AD-A243 421

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A RULE-BASED WEAPON SUGGESTION SYSTEM FOR SHIPBOARD THREE DIMENSIONAL DEFENSE

by

Weng, Wen-I

December 1990

Thesis Advisor: Yuh-jeng Lee

Approved for public release; distribution is unlimited.

Unclassified

Security Classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report Approved for public release; distribution is unlimited.
2b Declassification/Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization Naval Postgraduate School / 6b Office Symbol (If Applicable) xx	7a Name of Monitoring Organization Naval Postgraduate School
6c Address (city, state, and ZIP code) Monterey, CA 93943-5000	7b Address (city, state, and ZIP code) Monterey, CA 93943-5000
8a Name of Funding/Sponsoring Organization / 8b Office Symbol (If Applicable)	9 Procurement Instrument Identification Number
8c Address (city, state, and ZIP code)	10 Source of Funding Numbers Program Element Number / Project No / Task No / Work Unit Accession No

11 Title (Include Security Classification) A RULE-BASED WEAPON SUGGESTION SYSTEM FOR SHIPBOARD THREE DIMENSIONAL DEFENSE

12 Personal Author(s) Weng, Wen-I

13a Type of Report	13b Time Covered	14 Date of Report (year, month,day)	15 Page Count
Master's Thesis	From To	December 1990	88

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

17 Cosati Codes			18 Subject Terms (continue on reverse if necessary and identify by block number)
Field	Group	Subgroup	

19 Abstract (continue on reverse if necessary and identify by block number

This thesis examines the feasibility of using an expert system approach to design an intelligent Weapon Suggestion System (WSS) to assist the Weapons Department Head (WDH) on board a naval warship in making accurate and efficient decisions in critical battle situations.

We have analyzed the constraints of a WSS and the performance of on board weapons. We have also reviewed the related material previously published and discussed the implementation environment in this thesis. The system is supported by the Knowledge Engineering Environment (KEE), often referred to as an expert system shell since it provides a comprehensive set of expert system building tools to facilitate the development of expert systems.

The WSS receives preprocessed sensor input, determines what contacts are present, performs target analysis and correlation based upon the current tactical situation, and suggests the most effective weapon(s) to deploy against various hostile targets. Simulation results have shown that the system can provide timely decision support in a time-critical combat environment.

20 Distribution/Availability of Abstract [X] unclassified/unlimited [ ] same as report [ ] DTIC users	21 Abstract Security Classification Unclassified
22a Name of Responsible Individual Yuh-jeng Lee	22b Telephone (Include Area code) (408) 646-2361 / 22c Office Symbol CS/LE

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted — security classification of this page

All other editions are obsolete — Unclassified

Approved for public release; distribution is unlimited.

A RULE-BASED WEAPON SUGGESTION SYSTEM FOR SHIPBOARD THREE DIMENSIONAL DEFENSE

by

Weng, Wen-I
Lieutenant, Republic of China Navy
B.S., Chinese Naval Academy, 1990

Submitted in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1990

Author: Weng Wen-I
Weng, Wen-I

Approved by: Y J Lee
Yuh-jeng Lee, Thesis Advisor

B. B. Giannotti
B. B. Giannotti, Second Reader

Robert B. McGhee
Robert B. McGhee, Chairman,
Department of Computer Science

ABSTRACT

This thesis examines the feasibility of using an expert system approach to design an intelligent Weapon Suggestion System (WSS) to assist the Weapons Department Head (WDH) on board a naval warship in making accurate and efficient decisions in critical battle situations.

We have analyzed the constraints of a WSS and the performance of the on board weapons. We have also reviewed the related material previously published and discussed the implementation environment in this thesis. The system is supported by the Knowledge Engineering Environment (KEE), often referred to as an expert system shell since it provides a comprehensive set of expert system building tools to facilitate the development of expert systems.

The WSS receives preprocessed sensor input, determines what contacts are present, performs target analysis and correlation based upon the current tactical situation, and suggests the most effective weapon(s) to deploy against various hostile targets. Simulation results have shown that the system can provide timely decision support in a time-critical combat environment.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The assistance and support provided by Professor Yuh-jeng Lee and Commander B. B. Giannotti are gratefully acknowledge.

Additionally, I would like to give a thanks to my wife--Pai Fang, and my son for their patience and support during the writing of this thesis.

I. INTRODUCTION

A. THE PROBLEM STATEMENT

At present, navies around the world are being challenged by the changing face of modern warfare. With the development of missile technology in 1944, the patterns of naval operations have been totally transformed in the post World War II era. Today, almost every nation around the world that aspires to military power can be characterized by an extraordinary concentration of resources centered on weapons development. Weapons that are more powerful, faster, and more accurate are successfully implemented one after another. Thus we can imagine that the warfare of the future will be a rapid-fire affair and that the type of warfare will become more complex and technology dependent.

Naval warships confronting hostile contacts can choose between a "Soft-kill" or a "hard-kill." "Soft-kill" implies employing decoys such as chaff, flares, and other electronic devices to interfere with and "confuse" hostile targets. "Hard-kill" is to employ weapons on board to destroy the hostile targets. Though the utilization of both methods is vital for a vessel to counterattack, in this thesis we concentrate only on the "hard-kill" responses against incoming targets.

As a general rule, any naval force cruising at sea must always be prepared to encounter the three dimensional threat of air, surface, and subsurface attack. However, as was demonstrated in recent naval encounters, such as the Falkland Conflict, USS Stark in the Persian Gulf, and the Israeli destroyer

Eilat which was sunk during an encounter with the Egyptian Navy in 1967, it is obvious that in confined waters the threat from the air attack poses the highest damage threat potential. Hence, we have extended this assumption into the structure of our paper as well as into our simulations.

Drawing further upon the naval engagements in the post World War II, it can be decisively illustrated that the defensive responses of the vessel under attack must be expeditions and accurate in order to avoid serious damage and/or sinking (i.e. British destroyer Sheffied during the Falkland Conflict in 1982 and the US frigate Stark in the Persian Gulf in 1987).

B. OBJECTIVES

The purpose and intent of this thesis is to demonstrate that the Weapon Suggestion System (WSS) can provide the Weaponry Department Head (WDH) reliable weapon suggestion instructions. The system will decrease the reaction time to suggest weapons against targets and maximize the efficiency of weapon utilization on board. We have designed and implemented the Weapon Suggestion System using Knowledge Engineering Environment (KEE). The reasons for using an expert system (WSS) to aid decision-support process are as follows:

- Within the scope of a tactical military operation, knowledge, data and the decision-support process are so closely related as to be essentially one function.
- The increased speed of the incoming threats have reduced the time available for human formulation of tactical decisions.
- A tactical situation is always fluid and volatile. A well designed expert system can maintain the maximum efficiency of on board weapons within fluid strategic parameters.

C. ORGANIZATION OF THE THESIS

The above discussion illustrates the need of using an intelligent expert system to help the decision-support process. The process in our design includes data acquisition, analysis and a suggestion phase. During the acquisition phase, the WSS receives target data from various sensors, information networks, or human interface, and stores that information in its dynamic database. In the analysis phase, the WSS scans its database to identify, classify and calculate the comparative threat and class of the target. In the suggestion phase, the WSS will survey the weaponry available on board and suggest the optimum weapons engagement against hostile targets.

The remainder of this thesis is organized as follows: Chapters II provides some basic background about Artificial Intelligence (AI), expert systems, and general military applications. In Chapter III we review previous research related to this thesis and analyze the results. Chapter IV describes the functions and limitations of the Weapon Suggestion System. The peripheral devices of WSS are also discussed in this chapter. Chapter V presents the software architecture knowledge rules, and simulation process and their results in detail. Finally, Chapter VI discusses the possibilities of future enhancement to the performance and power of the system.

II. GENERAL BACKGROUND

A. ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS

1. Introduction

Artificial intelligence (AI) is a branch of computer science dedicateed to the study of computational machinery that exhibits intelligent behavior. During the past 20 years, AI research has evolved from a purely academic activity into a major business involving both government and commercial applications. In the past few years, there has been an explosive growth in the number of AI systems oriented toward providing users with systems capable of offering advice on a variety of problems such as classification, diagnosis, and planning. This chapter explores AI concepts, technologies, and military applications.

2. Basics of Expert Systems

Of the various types of AI systems, expert systems have received the most public exposure. Expert system technology is widely perceived as the AI technology with the most potential for the development of near-term applications. Expert systems are computer programs that are equipped with expert knowledge to help users solve real-world problems. For example, an expert system called MYCIN provides expert advice to medical doctors on the diagnosis and treatment of various types of bacterial infections. It is considered an "expert" system because its procedures for diagnosing and recommending treatment are modeled after judgmental heuristics employed by human experts. Emulating human expert behavior is often considered an essential characteristic of an expert system. In the following section, a basic introduction to expert system technology is provided.

a. *Knowledge Base and Inference Engine*

Virtually all expert systems contain three basic components: a knowledge base, an inference engine, and a user interface. In the knowledge base, domain-specific knowledge is expressed as a set of condition-action pairs referred to as production rules that specify the action to be carried out if the prerequisite conditions are satisfied. The role of the inference engine is to control the order of rule activation and to update the belief value of the hypotheses based upon acquired evidence. A user interface caters to a smooth communication between the user and the system. It may also provide the user with an insight into the problem-solving process carried out by the inference engine. It is convenient to view the inference engine and the interface as one module, usually called an expert system shell, or shell. Figure 2-1 illustrates the basic expert system architecture.

The advantages of separating knowledge base from inference engine are:

- Knowledge can be represented in a uniform fashion (i.e. If...then... style).
- The same inference engine and user interface can be applied to different problem domains (one only needs to add new knowledge).
- It allows modifications of one part without creating side effects in other parts of the code.
- System builders can focus directly on capturing and organizing problem-solving knowledge rather than on the details of low level implementations.
- It allows experimentation with alternative control regimes for the same rule base.

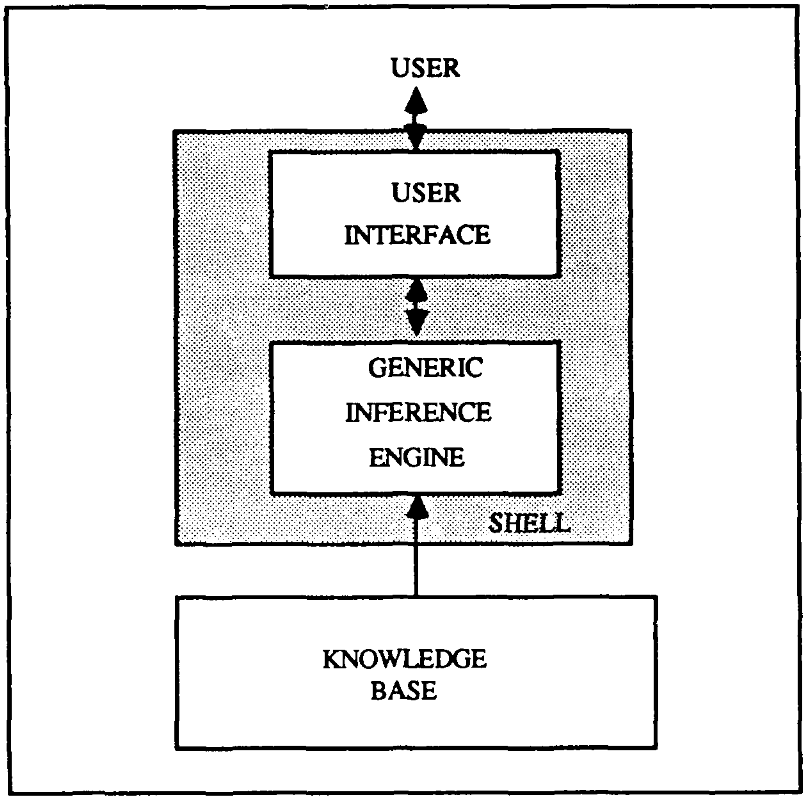

Figure 2-1. A Simplified View of Expert System Architecture

Most expert systems deal with various classes of inference problems, where the expert system must draw conclusions from various evidence or data inputs. In these types of inference problems, the set of rules (just like Figure 2-2) in rule-base can be graphically represented in the form of a set of inference networks. As illustrated in Figure 2-3, an inference networks contains top-level hypotheses, called goal hypotheses, that are decomposed into various levels of subhypotheses. The subhypotheses, in turn, are further broken down into specific items of evidence that can support those hypotheses. With each node there is usually an associated prior degree of belief and a rule for combining subnode belief values into an updated degree of belief for the node.

b. Control Strategies

The inference engine described above is theoretically sufficient for processing rule hierarchies of any size. However, as the number of rules in a rule base increases, the behavior of acquiring all available evidence or primitive values would become very inefficient, as well as frustrating to a user if he or she must enter all the data. To efficiently manage the application of domain knowledge to specific problems, inference engines apply a control strategy that carefully controls the order of rule activation.

IF:	The exhaust is smoky, and The car is backfiring, and There is a lack of power,
THEN :	The carburetor fuel mix is too rich.
IF:	There is a lack of power, and There is a gray deposit on the spark plugs,and The engine overheats,
THEN :	The carburetor fuel mix is too weak.
IF:	The carburetor fuel mix is too rich, or The carburetor fuel mix is too weak,
THEN :	The carburetor fuel mix needs to be adjusted.

Figure 2-2. Sample Production Rules

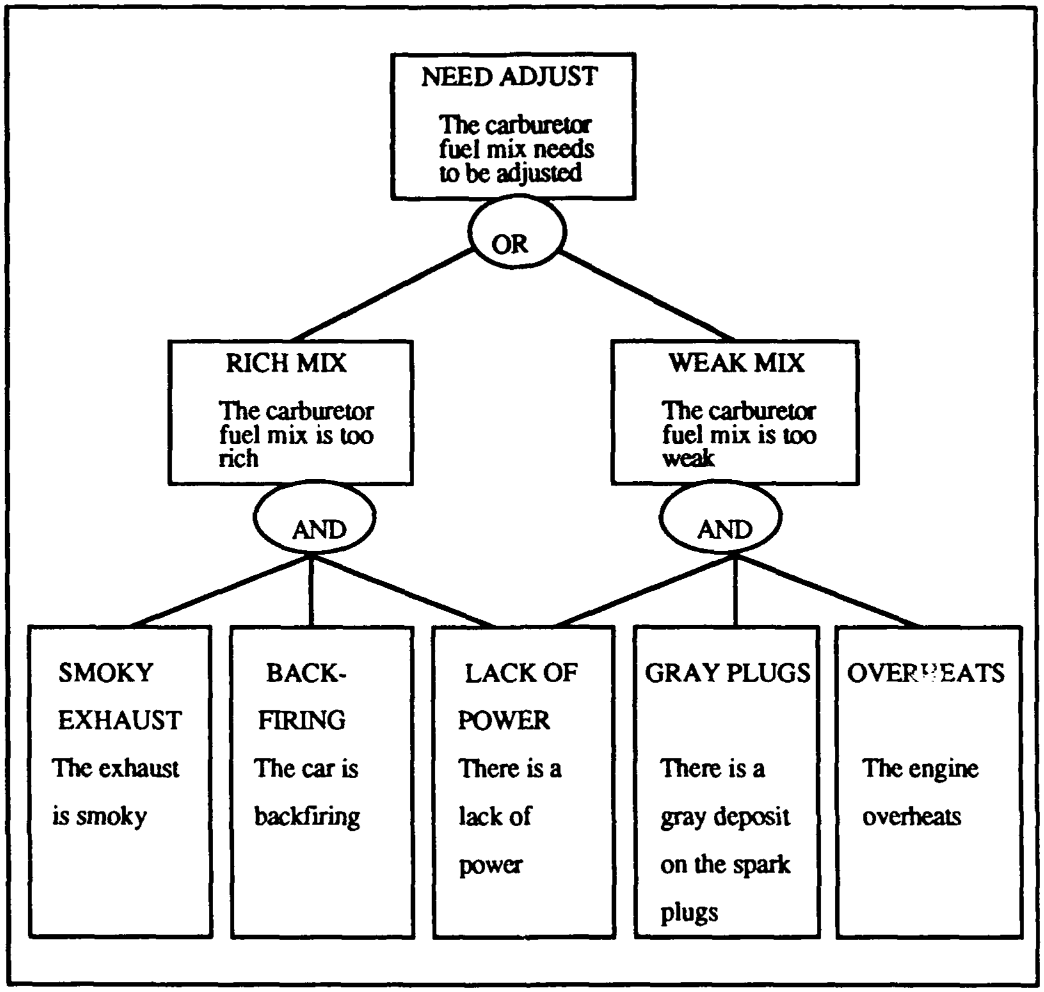

Figure 2-3. Sample Inference Network

One common strategy is to select a goal hypothesis, usually a top level hypothesis in the rule hierarchy, and to chain down the hierarchy one rule at a time to identify intermediate and primitive clauses that impact the selected goal hypothesis. The expert system then gathers data about evidence items specifically related to the goal hypothesis of interest. This approach of managing the utilization of rules is called a goal driven control strategy, or backward chaining, inasmuch as it selectively pursues one goal at a time.

