USACERL INTERIM REPORT P-90/06
January 1990
VIC-Based Engineer FAM (AMIP)

US Army Corps
of Engineers
Construction Engineering
Research Laboratory

The Enhanced Engineer Module of the Vector-in-Commander (VIC) Battle Simulation

by
Carol Subick

This report documents the enhancements made by the U.S. Army Construction Engineering Research Laboratory to the Vector-in-Commander (VIC) battle simulation during the first half of the VIC-EFAM portion of the Engineer Model Improvement Program (EMIP). The engineer module of VIC Version 2.0 has been completely redesigned and coded. This report describes the methodology, data structures, and input data requirements, summarizes routines and events, and presents a calling tree for the engineer module. This Interim Report will be replaced by a final Technical Report when the project is completed in 1991.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1990	Interim

4. TITLE AND SUBTITLE
The Enhanced Engineer Module of the Vector-in-Commander (VIC) Battle Simulation

5. FUNDING NUMBERS
PR 4A162734AT41
WU SE-Y29

6. AUTHOR(S)
Carol Subick

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Construction Engineering Research Laboratory
P.O. Box 4005
Champaign, IL 61824-4005

8. PERFORMING ORGANIZATION REPORT NUMBER
USACERL IR P-90/06

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of the Chief of Engineers
20 Massachusetts Avenue, NW.
Washington, DC 20314-1000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Copies are available from the National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161.

12a. DISTRIBUTIC ABILITY STATEMENT
Approved f .c release;
distributio limited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This report documents the enhancements made by the U.S. Army Construction Engineering Research Laboratory to the Vector-in-Commander (VIC) battle simulation during the first half of the VIC-EFAM portion of the Engineer Model Improvement Program (EMIP). The engineer module of VIC Version 2.0 has been completely redesigned and coded. This report describes the methodology, data structures, and input data requirements, summarizes routines and events, and presents a calling tree for the engineer module. This Interim Report will be replaced by a final Technical Report when the project is completed in 1991. Keywords:

14. SUBJECT TERMS
Vector-in-Commander,
military engineers,

15. NUMBER OF PAGES
91

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

FOREWORD

This work was performed for the Office of the Chief of Engineers (OCE) under Project 4A162734AT41, "Military Facilities Engineering Technology"; Work Unit SE-Y29, "VIC-Based Engineer FAM (AMIP)." The OCE Technical Monitor was MAJ T. Bauer, DAEN-ZCM.

The research was conducted by the Facility Systems Division, U.S. Army Construction Engineering Research Laboratory (USACERL-FS). Dr. Michael J. O'Connor is Chief of FS. The USACERL technical editor was Dana Finney, Information Management Office.

COL Carl O. Magnell is Commander and Director of USACERL, and Dr. L. R. Shaffer is Technical Director.

CONTENTS

CONTENTS (Cont'd)

THE ENHANCED ENGINEER MODULE OF THE VECTOR-IN-COMMANDER (VIC) BATTLE SIMULATION

1 INTRODUCTION

Background

The Army engineer community has long recognized that combat engineers' ability to meet the demands of the modern battlefield has suffered because of decisions based on the inadequate representation of their role in the battle simulations used to analyze force structure requirements and doctrine. Vector-In-Commander (VIC), the combined arms simulation at the Corps/Division level which is accredited for use in Army studies, is the focus of the Engineer Model Improvement Program's (EMIP's) early efforts at remedying this situation.

Work began in FY89 at the U.S. Army Construction Engineering Research Laboratory (USACERL) to redesign the engineer module of VIC. The engineer module in the reference version (2.0) of VIC was discarded, and a completely new module was implemented. The computer code, documentation, and required configuration control reports were delivered to the VIC model proponent, U.S. Army Training and Doctrine Command (TRADOC) Analysis Command at White Sands Missile Range (TRAC-WSMR), on 31 August 1989.

This report describes the version of VIC's new engineer module which was delivered to TRAC-WSMR. It provides interim information to familiarize programmers and model users with the basic features and requirements of the module. With the new module in place in VIC Version 3.0, USACERL's efforts during FY90 will concentrate on enhancing existing processes and adding new capabilities. A final report documenting the entire 2-year effort will replace this report in 1991.

Objective

The objective of this work is to ensure that VIC represents combat engineers well enough to allow analysts to accurately measure both the impact of changes in force structure on engineer capability and the contribution of engineers to the combined-arms team. The objective of this Interim Report is to describe the data structures and methodology of the new engineer module as implemented in Reference VIC 3.0.

Mode of Technology Transfer

The final product will be verified computer source code for the VIC engineer module with documentation as required by the VIC Configuration Management Charter.

2 METHODOLOGY

Overview

The engineer module (EN) of the Vector-In-Commander (VIC) battle simulation controls the explicit representation of engineers. The USACERL effort during the first half of the VIC-EFAM development work under the EMIP concentrated on a complete redesign of EN, changing Reference VIC's basic representation of engineers. The changes involve every aspect of that representation: engineer task capabilities, units, assets, methods, job performance, and command and control (C^2).

To say that engineers are "explicitly represented" means that each aspect of engineer operations is recognizably present in the model. Engineer units appear on the battlefield; their movement is affected by the terrain and the obstacles they encounter; they may suffer attrition from direct and indirect fire. The equipment and supplies that engineers use to accomplish their missions are available in known, limited quantities as determined by input data. The generation of engineer tasks, whether dynamically or by input data, and the assignment of those tasks to specific units follow well defined model rules. The type and quantity of equipment and supplies needed to do a specific task and the time required to do that task are all determined by input data. For performance of the task, each step is simulated: the formation of a work team, the movement of the work team to the job site, the impact of actual conditions on work time and effectiveness, and the return of the work team to its base. When the work is complete, the resulting change in the terrain has an impact on the way ground units move and/or suffer attrition.

The scenario developer determines the resolution of the simulation of engineer task performance by specifying the level of detail in the EN data module. The EN data describes engineer resources and techniques. In a low-resolution data set, the resources might be task-organized teams with only one simple technique for each type of task. In a high-resolution data set, Tables of Organization and Equipment (TOEs) can be used to determine the resources down to the specific equipment item, with each item having the possibility of being used for a number of different types of tasks. In this case, the number of techniques for any given task may be very large, with distinctions arising from different work times, equipment mixes, organizational variations, and end results.

Engineers are also "implicitly represented" in VIC. The EN data module focuses on specifying how a task is done and what resources are needed to do it. In certain situations, a maneuver unit may use this data to determine its own capabilities. In the current version of the engineer module, mobility tasks (obstacle and minefield breaching) can be accomplished by nonengineer units using organic assets. When such a unit encounters an obstacle or minefield and engineers are being played, the unit's weapon groups and on-hand supplies are compared with the equipment and supply requirements for the breaching techniques. If the unit owns all of the required equipment and supplies, the delay and attrition will be reduced according to the chosen technique's specifications. No simulation of task performance is done. In the case of line obstacles that require bridging, no breach point is left when the unit passes the obstacle. The assumption is that this would represent a river crossing in which retrievable bridging is used. If a scenario contains only mobility tasks for RED engineers, this new capability can be used to reduce the time required for explicit simulation of task performance by making all of the RED engineer resources the organic assets of non-engineers.

Engineer Task Types

Engineer activities revolve around the need to build, repair, breach, or remove one of the following physical features: minefields (alter movement and attrition), line obstacles (alter movement),

defensive positions (alter attrition), roads (alter movement), and facilities (alter specific nonengineer missions).

Each engineer task is defined by its task type and by a "feature type," with the distinguishing characteristics of each feature type determined by input data. For tasks related to minefields, the feature types are the mine types described in the minefield data module. For tasks related to line obstacles, the feature types are the line obstacle prototypes described in the terrain-barrier (TB) data module. For defensive position preparation, the feature types are the defensive position prototypes described in the EN data module. For example, any number of line obstacle feature types can be defined in the TB data module: ATD, Concertina wire, Embankment, 25M River, 50M River, etc. Each of these feature types has an associated set of delay factors established in the TB and EN input data and an associated set of breach techniques established in the EN input data. The task type and feature type together determine the engineer task (for example, bridging a 50M river has task type ..RIVER.CROSSING and feature type 50M River).

For the current release, the EN module has restored the engineer task capabilities present in earlier versions of VIC (placing minefields and linear obstacles, building defensive positions, and bridging linear obstacles) and added minefield breaching/clearing. By the end of the VIC-EFAM development, engineers will be able to perform all of the task types listed below.

MINEFIELD BREACH

Engineer effect: increases the MF.FRACTION.CLEARED for the given minefield, thereby reducing attrition and delay for crossing units. Engineer minefield breaching tasks are generated dynamically when a maneuver unit encounters an enemy minefield and the crossing delay is large enough to allow an engineer work team to complete a breaching operation before the unit moves through on its own.

MINEFIELD CLEAR

Engineer effect: removes the given minefield from the set of active minefields. Engineer minefield clearing tasks are generated dynamically when a maneuver unit encounters an enemy minefield and either bypasses it or uses its own resources to breach it.

LINE OBSTACLE BREACH (RIVER CROSSING)

Engineer effect: adds a new breach point to the segment's list of breach points, thereby reducing delay times for units using the breach to a data-specified delay time associated with the breach method. (In earlier versions of VIC, a line obstacle segment was either breached or unbreached. With the current implementation, each line segment has a list of breach points, each of which has a specific location and delay time.) Obstacle breach/bridge missions are generated dynamically when maneuver units encounter unbreached obstacles. In future releases, an enhanced river-crossing play will allow anticipatory engineer activity.

IMPROVE LINE BREACH

Engineer effect: used for follow-on improvements and to retrieve engineer assets like armored vehicle launched bridges (AVLBs) by replacing them with other forms of bridging--not currently implemented.

REPAIR ROAD CRATER

Engineer effect: depends on future implementation of roads--not currently implemented.

BUILD TRAIL

Engineer effect: depends on future implementation of roads--not currently implemented.

BUILD FALF (Forward Air-Landing Facility)

Engineer effect: depends on coordination with groups outside of the engineer community--not currently implemented.

EMPLACE MINEFIELD

Engineer effect: adds a new minefield to the list of active minefields, thereby causing delay and attrition to encountering units. The minefield delay and attrition factors are determined in data by the minefield prototype information. In the current version, minefield emplacement tasks arise only from data in Segment Five (Preplanned Minefield Emplacement) of the MF data module.

EMPLACE LINE OBSTACLE

Engineer effect: adds a new linear obstacle with "ACTIVE" engineer status to the set of linear obstacles, thereby causing a delay for encountering units. The line obstacle delay factors are determined in data by the linear obstacle prototype information. In the current version, line obstacle emplacement tasks arise only from data in Segment Four (Linear Obstacles and Line Features) of the TB data module.

REMOVE LINEAR BREACH/BRIDGE DEMOLITION

Engineer effect: removes a breach point from the list of breach points for a linear obstacle segment, possibly retrieving engineer assets if the side is the emplacing side and a retrievable asset has been left there--not currently implemented.

CRATER ROAD

Engineer effect: depends on future implementation of roads--not currently implemented.

PREPARE POSITION

Engineer effect: lowers attrition and visibility for protected units. In the current implementation, defensive positions are associated with a location rather than with a unit, and defensive position prototypes now exist to register differences in engineer techniques and effects. In the near future, the representation of defensive positions and their preparation will be expanded. For now the effect of position preparation is registered as in earlier versions of VIC. See FEATURE TYPES for new data structures. Defensive position preparation tasks arise from data in Segment Two (Ground Unit Data) of the global ground (GG) data module.

BUILD CSS FACILITY

Engineer effect: depends on coordination with groups outside of the engineer community--not currently implemented.

Engineer Missions

As the need for engineer activity arises, whether by specific battle plan in the input data or by dynamic generation, engineer missions are created to hold the information about what needs to be done and to link together jobs to be done as a single "mission." All of the jobs that arise from one mission must be of the same type, i.e., the task and feature type must be the same for all. A system-owned variable, EN.MISSION.COUNT, is used to assign a unique sequential number to each mission, linking the jobs arising from the mission by this common number. The missions are filed in the set of missions owned by the side and represent the unassigned engineer work to be done.

Periodically, the set of missions is processed, and those missions that can be assigned to a specific engineer unit are converted to jobs and removed from the mission set. A mission can be assigned as a job if the side has an engineer unit with the mission location in its tactical area and the ability to do the work. A mission that is impossible because the engineer input data has no technique for its task and feature type is removed from the mission set and discarded. Missions that are possible but for which no engineer unit currently has the site in its tactical area or a usable technique for the given circumstances remain in the mission set. They are reprocessed at cyclic intervals and whenever an engineer unit moves to a new tactical area. The first data item in the engineer data module determines the length of the reprocessing interval. To save run time, the interval should be in the range of 30 min to 1 hr.

Once a mission has been assigned to an engineer unit, a job is created and filed in the unit's pending list for each feature associated with the mission. The mission is removed from the side's mission list and destroyed. An engineer job is specified in terms of what needs to be done by the task type, feature type, and location (example: "Emplace a minefield of type A at location (x,y)"). The unit responsible for the job is determined by the command structure and unit capabilities and tactical areas, all of which are determined by input data. The priority ranking of the jobs is determined by the command structure and the situation. The possible methods for doing the job are determined by input data. The manner in which the job is actually done is determined by matching the equipment required by appropriate techniques for the given task/feature type with the equipment available to the responsible unit. The unit that does the work is a task-organized work unit created solely for the purpose of the job at hand using equipment owned by the responsible unit.

Engineer Units

The engineer module has three types of engineer units. The engineer headquarters (HQ) unit is a ground unit that is actually in the GG input data. The engineer work team (WT) is a task-organized work unit assigned to perform a specific set of jobs with equipment required for the jobs temporarily transferred to it by the HQ unit responsible for the jobs. The engineer staff unit is an HQ unit that has a nonengineer superior. The staff units are used primarily for modeling engineer command and control.

In VIC, only ground units move, fight, consume supplies, and interact. To model task-organized work units, extra BLUE and RED ground units of type engineer are created at the start of the simulation, marked as ..REMOVED so that they are invisible to the simulation until activated, and filed in a set of free units. Activating a free unit means removing it from the free set and using the associated ground unit for a job assignment. When the job is completed or the unit is destroyed, the associated ground unit is stripped of its attributes and removed; the work team is refiled in the free set for later use on other jobs. Two new data items in the GG data module specify the number of extra BLUE and RED engineer units to be created. Unlike earlier versions of VIC in which the engineer work units actually appeared in the ground unit input data, the new representation manages the organization and processing of work teams internally.

All engineer units are ground units, but only engineer HQ units have their attributes set directly by input data. As ground units that appear in the data modules, HQ units have their tactical command structure set in the ground movement (GM) data, their paths and tactical areas in the GG data, their assets in the front-line (FL) data, and their supply needs in the logistics (LO) data. HQ units do not do engineer work. They hold the engineer assets and are responsible for jobs as determined by simulation rules, but assign work teams to do the work with the HQ unit's assets. Each HQ unit keeps an inventory of owned equipment--one entry for each piece of equipment. For rapid processing, the HQ unit also keeps an equipment array that indicates at any given time the number of pieces of each asset prototype available at the HQ location. Engineer staff units are simply HQ units with certain characteristics (namely, their GM.UN.SUPERIOR is not an engineer).

The engineer WT is a mechanism to move the engineer assets around the battlefield in a way that makes them visible as GG.UNITs to the other modules in the simulation. WT units are created internally, and their attributes are determined by their current use. A WT unit is filed in its side's EN.SET.OF.FREE.UNITS when it is not active. When an HQ unit is ready to organize equipment to send to a job, a WT unit is removed from the free set and filed in the HQ unit's set of performing units. The necessary assets are transferred to the WT unit, which has its own inventory to keep a record of its assets so that they can be returned properly when the work is finished. In the course of processing the jobs on its list, a WT unit may own assets that do not belong to its HQ unit and may have some of its assets preempted before its jobs are complete.

A WT unit can have one of three possible destinations: the location of the job site for the first job on its job list, a rendezvous location for itself and other WT units assigned to the same job phase, or a rendezvous path point with its HQ unit. When its purpose has been fulfilled, the WT unit transfers its assets to another unit (either another WT unit that will move the assets to a new job or back to base or the HQ unit that owns the assets held by this WT unit) and becomes inactive again. It is removed from the set of performing units and refiled in the free set.

Engineer Command Structure and Decision-Making

Each ground unit, including engineer units, has a tactical superior, a list of subordinates, and a tactical area. The unit's tactical superior indicates where the unit fits in the Army organizational structure and is set by the command structure data in the GM input data. Each engineer unit must be in one of two categories:

1) has an engineer tactical superior
2) has a nonengineer tactical superior.

Engineer units in category 1 are in a command chain of engineer units that ultimately reaches a unit in category 2. Engineer units in category 2 are "engineer staff units." The types of jobs generated dynamically for this unit and its subordinates and the priorities attached to those jobs are determined by the activities of its nonengineer superior and that unit's nonengineer subordinates. In particular, job responsibility is first determined by looking at the staff units, and all jobs assigned to a staff unit and its subordinates are prioritized on one list before the individual engineer HQ unit job lists are processed. In addition, all engineer units under a given staff unit share equipment, perhaps combining equipment from several engineer units to accomplish one job.

A list of the staff units is constructed during the initialization process. This list is ranked by the level of the associated ground units, with the lowest echelon units first. This process ensures that engineer jobs are assigned at the proper level in the command chain; i.e., in the search for a job's responsible unit, the lowest level units are considered first. Each staff unit also keeps an array of the techniques for which the needed equipment is owned by its command chain so that a quick

determination can be made as to whether this unit or one of its subordinates will be able to do a specific job.

To accommodate the dual command link needed to represent engineers accurately, each staff engineer unit has a second pointer--this one to its engineer superior. Staff units may or may not have an engineer superior. The purpose of this link is to provide for the shifting of assets between the units on either end of the link; i.e., assets can flow down to the staff unit from its engineer superior or away from the staff unit to its engineer superior. A staff unit under operational control of a nonengineer unit would not have such an engineer link, while a unit in direct support of a nonengineer unit would have such a link. The current version of the engineer module does not take advantage of this extra command link, but the routines to load this data are present. In the next release, decision tables will be available to control the use of this feature.

Engineers may have only other engineer units in their list of subordinates. The list of subordinates is a data structure built using the tactical command link only. An engineer unit in the list of subordinates of another engineer unit has its jobs prioritized by the staff engineer at the top of the chain. A staff unit does not appear in the list of subordinates of its engineer superior. This is as it should be since the staff unit's activities are not determined by its engineer superior but by its maneuver superior.

All units have tactical areas, i.e., portions of the battlefield for which they are responsible. An important assumption is that tactical areas of units are nested to agree with the tactical command structure. An engineer unit can be placed in general support of a region by the appropriate specification of its tactical area and superiors. An engineer unit with an engineer tactical superior will work within the assigned tactical area on jobs generated by the activities of the units in its area which are in the command chain of its staff engineer's superior. Such an engineer unit could be moved to a new tactical area by its engineer superior. Engineer units in category 1 above generally behave as if they have been placed in a general support role in their tactical areas.

Engineer Assets

Engineer assets are originally defined in the FL data as weapon prototypes. This is because a unit's mass, attrition, and supply needs are determined by its weapon assets. The FL data also sets the number of each type of asset held by each ground unit, including engineer HQ units. The engineer asset prototype data in the EN data specifies those weapon prototypes which are engineer assets and adds more information about the engineer assets being modeled. Nonengineer units can own weapons identified as engineer asset types, and engineer units can own weapons not identified as engineer asset types.

The engineer assets are actually identified in the EN data, but each engineer asset must correspond to a specific weapon prototype in the FL data. The EN.ASSET.PR.WP.PTR identifies that weapon prototype. Each of these prototypes can be a specific equipment item (e.g., M9 ACE), a generic equipment item (e.g., dump truck), or an aggregate (e.g., bridge team). An engineer data set can contain a combination of specific equipment prototypes, generic prototypes, and aggregate prototypes.

To restrict the continuous use of assets, each asset has a data-specified maximum work period before rest and a data-specified length of required rest period. Each time an asset returns from a work assignment, its inventory record is updated. When its hours worked exceeds the maximu vork period for its prototype, the asset is made unavailable for assignment for the specified rest p

To allow for a job technique to specify equipment by function (digging, hauling, trenching, etc.) rather than specific prototype, each asset has an associated set of functions. The functions modeled and the particular functions of each prototype are determined by input data. Associated with each asset prototype and function is a relative capability which is used for computing work rates for different pieces of equipment having the same function. This concept is explained more fully in the discussion of engineer techniques below.

When the EN data is loaded, an inventory list is constructed for each HQ unit. This list has a record for each piece of engineer equipment owned by the unit. The HQ unit inventory is ordered first by asset prototype so that all items of the same type are listed together, then by low current job priority, and then by low hours worked since rest. Assets that are available have 0 for their current job pointer and 0.0 for their current job priority level. When an asset is assigned to a work team to work on a specific job, that asset is removed from the inventory and its new job priority is set. Then the asset is refiled in the inventory in a new position. This method keeps the most available assets at the top of the respective sections, thus allowing quick location of assets available for a new job.

A unit's inventory list grows or shrinks only when a formal transfer of an asset is made from or to another HQ unit. When a WT unit has control of the asset, the asset's record is still with the parent unit, but with appropriate attributes set to indicate the asset's status. Monitoring this inventory list will yield engineer output data regarding the use of assets.

Some engineer assets (e.g., AVLBs) may not return with the WT unit when a job is complete, though the asset still exists and can be reused if retrieved. The job technique information indicates which equipment fits this description. When such a job is completed, the WT unit returns to its HQ unit without the retrievable equipment.

Engineer Techniques

While an engineer job is determined by the task type, feature type, and location, the engineer effort to get the job done is determined by the technique chosen according to the equipment available. For each side and task type/feature type, input data specifies a set of techniques by which the task can be done. The order in which the techniques are listed in the input data determines the order in which the techniques will be considered for use, with the most preferred technique listed first.

Each technique has associated with it a command level that identifies the echelon normally tasked with this type of work. In searching for a responsible unit, an attempt will be made to assign a job, using this technique, to the unit closest to this level in the appropriate command chain.

Each technique has associated with it a minimum signed distance from the FEBA which is used to distinguish those techniques that would be used in an assault situation from those used in rear areas. If the job site is closer to the FEBA than this minimum value, the technique will not be used. This minimum distance can be negative.

Each technique has a point of organization, which is either 1 (at HQ) or 2 (at SITE). If a job is using a technique with "at HQ" specified, then all of the equipment needed for the current phase will move first to the responsible HQ unit's location and form one work unit to proceed to the job site. If a job is using a technique with "at SITE" specified, then each set of collocated equipment being used for the job moves as a unit directly to the job site, forming one unit there. Work begins as soon as the equipment required for the current segment is at the job site. In addition, when "at SITE" is specified, equipment leaves for its next destination as soon as it is no longer needed at its present job site. When "at HQ" is specified, the equipment working on a phase of the job stays together until the phase is complete.

Each technique has a relative effect associated with it, though the current engineer module uses the relative effect of a technique only for line obstacle breaches. This is intended to distinguish between the effects of different types of engineer techniques on the same feature type. For example, not all bridges have the same delay factor for crossing units. The feature type has associated with it a standard breached delay. The breached delay associated with a particular engineer breach is computed by dividing the standard delay by the relative effect. For example, if the breached delay for a given line obstacle is specified to be 20 min in the TB data, a bridge with a "relative effect" of 2 would have a delay of 10 min.

Each technique has a list of segments that divides the job into smaller pieces of work. With this arrangement, equipment used only at the beginning of the technique can be released for other jobs before the present job is finished and interrupted jobs can have a measured effect and can be resumed without starting from the beginning.

Each technique has a list of required equipment that determines the type (either specific prototype or function) of equipment to be used and the preferred number of pieces of each type. This list is used first when it is compared with a unit's assets to determine if the unit can do a job by this technique. If this technique is chosen, this list is used to detect when an asset is needed at the job site and to determine job segment duration after assets have been attrited.

The key structure for tracking a job through a multitude of steps is the job segment. The segments can be part of a continuous work effort that ties up all needed equipment for the duration or a disjoint set of activities using different equipment in phases that only tie up the equipment used in that phase. A phase is defined to be from the current segment to the next break point/end of job. This method also allows for a job to be preempted by a higher priority job and for WT units to work in the assault, where fluctuations in available equipment alter the duration of or cancel a job.

A job is divided into segments that mark discrete points at which an effect can be measured and a segment completion time can be predicted. When a segment is completed, the EN.END.OF.-SEGMENT.EFFECT specifies the fractional engineer effect of the work to this point. Between segments or if there is only one segment, the intrasegment effect is used.

