' posture. The potential loss rates are scaled to agree with this prediction.

The second, and final, adjustment scales the loss rates according to the intensity of combat, which is lowest when

feba = 0 and greatest when feba indicates that the attackers have penetrated to the depth of the defense.

5.2 FEBA MOVEMENT

As noted earlier in this section, in a given engagement the variable feba measures the attackers' progress. Its value always lies between 0 and 1. Its rate of change at any moment of the engagement depends upon a force ratio that includes the contribution of close air support (CAS). Air strikes are assessed periodically, and losses of ground-to-ground weapons inflicted by CAS are recorded for use by the ground combat procedure. To find a force ratio that reflects both the ground forces and the air forces in the engagement, it is necessary to value them consistently. The antipotential potential method offers a way of valuing the attackers' close air support that is completely consistent with the way the attackers' total value is determined (Section 5.1), and it offers a way of valuing the defenders' close air support that is completely consistent with the way the defenders' total value is determined. The combined ground-air force ratio is defined as

$$\frac{A1 + A2}{D1 + D2},$$

where A1 is the attackers' total ground value, A2 is the value of their CAS, D1 is the defenders' total ground value, and D2 is the value of their CAS. The rate of change of the variable feba depends upon the combined ground-air force ratio, the attackers' posture, the defenders' posture, and the side to which the attackers belong. This rate may be negative.

There is a game design datum called *febad*; $0 < febad \leq 1$. If, at some point in the engagement, feba becomes greater than or equal to *febad*, the defenders are declared defeated and required to retreat--to disengage and move to an adjacent cell. IDAHEX judges whether they have a line of retreat and, if so, selects the adjacent cell toward which they should go. If they have no line of retreat, they are destroyed. The procedure for finding and selecting a line of retreat is explained in Section 5.3 of Volume 2.

6.   AIR SUPPORT

Periodically, IDAHEX informs each player that he may make
air strikes. IDAHEX includes no air warfare model and therefore
has no way of ascertaining what air assets a side can allocate
against enemy ground forces. It assumes that any air strike a
player enters is within his side's capability. In practice the
game designer adopts either of two solutions: he gives each
player a list of the air assets available at various times in
the game for use against enemy ground forces, or he runs an
air warfare model concurrently with IDAHEX.

An air strike is specified in a sequence of requests by
IDAHEX and replies by the player. The player's replies are
interpreted according to the rules described at the beginning
of Section 4.

IDAHEX requests input of an air strike by requesting the
player to "Enter air strike role and target cell." The player's
reply is mapped into a list of two items. The first item is
the strike role, a character string variable. If the strike
role is "cas" or "CAS", the strike is close air support. If
not, the strike is air interdiction of battle units. The
second item is the identification number of the target cell,
the cell toward which the strike is directed. Making this
number nonpositive signals IDAHEX that the player is finished
entering air strikes at this time.

If the strike is interdiction, IDAHEX requests the player
to "Enter target posture classes in order of priority." His
reply must contain four items--the integers 1, 2, 3, and 4,
in any order. The reply is mapped into the list: $asprty(1)$,
$asprty(2)$, $asprty(3)$, $asprty(4)$. The highest priority posture
class is $asprty(1)$, and the lowest is $asprty(4)$.

Next, IDAHEX requests the player to "List, in order,
number of each type of aircraft in strike." His reply must
be a list of integers. The i-th item in the reply fixes the
number of type i aircraft participating in the strike.

The game design data fix for each side the number of types of aircraft; the number of types of air-to-ground weapons; and the notional load of each type of aircraft--the quantities of the various types of air-to-ground weapons that it carries. Knowing the strike's composition, IDAHEX can ascertain the total quantity of each type of air-to-ground weapon delivered. To assess the resulting damage, it must construct a set V, the set of battle units that are victims of the strike.

Suppose the strike role is CAS. If there is no engagement whose location is the target cell, the player is warned and no strike occurs. If such an engagement exists, let V be the set of every enemy unit in the engagement. If the enemy units are the engagement defenders and at least one of them is in a hold posture, then delete from V every unit in a disengagement posture.

On the other hand, suppose the strike role is interdiction. Let k be the smallest integer such that asprty(k) equals the posture class of some enemy unit located in the target cell. Thus, asprty(k) is the highest priority posture class that appears among enemy units in the target cell. Define V as the set of every active enemy unit whose location is the target cell and whose posture class is asprty(k). Behind this definition of V is the implicit assumption that the strike aircraft can only distinguish enemy units from each other by location (cell) or posture class.

Choose any unit in V. The quantity of the unit's materiel of a given type that is destroyed in the strike depends upon the composition of the strike; the allocation of the air-to-ground weapons' fire, which depends on the resource composition of V; the unit's posture; and the environment in the target cell. Destruction of a given type of materiel by a given type of air-to-ground weapon induces losses of the various types of personnel.

Example. The player inputs an air strike in support of his units engaged in cell 124:

Enter air strike role and target cell.
"cas 124"
List, in order, number of each type of aircraft in strike.
"25 25 0 13"

<u>Example</u>. The player wishes to attack moving enemy units
whose location is cell 45:

                Enter air strike role and target cell.
                "int 45"
                Enter target posture classes in
                order of priority.
                "3 2 1 4"
                List, in order, number of each type
                of aircraft in strike.
                "0 15"

7. SUPPLIES CONSUMPTION

Only an active unit consumes supplies. To be precise, the unit's resources consume supplies. Consumption of supplies of a given type by resources of a given type depends on the unit's posture and, if it is engaged, the fraction of the resources that are participating in combat. One qualification must be added: if the unit is a passenger in a stacked task force, it consumes supplies as though it were in a hold posture and not engaged. A unit that does not belong to a task force consumes supplies out of its own stocks. The supplies in a task force are pooled to meet the consumption of all its elements, and every element's stock of a given type of supplies is reduced by the same factor.

A unit's stocks of supplies cannot become negative. If the stock of a given type of supplies in a unit or task force falls to 0, the unit or task force simply does not consume any more. It suffers no immediate penalty, such as spontaneous attrition, as a result. Nevertheless, lacking supplies of some type might prevent it from moving (might make its movement delay infinite) and, should it become engaged, might prevent some or possibly all of its weapons from participating in combat.