A data driven control strategy, or forward chaining, is one in which rule activation is controlled by data available and not by the pursuit of particular goals. In the data driven approach the expert system awaits the input of new data. When new data is entered, the inference engine scans for rules that are impacted, applies the impacted rules to generate whatever conclusion it can, and then resumes waiting for input.

c. *Knowledge Engineering*

Expert systems can be described as computer-consultants that emulate human expert reasoning in a problem domain. The process of extracting and encoding domain knowledge held by human expertise is called knowledge engineering. Today, knowledge engineering remains a time-consuming and labor-intensive process wherein an AI technologist, called a knowledge engineer, must repeatedly interview one or more human experts over a long time period to extract the heuristics to be encoded in the expert system knowledge base.

d. *Limitations of Expert Systems Technology*

Expert system technology provides a powerful set of tools for developing systems that can generate expert advice to users for solving important and complex problems. Unfortunately, there are a number of practical difficulties exist in the system. The following, drawn from Barr, Cohen, and Feigenbaum (1989), represents a typical characterization of these limitations.

- **Narrow Domain of Expertise.** Expert systems work within narrow areas of expertise (Davis, 1982 and 1989). Technical domains, in which terms are well defined and in which subprograms can be solved separately, are more amenable to introduction of expert systems than more open-ended domains. Engineering and business are thus better problem domains than political science and sociology. When the limitations are well understood, there is little problem in using an expert system reliably for substantial gains in productivity, and many of the notable successes are of just this sort (Feigenbaum, 1989).

- **First Principles.** The domain models used by expert systems are not generally the theoretical first principles of textbooks, but are a looser collection of facts and associations (Davis, 1987). Expert systems rely more on special-case formulations of relations than on "first principles." Although a set of general principles such as Maxwell's equations governs the behavior of a large class of devices, designers of expert systems prefer to codify special cases, exceptions, and empirical associations, as well as some causal associations, in order to put the general principles in forms that can be applied more quickly and more precisely. As a result, they are unable to fall back on a better theory in some situations. There is substantial research in AI on using first principles in reasoning, much of it in the area of electronics troubleshooting (Davis, 1987). As this matures, it will allow us to build expert systems that blend the theoretical soundness of the first principles with the precision of special-case exception clauses that map the theory into the world of practical applications.

- **Limits of Knowledge.** Expert systems tend to perform well on the classes of cases that have been explicitly considered but may fail precipitously in new cases at the boundaries of their competence (Davies, 1987; Lenat, 1986). In part this is due to lack of knowledge of first principles. The performance of humans is more robust. As we reach the extent of what we know about a problem area, we often can give appropriate answers that are approximately correct, although not very precise, and we know that we have reached the limits of our knowledge. For expert systems, the standard solution today is to codify rules that screen out cases that are outside the intended scope in order to further ensure that the system is being used in an appropriate way.

- **Self-knowledge.** Expert systems have little or no self-knowledge, and thus do not have a sense of what they do not know (Lenat et al., 1983). Although expert systems can often give explanations of what they know, they do not have a general "awareness" of what the scope and limitations of their own knowledge are. Meta-level knowledge, such as rules of strategy, can offset this shortcoming in special situations but does not constitute a general capability.

- **Commonsense Knowledge.** Expert systems can only represent commonsense knowledge explicitly and do not use commonsense modes of reasoning such as analogical reasoning or reasoning from the most similar recent case (McCarthy, 1983). Designer of current expert systems resolve this by assuming that users can exercise some common sense, and by specifying common facts explicitly when needed.

- **Explicit Knowledge.** The knowledge of expert systems must be made explicit; they have no intuition (Dreyfus and Dreyfus, 1986). So far, the problems that have been most successfully solved with expert systems have been those in which inferential knowledge is easily formulated as rules and the organization of objects and concepts is easily formulated as taxonomic (class-subclass-instance) hierarchies and part-whole hierarchies. Reasoning by analogy of by intuition is still too unpredictable to use in high-performance systems. Because expert systems articulate what they know. Any task for which knowledge cannot be articulated for any reason is not a food candidate for an expert system.

- **Reusable Knowledge.** Knowledge bases are not reusable (Lenat, 1986). Since the cost of building a knowledge base is substantial, it is desirable to amortize it over several related expert systems, with unique extensions to cover unique circumstances. For example, many medical systems use facts about anatomy and physiology, yet often each encodes those facts specifically for use in a unique way. The challenge is to develop knowledge representations that can be used efficiently, independent of the specific context of use. By contrast, considerable progress has been made in building lower level components of expert systems that are reusable -- this has led to the widespread use of expert systems shells. Representing knowledge in structured objects improves the chances of reusability, and substantial current research is exploring this and other means of improving reusability of knowledge bases (see, for example, Lenat, 1986).

- **Learning.** Expert systems do not learn form experience (Schank, 1983). Research on machine learning is maturing to the point where expert systems will be able to learn from their mistakes and successes. Learning by induction from a large library of solved cases is already well enough understood to allow induction systems to learn classification rules that an expert system then uses (Michie et al., 1984; Michalski et al., 1986). Prototype systems have been built that emphasize learning in context, sometimes called explanation-based learning or apprentice learning, which appears to hold promise for expert systems (Mitchell et al., 1986). The challenge is to design learning mechanisms that are as accurate as knowledge engineering but are more cost effective.

- **Reasoning Methods.** It is generally not possible to prove theorems about the scope and limits of an expert system because the reasoning is not formal (Nilsson, 1982). Although some systems are implemented in a logic programming language such as Prolog, or otherwise use predicate calculus as a representation language, many systems are more "ad hoc." In this regard, though, expert systems are not in a much different state than other software in which complex reasoning with heuristics defies proofs of correctness. There is considerable research in formalizing the reasoning methods of AI programs and combining those with a predicate calculus representation of knowledge.

- **Knowledge Context.** Expert systems may fail if the user's conceptual framework is not the same as that of the expert and others on the design team (Winograd and Flores, 1986). Knowledge engineers work under the assumption that the experts they work with know the context of intended use and the intended users' terminology and point of view. This may result in misuse of a system when a user attaches different meaning to terms than did the expert who designed the knowledge base. There are no safeguards built into today's systems to test this assumption. Thus the challenge is to provide enough ways of explaining what is in a knowledge base to make its contents clear to all users. But a simple, more pragmatic remedy is to include members of the intended user community on the design team. A related problem is that the conceptual view of the design team -- even if only a single expert -- may change over time, and thus maintaining a knowledge base over time becomes difficult.

B. EXPERT SYSTEM IN MILITARY APPLICATIONS

The term "intelligence" as used in expert systems for military applications refers to the collection, correlation and analysis of information to support command decision-making (Lehner, 1989, p. 95). Time is always critical in a war at sea. A commanding officer in a combat situation must react efficiently and effectively in an environment that is both time critical and tactically complex. A well functioning expert system can render incalculable assistance to that commander by aiding in making both rapid and correct decisions, thereby significantly reducing the incidence of strategic and/or tactical errors in critical situations. In the following sections, we summarized some military applications using AI technology as detailed in (Lehner, 1989).

1. Large Area Sensor Surveillance System

The Automated Exploitation of Large Area Surveillance Sensor (AELASS) system is a production rule system developed by PAR Government Systems Corporation, for identifying the activities of military units based on surveillance data that is provided from a variety of collection devices (Lehner, 1989, p. 97).

2. Electronic Intelligence System

The Expert Prolog System (EXPRS) work, as described in Pecora (1984), is oriented toward developing a general class of techniques that can be applied to the full spectrum of problems in tactical fusion (Lehner, 1989, p. 101).

3. A Tactical Aid for Estimating Courses of Action

AI/ECONA is a prototype decision aid, developed by PAR Government Systems Corporation, that is designed to assist Army tactical intelligence analysis in evaluating alternative Enemy Course of Action (COAs). AI/ENCOA combines the use of additive multiattribute utility analysis (MAU) for course of action evaluations with rule-based procedures for assigning parameter values (scores and weight) to the MAU model (Lehner, 1989, p. 106).

4. Real-Time Advisory System

The Real-Time Advisory System (RTAS) is a prototype expert system that operates as an intelligent interface system. It is designed to provide real-time tactical advice for Airborne AntiSubmarine Warfare (AASW) problems (Lehner, 1989, p. 133).

5. A Fire Support Planning Aid

Battle is a decision aid designed to assist fire support planning, and was developed by Slagle and Hamburger (1985) while at the Navy Center for Applied Research in Artificial Intelligence (Lehner, 1989, p. 148).

III. RELATED WORK

A. INTRODUCTION

A great deal of energy and effort is presently directed towards research and application of AI technology. The effort involves development and refinement of both hardware and software that can be of assistance in overcoming the difficult and complex problems that are of immediate concern, and also those sets of problems which are anticipated for future encounters.

A number of officers who have studied at the Naval Postgraduate School are currently involved in the concentrated research that falls within the scope of AI technology. Using both experimental data and their on board training in an effort to further the efficacy of AI systems in battlefield applications.

B. REVIEW

1. Adaptation of a Knowledge Based Decision-Support System in the Tactical Environment

In a paper presented by Clair, and Danhof (1981), the authors describe an alternative to the system then in use (e.g the World Wide Military Command and Control System, and the Naval Tactical Data System), which the authors determined to be too slow, too large, and too expensive, therefore rendering it impractical for the tactical environment of 1981.

Clair and Danhof designed a replacement which they designated as a TAC* system, for "Tactical Adaptable Consultant." The new system incorporated a database, a knowledge base, their associated management systems, and a distributed interface, and could assist tactical commanders in decision making.

Many computer systems have been designed for specific problems. The result of such systems is limited application areas. While the original concept of TAC* was to provide a tool for the tactical commander, the ultimate design was conceived to be general in nature. Due to the method of treating data, rules, and changes as identical functions within the design structure, the basic system may be used in any number of other application areas.

2. TAC*II an Expert Knowledge Based System for Tactical Decision Making

TAC*II (Geschke, Bullock and Widmaier, 1983) is a redesign and partial implementation of an expert AI system for TAC*. The system receives preprocessed sensor inputs, determines what contacts are present, and suggests the best course of action to take. It performs target analysis and correlation based upon the current tactical situation. Production rules are used to discover which actions have been established by higher authority for the current tactical situation. In a highly dynamic tactical environment, the major emphasis is placed on one Naval Officer, the Tactical Action Officer (TAO). The TAO is required to respond to a vast amount of diverse information, received from a multitude of sources, in an extremely time critical situation. The system which they proposed is an automated aid to the TAO, a decision making system which can assist the officer to respond in a timely manner to the current situation.

To date it has not been sufficiently verified that the TAC*II system can operate within and meet the real time requirements of the tactical environment. Though the system designers believe that operational efficiency within real time is a realistic expectation of their system, their design goals did not incorporate real time functions within the prototype base, but rather designed a system that performed the required functions.

3. A Rule-Based System for Shipboard Air Defence

Wang (Wang, 1989) asserts that because of the changes in warfare that have occurred since War World II, today's navies are facing an unprecedented challenge at sea. The reasons for building and proliferating expert systems as aids to the decision making process in tactical air defense are as follows:

- The increased speed of weapons systems has reduced the time available for making tactical decisions by human decision makers. This requires greater capabilities to meet the incoming threats and can be partially automated through the use of computers.
- Weapons technology has progressed to a point where it is very difficult for a single human decision maker to be proficient in all offensive and defensive options; additionally, there may not be enough time for him to absorb all information and execute all decisions without any error.
- In the area of military tactical operations, knowledge and data are closely related in the decision making process.
- A tactical situation is usually presented to the OTC (Officer in Tactical Command) with a view of the "state of the world." This view can be inaccurate. Nonetheless, based on this incomplete information he must make decisions subject to the constraints imposed by preplanned actions.

The above statements illustrates the need for using an intelligent expert system to assist the OTC in executing the decision making process. The system, in common with the two systems previously discussed, receives preprocessed sensor input, performs target analysis and correlation based on current tactical situ. tion, and suggests the best optimal course of action.

The paper also shows the computer simulation results.

C. SUMMARY

All three papers discussed above address the issue of designing an expert system to aid the responsible party (i.e. TAO or OTC) in problem solving within the environment of hostile engagement.

We would like to point out and emphasize additional requisite characteristics of these systems as the following:

- All the systems need data input from the out side world (i.e sensors, information network, or human interface e.t.c).
- All the systems have to have the ability to analyze a very large amount of data in an extremely short time.
- All the systems must possess large memory to store the dynamic data in a complex environment.
- All the systems must not only give correct advice, but must make that advice available as fast as possible.

IV. DEFINING A WEAPON SUGGESTION SYSTEM

A. WEAPON SUGGESTION SYSTEMS

1. Definition

The Weapon Suggestion System (WSS) is a system to assist human analysts in difficult battle conditions. Inherent within both its design and capability is the capacity to process information and arrive at conclusions under conditions which are difficult for humans. Using relative data collected from the incoming target itself the WSS has the capacity to identify and classify the threat from that incoming target. Analyzing all the related data stored in the knowledge base the system is subsequently able to formulate and suggest an optimal response utilizing installed weapons systems.

2. Why Build the System?

At present, surface ships may well expect to confront a widely varying conformation of enemy strike weapons directed against it. This variety and complexity of offensive weapons is further compounded by the daily improvements made in offensive weapon's systems. These two factors may then again be multiplied by the potential of rapidly changing battle conditions due to the enhanced time frame delivery capabilities of todays modern weapons.

Putting these factors together and viewing them in a realistic light, it can be safely said that an officer's ability to assess all relevant technological data and arrive at the response within the critical time allotments has been outstripped by the mass of the technology itself.

The WSS is specifically designed to protect against that technological vulnerability. By assisting the WDH on board in optimal weapons selection against hostile targets the survivability of a ship and its crew can be greatly enhanced.

B. MAIN STRUCTURE OF WSS

1. Functions

The WSS, as previously mentioned, is used to suggest the best available weapons on board that can be used against the enemy targets within a specified period of time. Ideally, WSS can analyze any number of targets of different types, which may appear simultaneously or at varying times, then generate the appropriate suggestions and/or instructions regarding weapon deployment. The functions of WSS are:

- Decreasing the reaction time for weapon selection in time critical situations
- Generate a set of weapon suggestion instructions
- Producing a automatic weapon assignment system which significantly reduces reaction time and the probability of miscalculation when used with sensors, identification devices and an operational combat system
- Decreasing the probability of human error in the complex war at sea conditions

2. Peripheral Devices of the WSS

An ideal WSS is equipped with sensors, IFF devices, modern combat systems, and weapons. The combat system has the capability of arranging the firing order of the on board weapons, while the Identify Friend or Foe (IFF) device can assist in differentiating and classifying targets. The sensors are used primarily to detect targets.

Initially, the WSS receives the preprocessed input signals from the sensors (i.e. ESM, Radar, Sonar, vision etc.), performs target analysis and correlation based on the current tactical situation, and subsequently generates the optimal weapons selection instructions which feed into the combat system. Figure 4-1 showed the peripheral devices of the WSS.