Examples:

1) Emplace minefield: Number of segments = 1

Segment	End of Segment Effect/Action		Intrasegment
Effect			
1	1.0	3 (..END)	2 (..LINEAR)

(the effect is directly proportional to time worked; after working 1 hour on a 3-hour job, the effect is 0.33)

2) Emplace minefield: Number of segments = 1

Segment	End of Segment Effect/Action		Intrasegment
Effect			
1	1.0	3 (..END)	4 (..SKEW-RIGHT)

(the effect is computed by a function that makes the effect start slow and increase more rapidly as the end of job approaches; it is used for a system that is slow to get started but functions efficiently once it is going.)

3) Bridge a gap: Number of segments = 4 (reconnaissance [recon], near bank work, launch AVLB, far bank work)

Segment Effect	End of Segment Effect/Action		Intrasegment	
1	0.0	2 (..BREAK.POINT)	1	(..NO.CHANGE)
2	0.0	1 (..CONTINUE)	1	(..NO.CHANGE)
3	1.0	1 (..CONTINUE)	1	(..NO.CHANGE)
4	1.0	3 (..END)	1	(..NO.CHANGE)

(The recon is conducted by one work team. A new work team is formed to actually emplace the bridge after the recon is finished. The effect of the bridge occurs at the end of segment 3; jobs stopped before this point have no effect. The bridge is in effect after segment 3.)

When equipment is attrited, the time to complete the current and future segments will be adjusted to reflect the loss. If equipment is preempted by another job, that equipment leaves at the end of the current segment if it is in use. When equipment is attrited in the middle of a segment, no change in duration will occur if the lost equipment is not used in the segment. If only one type of equipment is lost, completion time for a segment will be adjusted to reflect the capacity of the current number based on the duration and capacity of the original number. If equipment of different types is lost, the previous computation will be done for each type and the maximum duration time used. If an equipment level falls below the minimum amount required by the technique, the job is interrupted at this segment.

Each technique has an associated required equipment list. The required equipment is specified by asset prototype or function and by amount. If a function is specified, the highest capability equipment available will be assigned to the job. A work unit whose equipment requirements are specified by function can use a mixture of asset prototypes, all of which serve the same function. The number of items with that function required for the work will be computed using the relative capability for that equipment. The relative capability can be an actual work rate or a scaled number. In forming a work unit to use this technique, items serving the given function will be added to the unit until the sum of their relative capabilities equals or exceeds the preferred amount. As long as the capability of the asset agrees in units of measure with the required equipment amount, several interpretations can be given. In particular, actual work rates for asset prototypes can be used for the capability, with total work capacity used for the amount of required equipment.

An equipment use indicator signals whether the equipment returns to base at the end of the job or stays at the site for later retrieval (e.g., AVLBs).

The base preparation time is used to determine prejob activity time for the WT unit. If any of the equipment requires base preparation time, the maximum of those amounts is used to determine the departure time.

Engineer Logistics

When logistics is being played, engineers have requirements for two kinds of supplies: equipment and job supplies. When a piece of equipment is transferred to a work team, the fuel and ammunition

required for that equipment is transferred. If the amount on hand is less than the amount needed, a report is sent to the engineer history file and the supply level is artificially raised to handle the deficit. This is not the case for job supplies. Indeed, a job is not processed until the responsible HQ unit has the supplies required by the job's technique. The logistics module links supplies to weapon types, with the type and amounts of supplies delivered to a unit dependent on the weapons that unit owns. For engineer job supplies, no corresponding weapon usually exists. An artificial weapon type may be created to handle this difficulty. The supply needs for this weapon type would be equal to the job supply requirements for the owning HQ unit.

Engineer Job Assignment

The following procedure is used to determine the engineer unit responsible for a job:

1) If the job has an associated "requesting unit," look for a staff engineer subordinate to that unit closest to the job site with the job site in its tactical area. If such a unit is found, go to 2. If no such staff unit exists, then compare the location of the job with the tactical area of each staff unit on the list of staff units, starting with the lowest echelon and moving up.

2) Upon finding a staff unit whose tactical area contains the job site, check the staff unit's list of possible techniques to see that this job could be done with equipment owned by its command chain.

3) If the job cannot be done in this command chain, move to the next staff unit and go to 1.

4) If the current staff unit has a technique by which the job can be done and has the least number of jobs of all such staff units, set the job's technique to this one and find a unit in this staff's command for the job. Moving recursively down the command chain, find the unit closest to the technique's specified level on the command chain that has the job in its tactical area and that has available the most equipment of the type required for the chosen technique.

5) Add the job to this unit's list of pending jobs.

The process of assigning the job and getting it done is as follows:

1) The responsible HQ unit looks for equipment needed for the current phase of the job in this order:

- Belonging to the HQ unit and available
- Belonging to HQ units subordinate to the current one and available
- Belonging to the HQ unit superior to the current one (and to its other subordinates) and available
- Belonging to the HQ unit and working on a lower priority job (considered in priority sequence)
- Belonging to HQ units subordinate to the current one and working on a lower priority job
- Belonging to the HQ unit superior to the current one (and to its subordinates) and working on a lower priority job.

2) If all of the needed equipment can be found and the responsible HQ unit has the required supplies on hand, the equipment is assembled by transferring each such block of equipment found to a WT unit and sending the resulting WT units to a rendezvous point as determined by the technique's organization point. Each asset has a record of who owns it so that it can return to that HQ unit when the phase is finished. Assets being taken from lower priority jobs leave at the end of the current segment if they are at work. The preempted job does not stop; at the beginning of the next segment, equipment levels will be assessed at that point and the job will continue if the minimum amount of each type is available.

3) All of the WT units join into one WT unit at the rendezvous point and, if necessary, proceed to the job site, with appropriate delays for base preparation. The WT unit has a list of jobs that are to be performed before it disbands. This list is constructed at the time the original WT units are being formed. Jobs on the responsible HQ unit's pending jobs list that have the same mission number are added to the WT unit's job list.

4) The job phase currently being processed is done one segment at a time according to the segment information associated with the technique chosen for this job. For a technique with organization point "at SITE," equipment that is no longer needed is assembled at the end of each segment into a new WT unit and sent to the next job on the list or to its owning HQ unit if all jobs on the list are done.

Feature Types

Since engineers alter the surface features of the simulation map, the engineer enhancements cannot be entirely contained in the EN module. In particular, some data structures that keep track of the surface features have been altered.

Line obstacle data structures required an expanded method for breach representation. Line obstacles are divided into individual line segments. Previously, if a segment was breached, its prototype number was negative and the entire segment was considered breached. In the current release, individual breach points are tracked, each of which may have a different breach delay determined by the breach technique. A unit encountering a line obstacle must be within a certain distance (currently the maximum of its radius of deployment and 5 km) of a breach point in order to use the breached delay time for that obstacle. Such a unit will use all of the breach points available within its radius of deployment to reduce its crossing delay.

Previously, defensive positions were associated with path points. The effect of emplacing a defensive position was to lower each occupying unit's exposure attribute which, in turn, lowered its visibility and attrition. No attempt was made to model congestion, with each unit occupying the path point receiving protection from the emplacement. Also no distinction was made in the amount of engineer effort required to dig in different types of units. To simulate defensive positions more realistically, defensive position prototypes are now defined to allow for variations in type, and defensive position data structures exist for tracking individual positions as changes in the terrain rather than in path point attributes. The current release uses the method of former releases to register the effect of defensive positions. The next release will contain enhancements to the representation of defensive position preparation and use.

3 THE ENGINEER MODULE: PRESENT AND FUTURE

Overview

The engineer module as implemented in Reference VIC 3.0 is the module as it exists at the midpoint of the EMIP development plans. Many refinements and enhancements will be implemented during FY90 and will be included in the next release of Reference VIC, version 4.0. The purpose of this chapter is to give VIC programmers and users a summary of what has been implemented and what remains to be done.

Engineer Unit and Asset Representation

The engineer module controls the explicit representation of combat engineers in VIC. That representation in VIC 3.0 differs substantially from earlier versions of the simulation. New structures have been created for modeling engineer units, assets, methods, and task performance. Almost without exception, these aspects of the engineer representation have been implemented as described in Chapter 2.

The internally organized engineer work team is the source of both the module's natural, flexible structure and most of its programming conflicts. This feature is the key to the module's generic mission processing, its natural input data structures, its variable levels of resolution, its balancing of resource allocations with mission requirements, and its multiple techniques for each task type. Within the engineer module itself, the work team structure is ideal. But the engineer work team must be a GG.UNIT if it is to move, fight, suffer attrition, and require supplies in the same way as other ground units do; and that is the source of the programming conflicts.

The EN module's engineer work teams do not follow several of VIC's basic assumptions about GG.UNITs. As GG.UNITs, work teams do not have a continuous role, though they are always present in the unit data base because GG.UNITs are permanent entities. Work teams can appear and disappear quickly, and they have a different role every time they reappear. When they are not active, the work teams have no superior and own no weapon groups. The weapon groups owned by engineer work teams are also reusable and have no type or owner when they are not being used. All of this means that every loop which cycles over all GG.UNITs on a side must be examined to determine if engineer work teams should be included or excluded; every loop that cycles over all FL.WPN.GROUPs must be examined to exclude those which might lead to a fatal crash from a zero entity pointer; and all events that might involve a scheduled action with a work team must be examined to ensure that the work team still exists when the event is to happen. For example, a scheduled artillery attack on a work team must first check to see if the work team is still at its expected location.

The dynamic "creation" of engineer work teams combined with the use of engineer HQs as the engineer resource managers requires extra steps in linking engineers with the front-line attrition, logistics, and return-to-duty modules. For all other types of units, a unit's strength is determined by the FL.WG.STRENGTHs of its weapon groups. The value of each FL.WG.STRENGTH is sufficient to determine attrition rates, supply needs, and return-to-duty demand. With engineers, this is not the case. Engineer HQ units temporarily transfer weapons and supplies to work teams that return the surviving weapons and remaining supplies at the completion of their mission. With such transfers, the value of the weapon group strengths for the HQ and work teams must rise and fall. These shifts are necessary for correct attrition calculations. However, the FL.WG.STRENGTH is not an appropriate measure for supply needs and return-to-duty priorities. A new routine to assess the true state of an engineer HQ's strength determines the unit's supply and repair needs.

In integrating the new engineer module with the rest of VIC, USACERL attempted to locate all processes that might be affected by the new interpretations discussed above. The pressure of using the new module in fully developed scenarios may force unexamined areas to the surface. This possibility was part of the strategy of introducing the new module at this stage, with the development team still readily available for quick fixes.

As experience with the new module grows, some effort will be devoted to refining its existing processes to improve their doctrinal acceptability and/or efficiency. High on the list for improvements are the processes of assigning a mission to an engineer unit and of prioritizing an engineer unit's missions. To improve efficiency, the HQ inventory will be restructured to have a separate list for each type of asset owned. Also, the search for required equipment for a particular mission will be changed. At present, a mission cannot begin unless the preferred amount of equipment is available. This will be changed to allow the mission to start if the minimum amount of equipment is available, though the work team will search for and use as much of the preferred amount as is available.

Engineer Task Capability

The EMIP plan approached the improvement of VIC's engineer representation by focusing on the types of engineer tasks that should be modeled in a simulation at VIC's level of resolution. The following engineer task capabilities are implemented in VIC 3.0:

Emplace Minefield

Emplace Line Obstacle

Prepare Position

Line Obstacle Breach/River Crossing

Minefield Clear

Minefield Breach.

The following engineer task capabilities will be implemented in FY90:

Improve Linear Breach

Remove Linear Breach/Bridge Demolition

Crater Road

Repair Road Crater

Build Trail

Build Forward Air-Landing Facility

Build CSS Facility

Maintain CSS Facility.

Except for minefield clear and breach, all of the engineer task capabilities in version 3.0 existed in earlier versions of VIC. Progress in adding new task capabilities depended on redesigning the representation of engineer units, assets, methods, and task performance. The FY89 effort concentrated on developing this new representation. With it in place, the new task capabilities in the second list above can be added and the existing task capabilities in the first list can be expanded to allow generation of the tasks dynamically, in addition to the present input data method. Because the formation of work teams and the simulation of task performance are done generically, i.e., without regard to the specific type of task, addition of new capabilities and expansion of existing ones depend on identifying trigger mechanisms for generating the tasks dynamically and on building the proper data structures to permit completion of the task to have the proper effect.

Dynamic generation of the existing countermobility and survivability tasks and addition of the two new tasks involving linear obstacle breaches can be accomplished as soon as trigger mechanisms are identified to generate missions of each type. This implementation will be in the form of new engineer decision tables.

Addition of the tasks involving roads depends on the implementation of maneuver unit road use being developed at the U.S. Army Waterways Experiment Station (WES) during FY90. Road damage will degrade the speed at which units move on the affected roads. Correspondingly, engineer effort to repair damage or improve trafficability (BUILD TRAIL) will increase that speed.

The tasks for facility maintenance and repair require construction of data structures to track the functionality of a facility, implementation of methods for degrading functionality and linking it to mission performance, and identification of trigger mechanisms for generating engineer activity to increase functionality. For example, air bases will have runways that can be damaged, and the damage of a runway will affect the number of sorties possible according to the amount of engineer effort expended to repair the damage.

Engineer Planning/Coordination

Modeling of engineer command and control is a major area for attention during FY90. Planning mobility tasks for known obstacles along a unit's path will be introduced. In version 3.0, maneuver units do not generate breaching missions until the obstacle is encountered. For major tasks, such as a river crossing, which would normally be known well in advance, a new planning path point for the unit will trigger engineer activity before the actual encounter. In addition, new engineer decision tables will allow HQ units to coordinate effort and the use of resources, and the data structures already in place for an engineer command chain will be used to dynamically shift assets to the locations where they are needed most.

Expansion of Implicit Representation

The implicit representation of engineers grew from an attempt to control the "magic carpet ride" often granted maneuver units. Since the engineer input data simply identify what types of engineer assets (weapons) are needed for each type of task, a true assessment of a maneuver unit's "engineer" capabilities can be made by comparing the unit's assets to the required equipment list. In VIC 3.0, this implicit representation is used for mobility tasks only. When a unit encounters a linear obstacle or minefield, an assessment of its own capability of dealing with the obstacle is made. A unit owning bridging assets will make its own way across a river in the time its takes to construct the bridge if that time is less than the unbreached delay for the river. Similarly, a unit owning minefield breaching equipment will reduce its delay and attrition according to the technique chosen.

During FY90, the implicit representation will be expanded to include countermobility and survivability missions. Maneuver units with the appropriate equipment and supplies will be able to emplace their own minefields and linear obstacles. Expansion of the representation of defensive position preparation will include a more detailed assessment of a unit's capability to prepare its own positions.

Representation of Defensive Position Preparation and Use

Earlier versions of VIC did not have structures that represented defensive positions. Maneuver unit and engineer effort to prepare positions were registered as a change in a path point attribute. This method did not take into account the different levels of engineer activity required to dig in different types and sizes of units. This method also did not account for congestion or multiple use of one site. VIC 3.0 has a new defensive position prototype structure and a new defensive position structure that will be used during FY90 to correct these problems. At present, position preparation is still registered by changing a unit path point attribute for exposure. This will change for the next release to represent a defensive position as a terrain feature.

Air and Artillery Mine Delivery

The VIC 3.0 implementation of mine delivery by artillery and air units is as it was in earlier versions of VIC. During FY90, this area will be completed. In addition, the dynamically generated countermobility plan of engineers will include emplacement of minefields by nonengineers and will track mine usage by all units.

Engineers and the Smoke Module

Engineers are not linked with the Smoke module in VIC 3.0. At the same time that the new engineer module was being developed, major changes were proposed for the Smoke module. During the coming year, an attempt will be made to relink the two modules.

Output Reports and the Postprocessor

The huge expansion in detail of the engineer module allows a considerable amount of output data of all kinds. The engineer output file is a chronological report sequence that must be processed by a postprocessor called ENPOST to produce usable reports. This postprocessor is in its infancy, but will grow with the addition of many improvements during FY90. See Chapter 29 of the *VIC Input and Methodology Manual*[1] for a key to reading the current engineer output file.

---

[1] *Vector-in-Commander (VIC) Documentation, Data Input and Methodology Manual*, TRAC-WSMR-TD-Draft (Department of the Army, Training and Doctrine Command [TRADOC] Analysis Command [TRAC], June 1987, revised 1989).

4 INPUT DATA DESCRIPTIONS

Coordination of the input data for several modules of VIC is necessary to take full advantage of the capabilities offered by the new engineer module. The EN data focuses primarily on task performance specifications, stating how engineer tasks are done and what equipment and supplies are required to do them. The engineer units, their organizational structure, equipment, supplies, and general characteristics are specified in the GG, GM, FL, and LO data modules. The terrain features--minefields, line obstacles, defensive positions--that are emplaced, removed, and breached by engineers are described by both type and occurrence in the MF, TB, GG, and EN modules.

As in earlier versions of VIC, the new engineer module has two types of engineer units: the HQ unit and the WT. In the current version, the engineer HQ units are very flexible entities. As ground units, they have no restrictions placed on them beyond those of any other ground unit. In addition, engineer HQ unit prototypes can be at any level except platoon, which is reserved for the work team prototype. The prototype specifications for GG units must contain at least one prototype with type engineer (ENG) and level platoon (PLT) to describe the engineer work team. The first such prototype listed in the ground prototype data of GG is assumed to be the BLUE engineer work team prototype. The second such prototype, if it exists, is assumed to be the RED engineer work team prototype. If the prototype data contains only one prototype of type engineer and level platoon, then this is assumed to be the RED engineer work team prototype as well. The unit data base of the GG data cannot contain any units with the engineer work team prototype as their prototype specification.

The new engineer module simplifies the engineer unit structure in VIC by removing the work teams from the GG unit data. Unlike earlier versions of VIC in which all of the work teams and their attributes were specified in the input data, the new engineer module "creates" the work teams internally as they are needed and task-organizes them for a specific set of jobs. Only the engineer HQ units are identified in the GG data. Consequently, HQ units are the only engineer units appearing in the unit data sections of GM, FL, and LO. The FL unit data specifies the types and quantities of equipment owned by the HQ units and the LO unit data specifies how the HQ units will be resupplied. The HQ units hold the only resources (equipment and supplies) available to engineers for the explicit simulation of task performance. They use their equipment and supplies for task performance by transferring what is needed to a newly created work team and sending the work team to the site. The EN data determines what is needed and how long the job will take.

To be able to "create" work teams, which must be GG.UNITs if they are to be visible to the rest of VIC, the model must create two pools of extra GG.UNITs, one for each side. The free GG.UNITs have their attributes set to make them invisible when they are not in use. "Creating" a work team simply means setting the attributes of an unused GG.UNIT so that it is visible and sending it on its way. After the work team has served its purpose, it returns to its invisible state to await being used again. This process allows the new version of VIC to accomplish more for engineers without increasing the number of GG.UNITs dramatically. Two numbers specifying how many extra BLUE and RED units to create are new data items in the GG module, immediately following the number of BLUE and RED units in the data base. Determining the number of extra units to create is not easy.

The number of extra GG.UNITs needed to simulate engineer task performance depends on many scenario variables: the total number of ground units; the level of detail of the engineer asset data; the complexity of the engineer technique data; the number of preplanned obstacles, minefields, and defensive positions; the number of natural obstacles in the terrain, etc. The engineer module cannot function if it depletes either of the sets of extra units. On the other hand, the run time will be affected if too many extra units are created. To aid in optimizing these two numbers, the minimum number of extra

BLUE and RED units available during a run is written as the last line of the engineer history file, **ENGR.LIS. The numbers in the GG data can be reduced by these values for subsequent runs of the same scenario.

The documentation in the Methodology and Input Data Manual describes the engineer-related data in the GG, GM, FL, LO, MF, and TB data modules. The input data required for the EN data module is described below. It is divided into nine segments:

Segment EN-ONE	Mission Update Interval
Segment EN-TWO	Defensive Position Data
Segment EN-THREE	Engineer Function Data
Segment EN-FOUR	Engineer Asset Prototype Data
Segment EN-FIVE	Engineer Command Links
Segment EN-SIX	Engineer Job Techniques
Segment EN-SEVEN	Required Equipment for Engineer Techniques
Segment EN-EIGHT	Required Supplies for Engineer Techniques
Segment EN-NINE	Translation of Coordinates

Mission Update Interval

This set of three integers (D H M) represents the maximum length in days, hours, and minutes of the interval for the reprocessing of the engineer mission list. Missions are processed initially as soon as they are generated. If a mission is assigned to a unit, it is removed from the mission list. The unassignable missions remain on the list for reprocessing, which occurs if an engineer unit moves to a new tactical area or when this interval of time has passed since the last update.

Defensive Position Data

NUMBER OF DEFENSIVE POSITION PROTOTYPES

This is the number of defensive position types to be defined in this data. This number of defensive position prototypes will be described below.

For each defensive position prototype, specify:

PROTOTYPE NAME (Character)

This character is the name for the prototype, to be used in output reports.

UNIT TYPE (Character)

This is the type of unit this defensive position is designed to protect. It may be any of the GG.UNIT.PROTOTYPE types of the GG data:

"TNK" = TANK UNIT

"MEC" = MECHANIZED UNIT

"INF" = INFANTRY UNIT

"CAV" = CAVALRY UNIT

"AH" = ATTACK HELICOPTER UNIT

"UH" = UTILITY HELICOPTER UNIT

"CH" = CARGO HELICOPTER UNIT

"HQV" = AVIATION HQ UNIT

"HQA" = ARTILLERY HQ UNIT

"TUB" = TUBE ARTILLERY UNIT

"RKT" = ROCKET ARTILLERY UNIT

"SSM" = MISSILE ARTILLERY UNIT

"ADA" = AIR DEFENSE UNIT

"HQ" = HEADQUARTERS UNIT

"SUP" = SUPPLY CONVOY UNIT

"FSA" = FORWARD (FWD) SUPPLY AREA

"RDU" = REPAIR UNIT

"ABN" = AIRBORNE UNIT

"ENG" = ENGINEER UNIT

"INT" = INTELLIGENCE UNIT

"JAM" = JAMMING UNIT

"EH" = ELECTRONIC HELICOPTER UNIT

UNIT LEVEL (Character)

This is the level of unit that this defensive position is designed to protect. It may be any of the GG.UNIT.PROTOTYPE levels of the GG data:

'FRT' = FRONT

'ARM' = ARMY

'COR' = CORPS

'DIV' = DIVISION

'RGT' = REGIMENT

'BDE' = BRIGADE

'BN' = BATTALION

'TF' = TASK FORCE

'CO' = COMPANY

'SQN' = SQUADRON

'TRP' = TROOP

'BTY' = BATTERY

'PLT' = PLATOON

UNIT EXPOSURE (Real Number Between 0.0 and 1.0)

This number is the fraction of unit exposed when a unit of the given type and level occupies a fully constructed position of this type.

Engineer Asset Function Data

NUMBER OF ASSET FUNCTIONS (Integer)

Each engineer asset can be classified and used by its function. Examples of functions are: hauling, digging, trenching, bridging. This data item specifies the total number of these different functions to be modeled in this scenario. The scenario developer then makes a list of the functions to be modeled. The order of the list determines the number to be associated with each function.

ASSET FUNCTION NAME (Character)

This character is the text name for each of the functions to be modeled in this scenario. The order in which the names are written will determine the number by which they are referenced in the asset prototype data.

Engineer Asset Prototype Data

NUMBER OF ENGINEER ASSETS (Integer)

This integer is the number of different types of engineer assets being modeled. Asset prototypes are first specified in the FL input data as weapon prototypes. This is the number of these weapon prototypes to be identified as engineer asset prototypes below.

Given the current structure of engineer HQ inventories and the list processing methods of Simscript, run time will be optimal if this list of asset prototypes is in ascending order of asset availability. That is, identify the scarcest resource first.

For each engineer asset prototype, specify the following attributes:

ASSET NAME (Character)

This character is the same name by which the corresponding weapon prototype was identified in the FL data. Each name here must match one of the weapon prototypes. This data item establishes the link between the engineer asset prototype's weapon data and the engineer asset prototype's engineer data.

ASSET SPEED (Real Number in km/hr)

This is the basic movement rate of this asset prototype across open country. The speed at which an engineer work team moves will be the minimum value of this speed for each of the assets owned by the work team.

ASSET DAMAGE THRESHOLD (Real Number Between 0.0 and 1.0)

This is the minimum portion of an asset of this type that must be functional for the asset to continue working. An asset of this type that has suffered attrition continues to work at a degraded level until its undamaged portion falls below this number. At that point, the asset is declared ..DAMAGED and is no longer available for assignment.

ASSET MAXIMUM WORK PERIOD (Real Number in Hours)

This number specifies the maximum amount of time that this asset can work continuously without a rest period.

ASSET REST PERIOD (Real Number in Hours)

This number specifies the amount of rest required after a maximum work period.

ASSET REPORT FLAG ("Y" or "N")

This flag indicates whether a detailed asset report is to be generated during the run for assets of this type. If "Y" is chosen, the model will print an asset report line to the engineer history file every time the status of an asset of this type changes. If "N" is chosen, no asset reports will be generated for assets of this type. If engineer assets are being modeled in detail with many engineer HQ units owning many assets of this type, the run time will be severely affected when the asset report flag is set to "Y".

NUMBER OF ASSET FUNCTIONS (Integer)

This specifies the number of functions this asset serves.

Specify the following attributes for each of the functions of this asset (number specified above):

ASSET FUNCTION (Integer)

This integer identifies the function number of a function to be associated with this asset. The function number is determined by the order in which the function names are written in the function data above.

ASSET RELATIVE FUNCTION CAPABILITY (Real Number)

Different engineer assets with the same function do not perform at the same level. This item allows the scenario developer to specify a relative work capability. When a technique's equipment requirement is given in terms of the function, a search for equipment to use will seek available equipment in the order of relative capability for serving the given function. A work unit whose equipment requirements are specified by function can use a mixture of asset prototypes, all of which serve the same function. The relative capability can be an actual work rate or a scaled number.

Engineer Command Links

The current version of the engineer module does not use the command links defined here. In the next release, decision tables will be implemented to improve the representation of engineer command and control, and this data structure will be a key element. The current release contains the code only for reading and storing this data.

NUMBER OF COMMAND LINKS BEING IDENTIFIED (Integer)

Each engineer staff unit in the GG data base of ground units can have an engineer superior in addition to the tactical superior identified in the GM data. An engineer unit that has an engineer unit as its tactical superior will, by default, have that engineer as its engineer superior. But a staff engineer, which has a nonengineer tactical superior, may have any engineer unit as its engineer superior. This engineer command link is used only for the reallocation of resources and supplies. No command links have to be defined here; the scenario builder is free to construct the command links as needed. This data item simply states the number of these links to be made below. As stated above, engineer command links are not used in the current version of the engineer module. For now, this data item should be set to "0."

ENGINEER SUPERIOR (a GG.UNIT Name)

This name is the engineer unit that is to be the engineer superior of the unit below.

ENGINEER SUBORDINATE (a GG.UNIT Name)

This name is the engineer staff unit having its engineer superior determined. Note that this unit must be an engineer unit and must have a nonengineer GM superior.

Engineer Job Techniques

This section of data allows the scenario builder to identify the different methods that engineers can use to accomplish a specific task, as determined by task type and feature type. The techniques may differ in several ways:

1) The effect of the end result

2) The equipment or supplies required to accomplish the work

3) The time required to accomplish the work

4) The work pattern established for the efficient use of the equipment

5) The suitability of the technique for the particular situation.

For example, if the task is "breach a linear obstacle" and the feature type is "50M River," then two different bridging techniques:

1) May produce bridges that have different capacities

2) May use different engineer assets in the work stages

3) May require different amounts of time to complete

4) May differ only in the way in which the work is organized

5) May identify one technique for assault locations and another for rear area locations.

NUMBER OF TECHNIQUES (Integer)

This integer is the total number of techniques to be specified in the data below for all of the task types and feature types. The engineer task types are identified below. The feature types are specified in other VIC data modules: minefield prototypes in MF, linear obstacle prototypes in TB, defensive position prototypes in EN, etc.

Techniques must be entered in order of most preferred to least preferred technique for each combination of side, task, and feature. For each technique:

TECHNIQUE NUMBER (Integer)

This integer is simply a sequential numbering of the techniques to give a reference number to the technique for the scenario builder and for the simulation.

TECHNIQUE SIDE ("B" or "R")

This letter specifies which side (BLUE or RED) uses this technique.

ENGINEER TASK NUMBER (Integer)

This integer specifies the task type of the given technique. The engineer tasks will be associated with the following numbers:

Mobility

..MINEFIELD.BREACH	= 1
..MINEFIELD.CLEAR	= 2
..LINE.OBSTACLE.BREACH	= 3
..RIVER.CROSSING	= 3
..IMPROVE.LINE.BREACH	= 4
..REPAIR.ROAD.CRATER	= 5
..BUILD.TRAIL	= 6
..BUILD.FALF	= 7

Countermobility:

..EMPLACE.MINEFIELD	= 8
..EMPLACE.LINE.OBSTACLE	= 9
..REMOVE.LINEAR.BREACH	= 10
..BRIDGE.DEMOLITION	= 10
..CRATER.ROAD	= 11

Survivability

..PREPARE.POSITION	= 12

Sustainment:

..BUILD.CSS.FACILITY	= 13
..MAINTAIN.CSS.FACILITY	= 14

FEATURE PROTOTYPE NAME (Character)

The feature prototype name corresponds to the name associated with this feature in the data module where the prototype description is given. For minefields, this is the mine name from the mine type input data in the MF module. For linear obstacles, this is the obstacle name from the line obstacle input data section of the TB module. For defensive positions, it is the name of the defensive position prototype in the EN data.

NORMAL COMMAND LEVEL (Character)

This specifies the echelon at which this type of technique is normally employed to do the task. In the search for a responsible unit for a job using this technique, an attempt will be made to assign the job to a unit at this level. The following text identifiers are used to designate the command level:

'FRT' = FRONT

'ARM' = ARMY

'COR' = CORPS

'DIV' = DIVISION

'RGT' = REGIMENT

'BDE' = BRIGADE

'BN' = BATTALION

'TF' = TASK FORCE

'CO' = COMPANY

'SQN' = SQUADRON

'TRP' = TROOP

'BTY' = BATTERY

Engineer work teams are assigned the level 'PLT' (PLATOON). No engineer units of that level can be in the GG unit data base, and "PLT" cannot be used as a level for this data entry.

MINIMUM DISTANCE FROM FEBA (Real Number in km)

This number specifies the minimum distance from the FEBA at which this technique will be used. It allows the scenario developer to distinguish techniques for assault situations from techniques used only in the rear. This number represents the signed distance from the FEBA for a side, with negative numbers indicating the distance from the FEBA on the enemy's side.

RELATIVE EFFECT (Real Number)

At present, this number is used primarily for breaching of linear obstacles. Each linear obstacle prototype has a standard breach point delay associated with it in the line obstacle prototype data. To distinguish between different breach technique results, this data item allows the delay time to be tailored to this type of breach by dividing the standard breach delay by the relative effect. The scenario developer must correlate the data in the TB module with the engineer data. This data item should be "1" for all other task types/feature types.

ORGANIZATION POINT

This item can have one of two values: 0 (at HQ) or 1 (at the SITE). A "0" indicates that when this technique is used, all equipment required for the current phase will rendezvous at the

responsible engineer headquarters and move to the work site as one unit. A "1" indicates that when this technique is used, equipment being used for the job but not currently located at the headquarters unit will move directly to the work site and form a single work unit there, with work beginning on the current segment as soon as the equipment required for that segment is at the site. Work units move to succeeding job sites and return to their bases in the same way as they traveled to the work site.

NUMBER OF SEGMENTS (Integer)

Each technique can divide the work into segments and phases. A segment is a distinct subsection of the total work effort, perhaps requiring only a subset of the equipment at the site and marking a point at which work might resume if it has been interrupted. A phase is a sequence of segments that are all to be done by one work unit in a continuous effort. This data item specifies the number of segments that the scenario builder chooses for this particular technique. The phases are determined by the end-of-segment action below, with a phase containing all segments from the starting segment until encountering a segment whose end action is "break point" or "end."

For each segment of the technique:

END-OF-SEGMENT EFFECT (Real Number)

This number is the fractional part of the work's effect at this point in the effort. If the work were interrupted at this point, the effect of the partial work effort could be registered. This number is between 0.0 (no effect) and 1.0 (total effect complete). Normally, each successive end-of-segment effect will be larger than its predecessor, but that need not always be the case.

INTRA-SEGMENT EFFECT (1, 2, 3, or 4)

This number is a pointer to one of four functions used to determine the effect of work interrupted in the middle of the segment.

1 = NO.CHANGE (use the effect at start of the phase)

2 = LINEAR (effect directly proportional to time worked)

3 = SKEW.LEFT (largest effect done early)

4 = SKEW.RIGHT (largest effect at end of job).

SEGMENT DURATION (Real Number in Hours)

This number is the length of time needed for the full complement of required equipment to complete the segment. If equipment has been lost or damaged, or conditions are worse than normal, this duration is increased. This attribute is used for those segments whose completion time is independent of the size of the job. For each segment, if the duration is specified, then the rate must be zero. If the rate is nonzero, then the duration must be zero.

SEGMENT RATE (Real Number; Units of Measure Vary With Task Type)

For segments whose duration depends on the size of the job, the duration is calculated by dividing the job size by the rate. For minefield emplacement, the job size is the number of mines and the rate is the number of mines per hour. For linear obstacle emplacement and building combat trails, the job size is the length in kilometers and the rate is in kilometers per hour. For minefield breaching,

the size is in kilometers of depth of the minefield and the rate is in kilometers per hour. All other task types are excluded from specifying a segment rate.

VISIBILITY FACTOR, WEATHER FACTOR, COMBAT FACTOR (Real Numbers)

These three data items account for the degradation of work rates because of nightfall, weather, and combat. The segment duration applies to work during daylight, good weather, and no combat. For nighttime, for each drop in the level of weather, and for combat situations, these factors multiply the duration in a compound effect.

END-OF-SEGMENT ACTION (1, 2, or 3)

This item can have one of three values:

1 = CONTINUE, i.e., go on to the next segment

2 = BREAK POINT, i.e., register the effect, put the remainder of the job back on the pending job list, and go back to base

3 = END, i.e., the job is complete at the end of this segment.

A job "phase" consists of those segments that have an end-of-segment action of "continue" except for the last segment, which has an end-of-segment action of "break point" or "end." Each new job phase is done by a different work unit with the equipment used only in that phase.

Mark the end of the engineer technique section of the data with "*".

Equipment Required for Engineer Techniques

For each of the techniques identified above, this section lists the engineer assets required for that technique.

TECHNIQUE NUMBER (Integer)

This integer is the link between this data set and the techniques above.

ASSET NAME (Character)

This is the name of the asset prototype of this required equipment. This name must correspond to an asset name in the engineer asset prototype data above. If the scenario builder wishes to specify the equipment by function rather than type, then this entry is "-".

FUNCTION NAME (Character)

Equipment needed for a technique can be specified by asset name or by function name (not both). If this technique can use any equipment that has a certain function, then the function name is entered here. This name will link with the function names entered in the asset function data above. If an asset name has been entered for this line of data, then this item is "-".

PREFERRED AMOUNT (Integer)

This specifies the number of items of this type normally used for this technique. If equipment has been specified by function rather than type, then this number represents the number of items with capability 1. In actually forming a work unit to use this technique, items serving the given function will be added to the unit until the sum of their relative capabilities equals or exceeds this preferred amount. For example, if the technique requires 6 items serving function 1, then the work unit can be composed of:

6 items with capability 1 or

1 item with capability 6 or

3 items with capability 2 or

12 items with capability 0.5 or

2 items with capability 2 and 2 items with capability 1

3 items with capability 1 and 6 items with capability 0.5.

As long as the capability of the asset agrees in units of measure with this attribute, many interpretations can be given. In particular, actual work rates for asset prototypes can be used for the capability, with total work capacity used for the amount.

MINIMUM AMOUNT (Integer)

This integer specifies the minimum number of this item with which the technique can still be used. When work teams are formed, they are equipped with the preferred amount of an item. If equipment is lost or preempted along the way, this number is the break-off point. An equipment level below this amount would cause the unit to cancel its job.

JOB USE (1 or 2)

This item has one of two values:

1 = PERMANENT, i.e., this equipment returns from the work site

2 = RETRIEVABLE, i.e., this equipment stays at the work site but can possibly be retrieved later.

BASE PREPARATION TIME (Real Number in Minutes)

If this equipment requires preparation before it can depart from its base area, this value specifies the amount of time required.

SEGMENTS USED (1, 2,. ., Number of Segments for Technique)

This is a one-line list of segment numbers (separated by spaces) of the segments in which this equipment is used. It determines the composition of work units working on a job phase as well as the release point for this specific equipment. When the current job segment number is greater than the last segment used for this equipment and the technique's point of organization is at the site, the

equipment can be released from the job to be moved to another job or returned to its headquarters. Mark the end of the list of segments with a "*".

Mark the end of the list of required equipment with a "*".

Supplies Required for Engineer Techniques

For each of the techniques identified above, this section lists the engineer supplies required during the technique. These do not include petroleum-oil-lubricant (POL) and ammunition normally carried by units with the equipment specified above, but supplies such as mines or line charges that are used specifically for the task.

TECHNIQUE NUMBER (Integer)

This data item serves as the link between this data set and the techniques above.

SUPPLY NAME (Character)

This is the name of the supply type needed. This name must be identical to a corresponding supply name in the logistics prototype data of the LO module.

REQUIRED AMOUNT (Real)

This item specifies the amount of this type normally used for this technique.

UNIT OF MEASURE FOR SUPPLY (Real)

This item is used for jobs that have an associated size, allowing the amount of required supplies to be determined by that size. Computation of the total supply requirement will follow this formula:

$$\text{Total required supplies} = \frac{\text{Job size} \times \text{Required amount}}{\text{Unit of measure for supply}} \qquad \text{[Eq 1]}$$

For minefield emplacement, both the required amount and unit of measure should be 1 since the job size is the number of mines to emplace. For mine breaching, using a 150-m line charge, the required amount should be 1 and the unit of measure should be 0.15.

FIRST SEGMENT NEEDED (Integer)

This is the segment number of the technique in which this supply must be at the work site. Earlier segments completed by other work units may not have required this type of supply. After this segment, the supply is expended. If work is interrupted before this segment, the supplies will be returned.

Mark the end of this list with "*".

Translation of Coordinates

X,Y TRANSLATION (Kilometers--Two Real Numbers)

An (x,y) translation from the minefield coordinate system (see the MF module) is needed to bring it in line with the battlefield coordinates, in general (0.0,0.0). Battlefield coordinates = (MF x-coordinate + x, MF y-coordinate + y).

Mark the end of the engineer data with "END".

ENGINEER INPUT DATA FORMAT

```
$$$$ EN
$$$  ENGINEER MODULE DATA

''Update Interval

# # #

''Number of Defensive Positions

#

''                                Unit    Unit    Exposure
'' Position Name                  Type    Level    Level
'' ------------------------------ ------  -------- ----------


''Number of Asset Functions

#

''Names of Asset Functions


''Number of Engineer Assets

#

''
'' Asset  Name                   Speed   Damage   Max Work   Rest     Asset    No. of      Function    Relative
''                                       Thres.    Period    Period   Report   Functions     No.      Capability
''------------------------------ ------- -------- ---------- -------- -------- ----------  ---------- ----------
```

''Number of Command Links Being Identified

#

'' Superior Subordinate
'' ---------- ----------

''Number of Techniques

#

''
'' Tech Task Nor Min. Rel Org No. End Intra- Visib. Weather Combat EOS
2''No. Side No. Feature Name Ech Dist to FEBA Eff Pt. Seg Eff Segmnt Effect Duration Rate Factor Factor Factor Act
'' ------ ----- ------ ---------- ----- -------- ---- ---- ----- ----- --------- ---------- ------ -------- ---------- ---------- ----
-

''Technique Required Equipment

'' Tech Function Pref. Min. Job Prep
'' No. Asset Name No. Amount Amount Use Time Segment Used
'' ------ ---------- ---------- ---------- ---------- ----- ------ ----------
*
* *

''Techniques Required Supplies

'' Tech Req. Unit Segment
'' No. Supply Name Amount Meas Used
'' ------ ---------- ---------- ------ ----------
*

# #

END

5 ENGINEER DATA STRUCTURES

The data structures of the new engineer module are described below. Included are descriptions of each entity and its attributes and the sets that establish relationships between entities. The order of presentation follows the logical flow of the relationships rather than the formal structure of the Preamble. The units of measure associated with each attribute are the units used internally. Attributes whose values are determined by input data may have different units of measure than the input data required. For example, all times are converted to seconds internally, but input data may have specified a preparation time in minutes and a rest time in hours.

System Entities

THE SYSTEM

HAS A EN.UPDATE.INTERVAL. This input data item specifies the maximum length of time allowed between successive passes through the mission lists in an attempt to assign the missions. Each side maintains a list of its unassigned engineer missions. When the list is processed, all missions that can be assigned are converted to engineer jobs and are removed from the list and destroyed. If the processing interval is short and the mission lists contain a large number of missions that cannot be assigned because of current conditions, the run time may be affected. In contrast, the larger this interval is, the less reponsive engineers will be to requests for support.

HAS A EN.MISSION.COUNT. This number is used to attach a unique identifier to engineer missions as they are assigned to an engineer HQ unit, linking all of the jobs arising from the mission by a common number. As a mission is processed, each job in the mission has the current value of this number as its mission number. After a mission has been numbered and assigned, EN.MISSION.COUNT is incremented by 1.

HAS A EN.TECHNIQUE.ARRAY. This ragged integer array, dimensioned by side by task type by obstacle feature prototype by the number of corresponding techniques available, provides an index to the engineer techniques. It is built immediately after the technique data is read.

HAS A EN.ASSET.REPORT.ARRAY. This integer array, dimensioned by the number of engineer asset prototypes, keeps track of the input data regarding the production of asset reports for each engineer asset prototype. Each engineer asset prototype has an input item that allows the user to stop the printing of an asset report when the status of an asset of this type changes. If a large number of engineer assets are created, as when individual soldiers are identified as engineer assets, printing a line of output each time the asset's status changes may affect run time severely.

OWNS A EN.SET.OF.FREE.WPN.GROUPS. This set contains the FL.WPN.GROUPs not currently owned by a GG.UNIT. Since FL.WPN.GROUPs are permanent entities, extra weapon groups must be created at initialization time for use when work teams are created and must have weapon groups to hold their required equipment.

EVERY SS.SIDE

HAS A EN.MAX.FREE.WORK.TEAMS. This variable keeps track of the smallest number of units left in each side's pool of free work teams during a run. It is printed in the engineer history file at the end of a run. Each side has a pool of unused units from which to form engineer work teams as they are required. Determining the correct initial size of these pools is a difficult task. With the value of this

variable known for each side, successive runs of the same scenario may decrease the number of free units by this amount so as to optimize the number of extra units created and thereby decrease run time.

OWNS A EN.SET.OF.FREE.UNITS. This set keeps track of free work units. An engineer work team is either in this set or in the performing units set of the HQ unit responsible for the jobs assigned to the work team.

OWNS A EN.SET.OF.HQ.UNITS. This set keeps track of the engineer HQ units on a side.

OWNS A EN.SET.OF.STAFF.UNITS. This set keeps track of the engineer staff units on a side.

OWNS A EN.SET.OF.MISSIONS. This set holds the information about unassigned engineer activity. It grows and shrinks as engineer activities are generated and assigned. This set is processed at regularly scheduled intervals (see EN. UPDATE.INTERVAL). Each mission that can be assigned is removed from the set and corresponding jobs are created and filed in the responsible HQ's set of pending jobs. Missions that cannot be assigned remain in this set awaiting a change in conditions.

OWNS A EN.SET.OF.PROPOSED.MINEFIELDS. This set contains all of the MF.MINEFIELDs that are not active and that are to be emplaced by engineers.

OWNS A EN.SET.OF.PROPOSED.OBSTACLES. This set contains all of the TB.OBSTACLE.-SECTIONs that are not active and that are to be emplaced by engineers.

OWNS A EN.SET.OF.DEFENSIVE.POSITION. This set contains all of the defensive positions currently in place and those to be emplaced by engineers, with the EN.DP.EFFECTIVENESS ranging from ..UNPREPARED (0.0) for future positions to ..FULLY.PREPARED (1.0) for active positions.

OWNS A EN.SET.OF.RETRIEVABLE.ASSETS. This set keeps track of retrievable assets. EN.RE- QUIRED.EQUIPMENT with EN.EQ.USE set to ..RETRIEVABLE does not return from the job site but may be retrieved later for use on another job. This set provides the data necessary for that retrieval.

Engineer Task Structures

EVERY EN.DEF.POS.PROTO

This permanent entity is a new terrain feature prototype for VIC. It allows engineers to emplace defensive positions at a particular location and with a degree of protection commensurate with the effort expended. In its initial implementation, this structure replicates the representation of defensive positions in reference VIC. This representation will be enhanced in the second half of the EMIP development program.

HAS A EN.DP.PR.NAME. This item is the defensive position prototype name to be used for input and output.

HAS A EN.DP.PR.UNIT.TYPE. This item is the type of unit for which this position is prepared. Its possible values are in the set of defined-to-means for GG.TYPE.

HAS A EN.DP.PR.UNIT.LEVEL. This structure is the level of unit for which this position is prepared. Its possible values are in the set of defined-to-means for GG.LEVEL.

HAS A EN.DP.PR.UNIT.EXPOSURE. This item is the fraction of the unit exposed when the correct type and level of unit is occupying the position.

EVERY EN.DEFENSIVE.POSITION

This structure is an actual occurrence of a defensive position on the battlefield. It serves the same purpose for defensive positions as MF.MINEFIELD for minefields and TB.LINE.FEATURE.-SEGMENT for linear obstacles.

HAS A EN.DP.PROTO. This is the prototype number of the EN.DEF.POS.PROTO for this position.

HAS A EN.DP.X.LOCATION. This is the x location of the position.

HAS A EN.DP.Y.LOCATION. This is the y location of the position.

HAS A EN.DP.EFFECTIVENESS. This is the percentage effectiveness of the position. This number combines with the unit exposure of this prototype to determine the amount of protection afforded by this position. The effectiveness is governed by the level of completion of the position while the prototype exposure applies to a completely finished position of that prototype.

BELONGS TO A EN.SET.OF.DEFENSIVE.POSITIONS. This is the set of defensive positions that can be used by units as they move across the battlefield. For now, each unit marks its path points with the level of defensive preparation as in reference VIC 2.0. During the second half of EMIP, this representation will be changed so that defensive positions become part of the terrain and each unit must search for an unoccupied position at its defensive path points.

EVERY EN.MISSION

This is a structure to hold the data for a desired engineer activity until an engineer HQ unit can be found to do the work. Once the task is assigned to an HQ unit, the EN.JOB structure is created and the mission is removed from its side's set of missions and destroyed.

HAS A EN.MISSION.TASK. This item is the engineer task type associated with the mission, using defined-to-means.

HAS A EN.MISSION.ORIG.NBR. The original mission number assigned in the input data is recorded here to keep track of missions specified in the GG, MF, and TB input data. Using the task type and original mission number together will allow the user to track specific missions generated by input data. This number is not related to the actual mission number associated with the jobs arising from it. See EN.MISSION.COUNT.

HAS A EN.MISSION.METHOD. This structure is a defined-to-mean that specifies how to search for a technique to accomplish this mission. In choosing a technique, the search can be directed to follow one of three paths: ..PREFERRED.METHOD (the default), which takes the first usable technique listed in the input data; ..SHORTEST.TIME, which takes the technique that requires the shortest amount of time to complete; ..BEST.EFFECT, which takes the technique that yields the best end result. The situation that generated the mission will determine the method.

HAS A EN.MISSION.X.LOC. This is the x location of the mission. For some of the tasks, this information is redundant whereas for others, it is necessary. For example, while emplacing a set of minefields, this location will be ignored in favor of using the minefield centers as determined by the

EN.MISSION.FEATURE below. For breaching of linear obstacles, however, this location will identify the actual breach location, whereas the feature information specifies only the endpoints of the segment.

HAS A EN.MISSION.Y.LOC. This is the y location of the mission. See above.

HAS A EN.MISSION.REQUESTOR. This is the GG.UNIT for which this mission was generated. For minefield and obstacle emplacement missions created by input data, no requestor will be identified. For dynamically generated missions, however, the work will be done for a specific unit. That unit's GG.UNIT number is stored here and is used in searching for engineer support. An attempt will be made to assign an engineer mission to an engineer unit subordinate to the requestor.

HAS A EN.MISSION.CREATION.TIME. This is the time at which the mission was generated.