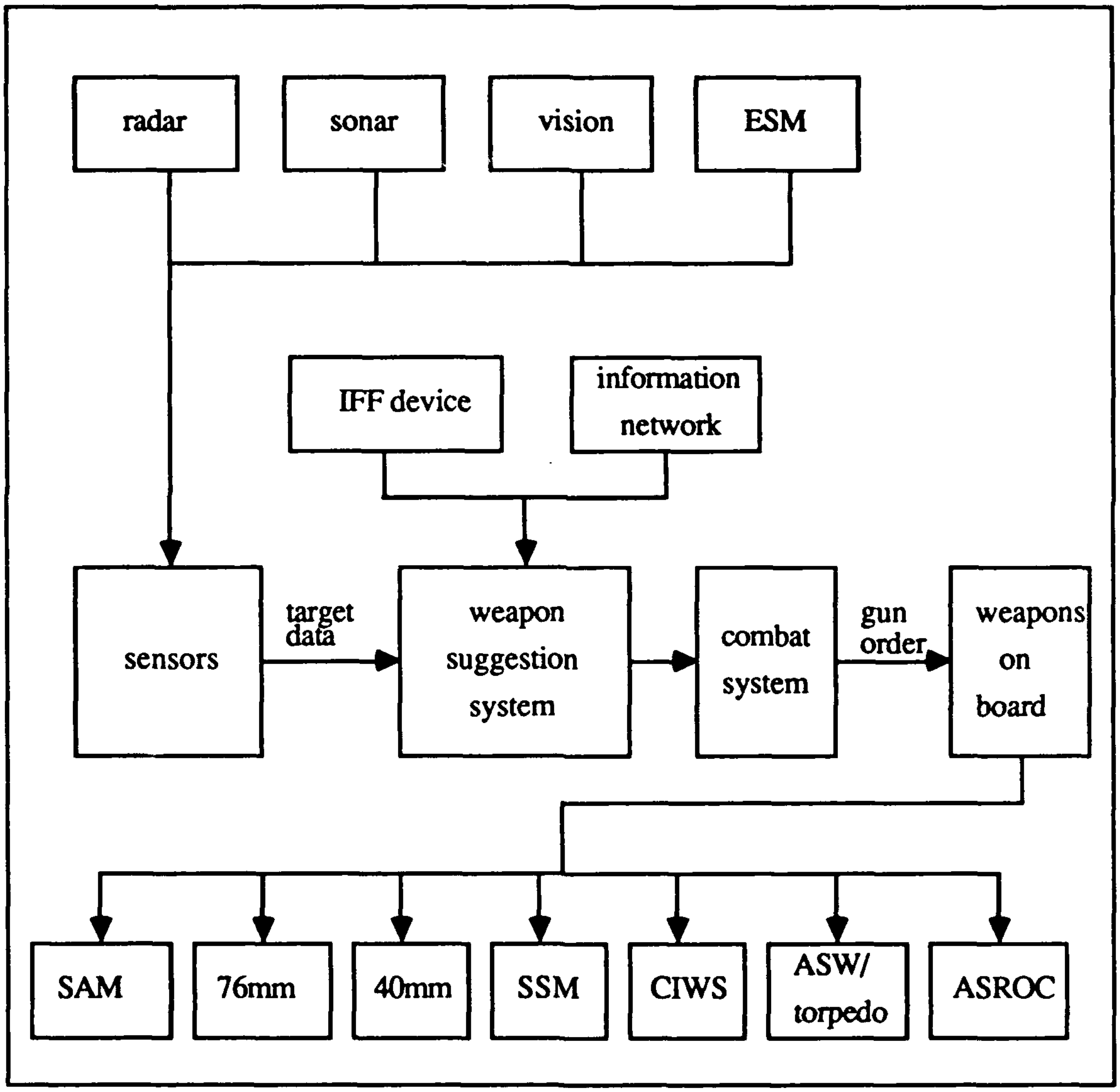

Figure 4-1. Peripheral Devices of WSS

a. Sensors

Sensors are devices that can detect, measure or record physical phenomena. In the simulation we assume we have the following sensors:

- **Radar**: a device using transmitted and reflected radio waves for detecting a reflecting object (such as an aircraft) and determining its direction, distance, height, or speed.
- **ESM** (Electrical Support Measure): a passive device employed to intercept the radio waves from the active sources.
- **Sonar**: an apparatus that transmits high-frequency sound waves through water and registers the vibrations reflected from an object, used in finding submarines, depths, etc.
- **Vision**: the act or power of seeing with the eye.

The parameters of each sensor are described in Table 4-1. The values in Table 4-1 are approximate number.

b. Weapons

Weapons are instruments or devices used to injure, kill or destroy an object. We assume we have the following weapons in WSS:

- **SAM** (Surface to Air Missile): an anti-aircraft and anti-missile weapon. It provides primary air defense for the warship.
- **76 mm** (76 mm gun): a gun whose main role is anti-missile defense but it is also effective in either anti-aircraft or anti-ship fire.
- **40 mm-1, 40 mm-2** (40 mm gun): effective in either anti-aircraft or anti-ship fire but the maximum firing range is significantly shorter than the 76 mm gun.
- **CIWS** (Close-In Weapon System): to provide last ditch defense against anti-ship missile. It also provides continuous surveillance and defense within its engagement envelope, independent of other weapon's systems.
- **SSM** (Surface to Surface Missile): an anti-ship weapon which can be launched from the surface ship.
- **ASROC** (Anti-Submarine Rocket): an ship-launched ballistic missile used as the primary anti-submarine warfare weapon.

- **ASW/torpedo-1, ASW/torpedo-2**: an anti-submarine weapon which can be launched from the surface ship. Its maximum firing range is significantly shorter than ASROC.

The parameters of each weapon are described in Table 4-2. The values in the table are approximate number.

TABLE 4-1. SENSOR PARAMETERS IN WSS

ITEM	detection range (mile)	surveillance sector (degree)	application
Radar	0 ~ 300	0 ~ 360	air, surface
ESM	0 ~ 400	0 ~ 360	air, surface
Sonar	0 ~ 30	0 ~ 360	subsurface
Vision	0 ~ 10	0 ~ 360	air, surface

TABLE 4-2. WEAPON PARAMETERS IN WSS

ITEM	firing range (mile)	firing sector (degree)	application
SAM	3 ~ 30	0 ~ 360	air
76 mm	0.1~ 8	0 ~ 360	air, surface
40 mm-1	0.01 ~ 3	60 ~ 120	air, surface
40 mm-2	0.01 ~ 3	240 ~ 300	air, surface
CIWS	0 ~ 1.5	0 ~ 360	air, surface
SSM	5 ~ 200	0 ~ 360	surface
ASROC	3 ~ 12	0 ~ 360	subsurface
ASW/torpedo-1	0 ~ 5	60 ~ 120	subsurface
ASW/torpedo-2	0 ~ 5	240 ~ 300	subsurface

3. Outline of the Weapon Suggestion System

The WSS is a software system which is implemented in KEE (see Chapter V for detail). We will discuss the implementation and simulation of the system in Chapter V. The flow chart of the system are presented in the Appendix. The working steps of WSS are described below:

- As the targets fall into the surveillance area of the sensors, the sensor network will generate such data as speed, distance, height, depth, and direction, for each potential target and input that information into the WSS database.
- Based on IFF signals and an interpretation of the information received, the WSS will identify the potential targets as friendly or hostile. If the target IFF signal corresponds to a known USN or allied reference, the target will be identified as such. Otherwise it will be automatically identified as a "hostile" target.
- Classification of the threat class of the hostile target. The threat class will be generated and determined from the threat value equation which is integrally programmed into WSS. (The threat value equation will be discussed in Chapter V.)
- Generate the weapon suggestion instruction. After the target has been identified as "hostile" and its potential threat assessed, the WSS will suggest the weapons to be most effectively deployed when the target distance has closed to within maximum firing range. Then, the WSS will suggest that weapon for use against the target. All the above procedures of the WSS are performed automatically.
- If a target has been neutralized or destroyed or any new targets appear, the system can immediately regenerate the new weapon suggestion instructions upon request.
- The WSS can generate an output signal from the selected weapon while it is being deployed against the hostile target, and the combat system will display the relative status of each weapon, whether it is deployed, in readiness or held in reserve. The combat system will generate and execute gun orders for the on board weapons, as well as performing the tracking and readiness inventory functions. Once the WDH has agreed to the weapon suggestion of the WSS, the weapon will be assigned and positioned to fire through the combat system.

4. Limitations

Although a WSS can perform many functions and solve innumerable problems, it is in fact limited by the fixed number of sensors and weapons on board any given vessel, as well as by the differing performance of those weapons and sensors. Consequently, we believe the following factors should be considered when designing the WSS for a specified platform.

a. Tactical Limitation

As we build the WSS we need to give detailed consideration towards defining the exact tactical parameters of that specified system. Each platform can and will be expected to perform entirely different functions within a context of hostile engagement. If the fundamental design and programming decisions have been ill-conceived with regard to tactical specialization, the systems will not only be useless but may also expose the platform to potentially disastrous consequences.

b. Sensor Limitation

Before we design a weapon suggestion system for a specific platform we need to give careful consideration to the following factors that limit sensors and which will directly impact upon a vessel capability to detect targets.

- The detection range of the sensor
- The blind zones of the sensor
- The capability of sensors for tracking multiple target simultaneously
- The number of sensors that can be installed in the platform
- The characteristics of the sensor (i.e for air, surface or subsurface surveillance)

c. Weapon Limitation

Weapons manufactured to the same mechanical specifications of exactly the same type and model, installed on exactly the same type of vessel can be expected to perform quite differently due to such minor variables as firing position. Because of this variability in weapons performance, the following limitation factors need to be considered as we build the system.

- The effective firing range of the weapon
- The "firing cut out" zones of the weapon

- The number of weapons in the platform
- The destructive capacity of the weapon
- The relative priority of the weapon for deployment against an enemy target
- The characteristic of the weapon (i.e Surface to Air, Surface to Surface, or Surface to Subsurface)

V. CONSTRUCTING A WEAPON SUGGESTION SYSTEM

A. THE APPLICATION DEVELOPMENT ENVIRONMENT

1. Introduction

The WSS was designed and implemented in the Knowledge Engineering Environment (KEE) on a Texas Instruments Explorer workstation. The tools and facilities in KEE that we used in the development are described below. KEE terminology is also essential for understanding later description of the applications.

2. Knowledge Engineering Environment

The KEE (KEE ActiveImage Reference Manual, 1989, KEE Interface Reference Manual, 1987, KEE TellAndAsk Reference Manual, 1989, KEE User's Manual, 1988) provides knowledge system developers with a set of programming tools and techniques for building applications to represent and utilize knowledge. Knowledge systems are an important part of programming technology. They have evolved to improve the way that people and computers interact. KEE is built upon Lisp and supports object oriented programming. It also includes a graphical user interface. The following sections introduce basic features of KEE.

a. Objects, Attributes and Values

To set up a knowledge base in KEE one must first define the objects and their attributes and values. An object is an entity or an item that represents a physical object such as a ship, a person, or a weapon; or it could be an abstract idea or a concept. Objects can be organized in hierarchical structures to represent classes and members. Attributes and values are properties that describe a class of objects, a subclass, or a member object. In KEE, the value of an attribute can be integers, symbols, objects, classes of objects, and methods (Lisp functions). In order for the value of an attribute to be a function, the value class of the attribute must be declared as a method. Methods are behavioral knowledge made up of Lisp code used to retrieve information or to modify the knowledge base. Once a method is written, it is stored as the value of an attribute. It does nothing until it is activated.

b. Inheritance

The units are grouped into hierarchies from more general objects, called classes, down to particular objects, called instances. A unit is made up of slots created to represent an object's attributes. Designers can describe a system by creating units that act as templates for a whole class of objects. When a new unit is designated a "child" of one or more units, it inherits slots and default information from its "parents." One can then add objects or modify the inherited information as needed to reflect the values of an object.

c. Rules

Rules are classified as a special type of object. Like other objects, rules have classes, subclasses, and members. Unlike other objects, we define the rules, not the attributes. In KEE, rules are written as if-then statements. The "if" part of a rule consists of a set of conditions, called the premises; when the premises are satisfied, facts contained in the "then" part of the rules, called the conclusion, are activated, and actions specified in the conclusion performed. KEE allows the user to partition rules by grouping them into rule classes. This increases the system efficiency by reducing CPU time for pattern-matching routines and user maintenance time.

KEE provides two types of rules for the designer: deduction rules and action-taking rules. Both types are needed depending upon whether the user requires rules that perform deduction and rules that take some kind of action on the system objects. Deduction rules are pure theorem-proving rules. They establish dependencies between a conclusion and its premises. If the rules produce a contradiction to the set of facts that the system started out with, then the original facts are inherently contradictory or the rules are inconsistent. Action-taking rules, on the other hand, change the knowledge base of the application.

d. TellAndAsk

TellAndAsk is an English-like language that can be used to interact with KEE. It can be used for a variety of purposes such as modifying knowledge bases, writing rules. creating justifications, and putting facts into worlds. TellAndAsk is a language with a syntax similar to English and provides a way of making statements about relationships between units, values of slots, and values of facets.

e. Active Values

Active values are "watchdog" units. They provide the facility to cause side effect behavior when accessing or modifying data in a knowledge base. By attaching an active value to a slot, the designer can specify that a particular action should occur every time that slot's value is accessed or modified.

f. ActiveImages

KEE's ActiveImages package provides a variety of graphic displays for both viewing and modifying slot values in KEE objects. These graphic images can be integrated into an application as an attractive interface to the system. They also can be used during application development as a convenient display aid for program debugging. There are many different kinds of images, including histograms, push buttons, thermometers, numeric-dials, and plots.

g. KEEworlds

KEE's multiple worlds facility enables one to assert a collection of facts into a single context, called the background, which can be used as a basis for the creation of additional worlds containing somewhat different facts and assumptions. KEEworlds is useful for modeling and exploring different hypothetical situations that might arise in a knowledge base. A new world represents an alternative state of new context of a knowledge base.

h. Truth Maintenance System

If a fact or assumption is no longer believed to be true, the Truth Maintenance System (TMS) makes sure any facts contingent upon the belief are retracted from the knowledge base. These contingencies are called justifications. A designer can use the Truth Maintenance System to perform bookkeeping for applications that incorporate information inferred from other facts and assumptions and to help with backtracking to previous states.

i. KEEpictures

The KEEpictures environment is an objective-oriented graphics tookit that enables designers to construct images or graphic systems, interfaces, and end-user applications. Its intended uses range from simple graphical displays to animation. KEEpictures provides a set of standard picture classes for picture construction. These include arcs, axes, box.string, bitmaps, circles, dials, lines, and rectangles. Each of these primitives can be shaped, enlarged, shrunk, and altered in a variety of ways. Pictures can be combined to form larger, composites pictures. Users can also draw their own pictures using bitmaps.

2. Texas Instruments Explorer

The Texas Instruments Explorer system is an advanced, single-user workstation that provides extensive support for the development of large-scale, complex programs and for research in new technologies, including artificial intelligence. The Explorer system is also an affordable delivery vehicle for end-user applications requiring symbolic processing, high-quality graphics, and special-purpose processors. The programming environment of the Explorer system includes the following:

- High-resolution, interactive display
- High-speed symbolic processing using the Lisp and Prolog language
- Integrated programming tools
- Extensive software
- Large memory capacity and sophisticated memory management
- Networking facilities

The Explorer system provides a Lisp environment of more than 14,000 Lisp functions. It features a highly productive programming environment that allows a user to develop very complex programs in incremental steps. One can use any of the system software, as well as optional toolkits and utilities, as building blocks to create application software. The

Explorer system uses a variety of windows and menus to present data and options. The window system provides a hierarchical input/output interface between a user and the Explorer system. Different windows have different characteristics: They can be of various sizes, occupy various positions on the video display, accept keyboard or mouse input, and display information.

The specially designed and microcoded Explorer system processors directly implement the software run-time environment, providing fast execution of large programs without sacrificing the dynamic nature of Lisp. The Explorer system is written entirely in Lisp.

B. SOFTWARE IMPLEMENTATION

1. Introduction

The WSS is designed using KEE. In the following sections we describe the software implementation architecture of the Weapon Suggestion System (WSS).