HAS A EN.MISSION.REQUIRED.COMPLETION.TIME. This is the latest acceptable completion time for the mission. In the reference VIC, input data specified a time at which to activate proposed obstacles. The new engineer implementation changes this interpretation. This time is now used as a bound for when the work must be done. For dynamically generated jobs, an analysis of the situation that generates the job will produce a time at which the work must be completed. This is an important decision variable, with a job being canceled if its estimated completion time exceeds this value.

OWNS A EN.SET.OF.MISSION.FEATURES. This is the set of features associated with the mission. One mission will generate several engineer jobs--one for each terrain feature being worked on by the work team created for this mission. Each job corresponds to a particular feature. For example, if the task is to emplace minefields, this set is the set of pointers to the proposed minefields that are to be constructed by one work team.

BELONGS TO A EN.SET.OF.MISSIONS. This set contains the data for all engineer activity that has been generated but not yet assigned to specific engineer HQ units. Each side owns a set of engineer missions.

EVERY EN.MISSION.FEATURE

This entity points to the terrain features being changed by the accomplishment of this mission.

HAS A EN.MISSION.FEATURE.PTR. This is the pointer to the terrain feature (the proposed MF.MINEFIELD, the proposed TB.LINE.FEATURE.SEGMENT, the TB.LINE.FEATURE.SEGMENT to be breached, the EN.DEFENSIVE.POSITION to be constructed, etc.).

BELONGS TO A EN.SET.OF.MISSION.FEATURES. This is the set of all features associated with the mission. This set provides the mechanism for linking similar jobs to be carried out by one work team.

Engineer Units

EVERY EN.HQ.UNIT

This entity is the basic engineer unit in the simulation. These units appear in the unit data base of the input data; they own the engineer assets and are responsible for engineer jobs. To do a job, the HQ unit creates a work team and transfers to it the equipment and supplies required by the job's chosen technique.

HAS A EN.HQ.UNIT.ID. This attribute stores the index of the GG.UNIT corresponding to this HQ unit.

HAS A EN.HQ.INDEX. This index eases processing of the engineer history file in the postprocessor. The engineer HQ units are numbered consecutively from 1 through the sets of BLUE and RED HQ units. VIC engineer output uses these numbers for reference and associates each engineer HQ unit with a permanent entity having this index in the postprocessor.

HAS A EN.HQ.LIST.PROCESS.FLAG. This flag indicates whether the pending job list of this HQ unit is to be processed. When new jobs are added to an HQ's list of pending jobs or engineer assets return to an HQ from an assignment, this flag is set to ..YES so that its list of pending jobs can be processed during the next update. This mechanism prevents difficulties that arise from attempting to process each job of a mission as it is added to the list or to process the job list repeatedly when a set of assets returns.

HAS A EN.HQ.ASSET.PTR. This is a pointer to a 1 by N.EN.ASSET.PROTO array, the nth element of which is the number of pieces of engineer equipment of asset prototype n currently available to this unit. When equipment is transferred to a work team, the appropriate array entry is decreased by the amount of the transfer.

OWNS A EN.HQ.INVENTORY. This inventory provides the set of records used to track engineer assets throughout the simulation. The HQ unit maintains an inventory of all of its engineer assets. An HQ's inventory changes only when assets are transferred to or from another HQ unit in a permanent transfer. Otherwise, each piece of equipment in the inventory has a record of which GG.UNIT currently has it, its current job, its current level of damage, etc.

OWNS A EN.SET.OF.HQ.PENDING.JOBS. This is the set of jobs assigned to an HQ unit but not currently active. When engineer jobs arise, they are assigned to an HQ unit according to a fixed set of rules, prioritized according to the situation, and placed on this list. This list is the one that is processed when the EN.HQ.LIST.PROCESS.FLAG is ..YES.

OWNS A EN.SET.OF.HQ.ACTIVE.JOBS. This is the set of jobs assigned to an HQ unit and assigned to work teams that will work on them. When the HQ unit is ready to process a pending job and determines that all required equipment for it is available for assignment, work teams are created to bring the equipment together to do the job, and the job is moved to the active list. When a phase is completed, the job can move back to the pending list to wait for assets for the next phase.

OWNS A EN.SET.OF.PERFORMING.UNITS. This is the set of all work teams created to work on jobs for which this HQ is responsible. Engineer work teams are either in the free pool (not working) or on this list of the HQ unit responsible for the job that the work team is performing.

BELONGS TO A EN.SET.OF.HQ.UNITS. This set keeps track of the HQ units, which are temporary entities. Each side in the simulation has such a set.

EVERY EN.STAFF.UNIT

This is the decision-making engineer unit. A GG.UNIT of type ..ENGINEER whose superior is not of type ..ENGINEER is an engineer staff unit.

HAS A EN.STAFF.UNIT.ID. This is the index of the corresponding GG.UNIT.

HAS A EN.STAFF.SUPERIOR. This is the engineer HQ unit that is the engineer superior to the given staff unit. Staff engineer units are allowed to have both a maneuver superior (GM.UN.SUPE-

RIOR) and an engineer superior. This link is used for reallocation of resources between engineer command elements.

HAS A EN.STAFF.ASSET.PTR. This item is the address of a 1 by N.EN.ASSET.PROTO array whose nth element is the total amount of equipment of type n owned by the command chain headed by this staff engineer. The amount changes only if equipment is attrited or formally transferred into or out of this command.

HAS A EN.STAFF.TECH.PTR. This holds the address of a two-dimensional array indexed by task type and feature type, whose entries are the numbers of the first corresponding techniques for which the required equipment is available to this staff unit. If the entry in element m,n is zero, this staff unit is not capable of performing a job of task type m, feature type n. Otherwise, the entry is the technique number of the most preferred technique available to this command for this task/feature combination.

HAS A EN.STAFF.RANKING.LEVEL. This is the level of the staff unit's GG.UNIT prototype and is used to rank the units in the set of staff units, with units of low rank being filed first.

OWNS A EN.SET.OF.STAFF.JOBS. This set of jobs contains one job for each of the jobs on the active and pending lists of each of the HQ units in this staff unit's command chain. This list is rebuilt when reprioritization of jobs is necessary. It is used as a working set, being constructed solely to store all current jobs in one place for prioritization and then to organize the HQ lists in priority order.

BELONGS TO A EN.SET.OF.STAFF.UNITS. This set keeps track of the staff units, which are temporary entities. Each side in the simulation has such a set.

EVERY EN.WT.UNIT. This engineer unit simulates the work process. Engineer work teams are GG.UNITs, being created with the rest of the GG.UNITs but not having their attributes set by input data. These units are treated as spare GG.UNITs when they are not assigned to a specific job. When they are assigned to a job, they are in the set of performing units for the HQ unit responsible for the job.

HAS A EN.WT.UNIT.ID. This is the index of the GG.UNIT associated with this work team.

HAS A EN.WT.RESPONSIBLE.HQ. This is the HQ unit responsible for the jobs assigned to the work team when it is active; 0 otherwise.

HAS A EN.WT.ASSET.PTR. This structure points to a 1 by N.EN.ASSET.PROTO array whose nth entry is the number of assets of type n held by the work team.

HAS A EN.WT.JOB.PRIORITY. This item is the priority of the first job on the work team's list or 0.0 if the work team is returning from a job. The set of performing units for an HQ unit is ranked by this attribute so that when a search is being made for equipment to preempt, the work teams working on the lowest priority jobs are examined first.

HAS A EN.WT.MAX.SPEED. This attribute holds the maximum speed an active work team can travel based on the equipment this team holds. It is computed after the work team is formed and is used to retard the ground unit prototype speed of an engineer work team if it owns equipment incapable of moving at that speed.

OWNS A EN.SET.OF.WT.ASSETS. This set is the work team's record-keeping device for the equipment that has been temporarily transferred to it by an HQ unit. Since each engineer asset is

being tracked throughout the simulation, when a piece of equipment is transferred between an HQ unit and a work team or between two work teams, a record must be kept showing who owns the equipment and with which piece of equipment it is associated in the owning HQ's inventory.

OWNS A EN.SET.OF.WT.JOBS. This set keeps track of the jobs that the work team must attempt to finish. A work team is activated to work on a specific phase of a set of jobs in one mission for which a particular HQ unit is responsible. For each of those jobs, the main information about the job is kept with the HQ unit as an EN.JOB, but a record of the crucial information about the job for the work team is kept in EN.WT.JOB. Each EN.WT.JOB must have a corresponding EN.JOB, whereas many EN.WT.JOBs may correspond to only one EN.JOB.

MAY BELONG TO A EN.SET.OF.FREE.UNITS.

MAY BELONG TO A EN.SET.OF.PERFORMING.UNITS. Work teams must be in one of two places. If not active, they are filed in the free pool for their side. When teams are working on a job (or set of jobs), they are filed in the performing unit's set of the HQ unit responsible for the job in the order of EN.WT.JOB.PRIORITY, with lowest priority teams at the top of the list to aid the search for assets.

Engineer Assets

EVERY EN.ASSET.PROTO

This entity is a type of engineer equipment. For each type of engineer equipment being modeled, an asset prototype is created and has attributes which hold information about that equipment type.

HAS A EN.ASSET.PR.WP.PTR. This attribute is the index of the weapon prototype corresponding to this engineer asset type. Input data specifies equipment by name in both the FL and the EN modules. This attribute is set by matching the names from the FL weapon prototype list with the names in the EN input data.

HAS A EN.ASSET.DAMAGE.THRESHOLD. This attribute is the minimum usable portion that must be functioning for a piece of this type of equipment to be active. This information is necessary for modeling attrition of engineer assets. An asset that has damage at a level above this threshold suffers a degradation of work capacity, which is reflected in the calculation of its job's duration. Below this threshold, a piece of equipment is declared ..DAMAGED and is no longer available for work.

HAS A EN.ASSET.PR.SPEED. This item is the ground speed of this equipment under good trafficability and good weather.

HAS A EN.ASSET.PR.MAX.WORK.PERIOD. This item is the maximum length of time in seconds that this piece of equipment can work before requiring a rest.

HAS A EN.ASSET.PR.REST.PERIOD. This item is the length of time in seconds that this piece of equipment must rest after a maximum work period before it can work again.

EVERY EN.FUNCTION

These functions are performed by the engineer assets, as determined by input data.

HAS A EN.FUNCTION.NAME. This data structure is the text name of the function and is used for output reports.

EVERY EN.ASSET.PROTO AND EN.FUNCTION

This compound entity is used to connect engineer assets with the functions they perform.

HAS A EN.CAPABILITY. This structure is the capability of the given asset prototype to perform the given function. It can be a relative index or an actual performance measure, as long as this number agrees with the corresponding functional equipment requirements in the technique data.

EVERY EN.ASSET

This entity represents an actual piece of engineer equipment. It is owned by an engineer HQ unit and is kept in that HQ unit's inventory.

HAS A EN.ASSET.PROTO.ID. This is the index of this asset's prototype.

HAS A EN.ASSET.UNIT.ASSIGNMENT. This attribute keeps track of the GG.UNIT that has physical possession of the asset. When this asset is transferred temporarily to another unit, this attribute changes.

HAS A EN.ASSET.STATUS. This attribute tracks the activity of the equipment, corresponding to defined-to-means:

..AVAILABLE	TO MEAN 1
..PRE.TASK	TO MEAN 2
..IN.TRANSIT	TO MEAN 3
..WAITING	TO MEAN 4
..WORKING	TO MEAN 5
..IN.REST.PERIOD	TO MEAN 6
..DAMAGED	TO MEAN 7
..LEFT.AT.SITE	TO MEAN 8

HAS A EN.ASSET.UNDAMAGED.PORTION. This attribute is the current usable portion of the asset. When attrition is assessed, this number decreases from 1.0 (undamaged) until the level of damage is below the damage threshold for this type of asset. Between those two levels, the performance capability of this asset decreases linearly.

HAS A EN.ASSET.LAST.UPDATE. This is the clock time at which the last update of attributes was done. It is used to measure hours worked, hours at rest, etc.

HAS A EN.ASSET.HOURS.WORK.SINCE.REST. This structure keeps a running total of the time worked since rest.

HAS A EN.ASSET.CURRENT.JOB. This attribute points to the EN.JOB for which this asset is currently being used. An asset can be working on only one job at a time. If the asset is ..AVAILABLE (not working on a job), this attribute is 0.

HAS A EN.ASSET.CURRENT.JOB.PRIORITY. This item is the EN.JOB.PRIORITY of this asset's current job. If the asset is available, it is 0.0.

BELONGS TO A EN.HQ.INVENTORY. Every asset must be owned by an HQ unit. The EN.ASSET and its attributes serve as an inventory record in that HQ unit's EN.HQ.INVENTORY.

EVERY EN.RETRIEVABLE.ASSET

When a technique has required equipment marked with an EN.EQ.USE = ..RETRIEVABLE, that equipment is removed from the work team at the job site and made a retrievable asset. This structure keeps a record of what type of asset it is, where it is, and who owns it.

HAS A EN.REA.PROTO. This item is the asset prototype index of the equipment.

HAS A EN.REA.FEATURE. This pointer indicates the actual linear obstacle, minefield, etc. containing the equipment.

HAS A EN.REA.X.LOCATION. This is the x-coordinate of the location.

HAS A EN.REA.Y.LOCATION. This is the y-coordinate of the location.

HAS A EN.REA.OWNER. This HQ unit owns the equipment.

HAS A EN.REA.INV.ID. This attribute is a pointer to the actual EN.ASSET in the EN.HQ.INVENTORY.

BELONGS TO A EN.SET.OF.RETRIEVABLE.ASSETS. This set keeps track of retrievable assets belonging to a side.

EVERY EN.WT.ASSET

This is a work team record for an engineer asset it possesses temporarily. When an HQ unit transfers an asset to a work team, the work team must create a record for that item so that all of the information necessary for using and returning the asset can be stored.

HAS A EN.WT.ASSET.PROTO. This is the index of the asset prototype for this asset.

HAS A EN.WT.ASSET.OWNER. This HQ unit owns the asset.

HAS A EN.WT.ASSET.INV.ID. This points to the EN.ASSET in the HQ's inventory corresponding to this asset.

HAS A EN.WT.ASSET.FUNCTION. This is the number of the function this asset is to perform on the jobs assigned to the owning WT unit. If the required asset has been specified by prototype rather than function, then this attribute is 0.

BELONGS TO A EN.SET.OF.WT.ASSETS. This structure serves as the work team's inventory.

Engineer Techniques

EVERY EN.TECHNIQUE

This entity's attributes specify a particular way in which to perform an engineer task as determined by side, task type, and feature type.

HAS A EN.TECH.SIDE. This indicates the side (..RED or ..BLUE) that uses this technique.

HAS A EN.TECH.TASK. This is the task type (a define-to-mean) for which this technique is used.

HAS A EN.TECH.FEATURE.PROTO. This is the feature prototype on which this technique is used.

HAS A EN.TECH.COMMAND.LEVEL. This is the command level that usually employs this technique (a defined-to-mean).

HAS A EN.TECH.MIN.DISTANCE.FROM.FEBA. This structure is the minimum signed distance in kilometers from the FEBA under which this technique cannot be used. This distance can be negative.

HAS A EN.TECH.ORGANIZATION.PT. This is a defined-to-mean that allows data input to dictate whether units coming from a number of locations to work on a specific job rendezvous at the responsible HQ unit's location or at the work site. Its possible values are:

..AT.HQ	TO MEAN 0
..AT.SITE	TO MEAN 1

HAS A EN.TECH.EFFECT. For tasks such as linear obstacle or minefield breach, this attribute allows for more than one breach effect; i.e., different types of bridges or breaches have different delay factors associated with them. This factor will divide a standard breach delay factor to produce a breach-specific effect; i.e., a technique with effect 2 is twice as good as a technique with effect 1 and produces a crossing delay which is half of the delay associated with the technique having effect 1.

HAS A EN.TECH.ASSET.PTR. This points to a 1 by N.EN.ASSET.PROTO array whose nth element is the number of assets of type n (EN.EQ.PROTO = n) required for this technique.

HAS A EN.TECH.FN.PTR. This points to a 1 by N.EN.ASSET.FUNCTION.TYPE array whose nth element is the number of assets performing function n (EN.EQ.FUNCTION = n) of capability 1 required for this technique.

OWNS A EN.SET.OF.TECH.SEGMENTS. The technique is broken into pieces called segments, each with its own duration, end effect, and timing factors. This set keeps track of the segments for this technique.

OWNS A EN.SET.OF.REQ.EQUIPMENT. This set keeps track of the equipment required for this technique.

OWNS A EN.SET.OF.REQ.SUPPLIES. This entity lists the supplies required for this technique. A distinction is made between the supplies needed for the equipment being used (fuel and ammunition) and the supplies specifically required for the task (mines, explosives, wire). Supplies in this list are limited to those required for the task. Equipment will refuel and rearm automatically.

EVERY EN.TECH.SEGMENT

This structure is the smallest subdivision of a technique.

HAS A EN.SEGMENT.NUMBER. The technique's segments are sequential and are numbered consecutively by this attribute.

HAS A EN.END.SEGMENT.EFFECT. This is the fraction of the total effect of this technique that is complete at the end of this segment. If the job is interrupted at the end of the segment, the effect of the job is adjusted according to this number.

HAS A EN.INTRA.SEGMENT.EFFECT. This attribute indicates the interpolation method for computing the effect of interrupting the job in the middle of the segment. The functions are specified by define-to-means:

..NO.CHANGE	TO MEAN 1
..LINEAR	TO MEAN 2
..SKEW.LEFT	TO MEAN 3
..SKEW.RIGHT	TO MEAN 4

HAS A EN.SEGMENT.DURATION. This item specifies the time in seconds for completion of this segment, given that all required equipment is present and conditions are normal. The actual job duration will be computed by adjusting this duration when equipment levels fall, equipment is damaged, or any of the factors below change. The following tasks must have non-zero segment durations and cannot have segment lengths determined by a rate:

..LINE.OBSTACLE.BREACH

..IMPROVE.LINE.BREACH

..REPAIR.ROAD.CRATER

..REMOVE.LINEAR.BREACH

..CRATER.ROAD

..PREPARE.POSITION

HAS A EN.SEGMENT.RATE. If duration is 0, then the length of the segment will be computed by dividing the size of the job by this rate. The units of measure for the rate vary with the task type: mines/sec for minefield emplacement, km depth/sec for minefield breaching, km/day for linear obstacle emplacement, km/sec for building combat trails.

HAS A EN.SEGMENT.VISIBILITY.FACTOR. This multiplier lengthens the segment duration to account for performance of the task at night.

HAS A EN.SEGMENT.WEATHER.FACTOR. This multiplier lengthens the segment duration to account for a decline of one level in weather (GOOD to FAIR, FAIR to POOR).

HAS A EN.SEGMENT.COMBAT.FACTOR. This multiplier lengthens the segment duration to account for the effects of working while under fire.

HAS A EN.END.SEGMENT.ACTION. When a segment is completed, one of three actions can be taken, as specified by defined-to-means:

..CONTINUE	TO MEAN 1
..BREAK.POINT	TO MEAN 2
..END	TO MEAN 3

BELONGS TO A EN.SET.OF.TECH.SEGMENTS. Each technique has a set of segments into which it is divided. Each EN.TECH.SEGMENT belongs to such a set.

EVERY EN.EQUIPMENT.USE.SEGMENT

This entity allows the EN.REQ.EQUIPMENT to keep a list of segments in which it is used. Input data specifies in which segments (by number) each type of required equipment is used.

HAS A EN.EQ.USE.SEGMENT.NBR. This is the segment number of one of the segments in which the equipment is used.

BELONGS TO A EN.SET.OF.EQUIPMENT.USE.SEGMENTS. This set contains the segments of a technique that use the required equipment.

EVERY EN.REQ.EQUIPMENT

For each type of equipment needed for a technique, one of these entities is created. The type of equipment can be specified by asset prototype or by function.

HAS A EN.EQ.PROTO. If the equipment is specified by asset prototype, this attribute is the asset type's index.

HAS A EN.EQ.FUNCTION. If the equipment is specified by function, this attribute is the function number. One and only one of EN.EQ.PROTO and EN.EQ.FUNCTION must be nonzero.

HAS A EN.EQ.AMOUNT. This specifies the number of pieces of equipment of this type required. If the type is specified by function, it is the number required with capability 1. The actual number of pieces required for a particular function will be computed from this number and the available asset's relative capability by taking the sum over all asset types being used of the product of the number of assets of the type by the capability of that type.

HAS A EN.EQ.MINIMUM. This item specifies the minimum number of pieces of equipment of this type that must be present for a job to continue. If the type is specified by function, this number will be used with the relative capability of the actual pieces used to determine whether the job can continue.

HAS A EN.EQ.USE. This defined-to-mean indicates whether the equipment is ..PERMANENT or ..RETRIEVABLE. A retrievable asset does not return from the job site and is essentially consumed unless a decision is made later to retrieve it.

HAS A EN.EQ.PREP.TIME. This is the amount of time (in seconds) required to prepare this piece of equipment at the engineer HQ before it can leave for the work site.

OWNS A EN.SET.OF.EQUIPMENT.USE.SEGMENTS. This set lists segments in which this equipment is used.

BELONGS TO A EN.SET.OF.REQ.EQUIPMENT. This set contains all of the required equipment for the given technique.

EVERY EN.REQ.SUPPLY

One of these entities is created for each type of supply needed for this technique.

HAS A EN.SUPPLY.PROTO. This is the supply prototype from the LO module.

HAS A EN.SUPPLY.AMOUNT. This is the amount of the supply required.

HAS A EN.SUPPLY.UNIT.OF.MEASURE. For jobs that have an associated size, this attribute is the unit of measure associated with the required amount. The following formula is used to compute the supply requirement:

$$\text{Required Supplies} = \frac{\text{Job size x Supply amount}}{\text{Supply unit of measure}} \qquad \text{[Eq 2]}$$

EXAMPLES: for emplacing minefields, supply amount and unit of measure for mines are both 1, so that the required supplies equal the job size (the number of mines). For breaching minefields (job size is depth in kilometers) with line charges, if each line charge clears 150 m, then supply amount is 1 and unit of measure is 0.15.

HAS A EN.SUPPLY.SEGMENT.USED. This is the segment in which the supplies are effectively consumed. Interruption of a job before this point will mean that the supplies will return to the HQ unit.

BELONGS TO A EN.SET.OF.REQ.SUPPLIES. This set contains all of the required supplies for the given technique.

Engineer Jobs

EVERY EN.JOB

This data structure is the master record keeper for specific engineer activity that has been assigned to an engineer HQ unit.

HAS A EN.JOB.TASK. This is the defined-to-mean task type associated with the job.

HAS A EN.JOB.FEATURE. This is the feature prototype associated with the job.

HAS A EN.JOB.FEATURE.PTR. This pointer indicates the actual feature involved.

HAS A EN.JOB.METHOD. This attribute has three possible defined-to-mean values that indicate how a technique is to be chosen for doing the job. ..PREFERRED.METHOD means that the first technique listed in the input data that is suitable for the job will be used. ..SHORTEST.TIME means that all usable techniques will be considered and the technique with the shortest completion time will be chosen. ..BEST.EFFECT means that all usable techniques will be considered and the technique with

the most effective end result will be chosen. The default is ..PREFERRED. METHOD, but in dynamically generated jobs, the combat status of the requesting unit may prompt a switch to one of the other two methods.

HAS A EN.JOB.TECHNIQUE. This is the index of the technique by which the work is to be done. It can change until the job has been started, at which time the technique must remain constant.

HAS A EN.JOB.SIZE. For jobs in which the duration is a function of the feature size, this attribute holds the necessary measurement: number of mines for emplacing and clearing of minefields, length in kilometers for emplacing and removing linear obstacles and building combat trails, and depth in kilometers for breaching minefields.

HAS A EN.JOB.PRIORITY. This numerical value reflects the priority of this job, to be compared with other jobs in determining the order in which jobs should be done.

HAS A EN.JOB.MISSION.CREATION.TIME. This is the time at which the original mission was created.

HAS A EN.JOB.ACTIVATION.TIME. This is the time at which the first segment of the job was activated, i.e., when the first work team unit was formed to work on the first phase.

HAS A EN.JOB.LATEST.COMPLETION.TIME. This is the latest time for completion of the job. If the job cannot be completed by this time, it will be canceled as soon as that fact is recognized.

HAS A EN.JOB.CURRENT.SEGMENT.START.TIME. This is the time at which the active segment started. The segments are sequential, so only one segment can be active at any given time.

HAS A EN.JOB.EST.DURATION. This is an estimate of when the job should be completed.

HAS A EN.JOB.LAST.SEGMENT.COMPLETED. This is the number of the last completed segment of the job.

HAS A EN.JOB.LAST.SEGMENT.COMPLETION.TIME. This is the time at which the last completed segment of the job was done.

HAS A EN.JOB.LAST.SEGMENT.ASSIGNED. This attribute keeps track of where the HQ unit is in assigning portions of the job to work teams.

HAS A EN.JOB.X.LOCATION. This is the x-coordinate of the work site.

HAS A EN.JOB.Y.LOCATION. This is the y-coordinate of the work site.

HAS A EN.JOB.REQUESTING.UNIT. This entity is the GG.UNIT for whom the work is being done, if one exists. Preplanned obstacles will not have requesting units, but defensive preparations and dynamically generated jobs will.

HAS A EN.JOB.HQ.UNIT. This is the HQ unit that is responsible for the job and that has this job on its pending or active job list.

HAS A EN.JOB.MISSION. This attribute is the model-assigned mission number for this job. Input data allows an "original mission number" for specifying when several jobs of one type will be done by one work team. Dynamically generated jobs will also be grouped together in a coordinated

plan. Because the engineer module mixes all types of jobs on the same list, the original mission number cannot be used for this attribute and has no relationship to it.

OWNS A EN.SET.OF.ACTIVE.PHASES. This set keeps track of which phases of a job have been assigned but not completed. Each phase of the job will be done by a different set of work teams. Since jobs are grouped by mission, several phases of a job may be active at once even though progress proceeds from one segment to the next in proper sequence.

MAY BELONG TO A EN.SET.OF.HQ.PENDING.JOBS.

MAY BELONG TO A EN.SET.OF.HQ.ACTIVE.JOBS. A job is either pending or active and is therefore on one of the above lists for the HQ unit responsible for it.

EVERY EN.ACTIVE.PHASE

The technique for the job divides the job into segments that are grouped together in phases according to the EN.END.SEGMENT.ACTION. Each phase is performed by a separate work team.

HAS A EN.PHASE.FIRST.SEGMENT. This pointer indicates the EN.TECH.SEGMENT with which this phase begins.

HAS A EN.PHASE.LAST.SEGMENT. This pointer indicates the EN.TECH.SEGMENT with which this phase ends.

HAS A EN.PHASE.ASSET.PTR. This pointer addresses a 1 by N.EN.ASSET.PROTO array whose nth element is the number of pieces of equipment of type n required during this phase of the job.

HAS A EN.PHASE.FN.PTR. This pointer addresses a N.EN.ASSET.PROTO by N.EN.ASSET.FUNCTION.TYPE array whose (m,n) element is the number of pieces of engineer asset type m performing function type n during this phase of the job.

OWNS A EN.SET.OF.PHASE.WT.UNITS. This set keeps track of the WT units created to carry equipment for this phase of the job.

BELONGS TO A EN.SET.OF.ACTIVE.PHASES. The job keeps this list of active phases.

EVERY EN.PHASE.WT.UNIT

This entity points to an engineer work team. The work teams themselves are filed in the EN.SET.OF.PERFORMING.UNITS of the HQ unit that owns the job.

HAS A EN.PHASE.WT.PTR. This pointer indicates the EN.WT.UNIT assigned to this phase.

BELONGS TO A EN.SET.OF.PHASE.WT.UNITS. This set keeps track of the work teams assigned to a particular phase of a job. The set is owned by a EN.ACTIVE.PHASE.

EVERY EN.WT.JOB

This entity points to a job assigned to a work team.

HAS A EN.WT.JOB.PTR. This attribute points to the corresponding EN.JOB.

HAS A EN.WT.JOB.PHASE. This attribute indicates the EN.ACTIVE.PHASE to which this work team is assigned.

BELONGS TO A EN.SET.OF.WT.JOBS. This set keeps track of jobs assigned to the work team.

EVERY EN.STAFF.JOB

The engineer staff unit is responsible for prioritizing the jobs in its entire command chain. It uses this entity to build a list of those jobs to use as a working set when determining priorities. An EN.STAFF.JOB is created for each job in the command chain.

HAS A EN.SJ.RESPONSIBLE.UNIT. This HQ unit is responsible for the corresponding EN.JOB.

HAS A EN.SJ.PTR. This pointer indicates the corresponding EN.JOB.

HAS A EN.SJ.ACTIVE.INDICATOR. This attribute is 0 if the job is pending and 1 if it is active.

HAS A EN.SJ.PRIORITY.LEVEL. This entity is the job's priority level that is determined in the prioritization process and is used to rank the individual unit job lists.

BELONGS TO A EN.SET.OF.STAFF.JOBS. This structure lists all of the jobs, both pending and active, for which units in this staff's command chain are responsible.

Define-to-Means for Engineers (EN)

The two extreme levels of position preparation are:

DEFINE ..UNPREPARED TO MEAN 0.0

DEFINE ..FULLY.PREPARED TO MEAN 1.0

The types of tasks engineers are capable of performing are:

DEFINE ..MINEFIELD.BREACH TO MEAN 1

DEFINE ..MINEFIELD.CLEAR TO MEAN 2

DEFINE ..LINE.OBSTACLE.BREACH TO MEAN 3

DEFINE ..RIVER.CROSSING TO MEAN 3

DEFINE ..IMPROVE.LINE.BREACH TO MEAN 4

DEFINE ..REPAIR.ROAD.CRATER TO MEAN 5

DEFINE ..BUILD.TRAIL TO MEAN 6

DEFINE ..BUILD.FALF TO MEAN 7

```
DEFINE ..EMPLACE.MINEFIELD                TO MEAN  8
DEFINE ..EMPLACE.LINE.OBSTACLE            TO MEAN  9
DEFINE ..REMOVE.LINEAR.BREACH             TO MEAN 10
DEFINE ..BRIDGE.DEMOLITION                TO MEAN 10
DEFINE ..CRATER.ROAD                      TO MEAN 11
DEFINE ..PREPARE.POSITION                 TO MEAN 12
DEFINE ..BUILD.CSS.FACILITY               TO MEAN 13
DEFINE ..MAINTAIN.CSS.FACILITY            TO MEAN 14
```

The number of tasks engineers are capable of performing is:

```
DEFINE ..NUMBER.TASKS                     TO MEAN 14
```

The set of values for EN.ASSET.STATUS is:

```
DEFINE ..AVAILABLE                        TO MEAN  1
DEFINE ..PRE.TASK                         TO MEAN  2
DEFINE ..IN.TRANSIT                       TO MEAN  3
DEFINE ..WAITING                          TO MEAN  4
DEFINE ..WORKING                          TO MEAN  5
DEFINE ..IN.REST.PERIOD                   TO MEAN  6
DEFINE ..DAMAGED                          TO MEAN  7
DEFINE ..LEFT.AT.SITE                     TO MEAN  8
```

The set of values for EN.TECH.ORGANIZATION.PT is:

```
DEFINE ..AT.HQ                            TO MEAN  0
DEFINE ..AT.SITE                          TO MEAN  1
```

The set of values for EN.INTRA.SEGMENT.EFFECT is:

```
DEFINE ..NO.CHANGE                        TO MEAN  1
DEFINE ..LINEAR                           TO MEAN  2
```

DEFINE ..SKEW.LEFT TO MEAN 3

DEFINE ..SKEW.RIGHT TO MEAN 4

The set of values for EN.END.OF.SEGMENT.ACTION is:

DEFINE ..CONTINUE TO MEAN 1

DEFINE ..BREAK.POINT TO MEAN 2

DEFINE ..END TO MEAN 3

The set of values for EN.EQ.USE is:

DEFINE ..PERMANENT TO MEAN 1

DEFINE ..RETRIEVABLE TO MEAN 2

The set of values used in the prioritization process for jobs is:

DEFINE ..VITAL TO MEAN 400.0

DEFINE ..CRITICAL TO MEAN 300.0

DEFINE ..ESSENTIAL TO MEAN 200.0

DEFINE ..NECESSARY TO MEAN 100.0

The set of values used in producing a job report for the engineer history file is:

DEFINE ..CREATE.JOB TO MEAN 1

DEFINE ..ACTIVATE.PHASE TO MEAN 2

DEFINE ..BEGIN.SEGMENT TO MEAN 3

DEFINE ..ADJUST.SEGMENT.DURATION TO MEAN 4

DEFINE ..END.SEGMENT TO MEAN 5

DEFINE ..COMPLETE.PHASE TO MEAN 6

DEFINE ..COMPLETE.JOB TO MEAN 7

DEFINE ..COMPLETE.MISSION TO MEAN 8

DEFINE ..PREEMPT.EQUIPMENT TO MEAN 9

```
DEFINE ..DISCONTINUE.ATTRITION                TO MEAN 10
DEFINE ..DISCONTINUE.LATE.FINISH              TO MEAN 11
DEFINE ..DISCONTINUE.RELATED.JOBS             TO MEAN 12
```

The set of values for EN.MISSION.METHOD and EN.JOB.METHOD is:

```
DEFINE ..PREFERRED.METHOD                     TO MEAN  0
DEFINE ..SHORTEST.TIME                        TO MEAN  1
DEFINE ..BEST.EFFECT                          TO MEAN  2
```

6 SUMMARY OF ROUTINES

Overview

Each routine of the engineer module is listed below with its given and yielded arguments. A brief description of the purpose of the routine follows. The routines are listed in alphabetical order.

Description

ROUTINE EN.ADD.AVAILABLE.ASSETS
GIVEN .UNIT, .ASSET.ARRAY, .PRIORITY

This recursive routine cycles through a staff unit and all of its HQ subordinates and their HQ subordinates down the command chain, adding up all of the engineer equipment available at HQ units and at those WT units working on lower priority jobs so that .ASSET.ARRAY indicates all of the assets available within that command chain to work on a job of the given .PRIORITY.

ROUTINE EN.ADD.JOBS.TO.LIST
GIVEN .STAFF, .UNIT

This recursive routine cycles through all of the .UNITs in the given .STAFF's command chain, adding a job to the staff unit's set of jobs for each active and pending job assigned to .UNIT. The jobs are removed from their HQ lists to be prioritized and refiled in the new priority order by the calling routine EN.PRIORITIZE.JOB.LIST.

ROUTINE EN.ARE.WTS.COLLOCATED
GIVEN .WT1, .WT2 YIELDING .ANSWER

This routine determines whether two work teams, .WT1 and .WT2, are at the same location, returning an .ANSWER of ..YES or ..NO.

ROUTINE EN.ASSESS.ENGINEER.SUPPORT.CAPABILITY
GIVEN .UNIT, .TASK, .FEATURE, .SIZE, .METHOD
YIELDING .MIN.DURATION, .TECHNIQUE

This routine is called to determine if a nonengineer .UNIT has explicit engineer support available to perform the engineer .TASK on a given .FEATURE with a given job .SIZE. If the capability is there, this routine returns the time .DURATION necessary to do the work and determines which engineer .TECHNIQUE best fits the given .METHOD, a defined-to-mean with three possible values corresponding to the preferred method as determined by input data, the method requiring the shortest time to complete, or the method yielding the most effective result.

ROUTINE EN.ASSESS.NON.ENGINEER.CAPABILITY
GIVEN .UNIT, .TASK, .FEATURE, .SIZE, .METHOD
YIELDING .MIN.DURATION, .TECHNIQUE

This routine is called to determine if a nonengineer .UNIT is capable of performing the engineer .TASK on a given .FEATURE with a given job .SIZE. This routine uses the technique data from the engineer module, comparing the engineer equipment owned by the given unit with the equipment required to perform the given work. If the equipment is owned, this routine returns the time

.MIN.DURATION necessary to do the work by the chosen .TECHNIQUE. The technique is chosen on the basis of the criteria determined by .METHOD (most preferred, shortest time, most effective).

ROUTINE EN.ASSIGN.ENGINEER.JOB
GIVEN .SIDE, .REQUEST.UNIT, .TASK, .FEATURE, .X.LOC, .Y.LOC,
.SIZE, .METHOD
YIELDING .HQ, .TECHNIQUE

This routine handles the search for an engineer HQ unit on the given .SIDE to be responsible for a possible job of the given .TASK, .FEATURE, .SIZE, and location. It first looks for a staff unit whose tactical area contains the job and whose command chain owns equipment by which the job can be done. Then it searches the staff unit's command chain for the unit at the command level specified by the chosen technique and closest to the job site. Engineer staff units supporting the job's .REQUESTING.UNIT are considered first if the requesting unit is known. The .METHOD indicates the type of technique being sought, with the usual case of taking the first technique found in the input data, but allowing for taking the technique with the shortest duration or the best effect.

ROUTINE EN.BEGIN.CURRENT.JOB.SEGMENT
GIVEN .WT

With the arrival of .WT at its destination, if it is at the work site of the first job on its job list, then this routine checks whether the segment of work that this unit is assigned is ready to begin and starts the job segment if it is. If the .WT is assigned to a phase of the job that is not ready to begin, this routine does nothing; completion of the preceding phase will trigger this .WT to begin work.

ROUTINE EN.BUILD.STAFF.ARRAYS
GIVEN .STAFF

This routine cycles through the techniques for each task and feature combination, comparing the total undamaged asset holdings of .STAFF and all of its engineer HQ subordinates with the required equipment for each technique to determine if this command chain can use the technique. This routine finds the most preferred technique possible for this unit and its subordinates using any combination of equipment available to the group as a whole. The resulting technique array owned by the staff unit indicates the most preferred technique its command chain can use for each type of job (task and feature combination).

ROUTINE EN.BUILD.WORK.UNIT.AT.HQ
GIVEN .JOB, .PHASE, .CONTRIBUTING.HQ, .NEEDED.ASSET, .NEEDED.FN, .IN.FLAG
YIELDING .FLAG

This recursive routine moves through .CONTRIBUTING.HQ and all of the HQ units subordinate to it searching for the engineer assets required for the given .PHASE of the given .JOB. The two arrays .NEEDED.ASSET and .NEEDED.FN indicate what is still needed. As long as .FLAG (.IN.FLAG) is ..YES, the search continues. .FLAG is ..NO when both arrays are zero. At each .CONTRIBUTING.HQ, if needed equipment is found, it is transferred to a newly created work team, which will move it to the job. If logistics is being played, each .CONTRIBUTING.HQ gives its equipment the fuel and ammunition required. At the very beginning of the search, a work team is created at the job's responsible HQ to carry the job supplies to the work site.

ROUTINE EN.BUILD.WORK.UNIT.AT.WT
GIVEN .JOB, .PHASE, .CONTRIBUTING.HQ, .NEEDED.ASSET, .NEEDED.FN, .IN.FLAG
YIELDING .FLAG

This recursive routine moves down the command chain from CONTRIBUTING.HQ searching for equipment needed for the given .PHASE of the given .JOB held by work teams working on jobs of lower priority. If needed equipment is found, a new work team is created and the equipment is transferred to it to move to the new job site. If the preempted work team is currently using the equipment, the equipment leaves at the end of the current segment. The two arrays .NEEDED.ASSET and .NEEDED.FN keep track of which equipment is still being sought. The search continues until .FLAG (.IN.FLAG) is ..NO or the bottom of the chain is reached. .FLAG becomes ..NO when both arrays are zero. Preprocessing should guarantee that all of the required equipment will be found.

ROUTINE EN.CALCULATE.NEW.FINISH.TIME
GIVEN .WT, .TECHNIQUE, .SEGMENT
YIELDING .DURATION

This routine computes the .DURATION of a .SEGMENT of a job by comparing the assets of .WT unit doing the work with the equipment required for this segment of the given .TECHNIQUE. This routine is used only when the work team has less equipment than was originally sent for the job. If the level of any of the required equipment is below the minimum specified by input data, then the duration is set high enough to cause the job to be discontinued. The asset array expresses in integer mode the number of pieces of equipment owned, while the EN.ASSET.UNDAMAGED.PORTION of each asset expresses the usable portion of that asset. Work time is degraded if the equipment is not at 100 percent.

ROUTINE EN.CHECK.FOR.SUPPLY.TYPES.IN.INVENTORY
GIVEN .HQ.GG, .TECHNIQUE
YIELDING .ANSWER

This routine compares the supply list for a .TECHNIQUE with the supply inventory for an .HQ unit and returns ..YES in .ANSWER if the unit receives each type of required supply.

ROUTINE EN.CHECK.FOR.AVAILABILITY.OF.JOB.SUPPLIES
GIVEN .UNIT, .TECHNIQUE, .START, .SIZE
YIELDING .ANSWER

This routine compares the supply list for a .TECHNIQUE with the supply inventory for a GG unit and returns ..YES in .ANSWER if the unit has each type of required supply on hand. .START is the segment number from which to begin checking and .SIZE is the size of the job being considered.

ROUTINE EN.CLOSE.OUT.JOB.PHASE
GIVEN .WT

This routine does the bookkeeping that is needed when a work team completes a phase of a job. The job attributes must be updated, and the WT unit sent to either its next job or home if it is finished.

ROUTINE EN.COMPARE.OWNED.TO.REQUIRED
GIVEN .UNIT.ASSET.ARRAY, .JB.ASSET.ARRAY, .JB.FN.ARRAY,
.PH.FN.ARRAY
YIELDING .ANSWER

This routine compares owned assets with required assets and is used in two ways: to decide which techniques a staff's command chain can use by comparing the total of its assets with the assets/functions required by the technique and to decide whether the assets owned by a command chain and available or working on a lower priority job are sufficient to meet the needs of the current phase of a job. In both instances, .UNIT.ASSET.ARRAY indicates what is owned, .JB.ASSET.ARRAY indicates the needed assets that have been specified by prototype, and .JB.FN.ARRAY indicates the functional needs. If all needed assets are available, the functional needs are assessed and a two-dimensional array .PH.FN.ARRAY is calculated to specify which assets will be needed to perform which functions. If all needs are met, then .ANSWER is ..YES, otherwise ..NO. When this routine is looking for equipment for a specific job phase, .PH.FN.ARRAY is further used as the functional equipment specifier for the phase.

ROUTINE EN.COMPUTE.DISTANCE.TO.JOB
GIVEN .UNIT, .JOB
YIELDING .DISTANCE

This utility routine computes the .DISTANCE between a given GG.UNIT called .UNIT and a given .JOB.

ROUTINE EN.COMPUTE.ENGINEERS.TRUE.STRENGTH
GIVEN .UNIT, .WG
YIELDING .STRENGTH

This routine determines the weapon group strength of an engineer unit in terms of what the unit owns versus what the unit currently has on hand, which is the value of FL.WG.STRENGTH. This routine is used for determining supply and maintenance needs.

ROUTINE EN.COMPUTE.WORK.TEAM.SPEED
GIVEN .WT.GG, .BASE.SPEED
YIELDING .SPEED

This routine computes the speed at which a work team can travel, subject to the speeds of the assets it owns.

ROUTINE EN.CONSTRUCT.A.MINEFIELD.MISSION
GIVEN .UNIT

This routine constructs a minefield mission in support of .UNIT, which is considered the requesting unit for this mission.

ROUTINE EN.CONSTRUCT.AN.OBSTACLE.MISSION
GIVEN .UNIT

Requires future expansion.

ROUTINE EN.CREATE.A.MISSION.REQUEST
GIVEN .TASK, .X.LOC, .Y.LOC, .UNIT, .TIME, .FEATURE

This routine creates an engineer mission for the given data.

ROUTINE EN.CREATE.A.WORK.UNIT
GIVEN .HQ, .X.LOCATION, .Y.LOCATION, .PRIORITY
YIELDING .WT

This routine retrieves an engineer work team unit .WT from the free pool and sets its attributes. .HQ is .WT's superior, and .WT's location is .X.LOCATION, .Y.LOCATION. Because .WT will be filed by the priority of its job in the EN.SET.OF.PERFORMING.UNITS, .PRIORITY is given.

ROUTINE EN.CREATE.HQ.UNITS

This routine builds the two lists of engineer HQ units in the data base and establishes their equipment inventories and asset arrays. It also builds the lists of engineer staff units, which are HQ units whose GM.UN.SUPERIOR is not an engineer.

ROUTINE EN.DELAY.ENGINEERS.IF.AT.TASK.SITE
GIVEN .UNIT, .DELAY.TIME

This routine resets the time an engineer work team spends at the task site given .DELAY.TIME imposed by current artillery or air activity.

ROUTINE EN.DETERMINE.FUNCTION.ARRAY
GIVEN .EXCESS.ARRAY, .JB.FN.ARRAY, .PH.FN.ARRAY
YIELDING .ANSWER

This routine works with an array of owned assets (.EXCESS.ARRAY) and an array of functional requirements (.JB.FN.ARRAY) to produce the elements of .PH.FN.ARRAY, a two-dimensional array whose m,n element is the number of engineer assets of type m to use for the function of type n. If enough assets are available to meet every functional need, the .ANSWER is ..YES and .PH.FN.ARRAY specifies how equipment is to be used.

ROUTINE EN.DETERMINE.JOB.LOAD
GIVEN .UNIT, .OLD.TOTAL
YIELDING .NEW.TOTAL

This recursive routine cycles through a staff unit and all of its HQ subordinates and their HQ subordinates down the command chain, adding up all of the engineer jobs, both pending and active, to return the total number of jobs assigned to this command chain.

ROUTINE EN.DETERMINE.REQUIRED.EQUIPMENT.ARRAYS
GIVEN .TECHNIQUE, .FIRST, .LAST, .NUMBER.OF.JOBS,
.ASSET.ARRAY, .FN.ARRAY

This routine retrieves the required equipment information for .TECHNIQUE for the phase of work from segment .FIRST to segment .LAST and builds the equipment and function arrays. Since jobs are assigned by mission, any equipment to be left at the job site (EN.EQ.USE = ..RETRIEVABLE) must be multiplied by the .NUMBER.OF.JOBS to be in adequate supply.

ROUTINE EN.DETERMINE.WT.UNIT.PATH
GIVEN .WT, .X.DESTINATION, .Y.DESTINATION,
.DEST.TYPE, .START.TIME

This routine builds the path of .WT from its present location to its .DESTINATION. This is a universal path builder, so the code needs to be as generic as possible. .DEST.TYPE specifies the GG.PP.TYPE of the destination, which can be either a base point or a task point. .START.TIME specifies when the move is to begin. In building the path, this routine inserts path points between the start point and the destination to route the unit around known obstacles.

ROUTINE EN.DISCONTINUE.ENGINEER.MISSION
GIVEN .WT.GG

This routine does the bookkeeping necessary to cancel the jobs assigned to .WT.

ROUTINE EN.DISSOLVE.WT.UNIT
GIVEN .WT

When a work team has reached its final destination and has transferred all of its assets to another unit, this routine takes care of the bookkeeping involved in emptying all of the unit's sets and refiling it in the set of free units.

ROUTINE EN.DOES.THIS.COMMAND.HAVE.SUPPLIES
GIVEN .UNIT, .TECHNIQUE
YIELDING .ANSWER

This recursive routine cycles through a staff unit and all of its subordinates looking for a unit whose supply inventory contains all of the required supplies to use the given .TECHNIQUE.

EVENT EN.END.OF.JOB.SEGMENT
GIVEN .WT

This event registers the completion of the current segment of the first job on .WT's job list and determines the next course of action.

EVENT EN.END.OF.REST.PERIOD
GIVEN .ASSET

This event resets an asset's attribute after its rest period so that the asset can be assigned to jobs again. When an asset is resting, its priority is set so that no job can preempt the rest.

ROUTINE EN.FILE.FREE.UNIT.IN.POOL
GIVEN .WT

This routine removes an engineer work team .WT from active status and files it in the free pool. Necessary GG.UNIT attributes are set so that the associated GG.UNIT is effectively removed from action.

ROUTINE EN.FILE.FREE.WEAPON.GROUP
GIVEN .WPN.GROUP

This routine clears the attributes of a weapon group and files it in the free set for reuse.

ROUTINE EN.FIND.BYPASS.FOR.LINE.OBSTACLE
GIVEN .UNIT, .FS, .X.INTER, .Y.INTER
YIELDING .X.C, .Y.C

This routine finds the bypass point C for the given line obstacle .FS, given that this .UNIT's path intersects the obstacle at .X.INTER, .Y.INTER.

ROUTINE EN.FIND.ENGINEER.HQ
GIVEN .UNIT
YIELDING .HQ

This routine links a GG.UNIT of type ..ENGINEER.UNIT with its associated .HQ unit, returning 0 if the unit is not in the set of engineer HQ units.

ROUTINE EN.FIND.OBSTACLE.BETWEEN.TWO.POINTS
GIVEN .UNIT, .X.A, .Y.A, .X.B, .Y.B
YIELDING .X.C, .Y.C

This routine finds the first obstacle that is either a line feature or a minefield located between A and B, and finds a point C by which the unit is able to bypass the obstacle. A is the starting point, and B is the destination point of the UNIT. If there is no obstacle between A and B, it returns B.