2. Knowledge Base

We created a knowledge base, and used an arbitrary name, in KEE for the storage of our WSS. The graph of the knowledge base is shown in Figure 5-1. In this section we focus on RULES.3, RULES.4, and RULES.5 rule class. There are four subclass rules in RULES.3 and RLUES.4. The major functions of the subclass rules in RULES.3 are detection, identification, and the classification of targets. The major function of the subclass rules in RULES.4 is the suggestion of weapons to be used against hostile contacts. There are five subclass rules in RULES.5. These subclass rules are used to process the target

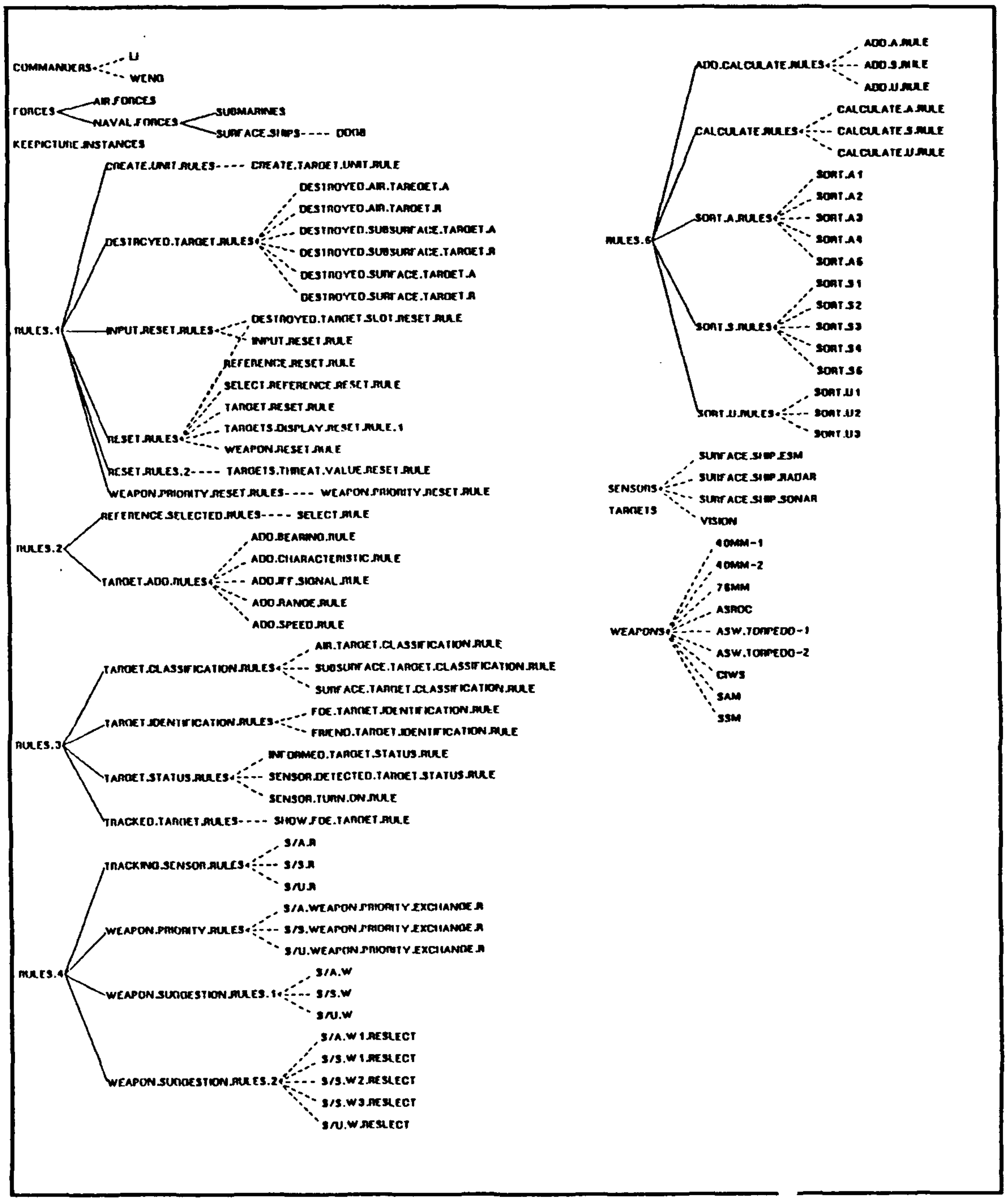

Figure 5-1. Graph Knowledge Base

data and generate the threat class of each target. The rest of the rules will be discussed in the following sections. As discussed in Chapter IV, the step for weapon suggestion is the following:

- Target detection
- Target identification
- Target classification
- Assignment of target threat class
- Suggestion of a proper weapon for use against target

The rules are designed based on the assumption that air defense operations have the highest priority. That is, if air and surface targets appear simultaneously, which theoretically could produce a conflict in regard to the weapon suggestion, the air target will be assigned priority.

a. RULES.3 in Knowledge Base

Figure 5-2 shows the TARGET.STATUS.RULES subclass rule in RULES.3. It contains three rule units. This subclass rule is used to decide whether the target is detected or not. For example, To apply the SENSOR.DETECTED.TARGET.STATUS.RULE rule, if the following conditions are true:

- **Air target A1** is in class targets, and
- **Surface ship Radar** is in class sensors, and
- The **status** of surface ship Radar is on, and
- The **characteristic** of air target A1 equal to the **characteristic** of surface ship Radar, and
- The **sensor** of DDGB (Guided Missile Destroyer of Blue unit) is surface ship Radar, and
- The **relative range** of air target A1 less than or equal to the **maximum detective range** of surface ship Radar,

then we will reach the conclusion "The status of air target A1 is detected."

```
((INFORMED.TARGET.STATUS.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE INFORMATION OF ?TARGET IS YES)
    THEN
    (THE STATUS OF ?TARGET IS DETECTED)))
 (SENSOR.DETECTED.TARGET.STATUS.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (?SENSOR IS IN CLASS SENSORS)
    (THE STATUS OF ?SENSOR IS ON)
    (LISP (EQUAL (THE CHARACTERISTIC OF ?TARGET) (THE CHARACTERISTIC OF ?SENSOR)))
    (THE SENSOR OF ?PLATFORM IS ?SENSOR)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.DETECTIVE.RANGE OF ?SENSOR)))
    THEN
    (THE STATUS OF ?TARGET IS DETECTED)))
 (SENSOR.TURN.ON.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (?SENSOR IS IN CLASS SENSORS)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.DETECTIVE.RANGE OF ?SENSOR)))
    (LISP (EQUAL (THE CHARACTERISTIC OF ?TARGET) (THE CHARACTERISTIC OF ?SENSOR)))
    (THE SENSOR OF ?PLATFORM IS ?SENSOR)
    THEN
    (LISP (PUT.VALUE ?SENSOR 'STATUS 'ON)))))
```

Figure 5-2. TARGET.STATUS.RULES in RULES.3

Figure 5-3 shows the TARGET.IDENTIFICATION.RULES subclass rule in RULES.3. This subclass rule is used to identify whether the target is a hostile or friendly. For example, to apple the FOE.TARGET.IDENTIFICATION.RULE rule, if the following conditions are true:

- **Air target A1** is in class targets, and
- The **status** of air target A1 is detected, and
- The **IFF signal** of air target A1 is different from DDGB,

then we can reach the conclusion "The identification of air target A1 is hostile."

```
((FOE.TARGET.IDENTIFICATION.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE STATUS OF ?TARGET IS DETECTED)
    (THE IFF.SIGNAL OF ?TARGET IS DIFFERENT)
    THEN
    (THE IDENTIFICATION OF ?TARGET IS FOE)))
 (FRIEND.TARGET.IDENTIFICATION.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE STATUS OF ?TARGET IS DETECTED)
    (THE IFF.SIGNAL OF ?TARGET IS SAME)
    THEN
    (THE IDENTIFICATION OF ?TARGET IS FRIEND))))
```

Figure 5-3. TARGET.IDENTIFICATION.RULES in RULES.3

Figure 5-4 shows the TARGET.CLASSIFICATION.RULES subclass rule of RULES.3. This subclass rule is used to classify the category of target. For example, to apply the AIR.TARGET.CLASSIFICATION.RULE rule, the following conditions must be true:

- Air target A1 is in class targets, and
- The status of air target A1 is detected, and
- The speed of air target A1 greater than 100 kts, and
- The characteristic of air target A1 is air,

then we can reach the conclusion "The classification of air target A1 is air."

```
((AIR.TARGET.CLASSIFICATION.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE STATUS OF ?TARGET IS DETECTED)
    (LISP (> (THE SPEED OF ?TARGET) 100))
    (THE CHARACTERISTIC OF ?TARGET IS AIR)
    THEN
    (THE CLASSIFICATION OF ?TARGET IS AIR)))
 (SUBSURFACE.TARGET.CLASSIFICATION.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE STATUS OF ?TARGET IS DETECTED)
    (LISP (<= (THE SPEED OF ?TARGET) 50))
    (THE CHARACTERISTIC OF ?TARGET IS SUBSURFACE)
    THEN
    (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)))
 (SURFACE.TARGET.CLASSIFICATION.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE STATUS OF ?TARGET IS DETECTED)
    (LISP (< (THE SPEED OF ?TARGET) 100))
    (THE CHARACTERISTIC OF ?TARGET IS SURFACE)
    THEN
    (THE CLASSIFICATION OF ?TARGET IS SURFACE))))
```

Figure 5-4. TARGET.CLASSIFICATION.RULES in RULES.3

b. *RULES.4 in Knowledge Base*

Figure 5-5 shows the TRACKING.SENSOR.RULES subclass rule in RULES.4. This subclass rule shows what kind sensor is tracking on the target. For example, to apply the S/A.R rule, if the following conditions are true:

- Surface ship Radar is in class sensors, and
- The status of surface ship Radar is on, and
- The status of air target A1 is detected, and

- The **relative range** of air target A1 less than or equal to the **maximum detective range** of surface ship Radar, and
- The **characteristic** of surface ship Radar is air, and
- DDGB is in class forces, and
- The **characteristic** of air target A1 is air, and
- The **sensor** of DDGB is surface ship Radar,

then we can reach the conclusion "The tracking sensor of DDGB is surface ship Radar."

```
((S/A.R
  (IF (?PLATFORM IS IN CLASS FORCES)
    (?SENSOR IS IN CLASS SENSORS)
    (THE STATUS OF ?SENSOR IS ON)
    (THE STATUS OF ?TARGET IS DETECTED)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.DETECTIVE.RANGE OF ?SENSOR)))
    (THE CHARACTERISTIC OF ?SENSOR IS AIR)
    (THE CHARACTERISTIC OF ?TARGET IS AIR)
    (THE SENSOR OF ?PLATFORM IS ?SENSOR)
    THEN
    (THE TRACKING.SENSOR OF ?PLATFORM IS ?SENSOR)))
 (S/S.R
  (IF (?PLATFORM IS IN CLASS FORCES)
    (?SENSOR IS IN CLASS SENSORS)
    (THE STATUS OF ?SENSOR IS ON)
    (THE STATUS OF ?TARGET IS DETECTED)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.DETECTIVE.RANGE OF ?SENSOR)))
    (LISP (< 0 (THE RELATIVE.RANGE OF ?TARGET)))
    (THE CHARACTERISTIC OF ?SENSOR IS SURFACE)
    (THE CHARACTERISTIC OF ?TARGET IS SURFACE)
    (THE SENSOR OF ?PLATFORM IS ?SENSOR)
    THEN
    (THE TRACKING.SENSOR OF ?PLATFORM IS ?SENSOR)))
 (S/U.R
  (IF (?PLATFORM IS IN CLASS FORCES)
    (?SENSOR IS IN CLASS SENSORS)
    (THE STATUS OF ?SENSOR IS ON)
    (THE STATUS OF ?TARGET IS DETECTED)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.DETECTIVE.RANGE OF ?SENSOR)))
    (THE CHARACTERISTIC OF ?SENSOR IS SUBSURFACE)
    (THE CHARACTERISTIC OF ?TARGET IS SUBSURFACE)
    (THE SENSOR OF ?PLATFORM IS ?SENSOR)
    THEN
    (THE TRACKING.SENSOR OF ?PLATFORM IS ?SENSOR))))
```

Figure 5-5. TRACKING.SENSOR.RULES in RULES.4

Figure 5-6 shows the WEAPON.SUGGESTION.RULES.1 subclass rule in RULES.4. This subclass rule is used to generate the weapon suggestion instructions. The suggested on board weapons are to be assigned to the targets. For example, to apply the S/A.W rule, if the following conditions are true:

- **Air target A1** is in class targets, and
- **DDGB** is in class forces, and
- The **identification** of air target A1 is hostile, and

```
((S/A.W
  (IF (?TARGET IS IN CLASS TARGETS)
    (?PLATFORM IS IN CLASS FORCES)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
    (THE CHARACTERISTIC OF ?WEAPON IS S/A)
    (THE REFERENCES OF ?PLATFORM IS INTERESTED)
    (THE WEAPON OF ?PLATFORM IS ?WEAPON)
    (THE WEAPON.STATUS OF ?WEAPON IS AVAILABLE)
    (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
    (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
    THEN
    (THE BEARING.SUGGESTION OF ?WEAPON IS ?BEARING)
    (THE RANGE.SUGGESTION OF ?WEAPON IS ?RANGE)
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/A))
    (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE))))
 (S/S.W
  (IF (?TARGET IS IN CLASS TARGETS)
    (?PLATFORM IS IN CLASS FORCES)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
    (THE CHARACTERISTIC OF ?WEAPON IS S/S)
    (THE REFERENCES OF ?PLATFORM IS INTERESTED)
    (THE WEAPON OF ?PLATFORM IS ?WEAPON)
    (THE WEAPON.STATUS OF ?WEAPON IS AVAILABLE)
    (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
    (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
    THEN
    (THE BEARING.SUGGESTION OF ?WEAPON IS ?BEARING)
    (THE RANGE.SUGGESTION OF ?WEAPON IS ?RANGE)
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/S))
    (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE))))
 (S/U.W
  (IF (?TARGET IS IN CLASS TARGETS)
    (?PLATFORM IS IN CLASS FORCES)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
    (THE CHARACTERISTIC OF ?WEAPON IS S/U)
    (THE REFERENCES OF ?PLATFORM IS INTERESTED)
    (THE WEAPON OF ?PLATFORM IS ?WEAPON)
    (THE WEAPON.STATUS OF ?WEAPON IS AVAILABLE)
    (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
    (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
    THEN
    (THE BEARING.SUGGESTION OF ?WEAPON IS ?BEARING)
    (THE RANGE.SUGGESTION OF ?WEAPON IS ?RANGE)
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/U))
    (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE)))))
```

Figure 5-6. Weapon Suggestion Rules for Single Target Appearing in Different Dimensions

- The classification of air target A1 is air, and
- The relative range of air target A1 less then or equal to the maximum firing range of SAM (Surface to Air Missile), and
- The relative range of air target A1 greater then or equal to the minimum firing range of SAM, and

- The **relative bearing** of air target A1 less than or equal to the **up limit firing sector** of SAM, and
- The **relative bearing** of air target A1 greater than or equal to the **down limit firing sector** of SAM, and
- The **characteristic** of SAM is s/a, and
- The **reference** of DDGB is interested, and
- The **weapon** of DDGB is SAM, and
- The **weapon status** of SAM is available, and
- The **relative bearing** of air target A1 is X degree and
- The **relative range** of air target A1 is Y mile,

then we can reach the conclusion "The SAM is suggested for use against air target A1."

Figure 5-7 and Figure 5-8 are the WEAPON.SUGGESTION.RULES.2 subclass rules. Figure 5-9 is WEAPON.PRIORITY.RULES subclass rules. Both are the subclass rules of RULES.4.