ROUTINE EN.FIND.SHORTEST.PATH.BETWEEN.TWO.POINTS
GIVEN .POINT.A, .POINT.B, .NEW.ROUTE

This routine recursively finds the shortest path from point A to point B.

ROUTINE EN.FIND.STAFF.ENGINEER
GIVEN .UNIT
YIELDING .STAFF

This routine links a GG.UNIT of type ..ENGINEER.UNIT to its associated .STAFF unit, returning 0 if the unit is not a staff engineer.

ROUTINE EN.FIND.STAFF.FOR.THIS.HQ
GIVEN .HQ
YIELDING .STAFF

This routine cycles up the command chain from .HQ until it locates .HQ's .STAFF unit.

ROUTINE EN.FORM.INITIAL.WT.UNITS
GIVEN .THIS.JOB, .PHASE, .NEEDED.ASSET, .NEEDED.FN

This routine searches .THIS.JOB'S responsible unit's command chain for the equipment needed for the given .PHASE. .NEEDED.ASSETS and .NEEDED.FN are the two arrays that indicate how much equipment is still needed. The elements of these arrays approach zero as more equipment is located. When all elements are zero, the equipment needs have been satisfied. Prior processing ensures that all of the needed equipment will be found.

ROUTINE EN.GET.TASK.NAME
GIVEN .TASK
YIELDING .NAME

This utility routine associates a text name with each task type being modeled. .TASK is the define-to-mean number for the task, and .NAME is its text counterpart.

ROUTINE EN.HOW.MUCH.TIME.IS.NEEDED
GIVEN .TECHNIQUE, .JOB.SIZE, .SEGMENT
YIELDING .EST.DURATION

This routine totals the segment times for using this .TECHNIQUE for a job of .JOB.SIZE and returns the estimated remaining duration, starting with .SEGMENT.

ROUTINE EN.HQ.DUMP
GIVEN .HQ

This routine produces a headquarters status report for one of the interactive menu options of EN.PRINT.CONTROL

ROUTINE EN.INITIALIZE.TASK.TECH.ARRAY

This routine builds a ragged array dimensioned by .SIDE, .TASK, .FEATURE, and .TECH.NO so that it will be known immediately which techniques are available for each engineer effort as determined by the first three dimensions.

ROUTINE EN.IS.THIS.STAFF.UNIT.CAPABLE
GIVEN .STAFF, .TASK, .FEATURE, .X.LOC, .Y.LOC, .SIZE, .METHOD
YIELDING .TECHNIQUE

This routine is part of the process for determining the engineer unit to be responsible for assigning a work team to do a possible job of the given .TASK and .FEATURE at the given location .X.LOC,.Y.LOC. If .STAFF has the given site in its tactical area and has a .TECHNIQUE for doing the job that satisfies the "minimum distance from FEBA" rule, then .TECHNIQUE is the technique number. Otherwise, .TECHNIQUE is 0. If logistics is played, a staff unit must have a unit in its command that receives the types of supplies required for the technique. .METHOD is one of three defined-to-means used in choosing the technique.

ROUTINE EN.IS.WT.AT.BASE
GIVEN .WT
YIELDING .ANSWER

This utility routine determines if the given .WT unit is at its base HQ location.

ROUTINE EN.IS.WT.AT.WORK.SITE
GIVEN .WT
YIELDING .ANSWER

This utility routine determines if the given .WT unit is at the site of its first job.

ROUTINE EN.JOB.DUMP
GIVEN .JOB

This routine produces a job status report for one of the interactive menu options of EN.PRINT.CONTROL

EVENT EN.JOB.UPDATE

This event schedules itself after its initial scheduling by EN.LOAD.AND.LIST.ENGINEER.DATA to periodically check the sets of engineer missions so that jobs can be created and processed.

ROUTINE EN.LIST.ASSET.DATA

This routine lists the engineer asset data during initialization.

ROUTINE EN.LIST.ENGINEER.COMMAND.LINKS

This routine lists the engineer command links that allow each staff unit to be linked to another engineer unit for the purpose of reallocating assets and supplies.

ROUTINE EN.LIST.JOB.PRIORITY.DATA

This routine lists engineer job prioritization data during initialization. It is not currently implemented.

ROUTINE EN.LIST.TECHNIQUE.DATA

This routine lists the engineer technique data.

ROUTINE EN.LOAD.AND.LIST.DEFENSIVE.POSITION.DATA

This routine loads and lists the defensive position data during initialization.

ROUTINE EN.LOAD.AND.LIST.ENGINEER.DATA

This routine loads the engineer input data during initialization, processes the engineer work specified by input data, and initiates the self-scheduling event EN.JOB.UPDATE.

ROUTINE EN.LOAD.ASSET.DATA

This routine loads the engineer asset data during initialization.

ROUTINE EN.LOAD.ENGINEER.COMMAND.LINKS

This routine loads the engineer command links that allow each staff unit to be linked to another engineer unit for the purpose of reallocating assets and supplies.

ROUTINE EN.LOAD.JOB.PRIORITY.DATA

This routine loads engineer job prioritization data during initialization. This data allows the scenario builder to determine how an engineer staff unit will prioritize the jobs within its command chain. This feature is not currently implemented.

ROUTINE EN.LOAD.TECHNIQUE.DATA

This routine loads the engineer technique data, preprocesses this data to allow quick searches, and builds the staff engineer asset and technique arrays.

ROUTINE EN.MISSION.DUMP
GIVEN .MISSION, .SIDE, .REQUESTED

This routine produces a mission status report for one of the interactive menu options of EN.PRINT.CONTROL if .REQUESTED is SS.THE.SCREEN and an output mission report if .REQUESTED is EN.ENGINEER.HISTORY.FILE.

ROUTINE EN.MOVE.DOWN.COMMAND.CHAIN
GIVEN .SUPERIOR, .TECH, .X.LOC, .Y.LOC
YIELDING .SUBORDINATE

This routine moves one step down the command chain from .SUPERIOR looking for the engineer HQ subordinate closest to the site with the site in its tactical area. It returns the GG.UNIT number of such a subordinate if it exists, otherwise, it is 0.

ROUTINE EN.ORDER.E.'GINEER.WORK
GIVEN .SIDE, .MISSION, .FLAG
YIELDING .OUTCOME, .HQ

This is the transition routine for moving an engineer mission to engineer jobs. If the work is impossible or unassignable, .OUTCOME indicates that and .HQ is 0. If the work involved in mission is assignable, then this routine creates a job for each mission feature in .MISSION, sets the job attributes, and assigns the jobs to an engineer HQ unit, .HQ. The .FLAG indicates that this is or is not a job that needs immediate processing and is determined by the calling routine. If .FLAG is yes, then a higher priority is assigned to the job at this point.

ROUTINE EN.PRINT.CONTROL

This routine provides gamer prints of engineer-related data.

ROUTINE EN.PRINT.FINAL.REPORT

This routine prints all necessary output to the engineer history file at the end of the simulation.

ROUTINE EN.PRINT.TASK.MENU

This routine displays the list of task names and the numbers corresponding to them.

ROUTINE EN.PRIORITIZE.INDIVIDUAL.JOB
GIVEN .JOB

This routine assigns a priority level for the given .JOB and files the job in the responsible engineer HQ unit's list of pending jobs by that priority level. This action occurs when a job is added to a pending job list after the initial jobs have been processed and jobs are coming in one at a time.

ROUTINE EN.PRIORITIZE.JOB.LIST
GIVEN .STAFF

This routine prioritizes the entire list of engineer jobs for .STAFF engineer and its command chain and then sorts all subordinate job lists by this prioritization. This is done at initialization after all data-specified jobs have been assigned and, after that, each time the combat status or tactical area of the staff unit's supported unit changes. The major work on this area is planned for the second half of EMIP; this routine is a simplified method for handling the prioritization.

ROUTINE EN.PROCESS.JOB
GIVEN .THIS.JOB

This routine takes a job from the pending list, attempts to find the assets and supplies needed for its accomplishment, and, if found, creates the work units needed to bring the equipment together and move to the work site.

ROUTINE EN.PROCESS.MISSION.LISTS

This routine periodically examines the mission list for each side, removing missions that are assignable.

ROUTINE EN.PROCESS.PENDING.LIST
GIVEN .HQ

This routine processes the entire list of .HQ's pending jobs. This task is done whenever there is a need to attempt starting new work: when equipment returns from an earlier job or completes a rest period and when a new job is added to the list.

ROUTINE EN.PROCESS.THIS.MISSION.NOW
GIVEN .SIDE, .MISSION

This routine processes dynamically generated missions that must be processed immediately.

ROUTINE EN.READ.FEATURE.NAME
GIVEN .TASK.TYPE
YIELDING .FEATURE.TYPE

When technique data is input, this routine reads the feature name and associates with it the corresponding feature prototype number. This will vary by the task type of the technique, so several cases must be handled.

ROUTINE EN.RECONCILE.ASSET.COUNTS
GIVEN .UNIT

This routine accounts for engineer attrition. Engineer assets are also weapons, and the FL module sees them only as weapons when it computes attrition. To bring the engineer bookkeeping in line with the weapon counts, this routine must compare totals and adjust accordingly. .UNIT is the GG.UNIT number of the engineer unit. This routine is called when attrition is assessed to engineer units.

ROUTINE EN.REGISTER.EFFECT.OF.JOB
GIVEN .JOB, .COMPLETED.EFFECT

When a job has reached and completed its final segment or has been interrupted at an intermediate segment with the potential that no more work will be done on it, the effect of the work to date is registered by this routine. The .COMPLETED.EFFECT is the portion completed and must be between 0 and 1.

EVENT EN.REGISTER.EFFECT.OF.NON.ENGINEER.JOB
GIVEN .TASK, .FEATURE, .TECH, .X.LOC, .Y.LOC, .UNIT

To register the effect of work done with a nonengineer unit's organic assets, this event is scheduled to occur at a time computed by using the engineer technique data.

ROUTINE EN.REMOVE.MINEFIELD
GIVEN .MINEFIELD

This routine removes a defunct minefield from its grid squares and from the EN.SET.OF.MINEFIELDS but does not destroy the minefield, so structures referencing the minefield are still valid.

ROUTINE EN.REQUEST.ENGINEER.PREPARATION
GIVEN .UNIT, .PATH.POINT

This routine creates an engineer mission to prepare a defensive position for .UNIT at .PATH.POINT.

ROUTINE EN.REQUEST.ENGINEER.SUPPORT
GIVEN .UNIT, .TASKTYPE

This routine issues a request for engineer support by requesting a mission of the given .TASK.TYPE.

ROUTINE EN.RESERVE.A.TECH.ARRAY
YIELDING .TECH.ARRAY

This routine builds a ragged array dimensioned by the number of tasks by the number of feature types. It is needed because the number of feature types for each type of task varies with each scenario. It is used whenever there is a need to reserve a technique array.

ROUTINE EN.SEND.WORK.CREW.HOME
GIVEN .WT

This routine breaks up a work team .WT owning equipment from more than one HQ unit by transferring all of the equipment owned by each of the HQ units to another work team and sending the newly created work team to the owning HQ's location.

ROUTINE EN.TRANSFER.ASSETS.HQ.TO.HQ
GIVEN .UNIT1, .UNIT2, .AMOUNT, .WPN.TYPE

This routine does all of the bookkeeping involved in transferring engineer assets from .UNIT1 TO .UNIT2. .AMOUNT specifies the number of assets being transferred. .WPN.TYPE specifies the weapon type being transferred.

ROUTINE EN.TRANSFER.ASSETS.HQ.TO.WT
GIVEN .HQ, .WT, .AMOUNT, .TYPE, .FUNCTION

This routine does all of the bookkeeping involved in transferring engineer assets from the owning .HQ to .WT. .TYPE specifies the engineer asset prototype being transferred, .AMOUNT specifies the number, and .FUNCTION specifies the function that this equipment serves. Only ..AVAILABLE assets at the .HQ can be transferred with this routine. Assets already assigned to a WT unit must be transferred with EN.TRANSFER.ASSETS.WT.TO.WT.

ROUTINE EN.TRANSFER.ASSETS.WT.TO.HQ
GIVEN .WT

This routine does the bookkeeping needed to transfer all of the engineer assets held by the work unit .WT to the owning .HQ.

ROUTINE EN.TRANSFER.ASSETS.WT.TO.WT
GIVEN .WT1, .WT2, .AMOUNT, .TYPE, .FN

This routine does all of the bookkeeping involved in transferring .AMOUNT of engineer assets of a .TYPE to perform function .FN owned by one work team to another, i.e., from .WT1 to .WT2. This routine is used when engineer assets originally at two locations are brought together in one unit to work on a particular job.

ROUTINE EN.TRANSFER.JOB.SUPPLIES.TO.WT
GIVEN JOB, .PHASE, .UNIT, .IN.WT
YIELDING .WT

This routine does the bookkeeping needed to transfer supplies required for the given .PHASE of the given .JOB from the given .UNIT to a work team. If the phase is just starting, and no work teams exist, .IN.WT is 0 and .WT is created here. When equipment is being moved to the next job in a mission, .IN.WT is a valid work team, which becomes .WT and receives the supplies for subsequent jobs.

ROUTINE EN.TRANSFER.SUPPLIES.FOR.EQUIPMENT
GIVEN .UNIT1, .UNIT2, .WG.PROTO, .WG.AMOUNT

This routine does all of the bookkeeping involved in transferring the fuel and ammunition requirements for equipment that has just been transferred from .UNIT1 to .UNIT2. .WG.PROTO is the weapon group type of the equipment, and .WG.AMOUNT is the change in the FL.STRENGTH on which the supply amounts are based.

ROUTINE EN.TRANSFER.THIS.ASSET.WT.TO.WT
GIVEN .WT1, .WT2, .WT.ASSET

This routine does all of the bookkeeping involved in transferring a particular engineer asset, .WT.ASSET, from one work team .WT1 to another .WT2. This action occurs when assets need to be moved to a new work site while .WT1 stays at its present site and when .WT1 is breaking up and needs to send all of its assets to their owning HQ units.

ROUTINE EN.UPDATE.ALL.DEFENSIVE.PREPARATIONS.IN.PROGRESS

This routine updates all ongoing defensive preparations being done by nonengineer units.

ROUTINE EN.UPDATE.ARRAYS.AND.SEND.EXCESS.ON
GIVEN .WT

At the completion of a segment, the phase arrays are adjusted to reflect the numbers of equipment by type and function required to complete the phase. When a job's technique specifies that the organization point is at the work site, equipment can leave the work site as soon as its last use-segment is completed without waiting for phase completion. This routine locates the equipment that is no longer needed and routes it to the next job or to its owning HQ unit.

ROUTINE EN.UPDATE.ASSET.REPORT
GIVEN .ASSET

This routine prints a standard format report to the engineer history file so that an .ASSET's .STATUS can be traced. Because of the possible impact on run time when a large number of engineer assets are played, engineer input allows this report to be for specified asset types.

ROUTINE EN.UPDATE.JOB.REPORT
GIVEN .JOB, .STATUS, .INFO

This routine prints a standard format report to the engineer history file so that the progress of .JOB can be traced. .JOB is the job pointer, .STATUS is a defined-to-mean integer indicating how far the job has progressed, and .INFO is a real argument which allows information that is not a part of the job's attributes to be passed to this routine.

ROUTINE EN.WT.ARRIVAL.AT.DESTINATION
GIVEN .WT.GG

When an engineer work team arrives at its destination path point, three situations may exist:

1. The unit is at its work site

2. The unit has returned to its .HQ after a job

3. The unit is rendezvousing with other work teams before proceeding to work.

This routine analyzes the situation and instigates the appropriate action. The given argument .WTGG is the GG.UNIT number of the work team.

ROUTINE EN.WT.DUMP
GIVEN .WT

This routine produces a work team status report for one of the interactive menu options of EN.PRINT.CONTROL.

7 ENGINEER MODULE CALLING TREE

The order of the routine calls in the new engineer module is summarized in the following set of calling trees. After a sequence has been given in full, only the first routine is listed in subsequent calls.

```
SS.LOAD.THE.DATABASE
   EN.LOAD.AND.LIST.ENGINEER.DATA
   .  SS.POSITION.INPUT.FILE
   .  EN.LOAD.AND.LIST.DEFENSIVE.POSITION.DATA
   .  EN.LOAD.JOB.PRIORITY.DATA
   .  EN.LIST.JOB.PRIORITY.DATA
   .  EN.LOAD.ASSET.DATA
   .  :  FL.READ.NAME.OF.WPN
   .  EN.LIST.ASSET.DATA
   .  EN.CREATE.HQ.UNITS
   .  :  EN.UPDATE.ASSET.REPORT
   .  EN.LOAD.ENGINEER.COMMAND.LINKS
   .  :  GG.READ.UNIT.NAME
   .  :  EN.FIND.STAFF.ENGINEER
   .  :  EN.FIND.ENGINEER.HQ
   .  :  SS.TRACE.AND.ABORT
   .  EN.LIST.ENGINEER.COMMAND.LINKS
   .  EN.LOAD.TECHNIQUE.DATA
   .  :  SS.TRACE.AND.ABORT
   .  :  EN.READ.FEATURE.NAME
   .  :  .  SS.TRACE.AND.ABORT
   .  :  EN.INITIALIZE.TASK.TECH.ARRAY
   .  :  EN.RESERVE.A.TECH.ARRAY
   .  :  EN.BUILD.STAFF.ARRAYS
   .  :  .  EN.ADD.AVAILABLE.ASSETS
   .  :  .  :  EN.FIND.ENGINEER.HQ
   .  :  .  :  EN.ADD.AVAILABLE.ASSETS
   .  :  .  EN.COMPARE.OWNED.TO.REQUIRED
   .  :  .  :  EN.DETERMINE.FUNCTION.ARRAY
   .  :  .  EN.DOES.THIS.COMMAND.HAVE.SUPPLIES
   .  :  .  :  EN.CHECK.FOR.SUPPLY.TYPES.IN.INVENTORY
   .  :  .  :  EN.DOES.THIS.COMMAND.HAVE.SUPPLIES
   .  EN.LIST.TECHNIQUE.DATA
   .  :  GG.PRINT.UNIT.LEVEL
   .  TB.CALCULATE.GRID.SQUARES.FOR.THIS.LINE.SEG

SS.SCHEDULE.INITIAL.EVENTS for initial scheduling
   EN.JOB.UPDATE
   .  EN.PROCESS.MISSION.LISTS
   .  :  EN.ORDER.ENGINEER.WORK
```

```
. : . EN.ASSIGN.ENGINEER.JOB
. : . : EN.FIND.STAFF.ENGINEER
. : . : EN.IS.THIS.STAFF.UNIT.CAPABLE
. : . : . UT.IS.POINT.INSIDE.POLYGON
. : . : . GG.COMPUTE.DISTANCE.TO.FEBA
. : . : . EN.COMPARE.OWNED.TO.REQUIRED
. : . : . ** see above
. : . : . EN.DOES.THIS.COMMAND.HAVE.SUPPLIES
. : . : . ** see above
. : . : EN.DETERMINE.JOB.LOAD
. : . : . EN.FIND.ENGINEER.HQ
. : . : . EN.DETERMINE.JOB.LOAD
. : . : EN.MOVE.DOWN.COMMAND.CHAIN
. : . : . EN.FIND.ENGINEER.HQ
. : . : . EN.CHECK.FOR.SUPPLY.TYPES.IN.INVENTORY
. : . : . UT.IS.POINT.INSIDE.POLYGON
. : . : EN.FIND.ENGINEER.HQ
. : . EN.HOW.MUCH.TIME.IS.NEEDED
. : . EN.PRIORITIZE.INDIVIDUAL.JOB
. : . : GG.COMPUTE.DISTANCE.TO.FEBA
. : . : EN.COMPUTE.DISTANCE.TO.JOB
. : . : EN.FIND.STAFF.FOR.THIS.HQ
. : . : . EN.FIND.STAFF.ENGINEER
. : . : . SS.TRACE.AND.ABORT
. : . EN.UPDATE.JOB.REPORT
. : EN.MISSION.DUMP
. : . EN.GET.TASK.NAME
. : EN.PRIORITIZE.JOB.LIST
. : . EN.ADD.JOBS.TO.LIST
. : . : EN.FIND.ENGINEER.HQ
. : . : EN.ADD.JOBS.TO.LIST
. : . EN.PRIORITIZE.INDIVIDUAL.JOB
. : . ** see above
. : EN.PROCESS.JOB
. : . EN.FIND.STAFF.FOR.THIS.HQ
. : . ** see above
. : . EN.IS.THIS.STAFF.UNIT.CAPABLE
. : . ** see above
. : . EN.HOW.MUCH.TIME.IS.NEEDED
. : . EN.COMPUTE.DISTANCE.TO.JOB
. : . EN.UPDATE.JOB.REPORT
. : . EN.DISCONTINUE.ENGINEER.MISSION
. : . : EN.FIND.ENGINEER.HQ
. : . : SS.CANCEL.EVENT.FOR.UNIT.IF.SCHEDULED
. : . : EN.UPDATE.ASSET.REPORT
. : . : EN.SEND.WORK.CREW.HOME
```

```
. : . : . EN.CREATE.A.WORK.UNIT
. : . : . : GG.COMPUTE.DISTANCE.TO.FEBA
. : . : . : TB.SET.TERRAIN.ATTRIBUTES.OF.UNIT
. : . : . EN.TRANSFER.THIS.ASSET.WT.TO.WT
. : . : . : SS.TRACE.AND.ABORT
. : . : . : EN.FILE.FREE.WEAPON.GROUP
. : . : . : EN.TRANSFER.SUPPLIES.FOR.EQUIPMENT
. : . : . : . LO.FIND.STOCK.POINTER
. : . : . : . LO.UPDATE.UNITS.SUPPLIES
. : . : . EN.DETERMINE.WT.UNIT.PATH
. : . : . : EN.COMPUTE.DISTANCE.TO.JOB
. : . : . : EN.FIND.SHORTEST.PATH.BETWEEN.TWO.POINTS
. : . : . : . EN.FIND.OBSTACLE.BETWEEN.TWO.POINTS
. : . : . : . : EN.FIND.BYPASS.FOR.LINE.OBSTACLE
. : . : . : . EN.FIND.SHORTEST.PATH.BETWEEN.TWO.POINTS
. : . : . : FL.MOUNT.OR.DISMOUNT.A.UNIT
. : . : . : GG.MAP.FROM.GROUND.COMBAT.STATUS
. : . : . : SS.CANCEL.EVENT.FOR.UNIT.IF.SCHEDULED
. : . : . : GG.END.OF.WAIT
. : . : . : EN.COMPUTE.WORK.TEAM.SPEED
. : . : . : . EN.FIND.ENGINEER.HQ
. : . : . EN.UPDATE.ASSET.REPORT
. : . : . EN.DISSOLVE.WT.UNIT
. : . : . : SS.CANCEL.EVENT.FOR.UNIT.IF.SCHEDULED
. : . : . : SS.TRACE.AND.ABORT
. : . : . : EN.FILE.FREE.UNIT.IN.POOL
. : . : . : . GG.DELETE.THE.PATH
. : . : . : . TB.SET.TERRAIN.ATTRIBUTES.OF.UNIT
. : . : . : . SS.TRACE.AND.ABORT
. : . : . EN.IS.WT.AT.BASE
. : . : . EN.TRANSFER.ASSETS.WT.TO.HQ
. : . : . : EN.END.OF.REST.PERIOD
. : . : . : . EN.FIND.ENGINEER.HQ
. : . : . : . EN.UPDATE.ASSET.REPORT
. : . : . : EN.UPDATE.ASSET.REPORT
. : . : . : SS.TRACE.AND.ABORT
. : . : . : LO.MERGE.WITH.ANOTHER.UNIT
. : . : . : EN.FILE.FREE.WEAPON.GROUP
. : . : . : FL.MOUNT.OR.DISMOUNT.A.UNIT
. : . : EN.DETERMINE.WT.UNIT.PATH
. : . : ** see above
. : . : EN.DISSOLVE.WT.UNIT
. : . : ** see above
. : . : EN.REGISTER.EFFECT.OF.JOB
. : . : . EN.REMOVE.MINEFIELD
. : . : . : TB.CALCULATE.GRID.SQUARES.FOR.THIS.LINE.SEG
```

```
. : . : . GM.DETERMINE.A.UNIT.VELOCITY
. : . : . TB.CALCULATE.GRID.SQUARES.FOR.THIS.LINE.SEG
. : . : EN.UPDATE.JOB.REPORT
. : . EN.DETERMINE.REQUIRED.EQUIPMENT.ARRAYS
. : . EN.ADD.AVAILABLE.ASSETS
. : . ** see above
. : . EN.COMPARE.OWNED.TO.REQUIRED
. : . ** see above
. : . EN.CHECK.FOR.AVAILABILITY.OF.JOB.SUPPLIES
. : . EN.FORM.INITIAL.WT.UNITS
. : . : EN.BUILD.WORK.UNIT.AT.HQ
. : . : . EN.TRANSFER.JOB.SUPPLIES.TO.WT
. : . : . : EN.CREATE.A.WORK.UNIT
. : . : . : ** see above
. : . : . : LO.FIND.STOCK.POINTER
. : . : . : LO.UPDATE.UNITS.SUPPLIES
. : . : . EN.CREATE.A.WORK.UNIT
. : . : . ** see above
. : . : . EN.TRANSFER.ASSETS.HQ.TO.WT
. : . : . : SS.TRACE.AND.ABORT
. : . : . : EN.UPDATE.ASSET.REPORT
. : . : . : EN.TRANSFER.SUPPLIES.FOR.EQUIPMENT
. : . : . : . LO.FIND.STOCK.POINTER
. : . : . : . LO.UPDATE.UNITS.SUPPLIES
. : . : . : FL.MOUNT.OR.DISMOUNT.A.UNIT
. : . : . EN.DETERMINE.WT.UNIT.PATH
. : . : . ** see above
. : . : . EN.FIND.ENGINEER.HQ
. : . : . EN.BUILD.WORK.UNIT.AT.HQ
. : . : EN.FIND.ENGINEER.HQ
. : . : EN.BUILD.WORK.UNIT.AT.WT
. : . : . EN.CREATE.A.WORK.UNIT
. : . : . ** see above
. : . : . EN.TRANSFER.ASSETS.WT.TO.WT
. : . : . : SS.TRACE.AND.ABORT
. : . : . : EN.FILE.FREE.WEAPON.GROUP
. : . : . : EN.TRANSFER.SUPPLIES.FOR.EQUIPMENT
. : . : . : . LO.FIND.STOCK.POINTER
. : . : . : . LO.UPDATE.UNITS.SUPPLIES
. : . : . EN.DETERMINE.WT.UNIT.PATH
. : . : . ** see above
. : . : . EN.UPDATE.JOB.REPORT
. : . : . LO.MERGE.WITH.ANOTHER.UNIT
. : . : . EN.DISSOLVE.WT.UNIT
. : . : . ** see above
. : . : . EN.FIND.ENGINEER.HQ
```

```
.  :  .  :  .  EN.BUILD.WORK.UNIT.AT.WT
.  :  .  EN.TRANSFER.JOB.SUPPLIES.TO.WT
.  :  .  :  EN.CREATE.A.WORK.UNIT
.  :  .  :  ** see above
.  :  .  :  LO.FIND.STOCK.POINTER
.  :  .  :  LO.UPDATE.UNITS.SUPPLIES
.  :  EN.PROCESS.PENDING.LIST
.  :  .  EN.PROCESS.JOB
.  :  .  ** see above
.  EN.JOB.UPDATE

GG.PATH.POINT.ARRIVAL
   EN.WT.ARRIVAL.AT.DESTINATION
   .  EN.FIND.ENGINEER.HQ
   .  EN.ARE.WTS.COLLOCATED
   .  EN.TRANSFER.THIS.ASSET.WT.TO.WT
   .  **  ee above
   .  LO.MERGE.WITH.ANOTHER.UNIT
   .  EN.DISSOLVE.WT.UNIT
   .  ** see above
   .  EN.BEGIN.CURRENT.JOB.SEGMENT
   .  :  EN.CALCULATE.NEW.FINISH.TIME
   .  :  GM.SET.UNITS.WEATHER.TRAFFICABILITY.LEVEL
   .  :  SS.CANCEL.EVENT.FOR.UNIT.IF.SCHEDULED
   .  :  EN.END.OF.JOB.SEGMENT
   .  :  .  EN.UPDATE.JOB.REPORT
   .  :  .  LO.UPDATE.UNITS.SUPPLIES
   .  :  .  EN.UPDATE.ARRAYS.AND.SEND.EXCESS.ON
   .  :  .  :  EN.UPDATE.ASSET.REPORT
   .  :  .  :  EN.CREATE.A.WORK.UNIT
   .  :  .  :  ** see above
   .  :  .  :  EN.TRANSFER.THIS.ASSET.WT.TO.WT
   .  :  .  :  ** see above
   .  :  .  :  EN.DETERMINE.WT.UNIT.PATH
   .  :  .  :  ** see above
   .  :  .  :  EN.SEND.WORK.CREW.HOME
   .  :  .  :  ** see above
   .  :  .  EN.BEGIN.CURRENT.JOB.SEGMENT
   .  :  .  EN.CLOSE.OUT.JOB.PHASE
   .  :  .  :  EN.UPDATE.JOB.REPORT
   .  :  .  :  EN.REGISTER.EFFECT.OF.JOB
   .  :  .  :  ** see above
   .  :  .  :  EN.GET.TASK.NAME
   .  :  .  :  SS.TRACE.AND ABORT
   .  :  .  :  EN.UPDATE.ASSET.REPORT
   .  :  .  :  EN.DETERMINE.WT.UNIT.PATH
```

```
.  :  .  :  ** see above
.  :  .  :  EN.SEND.WORK.CREW.HOME
.  :  .  :  ** see above
.  :  .  EN.PROCESS.JOB
.  :  .  ** see above
.  :  .  EN.IS.WT.AT.WORK.SITE
.  :  EN.UPDATE.JOB.REPORT
.  :  EN.UPDATE.ASSET.REPORT
.  :  EN.DISCONTINUE.ENGINEER.MISSION
.  :  ** see above
.  EN.DETERMINE.WT.UNIT.PATH
.  ** see above
.  EN.UPDATE.ASSET.REPORT
.  EN.SEND.WORK.CREW.HOME
.  ** see above
.  SS.TRACE.AND.ABORT

AG.AREA.DAMAGE
AG.POINT.DAMAGE
AT.AREA.DAMAGE
AT.POINT.DAMAGE
CH.CHANGE.MASS.OF.UNIT
FL.UPDATE.MASS.OF.UNIT
GG.CHANGE.MASS.OF.UNIT
GG.CHECK.FOR.BREAK
MF.ACCOUNT.FOR.MINEFIELD.CROSSING.LOSSES
MF.ACCOUNT.FOR.MINEFIELD.DISCOVERY.LOSSES
RD.ALLOCATE.WPN.REPAIRS
RD.DETERMINE.RAM.FAILURES
   EN.RECONCILE.ASSET.COUNTS
   .  EN.FIND.STAFF.FOR.THIS.HQ
   .  :  ** see above
   .  EN.UPDATE.ASSET.REPORT
   .  EN.FIND.ENGINEER.HQ
   .  RD.PLACE.LOSSES.IN.RECOVERY.SYSTEM
   .  LO.UPDATE.UNIT.SUPPLIES.DUE.TO.LOSS
   .  EN.FILE.FREE.WEAPON.GROUP
   .  EN.UPDATE.JOB.REPORT
   .  EN.DISCONTINUE.ENGINEER.MISSION
   .  ** see above
   .  EN.CALCULATE.NEW.FINISH.TIME
   .  SS.TRACE.AND.ABORT
   .  EN.BUILD.STAFF.ARRAYS
   .  ** see above

GG.FILL.THE.PATH
```

```
   EN.REQUEST.ENGINEER.PREPARATION
   DT.PERFORM.ACTION
   EN.REQUEST.ENGINEER.SUPPORT
   .  EN.CONSTRUCT.A.MINEFIELD.MISSION
   .  :  MF.CREATE.A.MINEFIELD
   .  :  LO.UPDATE.UNITS.AMMO.SUPPLIES
   .  EN.CONSTRUCT.AN.OBSTACLE.MISSION

Future expansion
   EN.ASSESS.ENGINEER.SUPPORT.CAPABILITY
   .  EN.FIND.STAFF.ENGINEER
   .  EN.ADD.AVAILABLE.ASSETS
   .  ** see above
   .  EN.COMPARE.OWNED.TO.REQUIRED
   .  ** see above
   .  EN.CHECK.FOR.AVAILABILITY.OF.JOB.SUPPLIES

MF.UNIT.HITS.MINEFIELD
   EN.ASSESS.NON.ENGINEER.CAPABILITY
   .  EN.COMPARE.OWNED.TO.REQUIRED
   .  ** see above
   .  EN.CHECK.FOR.AVAILABILITY.OF.JOB.SUPPLIES

GX.DISPLAY.BARRIERS.ON.RAMTEK
TB.CROSSING FEATURE
   EN.CREATE.A.MISSION.REQUEST
   .  EN.PROCESS.THIS.MISSION.NOW
   .  :  EN.ORDER.ENGINEER.WORK
   .  :  ** see above
   .  :  EN.PROCESS.PENDING.LIST
   .  :  ** see above

GM.HANDLE.DELAY.FROM.INDIRECT.FIRE
   EN.DELAY.ENGINEERS.IF.AT.TASK.SITE

FL.SHIFT.EQUIPMENT.FROM.UNIT.TO.UNIT
   EN.TRANSFER.ASSETS.HQ.TO.HQ
   .  EN.FIND.ENGINEER.HQ
   .  FL.MOUNT.OR.DISMOUNT.A.UNIT
   .  PT.WRITE.TYPE.10.RECORD
   .  FL.PRINT.A.UNITS.WPN.GROUP.ATTRIBUTES
   .  EN.FIND.STAFF.FOR.THIS.HQ
   .  ** see above
   .  EN.BUILD.STAFF.ARRAYS
   .  ** see above
```

```
SS.GAMER.CONTROL
   EN.PRINT.CONTROL
   .  EN.PRINT.TASK.MENU
   .  :  EN.GET.TASK.NAME
   .  EN.MISSION.DUMP
   .  ** see above
   .  UT.SELECT.COLOR
   .  GR.DRAW.CIRCLE
   .  GR.DRAW.ORIENTED.RECTANGLE
   .  GR.MOVE
   .  GR.DRAW
   .  GR.MAP
   .  UT.DRAW.CIRCLE
   .  GR.DRAW.RECTANGLE
   .  GR.WRITE.TEXT
   .  UT.FLUSH.GRAPHICS.OUTPUT
   .  SS.READ.UNIT.NUMBER
   .  EN.FIND.ENGINEER.HQ
   .  EN.HQ.DUMP
   .  :  EN.GET.TASK.NAME
   .  EN.WT.DUMP
   .  :  GM.PRINT.CMBT.STATUS
   .  :  EN.GET.TASK.NAME
   .  EN.JOB.DUMP
   .  :  EN.GET.TASK.NAME

Future expansion
   EN.COMPUTE.ENGINEERS.TRUE.STRENGTH
   .  EN.FIND.ENGINEER.HQ

SS.STOP.SIMULATION
   EN.PRINT.FINAL.REPORT
   .  EN.MISSION.DUMP
   .  ** see above
```