WEAPON.ASSIGNMENT.RULES.2 are used when there is an occurrence of multiple targets. The weapon suggestions are based on the target threat classes in conjunction with the weapon priority to formulate optimal weaponry deployment against the targets. The WEAPON.PRIORITY.RULES are designed to be employed when the relative range of the target is less than the efficient minimum firing range of the assigned weapon. It changed so as to maintain defensive integrity.

```
((S/A.W.RESLECT
  (IF (?TARGET IS IN CLASS TARGETS)
    (?PLATFORM IS IN CLASS FORCES)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
    (THE CHARACTERISTIC OF ?WEAPON IS S/A)
    (THE REFERENCES OF ?PLATFORM IS INTERESTED)
    (THE WEAPON OF ?PLATFORM IS ?WEAPON)
    (THE WEAPON.STATUS OF ?WEAPON IS NOT.AVAILABLE)
    (LISP (EQUAL (THE TOTAL.A.THREAT OF ?TARGET) (THE WEAPON.PRIORITY OF ?WEAPON)))
    (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
    (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
    (THE WEAPON.ASSIGN OF ?WEAPON IS S/A)
    THEN
    (LISP (PUT.VALUE ?WEAPON 'BEARING.SUGGESTION ?BEARING))
    (LISP (PUT.VALUE ?WEAPON 'RANGE.SUGGESTION ?RANGE))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/A))
    (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE))))
 (S/U.W.RESLECT
  (IF (?TARGET IS IN CLASS TARGETS)
    (?PLATFORM IS IN CLASS FORCES)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
    (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
    (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
    (THE CHARACTERISTIC OF ?WEAPON IS S/U)
    (THE REFERENCES OF ?PLATFORM IS INTERESTED)
    (THE WEAPON OF ?PLATFORM IS ?WEAPON)
    (THE WEAPON.STATUS OF ?WEAPON IS NOT.AVAILABLE)
    (LISP (EQUAL (THE TOTAL.U.THREAT OF ?TARGET) (THE WEAPON.PRIORITY OF ?WEAPON)))
    (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
    (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
    THEN
    (LISP (PUT.VALUE ?WEAPON 'BEARING.SUGGESTION ?BEARING))
    (LISP (PUT.VALUE ?WEAPON 'RANGE.SUGGESTION ?RANGE))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/U))
    (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
    (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE)))))
```

Figure 5-7. Weapon Suggestion Rules for Multiple Air and Subsurface Targets

```
(S/S.W1.RESLECT
 (IF (?TARGET IS IN CLASS TARGETS)
   (?PLATFORM IS IN CLASS FORCES)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS AIR)
   (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
   (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
   (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
   (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
   (THE CHARACTERISTIC OF ?WEAPON IS S/A)
   (THE REFERENCES OF ?PLATFORM IS INTERESTED)
   (THE WEAPON OF ?PLATFORM IS ?WEAPON)
   (THE WEAPON.STATUS OF ?WEAPON IS NOT.AVAILABLE)
   (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
   (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
   (THE WEAPON.ASSIGN OF ?WEAPON IS S/S)
   THEN
   (LISP (PUT.VALUE ?WEAPON 'BEARING.SUGGESTION ?BEARING))
   (LISP (PUT.VALUE ?WEAPON 'RANGE.SUGGESTION ?RANGE))
   (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/A))
   (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
   (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE))))
(S/S.W2.RESLECT
 (IF (?TARGET IS IN CLASS TARGETS)
   (?PLATFORM IS IN CLASS FORCES)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS AIR)
   (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
   (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
   (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
   (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON))
)
   (THE CHARACTERISTIC OF ?WEAPON IS S/A)
   (THE REFERENCES OF ?PLATFORM IS INTERESTED)
   (THE WEAPON OF ?PLATFORM IS ?WEAPON)
   (THE WEAPON.STATUS OF ?WEAPON IS NOT.AVAILABLE)
   (LISP (EQUAL (THE TOTAL.S.THREAT OF ?TARGET) (THE WEAPON.PRIORITY OF ?WEAPON)))
   (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
   (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
   (THE WEAPON.ASSIGN OF ?WEAPON IS S/S)
   THEN
   (LISP (PUT.VALUE ?WEAPON 'BEARING.SUGGESTION ?BEARING))
   (LISP (PUT.VALUE ?WEAPON 'RANGE.SUGGESTION ?RANGE))
   (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/A))
   (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
   (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE))))
(S/S.W3.RESLECT
 (IF (?TARGET IS IN CLASS TARGETS)
   (?PLATFORM IS IN CLASS FORCES)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS SURFACE)
   (LISP (<= (THE RELATIVE.RANGE OF ?TARGET) (THE MAX.FIRING.RANGE OF ?WEAPON)))
   (LISP (>= (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
   (LISP (<= (THE RELATIVE.BEARING OF ?TARGET) (THE UP.LIMIT.FIRING.SECTOR OF ?WEAPON)))
   (LISP (>= (THE RELATIVE.BEARING OF ?TARGET) (THE DOWN.LIMIT.FIRING.SECTOR OF ?WEAPON)) !
)
   (THE CHARACTERISTIC OF ?WEAPON IS S/S)
   (THE REFERENCES OF ?PLATFORM IS INTERESTED)
   (THE WEAPON OF ?PLATFORM IS ?WEAPON)
   (THE WEAPON.STATUS OF ?WEAPON IS NOT.AVAILABLE)
   (LISP (EQUAL (THE TOTAL.S.THREAT OF ?TARGET) (THE WEAPON.PRIORITY OF ?WEAPON)))
   (THE RELATIVE.BEARING OF ?TARGET IS ?BEARING)
   (THE RELATIVE.RANGE OF ?TARGET IS ?RANGE)
   (THE WEAPON.ASSIGN OF ?WEAPON IS S/S)
   THEN
   (LISP (PUT.VALUE ?WEAPON 'BEARING.SUGGESTION ?BEARING))
   (LISP (PUT.VALUE ?WEAPON 'RANGE.SUGGESTION ?RANGE))
   (LISP (PUT.VALUE ?WEAPON 'WEAPON.ASSIGN 'S/S))
   (LISP (PUT.VALUE ?WEAPON 'SUGGEST.TO ?TARGET))
   (LISP (PUT.VALUE ?WEAPON 'WEAPON.STATUS 'NOT.AVAILABLE))))
```

Figure 5-8. Weapon Suggestion Rules for Multiple Surface Targets

```
((S/A.WEAPON.PRIORITY.EXCHANGE.R
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (?WEAPON IS IN CLASS WEAPONS)
    (THE CHARACTERISTIC OF ?WEAPON IS S/A)
    (LISP (< (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    THEN
    (LISP (PUT.VALUE 'SAM 'WEAPON.PRIORITY 'AIR2))
    (LISP (PUT.VALUE '76MM 'WEAPON.PRIORITY 'AIR1))
    (LISP (ADD.VALUE '76MM 'WEAPON.PRIORITY 'SUF1))))
 (S/S.WEAPON.PRIORITY.EXCHANGE.R
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (?WEAPON IS IN CLASS WEAPONS)
    (THE CHARACTERISTIC OF ?WEAPON IS S/S)
    (LISP (< (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    THEN
    (LISP (PUT.VALUE 'SSM 'WEAPON.PRIORITY 'SUF2))
    (LISP (PUT.VALUE '76MM 'WEAPON.PRIORITY 'SUF1))
    (LISP (ADD.VALUE '76MM 'WEAPON.PRIORITY 'AIR1))))
 (S/U.WEAPON.PRIORITY.EXCHANGE.R
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
    (?WEAPON IS IN CLASS WEAPONS)
    (THE CHARACTERISTIC OF ?WEAPON IS S/U)
    (LISP (< (THE RELATIVE.RANGE OF ?TARGET) (THE MIN.FIRING.RANGE OF ?WEAPON)))
    THEN
    (LISP (PUT.VALUE 'ASROC 'WEAPON.PRIORITY 'SUB2))
    (LISP (PUT.VALUE 'ASW.TORPEDO-1 'WEAPON.PRIORITY 'SUB1))
    (LISP (PUT.VALUE 'ASW.TORPEDO-2 'WEAPON.PRIORITY 'SUB1)))))
```

Figure 5-9. WEAPON.PRIORITY.RULES in RULES.4

c. RULES.5 in Knowledge Base

There are five subclass rules in RULES.5 (see Figure 5-1). The major function of RULES.5 is to calculate the threat class of the target and assign a threat class to the corresponding target. This rule is utilized in those situations where multiple targets appear, because of the obvious need in such a context to use the appropriate weapon against the respective targets. The following example shows how the SORT.A.RULES rule in RULES.5 works in a given condition. The SORT.A.RULES in RULES.5 is shown in Figure 5-10. If the following conditions are true:

- **Air target A1** is in class targets, and
- The **identification** of air target A1 is hostile, and

- The classification of air target A1 is air, and
- Sort (from large to small) threat.a.value of unit class targets and store into variable AW, and
- The slot **threat.1** of unit air target A1 equal to the first value of variable AW,

then we can reach the conclusion "The threat class of air target A1 is AIR1."

```
((SORT.A1
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.A.VALUE) NIL)))
    (LISP (SETF AW (GET.VALUES 'TARGETS 'THREAT.A.VALUE)))
    (LISP (SORT AW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.1) (FIRST AW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.A.THREAT 'AIR1))))
 (SORT.A2
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.A.VALUE) NIL)))
    (LISP (SETF AW (GET.VALUES 'TARGETS 'THREAT.A.VALUE)))
    (LISP (SORT AW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.1) (SECOND AW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.A.THREAT 'AIR2))))
 (SORT.A3
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.A.VALUE) NIL)))
    (LISP (SETF AW (GET.VALUES 'TARGETS 'THREAT.A.VALUE)))
    (LISP (SORT AW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.1) (THIRD AW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.A.THREAT 'AIR3))))
 (SORT.A4
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.A.VALUE) NIL)))
    (LISP (SETF AW (GET.VALUES 'TARGETS 'THREAT.A.VALUE)))
    (LISP (SORT AW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.1) (FOURTH AW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.A.THREAT 'AIR4))))
 (SORT.A5
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.A.VALUE) NIL)))
    (LISP (SETF AW (GET.VALUES 'TARGETS 'THREAT.A.VALUE)))
    (LISP (SORT AW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.1) (FIFTH AW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.A.THREAT 'AIR5)))))
```

Figure 5-10. SORT.A.RULES in RULES.5

It is important that we sort different types of targets by different rules. SORT.A.RULES are for air targets, SORT.U.RULES are for subsurface targets (see Figure 5-11) and SORT.S.RULES are for surface targets (see Figure 5-12). In order to efficiently match the available weaponry in the platform

DDGB (Guided Missile Destroyer) we use three different sort rules in the performance of the sorting function. From Chapter IV we have five weapons available for air targets, five weapons available for surface targets and three weapons available for subsurface targets. Therefore, we only assign threat class for air targets from AIR1 ~ AIR5, surface targets from SUF1 ~ SUF5 and subsurface targets from SUB1 ~ SUB3. The threat value is calculated by the following equation:

threat value = (target's speed) + (2500/(target's relative range + 0.01))

+ ((target's relative bearing)/1000)

```
(SORT.U1
 (IF (?TARGET IS IN CLASS TARGETS)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
   (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.U.VALUE) NIL)))
   (LISP (SETF UW (GET.VALUES 'TARGETS 'THREAT.U.VALUE)))
   (LISP (SORT UW #'>))
   (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.3) (FIRST UW)))
   THEN
   (LISP (PUT.VALUE ?TARGET 'TOTAL.U.THREAT 'SUB1))))
(SORT.U2
 (IF (?TARGET IS IN CLASS TARGETS)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
   (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.U.VALUE) NIL)))
   (LISP (SETF UW (GET.VALUES 'TARGETS 'THREAT.U.VALUE)))
   (LISP (SORT UW #'>))
   (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.3) (SECOND UW)))
   THEN
   (LISP (PUT.VALUE ?TARGET 'TOTAL.U.THREAT 'SUB2))))
(SORT.U3
 (IF (?TARGET IS IN CLASS TARGETS)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
   (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.U.VALUE) NIL)))
   (LISP (SETF UW (GET.VALUES 'TARGETS 'THREAT.U.VALUE)))
   (LISP (SORT UW #'>))
   (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.3) (THIRD UW)))
   THEN
   (LISP (PUT.VALUE ?TARGET 'TOTAL.U.THREAT 'SUB3)))))
```

Figure 5-11. SORT.U.RULES in RULES.5

This is a general equation based on the target's data and we assume it can satisfy our operational considerations (in fact, for different operational considerations the threat value equation must be different). Users can change the constant on each term. For example, if speed is an especially salient factor, then it can be assigned a greater relative value than range in the equation. The bearing term is used to guarantee that no more than one object will be assigned location at the same point and time. We will discuss how the threat value been calculated in weapon suggestion system in section D (see p. 54).

```
((SORT.S1
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.S.VALUE) NIL)))
    (LISP (SETF SW (GET.VALUES 'TARGETS 'THREAT.S.VALUE)))
    (LISP (SORT SW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.2) (FIRST SW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.S.THREAT 'SUF1))))
 (SORT.S2
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.S.VALUE) NIL)))
    (LISP (SETF SW (GET.VALUES 'TARGETS 'THREAT.S.VALUE)))
    (LISP (SORT SW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.2) (SECOND SW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.S.THREAT 'SUF2))))
 (SORT.S3
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.S.VALUE) NIL)))
    (LISP (SETF SW (GET.VALUES 'TARGETS 'THREAT.S.VALUE)))
    (LISP (SORT SW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.2) (THIRD SW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.S.THREAT 'SUF3))))
 (SORT.S4
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.S.VALUE) NIL)))
    (LISP (SETF SW (GET.VALUES 'TARGETS 'THREAT.S.VALUE)))
    (LISP (SORT SW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.2) (FOURTH SW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.S.THREAT 'SUF4))))
 (SORT.S5
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (LISP (NOT (EQUAL (GET.VALUES 'TARGETS 'THREAT.S.VALUE) NIL)))
    (LISP (SETF SW (GET.VALUES 'TARGETS 'THREAT.S.VALUE)))
    (LISP (SORT SW #'>))
    (LISP (EQUAL (GET.VALUE ?TARGET 'THREAT.2) (FIFTH SW)))
    THEN
    (LISP (PUT.VALUE ?TARGET 'TOTAL.S.THREAT 'SUF5)))))
```

Figure 5-12. SORT.S.RULES in RULES.5

3. User Interface

KEE provides graphical tools, called KEEpictures, for interface design. They simplify data input and generate output display that allows the user to interact with the system easily. Figure 5-13 shows the user interface of the

WSS. Each block in Figure 5-13 is called a unit. The units for data input (in the bottom of Figure 5-13) can be accessed by mouse or keyboard. The reminder units serve as output display (i.e. the suggest instructions for a specified case). Generally, we can input the target data by clicking on the data input unit and then select the data from the submenu, or type in the data from the keyboard if there are no submenu appear. The functions of each of the data input and output display units are described below:

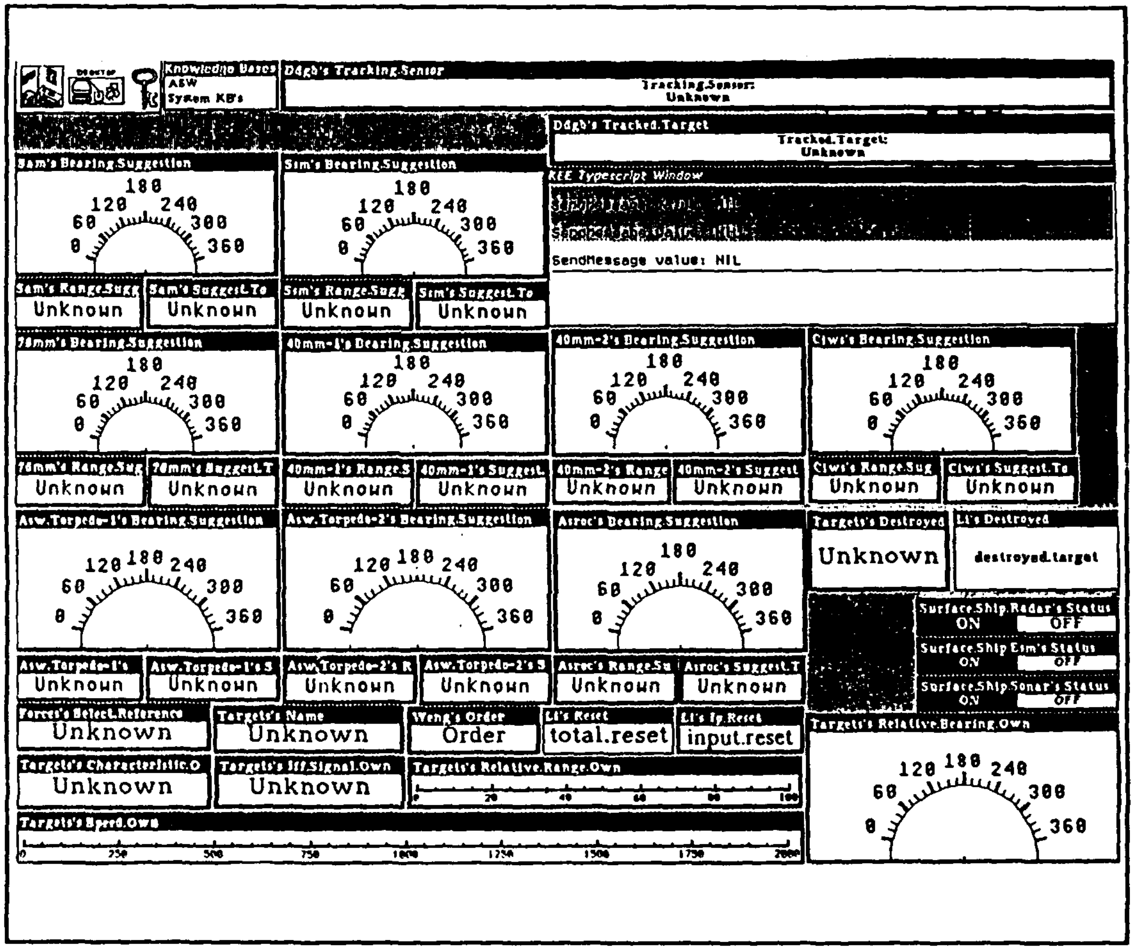

Figure 5-13. User Interface of WSS

a. Data Input Units

- *Forces's Select.Reference* unit: to engage the selected platform in the knowledge base. For the purposes of this thesis we have only used the DDGB in the knowledge base.
- *Target's Name* unit: to designate a name to the target appearing on the sensor. The name can be any characters or symbols.
- *Weng's Order* unit: to start our simulation for a given data input.
- *Li's Reset unit:* to reset the weapon suggestion system by clicking it.
- *Li's IP.Reset* unit: to reset the data input unit by clicking it.
- *Targets's Characteristic.Own* unit: the characteristic of the target (i.e air, surface or subsurface). In fact, we can deduce the target characteristic from the sensor input. For example, if the target is detected by Sonar then we can assume the target characteristic should be subsurface. I^f the target is detected by s/a Radar then the target characteristic should be air, etc.
- *Targets's Iff.Signal.Own* unit: the IFF signal identification of the target. If it is a hostile target the IFF signal should be different, otherwise the IFF signal will be same. Both this and *Targets's Characteristic.Own* units, can be accessed by first clicking on it then selecting the input data from the submenu.
- *Targets's Relative.Range.Own* unit: the range (in miles) between the ship and target.
- *Targets's Speed.Own* unit: the speed (in knots) of the target.
- *Targets's Relative.Bearing.Own* unit: the bearing between our own ships heading and the target (the unit is degree). This unit and *Targets's Relative.Range.Own* as well as *Targets's Speed.Own* units can be accessed by clicking on the horizontal bar or the dial at any scale.
- *Surface.Ship.Radar's Status* unit: to set the surface ship Radar (on or off).
- *Surface.Ship.Esm's Status* unit: to set the surface ship ESM (on or off).
- *Surface.Ship.Sonar's Status* unit: to set the surface ship Sonar (on or off). The three status units can be accessed by clicking it on. If the user wishes to activate the surface ship Radar they can use the mouse to click the 'on' part of the screen to the corresponding units. The device is 'on' when the background screen has become white.