8 ENGINEER PREAMBLE

The data structures designed for the new engineer module are declared in two sections of the VIC PREAMBLE. These two sections are reproduced here so that the reader can verify entity attributes, set relationships, modes, and define-to-means.

```
''                          *************************
''                          *     ENGINEERS (EN)    *
''                          *************************

''     "IN WORD" ASSIGNMENT RANGE 2930-2995

    NORMALLY MODE IS INTEGER
    NORMALLY DIMENSION IS 0

      THE SYSTEM
         HAS A EN.UPDATE.INTERVAL                              IN WORD 2929
         HAS A EN.MISSION.COUNT                                IN WORD 2930
         HAS A EN.TECHNIQUE.ARRAY                              IN WORD 2931
         HAS A EN.ASSET.REPORT.ARRAY                           IN WORD 2932
         HAS A F.EN.SET.OF.FREE.WPN.GROUPS                     IN WORD 2933
         HAS A L.EN.SET.OF.FREE.WPN.GROUPS                     IN WORD 2934
         HAS A N.EN.SET.OF.FREE.WPN.GROUPS                     IN WORD 2935
         AND OWNS A EN.SET.OF.FREE.WPN.GROUPS

         DEFINE EN.UPDATE.INTERVAL AS A REAL VARIABLE
         DEFINE EN.TECHNIQUE.ARRAY AS A 4-DIMENSIONAL INTEGER ARRAY
         DEFINE EN.ASSET.REPORT.ARRAY AS A 1-DIMENSIONAL INTEGER ARRAY

      EVERY SS.SIDE
         HAS A EN.MAX.FREE.WORK.TEAMS                          IN WORD 2936
         HAS A F.EN.SET.OF.FREE.UNITS                          IN WORD 2937
         HAS A L.EN.SET.OF.FREE.UNITS                          IN WORD 2938
         HAS A N.EN.SET.OF.FREE.UNITS                          IN WORD 2939
         HAS A F.EN.SET.OF.HQ.UNITS                            IN WORD 2940
         HAS A L.EN.SET.OF.HQ.UNITS                            IN WORD 2941
         HAS A N.EN.SET.OF.HQ.UNITS                            IN WORD 2942
         HAS A F.EN.SET.OF.STAFF.UNITS                         IN WORD 2943
         HAS A L.EN.SET.OF.STAFF.UNITS                         IN WORD 2944
         HAS A N.EN.SET.OF.STAFF.UNITS                         IN WORD 2945
         HAS A F.EN.SET.OF.MISSIONS                            IN WORD 2946
         HAS A L.EN.SET.OF.MISSIONS                            IN WORD 2947
         HAS A N.EN.SET.OF.MISSIONS                            IN WORD 2948
         HAS A F.EN.SET.OF.PROPOSED.MINEFIELDS                 IN WORD 2949
         HAS A L.EN.SET.OF.PROPOSED.MINEFIELDS                 IN WORD 2950
         HAS A N.EN.SET.OF.PROPOSED.MINEFIELDS                 IN WORD 2951
         HAS A F.EN.SET.OF.PROPOSED.OBSTACLES                  IN WORD 2952
         HAS A L.EN.SET.OF.PROPOSED.OBSTACLES                  IN WORD 2953
         HAS A N.EN.SET.OF.PROPOSED.OBSTACLES                  IN WORD 2954
         HAS A F.EN.SET.OF.DEFENSIVE.POSITIONS                 IN WORD 2955
         HAS A L.EN.SET.OF.DEFENSIVE.POSITIONS                 IN WORD 2956
```

```
      HAS A N.EN.SET.OF.DEFENSIVE.POSITIONS                IN WORD 2957
      HAS A F.EN.SET.OF.RETRIEVABLE.ASSETS                 IN WORD 2958
      HAS A L.EN.SET.OF.RETRIEVABLE.ASSETS                 IN WORD 2959
      HAS A N.EN.SET.OF.RETRIEVABLE.ASSETS                 IN WORD 2960
      OWNS A EN.SET.OF.FREE.UNITS
      OWNS A EN.SET.OF.HQ.UNITS
      OWNS A EN.SET.OF.STAFF.UNITS
      OWNS A EN.SET.OF.MISSIONS
      OWNS A EN.SET.OF.PROPOSED.MINEFIELDS
      OWNS A EN.SET.OF.PROPOSED.OBSTACLES
      OWNS A EN.SET.OF.DEFENSIVE.POSITIONS
      OWNS A EN.SET.OF.RETRIEVABLE.ASSETS

PERMANENT ENTITIES

   EVERY EN.DEF.POS.PROTO
      HAS A EN.DP.PR.NAME                                  IN WORD 2961
      HAS A EN.DP.PR.UNIT.TYPE                             IN WORD 2962
      HAS A EN.DP.PR.UNIT.LEVEL                            IN WORD 2963
      HAS A EN.DP.PR.UNIT.EXPOSURE                         IN WORD 2964

      DEFINE EN.DP.PR.NAME AS A TEXT VARIABLE
      DEFINE EN.DP.PR.UNIT.EXPOSURE AS A REAL VARIABLE

   EVERY EN.ASSET.PROTO
      HAS A EN.ASSET.PR.WP.PTR                             IN WORD 2965
      HAS A EN.ASSET.PR.SPEED                              IN WORD 2966
      HAS A EN.ASSET.DAMAGE.THRESHOLD                      IN WORD 2967
      HAS A EN.ASSET.PR.MAX.WORK.PERIOD                    IN WORD 2968
      HAS A EN.ASSET.PR.REST.PERIOD                        IN WORD 2969

      DEFINE EN.ASSET.PR.SPEED AS A REAL VARIABLE
      DEFINE EN.ASSET.DAMAGE.THRESHOLD AS A REAL VARIABLE
      DEFINE EN.ASSET.PR.MAX.WORK.PERIOD AS A REAL VARIABLE
      DEFINE EN.ASSET.PR.REST.PERIOD AS A REAL VARIABLE

   EVERY EN.FUNCTION
      HAS A EN.FUNCTION.NAME                               IN WORD 2970

      DEFINE EN.FUNCTION.NAME AS A TEXT VARIABLE

   EVERY EN.ASSET.PROTO AND EN.FUNCTION
      HAS A EN.CAPABILITY                                  IN WORD 2971

      DEFINE EN.CAPABILITY AS A REAL VARIABLE

   EVERY EN.TECHNIQUE
      HAS A EN.TECH.SIDE                                   IN WORD 2972
      HAS A EN.TECH.TASK                                   IN WORD 2973
      HAS A EN.TECH.FEATURE.PROTO                          IN WORD 2974
      HAS A EN.TECH.COMMAND.LEVEL                          IN WORD 2975
      HAS A EN.TECH.MIN.DISTANCE.FROM.FEBA                 IN WORD 2976
      HAS A EN.TECH.ORGANIZATION.PT                        IN WORD 2977
```

```
      HAS A EN.TECH.EFFECT                                  IN WORD 2978
      HAS A EN.TECH.ASSET.PTR                               IN WORD 2979
      HAS A EN.TECH.FN.PTR                                  IN WORD 2980
      HAS A F.EN.SET.OF.TECH.SEGMENTS                       IN WORD 2981
      HAS A L.EN.SET.OF.TECH.SEGMENTS                       IN WORD 2982
      HAS A N.EN.SET.OF.TECH.SEGMENTS                       IN WORD 2983
      HAS A F.EN.SET.OF.REQ.EQUIPMENT                       IN WORD 2984
      HAS A L.EN.SET.OF.REQ.EQUIPMENT                       IN WORD 2985
      HAS A N.EN.SET.OF.REQ.EQUIPMENT                       IN WORD 2986
      HAS A F.EN.SET.OF.REQ.SUPPLIES                        IN WORD 2987
      HAS A L.EN.SET.OF.REQ.SUPPLIES                        IN WORD 2988
      HAS A N.EN.SET.OF.REQ.SUPPLIES                        IN WORD 2989
      OWNS A EN.SET.OF.TECH.SEGMENTS
      OWNS A EN.SET.OF.REQ.EQUIPMENT
      OWNS A EN.SET.OF.REQ.SUPPLIES

      DEFINE EN.TECH.MIN.DISTANCE.FROM.FEBA AS A REAL VARIABLE
      DEFINE EN.TECH.EFFECT AS A REAL VARIABLE

TEMPORARY ENTITIES

   EVERY EN.DEFENSIVE.POSITION
      HAS A EN.DP.PROTO                                     IN WORD 1
      HAS A EN.DP.X.LOCATION                                IN WORD 2
      HAS A EN.DP.Y.LOCATION                                IN WORD 3
      HAS A EN.DP.EFFECTIVENESS                             IN WORD 4
      HAS A P.EN.SET.OF.DEFENSIVE.POSITIONS                 IN WORD 5
      HAS A S.EN.SET.OF.DEFENSIVE.POSITIONS                 IN WORD 6
      HAS A M.EN.SET.OF.DEFENSIVE.POSITIONS                 IN WORD 7
      BELONGS TO A EN.SET.OF.DEFENSIVE.POSITIONS

      DEFINE EN.DP.X.LOCATION AS A REAL VARIABLE
      DEFINE EN.DP.Y.LOCATION AS A REAL VARIABLE
      DEFINE EN.DP.EFFECTIVENESS AS A REAL VARIABLE

   EVERY EN.HQ.UNIT
      HAS A EN.HQ.UNIT.ID                                   IN WORD 1
      HAS A EN.HQ.INDEX                                     IN WORD 2
      HAS A EN.HQ.LIST.PROCESS.FLAG                         IN WORD 3
      HAS A EN.HQ.ASSET.PTR                                 IN WORD 4
      HAS A F.EN.HQ.INVENTORY                               IN WORD 5
      HAS A L.EN.HQ.INVENTORY                               IN WORD 6
      HAS A N.EN.HQ.INVENTORY                               IN WORD 7
      HAS A F.EN.SET.OF.HQ.PENDING.JOBS                     IN WORD 8
      HAS A L.EN.SET.OF.HQ.PENDING.JOBS                     IN WORD 9
      HAS A N.EN.SET.OF.HQ.PENDING.JOBS                     IN WORD 10
      HAS A F.EN.SET.OF.HQ.ACTIVE.JOBS                      IN WORD 11
      HAS A L.EN.SET.OF.HQ.ACTIVE.JOBS                      IN WORD 12
      HAS A N.EN.SET.OF.HQ.ACTIVE.JOBS                      IN WORD 13
      HAS A F.EN.SET.OF.PERFORMING.UNITS                    IN WORD 14
      HAS A L.EN.SET.OF.PERFORMING.UNITS                    IN WORD 15
      HAS A N.EN.SET.OF.PERFORMING.UNITS                    IN WORD 16
      HAS A P.EN.SET.OF.HQ.UNITS                            IN WORD 17
```

```
   HAS A S.EN.SET.OF.HQ.UNITS                          IN WORD 18
   HAS A M.EN.SET.OF.HQ.UNITS                          IN WORD 19
   OWNS A EN.HQ.INVENTORY
   OWNS A EN.SET.OF.HQ.PENDING.JOBS
   OWNS A EN.SET.OF.HQ.ACTIVE.JOBS
   OWNS A EN.SET.OF.PERFORMING.UNITS
   BELONGS TO A EN.SET.OF.HQ.UNITS

EVERY EN.STAFF.UNIT
   HAS A EN.STAFF.UNIT.ID                              IN WORD 1
   HAS A EN.STAFF.SUPERIOR                             IN WORD 2
   HAS A EN.STAFF.ASSET.PTR                            IN WORD 3
   HAS A EN.STAFF.TECH.PTR                             IN WORD 4
   HAS A EN.STAFF.RANKING.LEVEL                        IN WORD 5
   HAS A F.EN.SET.OF.STAFF.JOBS                        IN WORD 6
   HAS A L.EN.SET.OF.STAFF.JOBS                        IN WORD 7
   HAS A N.EN.SET.OF.STAFF.JOBS                        IN WORD 8
   HAS A P.EN.SET.OF.STAFF.UNITS                       IN WORD 9
   HAS A S.EN.SET.OF.STAFF.UNITS                       IN WORD 10
   HAS A M.EN.SET.OF.STAFF.UNITS                       IN WORD 11
   OWNS A EN.SET.OF.STAFF.JOBS
   BELONGS TO A EN.SET.OF.STAFF.UNITS

EVERY EN.WT.UNIT
   HAS A EN.WT.UNIT.ID                                 IN WORD 1
   HAS A EN.WT.RESPONSIBLE.HQ                          IN WORD 2
   HAS A EN.WT.ASSET.PTR                               IN WORD 3
   HAS A EN.WT.JOB.PRIORITY                            IN WORD 4
   HAS A EN.WT.MAX.SPEED                               IN WORD 5
   HAS A F.EN.SET.OF.WT.ASSETS                         IN WORD 6
   HAS A L.EN.SET.OF.WT.ASSETS                         IN WORD 7
   HAS A N.EN.SET.OF.WT.ASSETS                         IN WORD 8
   HAS A F.EN.SET.OF.WT.JOBS                           IN WORD 9
   HAS A L.EN.SET.OF.WT.JOBS                           IN WORD 10
   HAS A N.EN.SET.OF.WT.JOBS                           IN WORD 11
   HAS A P.EN.SET.OF.FREE.UNITS                        IN WORD 12
   HAS A S.EN.SET.OF.FREE.UNITS                        IN WORD 13
   HAS A M.EN.SET.OF.FREE.UNITS                        IN WORD 14
   HAS A P.EN.SET.OF.PERFORMING.UNITS                  IN WORD 15
   HAS A S.EN.SET.OF.PERFORMING.UNITS                  IN WORD 16
   HAS A M.EN.SET.OF.PERFORMING.UNITS                  IN WORD 17
   OWNS A EN.SET.OF.WT.ASSETS
   OWNS A EN.SET.OF.WT.JOBS
   MAY BELONG TO A EN.SET.OF.FREE.UNITS
   MAY BELONG TO A EN.SET.OF.PERFORMING.UNITS

   DEFINE EN.WT.JOB.PRIORITY AS A REAL VARIABLE
   DEFINE EN.WT.MAX.SPEED AS A REAL VARIABLE

EVERY EN.MISSION
   HAS A EN.MISSION.TASK                               IN WORD 1
   HAS A EN.MISSION.ORIG.NBR                           IN WORD 2
   HAS A EN.MISSION.METHOD                             IN WORD 3
```

```
   HAS A EN.MISSION.X.LOC                                IN WORD 4
   HAS A EN.MISSION.Y.LOC                                IN WORD 5
   HAS A EN.MISSION.REQUESTOR                            IN WORD 6
   HAS A EN.MISSION.CREATION.TIME                        IN WORD 7
   HAS A EN.MISSION.REQUIRED.COMPLETION.TIME             IN WORD 8
   HAS A P.EN.SET.OF.MISSIONS                            IN WORD 9
   HAS A S.EN.SET.OF.MISSIONS                            IN WORD 10
   HAS A M.EN.SET.OF.MISSIONS                            IN WORD 11
   HAS A F.EN.SET.OF.MISSION.FEATURES                    IN WORD 12
   HAS A L.EN.SET.OF.MISSION.FEATURES                    IN WORD 13
   HAS A N.EN.SET.OF.MISSION.FEATURES                    IN WORD 14
   OWNS A EN.SET.OF.MISSION.FEATURES
   BELONGS TO A EN.SET.OF.MISSIONS

   DEFINE EN.MISSION.X.LOC AS A REAL VARIABLE
   DEFINE EN.MISSION.Y.LOC AS A REAL VARIABLE
   DEFINE EN.MISSION.CREATION.TIME AS A REAL VARIABLE
   DEFINE EN.MISSION.REQUIRED.COMPLETION.TIME AS A REAL VARIABLE

EVERY EN.MISSION.FEATURE
   HAS A EN.MISSION.FEATURE.PTR                          IN WORD 1
   HAS A P.EN.SET.OF.MISSION.FEATURES                    IN WORD 2
   HAS A S.EN.SET.OF.MISSION.FEATURES                    IN WORD 3
   HAS A M.EN.SET.OF.MISSION.FEATURES                    IN WORD 4
   BELONGS TO A EN.SET.OF.MISSION.FEATURES

EVERY EN.ASSET
   HAS A EN.ASSET.PROTO.ID                               IN WORD 1
   HAS A EN.ASSET.UNIT.ASSIGNMENT                        IN WORD 2
   HAS A EN.ASSET.STATUS                                 IN WORD 3
   HAS A EN.ASSET.UNDAMAGED.PORTION                      IN WORD 4
   HAS A EN.ASSET.LAST.UPDATE                            IN WORD 5
   HAS A EN.ASSET.HOURS.WORK.SINCE.REST                  IN WORD 6
   HAS A EN.ASSET.CURRENT.JOB                            IN WORD 7
   HAS A EN.ASSET.CURRENT.JOB.PRIORITY                   IN WORD 8
   HAS A P.EN.HQ.INVENTORY                               IN WORD 9
   HAS A S.EN.HQ.INVENTORY                               IN WORD 10
   HAS A M.EN.HQ.INVENTORY                               IN WORD 11
   BELONGS TO A EN.HQ.INVENTORY

   DEFINE EN.ASSET.UNDAMAGED.PORTION AS A REAL VARIABLE
   DEFINE EN.ASSET.LAST.UPDATE AS A REAL VARIABLE
   DEFINE EN.ASSET.HOURS.WORK.SINCE.REST AS A REAL VARIABLE
   DEFINE EN.ASSET.CURRENT.JOB.PRIORITY AS A REAL VARIABLE

EVERY EN.RETRIEVABLE.ASSET
   HAS A EN.REA.PROTO                                    IN WORD 1
   HAS A EN.REA.FEATURE                                  IN WORD 2
   HAS A EN.REA.X.LOCATION                               IN WORD 3
   HAS A EN.REA.Y.LOCATION                               IN WORD 4
   HAS A EN.REA.OWNER                                    IN WORD 5
   HAS A EN.REA.INV.ID                                   IN WORD 6
   HAS A P.EN.SET.OF.RETRIEVABLE.ASSETS                  IN WORD 7
```

```
   HAS A S.EN.SET.OF.RETRIEVABLE.ASSETS                    IN WORD 8
   HAS A M.EN.SET.OF.RETRIEVABLE.ASSETS                    IN WORD 9
   BELONGS TO A EN.SET.OF.RETRIEVABLE.ASSETS

   DEFINE EN.REA.X.LOCATION AS A REAL VARIABLE
   DEFINE EN.REA.Y.LOCATION AS A REAL VARIABLE

EVERY EN.WT.ASSET
   HAS A EN.WT.ASSET.PROTO                                 IN WORD 1
   HAS A EN.WT.ASSET.OWNER                                 IN WORD 2
   HAS A EN.WT.ASSET.INV.ID                                IN WORD 3
   HAS A EN.WT.ASSET.FUNCTION                              IN WORD 4
   HAS A P.EN.SET.OF.WT.ASSETS                             IN WORD 5
   HAS A S.EN.SET.OF.WT.ASSETS                             IN WORD 6
   HAS A M.EN.SET.OF.WT.ASSETS                             IN WORD 7
   BELONGS TO A EN.SET.OF.WT.ASSETS

EVERY EN.TECH.SEGMENT
   HAS A EN.SEGMENT.NUMBER                                 IN WORD 1
   HAS A EN.END.SEGMENT.EFFECT                             IN WORD 2
   HAS A EN.INTRA.SEGMENT.EFFECT                           IN WORD 3
   HAS A EN.SEGMENT.DURATION                               IN WORD 4
   HAS A EN.SEGMENT.RATE                                   IN WORD 5
   HAS A EN.SEGMENT.VISIBILITY.FACTOR                      IN WORD 6
   HAS A EN.SEGMENT.WEATHER.FACTOR                         IN WORD 7
   HAS A EN.SEGMENT.COMBAT.FACTOR                          IN WORD 8
   HAS A EN.END.SEGMENT.ACTION                             IN WORD 9
   HAS A P.EN.SET.OF.TECH.SEGMENTS                         IN WORD 10
   HAS A S.EN.SET.OF.TECH.SEGMENTS                         IN WORD 11
   HAS A M.EN.SET.OF.TECH.SEGMENTS                         IN WORD 12
   BELONGS TO A EN.SET.OF.TECH.SEGMENTS

   DEFINE EN.END.SEGMENT.EFFECT AS A REAL VARIABLE
   DEFINE EN.SEGMENT.DURATION AS A REAL VARIABLE
   DEFINE EN.SEGMENT.RATE AS A REAL VARIABLE
   DEFINE EN.SEGMENT.VISIBILITY.FACTOR AS A REAL VARIABLE
   DEFINE EN.SEGMENT.WEATHER.FACTOR AS A REAL VARIABLE
   DEFINE EN.SEGMENT.COMBAT.FACTOR AS A REAL VARIABLE

EVERY EN.EQUIPMENT.USE.SEGMENT
   HAS A EN.EQ.USE.SEGMENT.NBR                             IN WORD 1
   HAS A P.EN.SET.OF.EQUIPMENT.USE.SEGMENTS                IN WORD 2
   HAS A S.EN.SET.OF.EQUIPMENT.USE.SEGMENTS                IN WORD 3
   HAS A M.EN.SET.OF.EQUIPMENT.USE.SEGMENTS                IN WORD 4
   BELONGS TO A EN.SET.OF.EQUIPMENT.USE.SEGMENTS

EVERY EN.REQ.EQUIPMENT
   HAS A EN.EQ.PROTO                                       IN WORD 1
   HAS A EN.EQ.FUNCTION                                    IN WORD 2
   HAS A EN.EQ.AMOUNT                                      IN WORD 3
   HAS A EN.EQ.MINIMUM                                     IN WORD 4
   HAS A EN.EQ.USE                                         IN WORD 5
   HAS A EN.EQ.PREP.TIME                                   IN WORD 6
```

```
   HAS A F.EN.SET.OF.EQUIPMENT.USE.SEGMENTS          IN WORD 7
   HAS A L.EN.SET.OF.EQUIPMENT.USE.SEGMENTS          IN WORD 8
   HAS A N.EN.SET.OF.EQUIPMENT.USE.SEGMENTS          IN WORD 9
   HAS A P.EN.SET.OF.REQ.EQUIPMENT                   IN WORD 10
   HAS A S.EN.SET.OF.REQ.EQUIPMENT                   IN WORD 11
   HAS A M.EN.SET.OF.REQ.EQUIPMENT                   IN WORD 12
   OWNS A EN.SET.OF.EQUIPMENT.USE.SEGMENTS
   BELONGS TO A EN.SET.OF.REQ.EQUIPMENT

   DEFINE EN.EQ.PREP.TIME AS A REAL VARIABLE

EVERY EN.REQ.SUPPLY
   HAS A EN.SUPPLY.PROTO                             IN WORD 1
   HAS A EN.SUPPLY.AMOUNT                            IN WORD 2
   HAS A EN.SUPPLY.UNIT.OF.MEASURE                   IN WORD 3
   HAS A EN.SUPPLY.SEGMENT.USED                      IN WORD 4
   HAS A P.EN.SET.OF.REQ.SUPPLIES                    IN WORD 5
   HAS A S.EN.SET.OF.REQ.SUPPLIES                    IN WORD 6
   HAS A M.EN.SET.OF.REQ.SUPPLIES                    IN WORD 7
   BELONGS TO A EN.SET.OF.REQ.SUPPLIES

   DEFINE EN.SUPPLY.AMOUNT AS A REAL VARIABLE
   DEFINE EN.SUPPLY.UNIT.OF.MEASURE AS A REAL VARIABLE

EVERY EN.JOB
   HAS A EN.JOB.TASK                                 IN WORD 1
   HAS A EN.JOB.FEATURE                              IN WORD 2
   HAS A EN.JOB.FEATURE.PTR                          IN WORD 3
   HAS A EN.JOB.TECHNIQUE                            IN WORD 4
   HAS A EN.JOB.METHOD                               IN WORD 5
   HAS A EN.JOB.SIZE                                 IN WORD 6
   HAS A EN.JOB.PRIORITY                             IN WORD 7
   HAS A EN.JOB.MISSION.CREATION.TIME                IN WORD 8
   HAS A EN.JOB.ACTIVATION.TIME                      IN WORD 9
   HAS A EN.JOB.LATEST.COMPLETION.TIME               IN WORD 10
   HAS A EN.JOB.CURRENT.SEGMENT.START.TIME           IN WORD 11
   HAS A EN.JOB.EST.DURATION                         IN WORD 12
   HAS A EN.JOB.LAST.SEGMENT.COMPLETED               IN WORD 13
   HAS A EN.JOB.LAST.SEGMENT.COMPLETION.TIME         IN WORD 14
   HAS A EN.JOB.LAST.SEGMENT.ASSIGNED                IN WORD 15
   HAS A EN.JOB.X.LOCATION                           IN WORD 16
   HAS A EN.JOB.Y.LOCATION                           IN WORD 17
   HAS A EN.JOB.REQUESTING.UNIT                      IN WORD 18
   HAS A EN.JOB.HQ.UNIT                              IN WORD 19
   HAS A EN.JOB.MISSION                              IN WORD 20
   HAS A EN.JOB.ORIGINAL.MISSION                     IN WORD 21
   HAS A F.EN.SET.OF.ACTIVE.PHASES                   IN WORD 22
   HAS A L.EN.SET.OF.ACTIVE.PHASES                   IN WORD 23
   HAS A N.EN.SET.OF.ACTIVE.PHASES                   IN WORD 24
   HAS A P.EN.SET.OF.HQ.PENDING.JOBS                 IN WORD 25
   HAS A S.EN.SET.OF.HQ.PENDING.JOBS                 IN WORD 26
   HAS A M.EN.SET.OF.HQ.PENDING.JOBS                 IN WORD 27
   HAS A P.EN.SET.OF.HQ.ACTIVE.JOBS                  IN WORD 28
```

```
   HAS A S.EN.SET.OF.HQ.ACTIVE.JOBS                          IN WORD 29
   HAS A M.EN.SET.OF.HQ.ACTIVE.JOBS                          IN WORD 30
   OWNS A EN.SET.OF.ACTIVE.PHASES
   MAY BELONG TO A EN.SET.OF.HQ.PENDING.JOBS
   MAY BELONG TO A EN.SET.OF.HQ.ACTIVE.JOBS

   DEFINE EN.JOB.SIZE AS A REAL VARIABLE
   DEFINE EN.JOB.PRIORITY AS A REAL VARIABLE
   DEFINE EN.JOB.MISSION.CREATION.TIME AS A REAL VARIABLE
   DEFINE EN.JOB.ACTIVATION.TIME AS A REAL VARIABLE
   DEFINE EN.JOB.LATEST.COMPLETION.TIME AS A REAL VARIABLE
   DEFINE EN.JOB.CURRENT.SEGMENT.START.TIME AS A REAL VARIABLE
   DEFINE EN.JOB.EST.DURATION AS A REAL VARIABLE
   DEFINE EN.JOB.LAST.SEGMENT.COMPLETION.TIME AS A REAL VARIABLE
   DEFINE EN.JOB.X.LOCATION AS A REAL VARIABLE
   DEFINE EN.JOB.Y.LOCATION AS A REAL VARIABLE

EVERY EN.ACTIVE.PHASE
   HAS A EN.PHASE.FIRST.SEGMENT                              IN WORD 1
   HAS A EN.PHASE.LAST.SEGMENT                               IN WORD 2
   HAS A EN.PHASE.ASSET.PTR                                  IN WORD 3
   HAS A EN.PHASE.FN.PTR                                     IN WORD 4
   HAS A F.EN.SET.OF.PHASE.WT.UNITS                          IN WORD 5
   HAS A L.EN.SET.OF.PHASE.WT.UNITS                          IN WORD 6
   HAS A N.EN.SET.OF.PHASE.WT.UNITS                          IN WORD 7
   HAS A P.EN.SET.OF.ACTIVE.PHASES                           IN WORD 8
   HAS A S.EN.SET.OF.ACTIVE.PHASES                           IN WORD 9
   HAS A M.EN.SET.OF.ACTIVE.PHASES                           IN WORD 10
   OWNS A EN.SET.OF.PHASE.WT.UNITS
   BELONGS TO A EN.SET.OF.ACTIVE.PHASES

EVERY EN.PHASE.WT.UNIT
   HAS A EN.PHASE.WT.PTR                                     IN WORD 1
   HAS A P.EN.SET.OF.PHASE.WT.UNITS                          IN WORD 2
   HAS A S.EN.SET.OF.PHASE.WT.UNITS                          IN WORD 3
   HAS A M.EN.SET.OF.PHASE.WT.UNITS                          IN WORD 4
   BELONGS TO A EN.SET.OF.PHASE.WT.UNITS

EVERY EN.WT.JOB
   HAS A EN.WT.JOB.PTR                                       IN WORD 1
   HAS A EN.WT.JOB.PHASE                                     IN WORD 2
   HAS A P.EN.SET.OF.WT.JOBS                                 IN WORD 3
   HAS A S.EN.SET.OF.WT.JOBS                                 IN WORD 4
   HAS A M.EN.SET.OF.WT.JOBS                                 IN WORD 5
   BELONGS TO A EN.SET.OF.WT.JOBS

EVERY EN.STAFF.JOB
   HAS A EN.SJ.RESPONSIBLE.UNIT                              IN WORD 1
   HAS A EN.SJ.PTR                                           IN WORD 2
   HAS A EN.SJ.ACTIVE.INDICATOR                              IN WORD 3
   HAS A EN.SJ.PRIORITY.LEVEL                                IN WORD 4
   HAS A P.EN.SET.OF.STAFF.JOBS                              IN WORD 5
   HAS A S.EN.SET.OF.STAFF.JOBS                              IN WORD 6
```

```
        HAS A M.EN.SET.OF.STAFF.JOBS                          IN WORD 7
        BELONGS TO A EN.SET.OF.STAFF.JOBS

        DEFINE EN.SJ.PRIORITY.LEVEL AS A REAL VARIABLE

'' DEFINITION OF SETS

        DEFINE EN.SET.OF.FREE.WPN.GROUPS AS A SET
        DEFINE EN.SET.OF.FREE.UNITS AS A SET
        DEFINE EN.SET.OF.HQ.UNITS AS A SET
        DEFINE EN.SET.OF.STAFF.UNITS AS A SET RANKED BY
           HIGH EN.STAFF.RANKING.LEVEL
        DEFINE EN.SET.OF.MISSIONS AS A SET
        DEFINE EN.SET.OF.MISSION.FEATURES AS A SET
        DEFINE EN.SET.OF.DEFENSIVE.POSITIONS AS A SET
        DEFINE EN.SET.OF.RETRIEVABLE.ASSETS AS A SET
        DEFINE EN.SET.OF.TECH.SEGMENTS AS A SET
        DEFINE EN.SET.OF.REQ.EQUIPMENT AS A SET
        DEFINE EN.SET.OF.REQ.SUPPLIES AS A SET
        DEFINE EN.HQ.INVENTORY AS A SET RANKED BY
           LOW EN.ASSET.PROTO.ID, THEN BY
           LOW EN.ASSET.CURRENT.JOB.PRIORITY, THEN BY
           LOW EN.ASSET.HOURS.WORK.SINCE.REST
        DEFINE EN.SET.OF.HQ.PENDING.JOBS AS A SET RANKED BY
           HIGH EN.JOB.PRIORITY
        DEFINE EN.SET.OF.HQ.ACTIVE.JOBS AS A SET RANKED BY
           HIGH EN.JOB.PRIORITY
        DEFINE EN.SET.OF.PERFORMING.UNITS AS A SET RANKED BY
           LOW EN.WT.JOB.PRIORITY
        DEFINE EN.SET.OF.STAFF.JOBS AS A SET
        DEFINE EN.SET.OF.WT.ASSETS AS A SET
        DEFINE EN.SET.OF.WT.JOBS AS A SET
        DEFINE EN.SET.OF.EQUIPMENT.USE.SEGMENTS AS A SET
        DEFINE EN.SET.OF.ACTIVE.PHASES AS A FIFO SET
        DEFINE EN.SET.OF.PHASE.WT.UNITS AS A SET

    EVENT NOTICES

      EVENT NOTICES INCLUDE EN.JOB.UPDATE

      EVERY EN.END.OF.JOB.SEGMENT
         HAS AN X                                             IN WORD 6

      EVERY EN.END.OF.REST.PERIOD
         HAS AN X                                             IN WORD 6

      EVERY EN.REGISTER.EFFECT.OF.NON.ENGINEER.JOB
         HAS A EN.NE.TASK                                     IN WORD 6
         HAS A EN.NE.FEATURE                                  IN WORD 7
         HAS A EN.NE.TECHNIQUE                                IN WORD 8
         HAS A EN.NE.X.LOC                                    IN WORD 9
         HAS A EN.NE.Y.LOC                                    IN WORD 10
         HAS A EN.NE.UNIT                                     IN WORD 11
```

```
      DEFINE EN.NE.X.LOC AS A REAL VARIABLE
      DEFINE EN.NE.Y.LOC AS A REAL VARIABLE


''DEFINE TO MEANS FOR ENGINEERS (EN)

     DEFINE ..UNPREPARED              TO MEAN 0.0
     DEFINE ..FULLY.PREPARED          TO MEAN 1.0

     DEFINE ..MINEFIELD.BREACH              TO MEAN  1
     DEFINE ..MINEFIELD.CLEAR               TO MEAN  2
     DEFINE ..LINE.OBSTACLE.BREACH          TO MEAN  3
     DEFINE ..RIVER.CROSSING                TO MEAN  3
     DEFINE ..IMPROVE.LINE.BREACH           TO MEAN  4
     DEFINE ..REPAIR.ROAD.CRATER            TO MEAN  5
     DEFINE ..BUILD.TRAIL                   TO MEAN  6
     DEFINE ..BUILD.FALF                    TO MEAN  7
     DEFINE ..EMPLACE.MINEFIELD             TO MEAN  8
     DEFINE ..EMPLACE.LINE.OBSTACLE         TO MEAN  9
     DEFINE ..REMOVE.LINEAR.BREACH          TO MEAN 10
     DEFINE ..BRIDGE.DEMOLITION             TO MEAN 10
     DEFINE ..CRATER.ROAD                   TO MEAN 11
     DEFINE ..PREPARE.POSITION              TO MEAN 12
     DEFINE ..BUILD.CSS.FACILITY            TO MEAN 13
     DEFINE ..MAINTAIN.CSS.FACILITY         TO MEAN 14

     DEFINE ..NUMBER.TASKS                  TO MEAN 14

''   DEFINE ..AVAILABLE                     TO MEAN  1
     DEFINE ..PRE.TASK                      TO MEAN  2
     DEFINE ..IN.TRANSIT                    TO MEAN  3
''   DEFINE ..WAITING                       TO MEAN  4
     DEFINE ..WORKING                       TO MEAN  5
     DEFINE ..IN.REST.PERIOD                TO MEAN  6
     DEFINE ..DAMAGED                       TO MEAN  7
     DEFINE ..LEFT.AT.SITE                  TO MEAN  8

     DEFINE ..AT.HQ                         TO MEAN  0
     DEFINE ..AT.SITE                       TO MEAN  1

     DEFINE ..NO.CHANGE                     TO MEAN  1
     DEFINE ..LINEAR                        TO MEAN  2
     DEFINE ..SKEW.LEFT                     TO MEAN  3
     DEFINE ..SKEW.RIGHT                    TO MEAN  4

     DEFINE ..CONTINUE                      TO MEAN  1
     DEFINE ..BREAK.POINT                   TO MEAN  2
     DEFINE ..END                           TO MEAN  3

     DEFINE ..PERMANENT                     TO MEAN  1
     DEFINE ..RETRIEVABLE                   TO MEAN  2

     DEFINE ..VITAL                         TO MEAN 400.0
```

```
DEFINE ..CRITICAL                       TO MEAN 300.0
DEFINE ..ESSENTIAL                      TO MEAN 200.0
DEFINE ..NECESSARY                      TO MEAN 100.0

DEFINE ..CREATE.JOB                     TO MEAN 1
DEFINE ..ACTIVATE.PHASE                 TO MEAN 2
DEFINE ..BEGIN.SEGMENT                  TO MEAN 3
DEFINE ..ADJUST.SEGMENT.DURATION        TO MEAN 4
DEFINE ..END.SEGMENT                    TO MEAN 5
DEFINE ..COMPLETE.PHASE                 TO MEAN 6
DEFINE ..COMPLETE.JOB                   TO MEAN 7
DEFINE ..COMPLETE.MISSION               TO MEAN 8
DEFINE ..PREEMPT.EQUIPMENT              TO MEAN 9
DEFINE ..DISCONTINUE.ATTRITION          TO MEAN 10
DEFINE ..DISCONTINUE.LATE.FINISH        TO MEAN 11
DEFINE ..DISCONTINUE.RELATED.JOBS       TO MEAN 12

DEFINE ..PREFERRED.METHOD               TO MEAN 0
DEFINE ..SHORTEST.TIME                  TO MEAN 1
DEFINE ..BEST.EFFECT                    TO MEAN 2
```

DISTRIBUTION

Chief of Engineers
ATTN: CEIM-SL (2)
ATTN: CERD-M
ATTN: DAEN-ZCM

US Army Schools
ATTN: ATSE-CDC-M (10)

US Army, Ft. Belvoir 22060
ATTN: Engr Studies Ctr Library
ATTN: CEESC-MD
ATTN: Engr Library

TRADOC Analysis Command
ATTN: ATRC-FM 66027
ATTN: ATRC-TD 66027
ATTN: ATRC-WEA 88002 (10)

US Army MISMA 20324
ATTN: ODUSA-OR

CEWES
ATTN: CEWES-GV-A
ATTN: CEWES-EN-A
ATTN: CEWES-GM-L (2)
ATTN: Library

USAMSAA Aberdeen Proving Grounds
ATTN: AMXSY-CC

CECRL ATTN: Library 03755

NCEL ATTN: Library 93043

US Military Academy 10996
ATTN: Dept. of Military Science

US Army Command & General Staff College
ATTN: ATZL-SWH 66027

US Army Armor School 40121
ATTN: ATSB-DOTD-CBT

US Army Combined Arms Center
ATTN: ATZL-CAS 66027

Army War College 17013
ATTN: Library

US Government Printing Office 22304
Receiving/Depository (2)

Defense Technical Info. Center 22314
ATTN: DDA (2)