- ***Targets's Destroyed*** unit: to input the name of a target that has been destroyed. This allows the target name to be deleted from the knowledge base.
- ***Li's Destroyed*** unit: an actuator to start the delete procedure of the destroyed target.

b. Output Display Units

- ***Sam's Bearing.Suggestion*** unit: the display of bearing suggestion of SAM.
- ***Sam's Range.Suggestion*** unit: the display of range suggestion of SAM.
- ***Sam's Suggest.To*** unit: the display of the target to which SAM is suggested.
- ***Ssm's Bearing.Suggestion*** unit: the display of bearing suggestion of SSM.
- ***Ssm's Range.Suggestion*** unit: the display of range suggestion of SSM.
- ***Ssm's Suggest.To*** unit: the display of the target to which SSM is suggested.
- ***76mm's Bearing.Suggestion*** unit: the display of bearing suggestion of 76mm.
- ***76mm's Range.Suggestion*** unit: the display of range suggestion of 76mm.
- ***76mm's Suggest.To*** unit: the display of the target to which 76mm is suggested.
- ***40mm-1's Bearing.Suggestion*** unit: the display of bearing suggestion of 40mm-1.
- ***40mm-1's Range.Suggestion*** unit: the display of range suggestion of 40mm-1.
- ***40mm-1's Suggest.To*** unit: the display of the target to which 40mm-1 is suggested.
- ***40mm-2's Bearing.Suggestion*** unit: the display of bearing suggestion of 40mm-2.
- ***40mm-2's Range.Suggestion*** unit: the display of range suggestion of 40mm-2.
- ***40mm-2's Suggest.To*** unit: the display of the target to which 40mm-2 is suggested.

- ***Ciws's Bearing.Suggestion*** unit: the display of bearing suggestion of CIWS.
- ***Ciws's Range.Suggestion*** unit: the display of range suggestion of CIWS.
- ***Ciws's Suggest.To*** unit: the display of the target to which CIWS is suggested.
- ***Asw.Torpedo-1's Bearing.Suggestion*** unit: the display of bearing suggestion of ASW/Torpedo-1.
- ***Asw.Torpedo-1's Range.Suggestion*** unit: the display of range suggestion of ASW/Torpedo-1.
- ***Asw.Torpedo-1's Suggest.To*** unit: the display of the target to which ASW/Torpedo-1 is suggested.
- ***Asw.Torpedo-2's Bearing.Suggestion*** unit: the display of bearing suggestion of ASW/Torpedo-2.
- ***Asw.Torpedo-2's Range.Suggestion*** unit: the display of range suggestion of ASW/Torpedo-2.
- ***Asw.Torpedo-2's Suggest.To*** unit: the display of the target to which ASW/Torpedo-2 is suggested.
- ***Asroc's Bearing.Suggestion*** unit: the display of bearing suggestion of ASROC.
- ***Asroc's Range.Suggestion*** unit: the display of range suggestion of ASROC.
- ***Asroc's Suggest.To*** unit: the display of the target to which ASROC is suggested.

C. SIMULATIONS

In this section we set up several test cases to demonstrate our weapon suggestion system. The targets will be added into the system one by one and the system will suggest the best response to each situation.

1. Case 1: Three Hostile Targets Appearing in Different Dimensions

For this situation the system suggestion is shown in Figure 5-14. In this case our sensors detect three unknown targets, one each from air, surface, and subsurface, which are approaching the ships DDGB. The targets are analyzed by sensors and the IFF device on board, as shown in Table 5-1, and the results are then fed to the WSS.

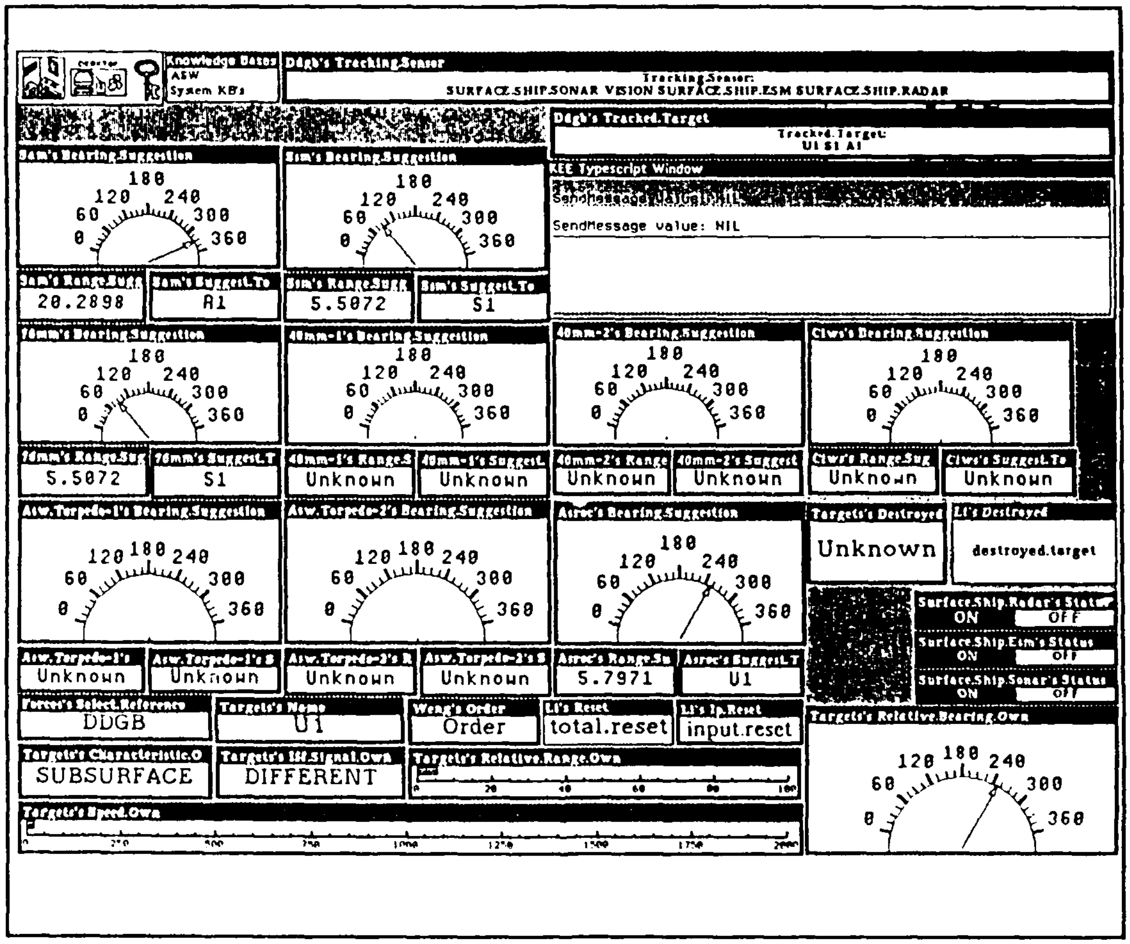

Figure 5-14. Simulation Display of Case 1

TABLE 5-1. TARGET PARAMETERS OF CASE 1

	target characteristic	IFF signal	relative range (mile)	relative bearing (degree)	target speed (knots)	threat class
A1	air	different	20.29	330	750	air1
S1	surface	different	5.22	091	48	suf1
U1	subsurface	different	6.1	246	25	sub1

We used the results, calculated by threat value equation (see Chapter IV p. 49), to decide the threat class for each target. The threat classes are listed in the table. Based on target and weapon parameters listed in Table 4-2, WSS display is shown Figure 5-14. SAM is suggested for use against A1, since it is the only weapon available and able to reach an air target. For surface target S1, because it is within range of both 76mm and SSM both weapons are suggested. For subsurface target U1 only the ASROC can be used.

2. Case 2: Multiple Hostile Targets Appearing in Air and Surface

In this case we have three air (A1~A3) and three surface (S1~S3) targets. The target parameters are listed in Table 5-2. The simulation results are displayed in Figure 5-15. The threat values have been calculated by the threat value equation. Based on the threat values for each targets we list the threat class in Table 5-2. The weapon suggestion for each target is described below:

- SAM was suggested for use against A2.
- 76mm, 40mm-2, and CIWS were suggested for use against A3.
- SSM was suggested for use against S3.
- 40mm-1 was suggested for use against S2.

TABLE 5-2. TARGET PARAMETERS OF CASE 2

	target characteristic	IFF signal	relative range (mile)	relative bearing (degree)	target speed (knots)	threat class
A1	air	different	10	030	750	air3
A2	air	different	5.2	097	750	air2
A3	air	different	1.45	285	750	air1
S1	surface	different	40	015	50	suf3
S2	surface	different	2.02	106	35	suf1
S3	surface	different	5.8	180	35	suf2

Since the system was defined to assign a deployment priority to air targets, the 76mm was used to counterstrike A3. On the other hand, because A1 and S1 do not satisfy the premises (e.g. they might be out of the maximum firing range of weapons or no weapons are available) for the rules, consequently there are no optimal weapons that can be suggested.

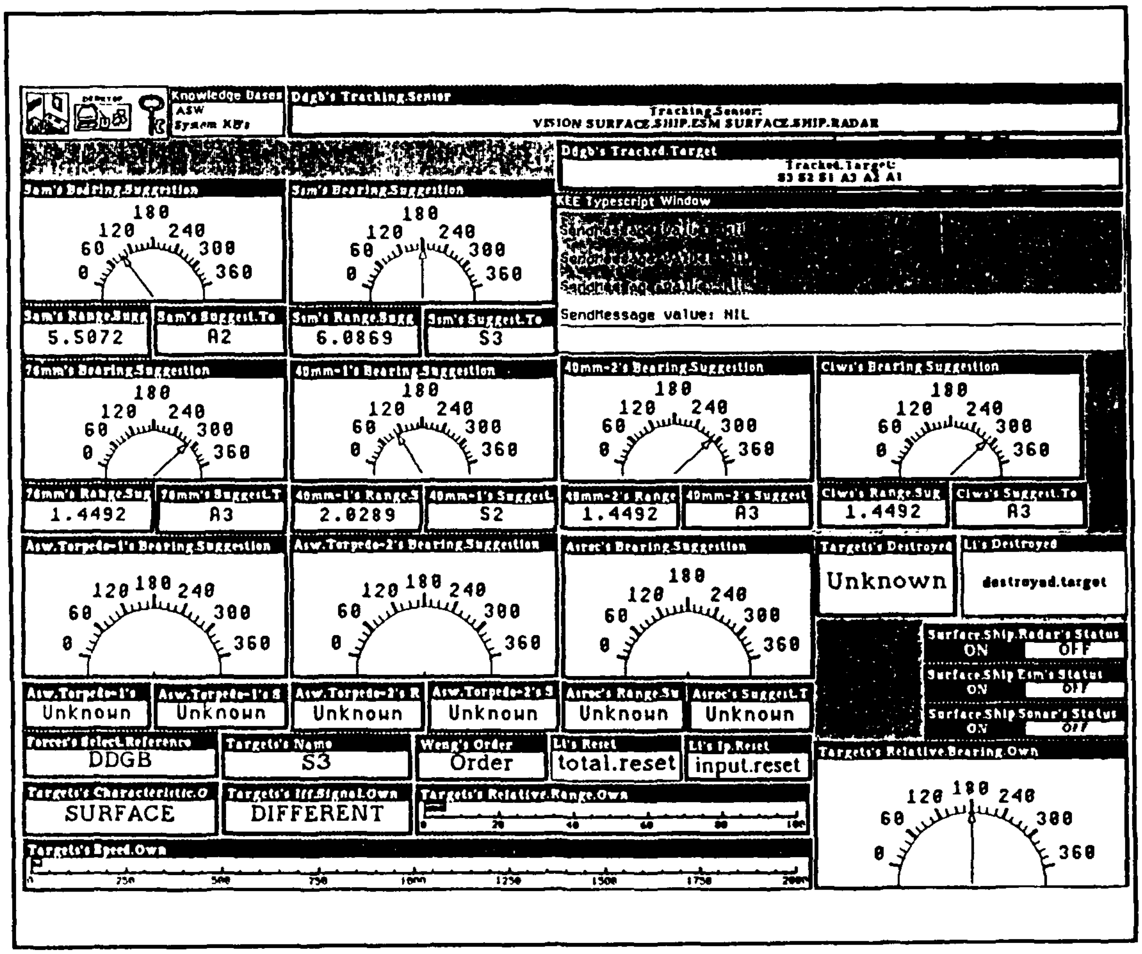

Figure 5-15. Simulation Display of Case 2

3. Case 3: Multiple Hostile Targets Appearing in Three Dimensions

In this case four air (A1~A4), four surface (S1~S4), and three subsurface (U1~U3) targets were input into the WSS. The target parameters are listed in Table 5-3. The simulation results are displayed in Figure 5-16. The threat class for each of the targets are listed in Table 5-3 also. The weapon deployment of this case is described below:

- SAM was suggested for use against A3.
- 76mm, 40mm-1, and CIWS were suggested for use against A4.
- SSM was suggested for use against S4.
- 40mm-2 was suggested for use against S3.
- ASROC and ASW/Torpedo-1 were suggested for use against U3.
- ASW/Torpedo-2 was suggested for use against U2.

For targets A1, A2, S1, S2, and U1 no weapons can be suggested, since they do not satisfy the premises for the rules (e.g. they might be a friend or no weapons available for use against them).

TABLE 5-3. TARGET PARAMETERS OF CASE 3

	target characteristic	IFF signal	relative range (mile)	relative bearing (degree)	target speed (knots)	threat class
A1	air	different	10	030	750	air3
A2	air	same	5	180	750	no
A3	air	different	4	285	750	air2
A4	air	different	1.5	090	700	air1
S1	surface	different	40	015	50	suf3
S2	surface	same	2.5	255	35	no
S3	surface	different	1.5	255	50	suf1
S4	surface	different	6	180	35	suf2
U1	subsurface	different	5	090	25	sub3
U2	subsurface	different	2.5	285	40	sub1
U3	subsurface	different	4	105	25	sub2

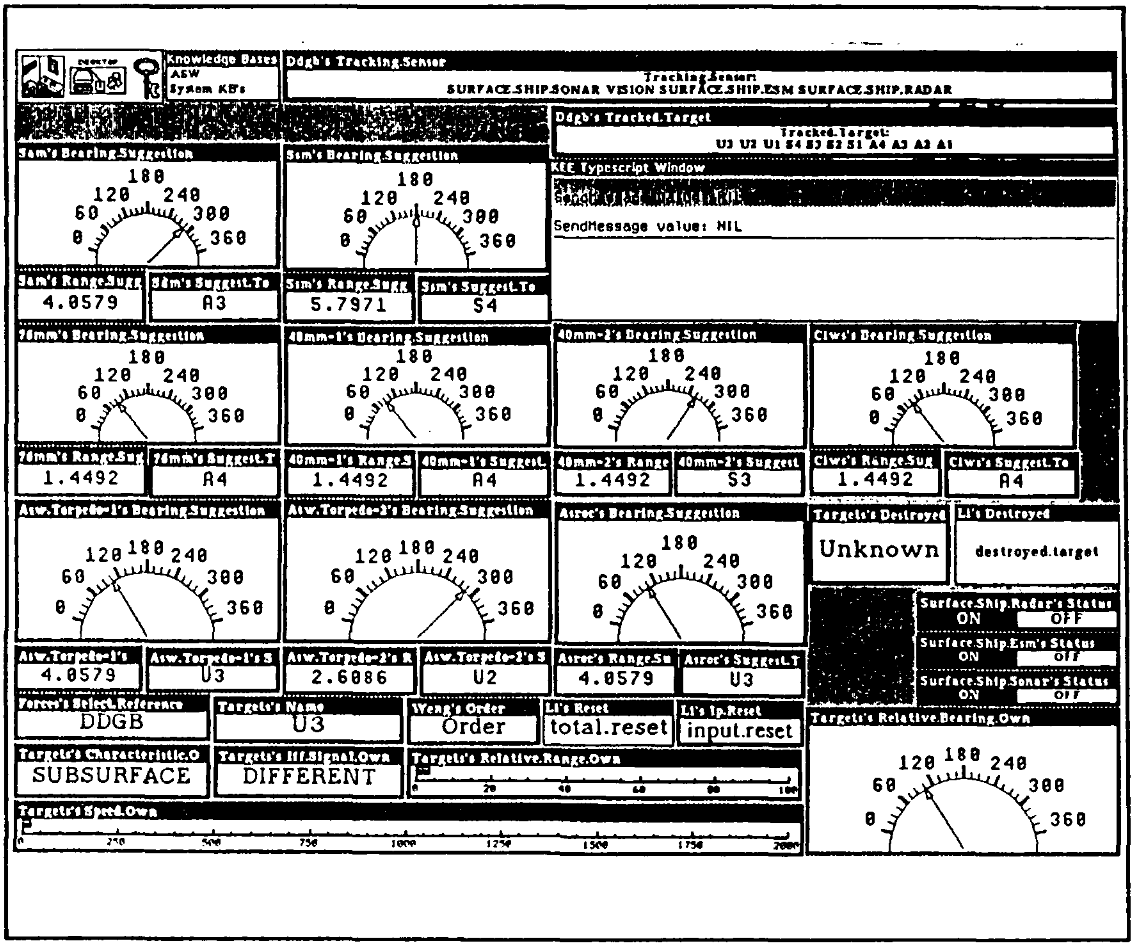

Figure 5-16. Simulation Display of Case 3

D. SOME IMPLEMENTATION PROBLEMS

1. How to Decide the Threat Class for Targets?

In section *b* we gave the threat value equation which is stored in the slot CALCULATE.A, CALCULATE.S and CALCULATE.U for air, surface, and subsurface targets, respectively. The CALCULATE.A.RULE, CALCULATE.S.RULE and CALCULATE.U.RULE in RULES.5 are used to collect the target data and send them to the corresponding slot used in the calculation of the threat value for the target. The threat value is then stored in the slot THREAT.1 (for air target), THREAT.2 (for surface target) or THREAT.3 (for subsurface target). Finally, all air target threat values will be collected into slot TOTAL.A.VALUE. Then, the values in TOTAL.A.VALUE slot will be sorted from large to small. The largest value in the slot means the corresponding target has the highest threat class. Similarly, the smallest value in TOTAL.A.VALUE slot means the corresponding target has the lowest threat class. For surface and subsurface targets the threat values will be collected into slot TOTAL.S.VALUE or TOTAL.U.VALUE. Using the same procedures we can generate and designate threat class to each corresponding targets. The CALCULATE.RULES and ADD.CALCULATE.RULES subclass rules are shown in Figure 5-17 and Figure 5-18.

```
(CALCULATE.A.RULE
 (IF (?TARGET IS IN CLASS TARGETS)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS AIR)
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'RELATIVE.RANGE) NIL)))
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'SPEED) NIL)))
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'RELATIVE.BEARING) NIL)))
   THEN
   (LISP (UNITMSG ?TARGET 'CALCULATE.A))))
(CALCULATE.S.RULE
 (IF (?TARGET IS IN CLASS TARGETS)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS SURFACE)
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'RELATIVE.RANGE) NIL)))
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'SPEED) NIL)))
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'RELATIVE.BEARING) NIL)))
   THEN
   (LISP (UNITMSG ?TARGET 'CALCULATE.S))))
(CALCULATE.U.RULE
 (IF (?TARGET IS IN CLASS TARGETS)
   (THE IDENTIFICATION OF ?TARGET IS FOE)
   (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'RELATIVE.RANGE) NIL)))
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'SPEED) NIL)))
   (LISP (NOT (EQUAL (GET.VALUES ?TARGET 'RELATIVE.BEARING) NIL)))
   THEN
   (LISP (UNITMSG ?TARGET 'CALCULATE.U)))))
```

Figure 5-17. CALCULATE.RULES in RULES.5

2. How to Create and Delete a Target from the Knowledge Base?

All the targets have been added into the knowledge base by means of the user interface (i.e. Targets's Name unit). From the Targets's Name unit we can input the target's name. Then, the RULES.1 rule will create the name as a instance which will be linked as a "child" of the class TARGETS in the knowledge base. Similarly, we use the same method to delete the destroyed targets from the knowledge base.

```
((ADD.A.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS AIR)
    (THE THREAT.1 OF ?TARGET IS ?VALUE)
    THEN
    (LISP (ADD.VALUE 'TARGETS 'THREAT.A.VALUE ?VALUE))))
 (ADD.S.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SURFACE)
    (THE THREAT.2 OF ?TARGET IS ?VALUE)
    THEN
    (LISP (ADD.VALUE 'TARGETS 'THREAT.S.VALUE ?VALUE))))
 (ADD.U.RULE
  (IF (?TARGET IS IN CLASS TARGETS)
    (THE IDENTIFICATION OF ?TARGET IS FOE)
    (THE CLASSIFICATION OF ?TARGET IS SUBSURFACE)
    (THE THREAT.3 OF ?TARGET IS ?VALUE)
    THEN
    (LISP (ADD.VALUE 'TARGETS 'THREAT.U.VALUE ?VALUE)))))
```

Figure 5-18. ADD.CALCULATE.RULES in RULES.5

VI. CONCLUSIONS

A. SUMMARY

This thesis focuses on using an expert system approach in designing an intelligent weapon suggestion system capable of assisting the Weaponry Department Head in making efficient and accurate decisions in critical war at sea scenarios.

In Chapter I we described the objectives of building such a system, as well as the attendant problems. We discussed some general background and related work in Chapter II and III. In Chapter IV we explained the basic concepts pertaining to the structure, design, and limitations of a WSS. In Chapter V we explored the developmental environment, knowledge base, and some special problems in the implementation and treatment of the system. The software implementation and simulation results were also presented in Chapter V as well.

B. FUTURE WORK

We, and many others, have shown that the tactical knowledge, reasoning, and decision making process within combat situations can be modelled by production rules and expert systems technology. However, it is beyond the scope of our design prototype to approach a fully integrated paradigm that would have included a more detailed analysis of the crucial interactions of the system with the combat system and its weaponry, which holds the promise of forming the basis of an automatic weapon assignment system a more powerful than the existing prototypes. In fact, the unstated goal of our design work on the WSS is to progress and evolve towards a fully integrated automatic weapon assignment system.

In keeping with the aspiration to develop an operational automatic weapon assignment system is the future, the following considerations necessitate solutions:

- How to establish the performance and data link between the WSS and its periphery?
- How to design and implement an automatic weapons system on board that achieves full utilization of the expert system?
- How to join and wholly integrate the considerations of tactical operations into the system?
- How to establish all the connections and linkages of electronic warfare into mechanical functions of the system?

APPENDIX. THE WSS FLOW CHART

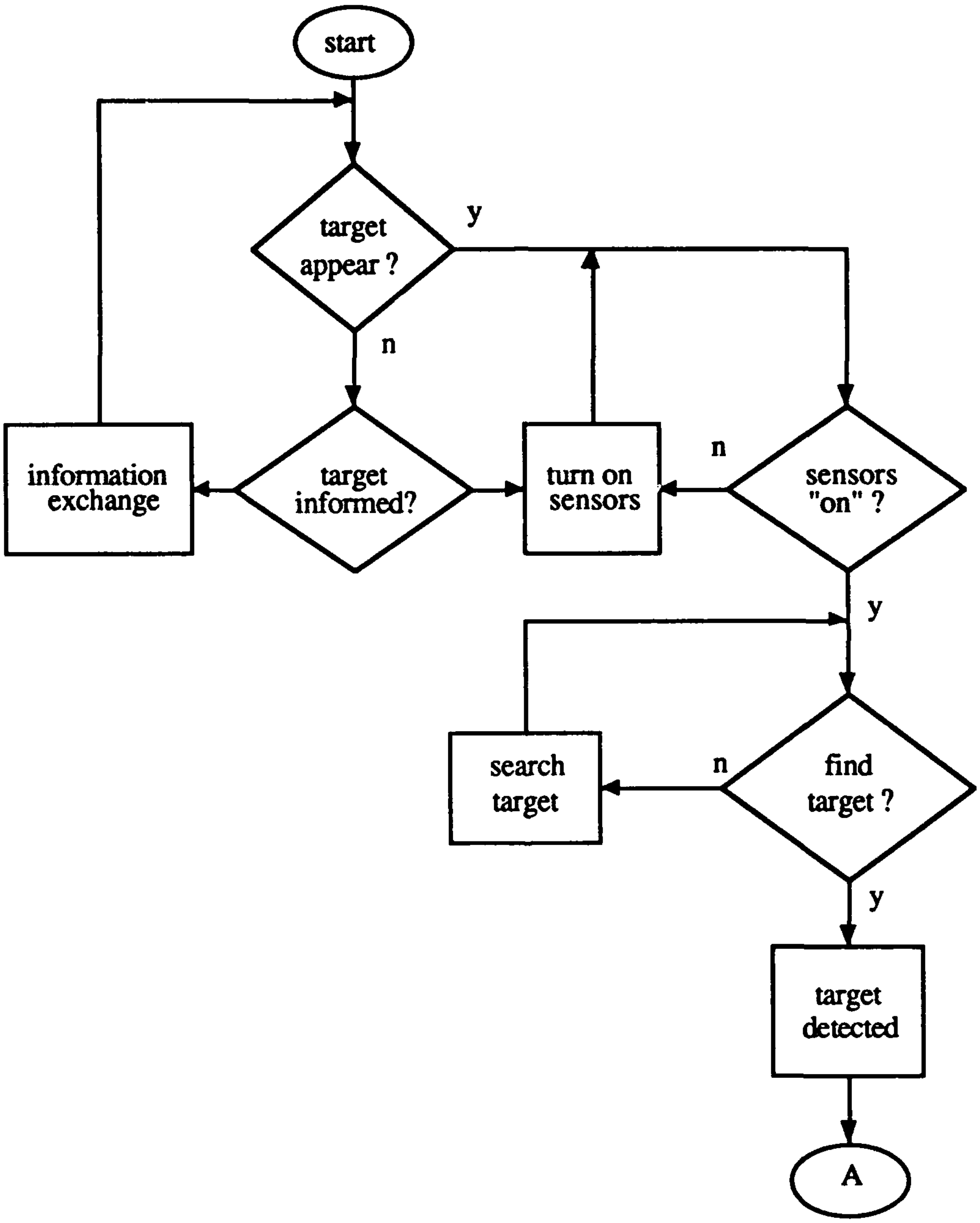

Figure A-1. Target Detected Flow Chart

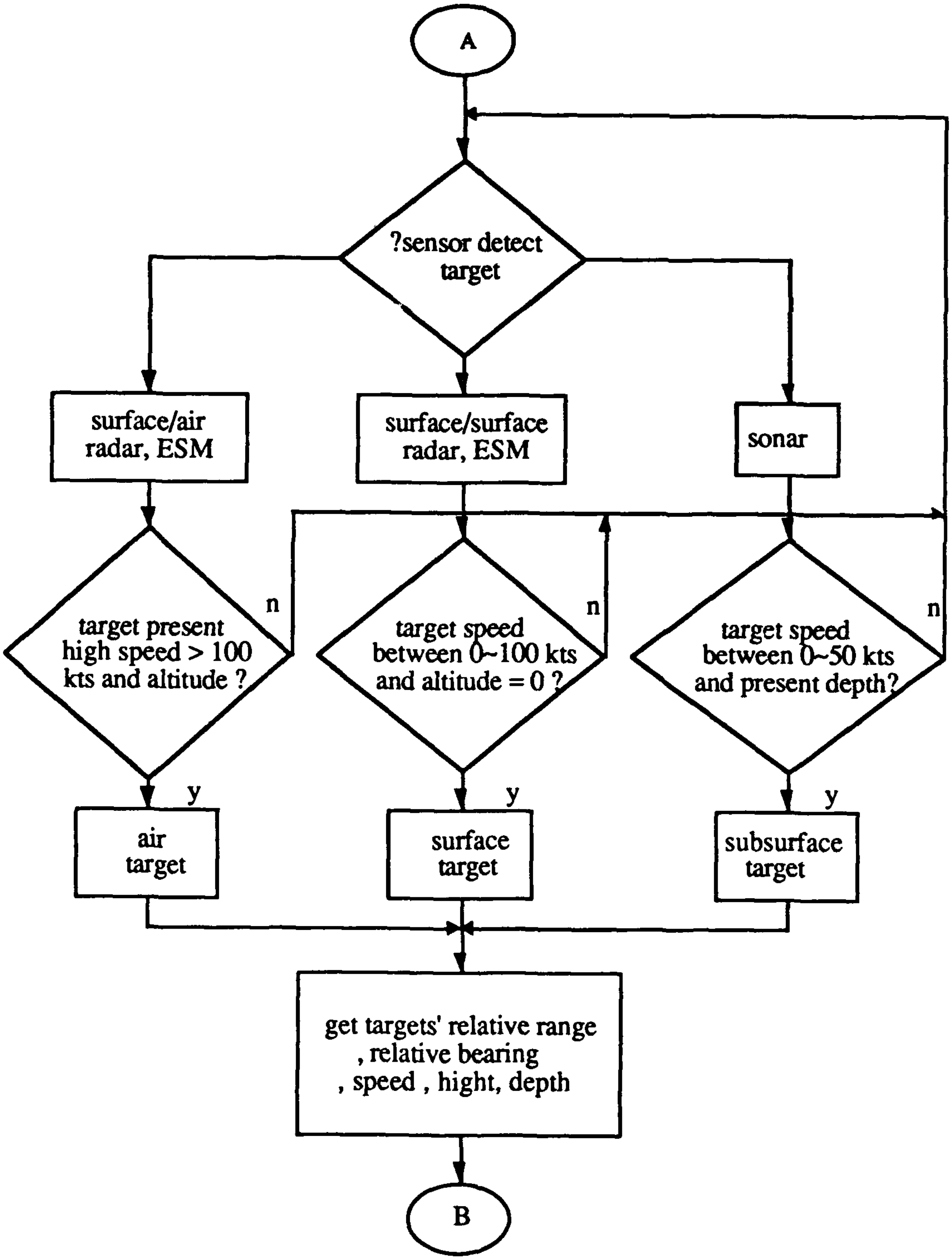

Figure A-2. Get Target Data Flow Chart

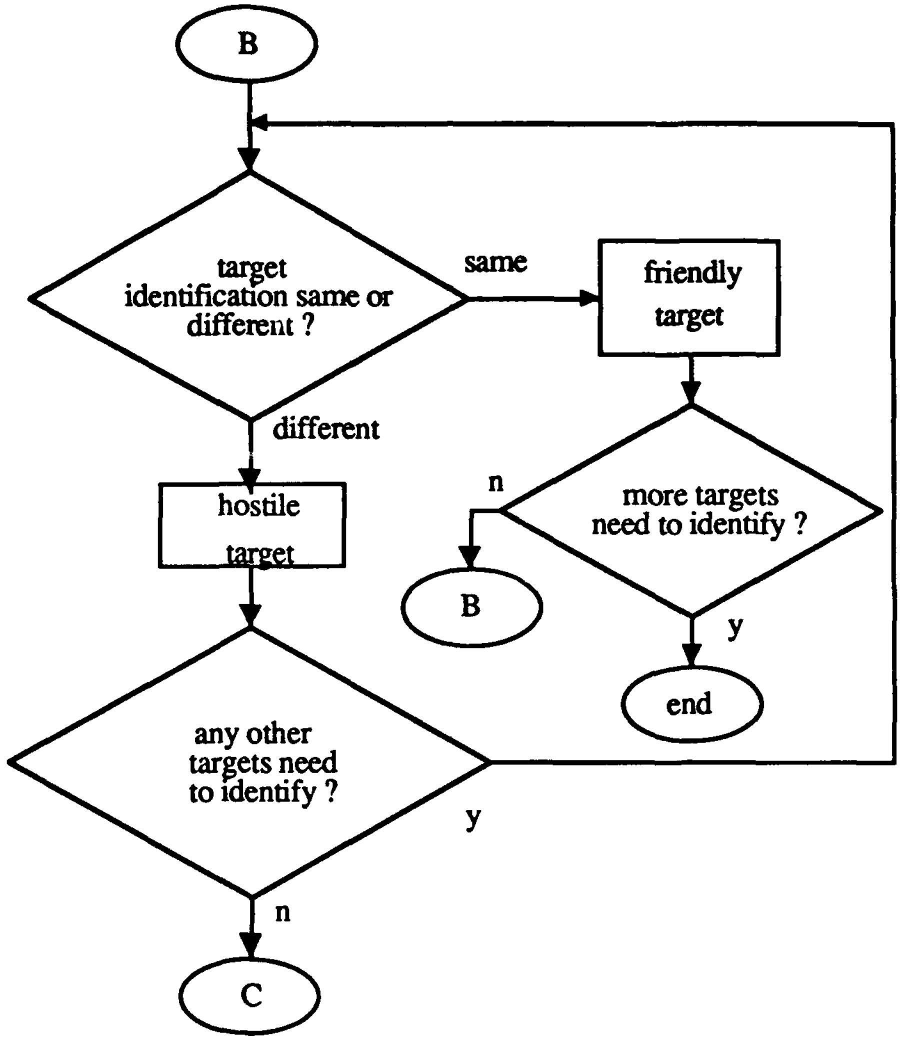

Figure A-3. Target Identification Flow Chart

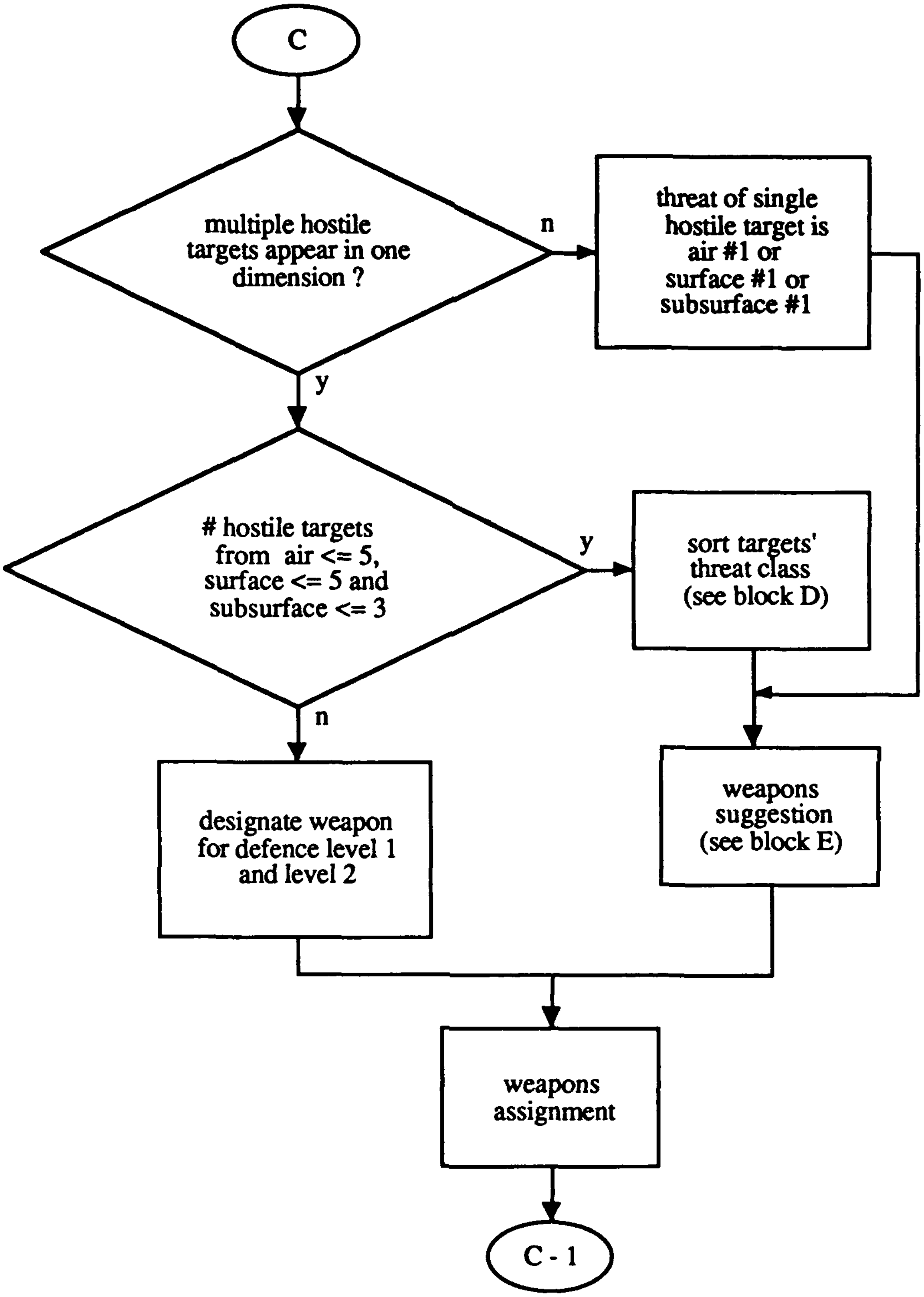

Figure A-4. Decide Number of Targets Flow Chart

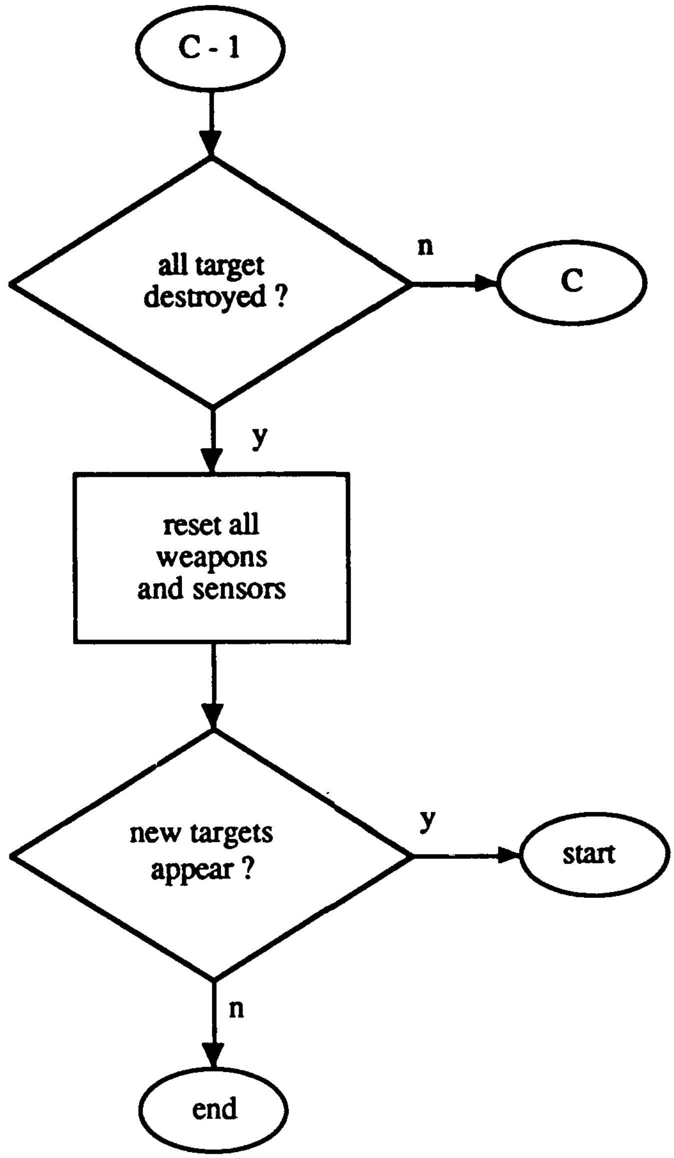

Figure A-5. Target Destroyed Flow Chart

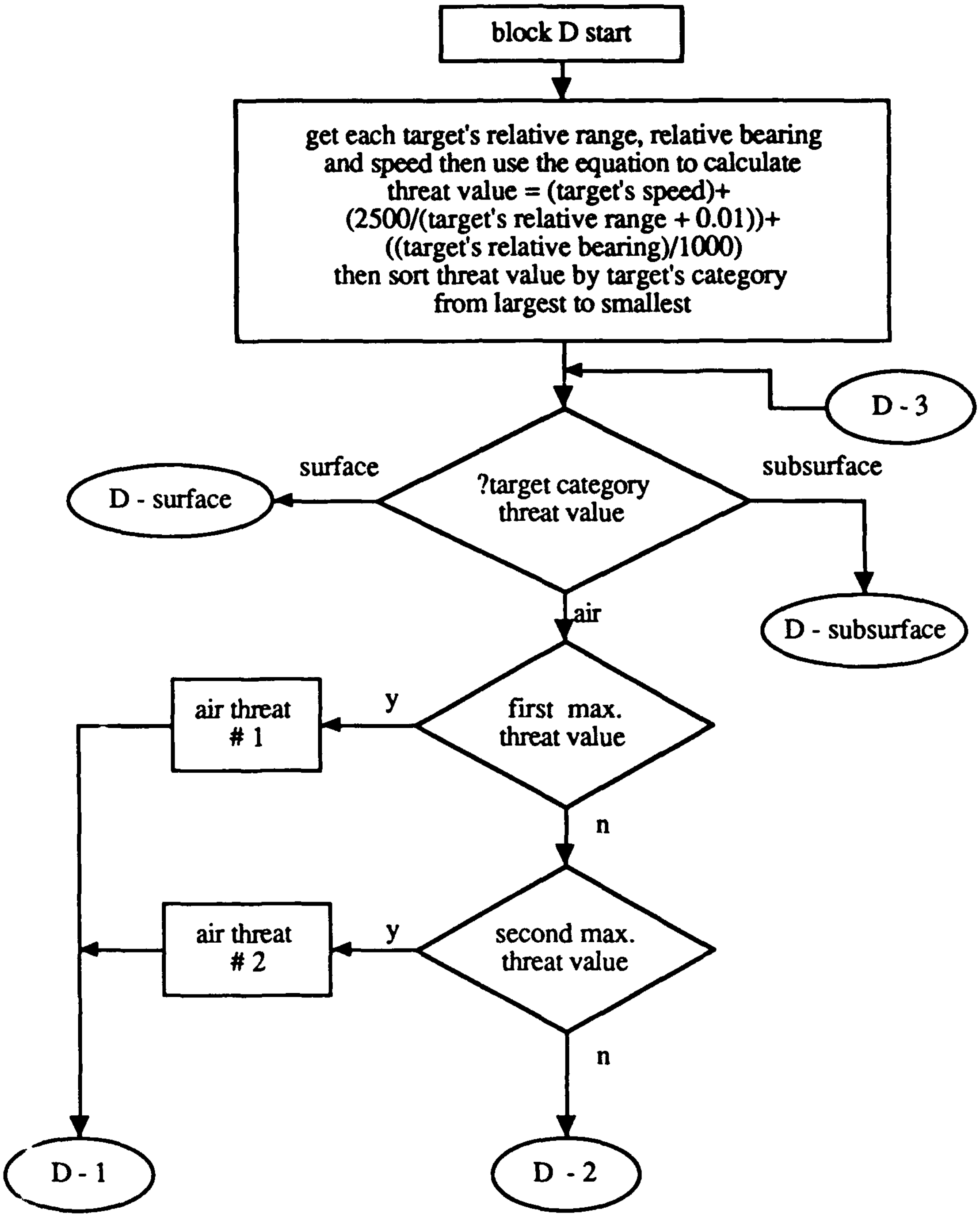

Figure A-6. Sort Air Target Threat Class Flow Chart

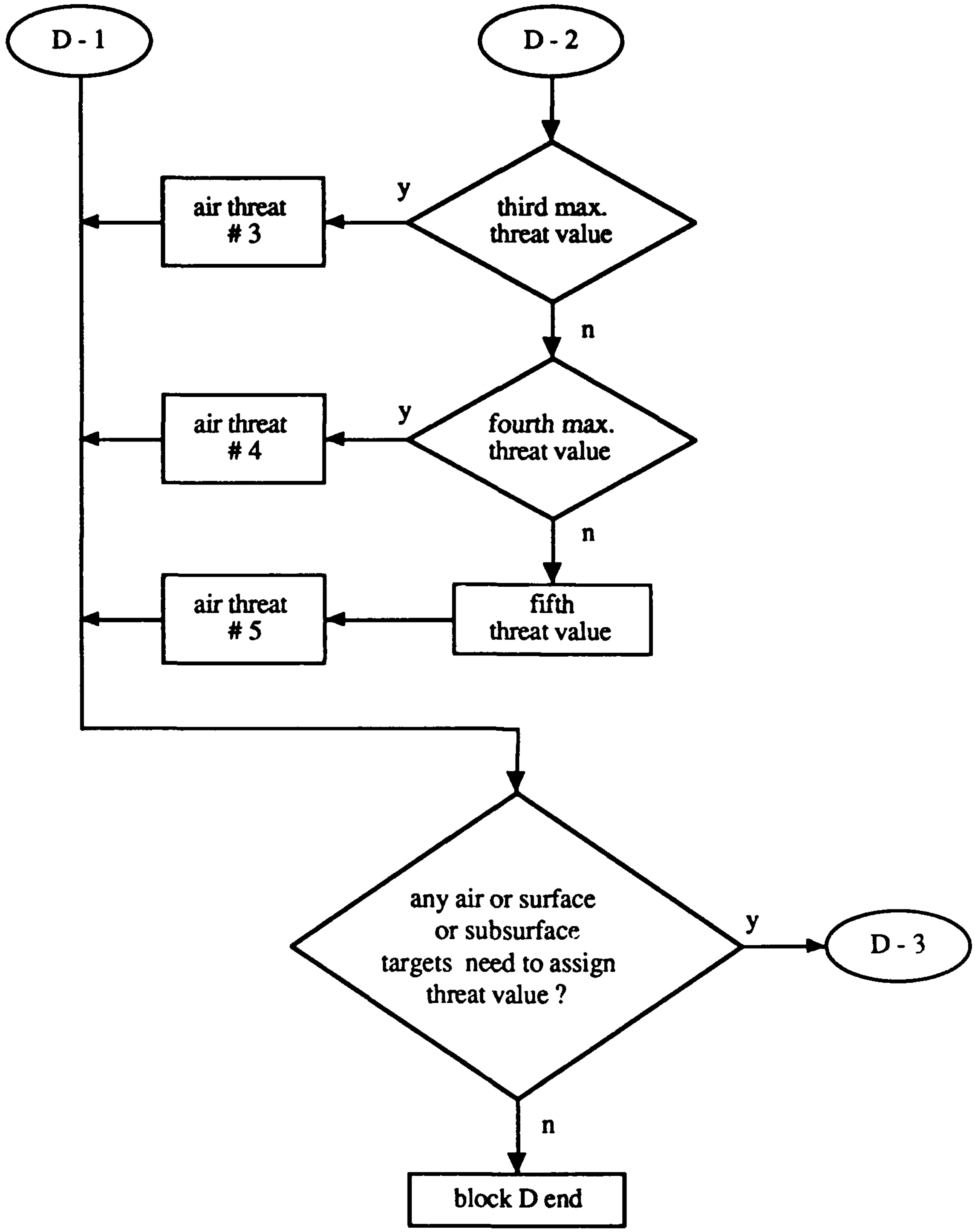

Figure A-7. Sort Air Target Threat Class Flow Chart (continued)

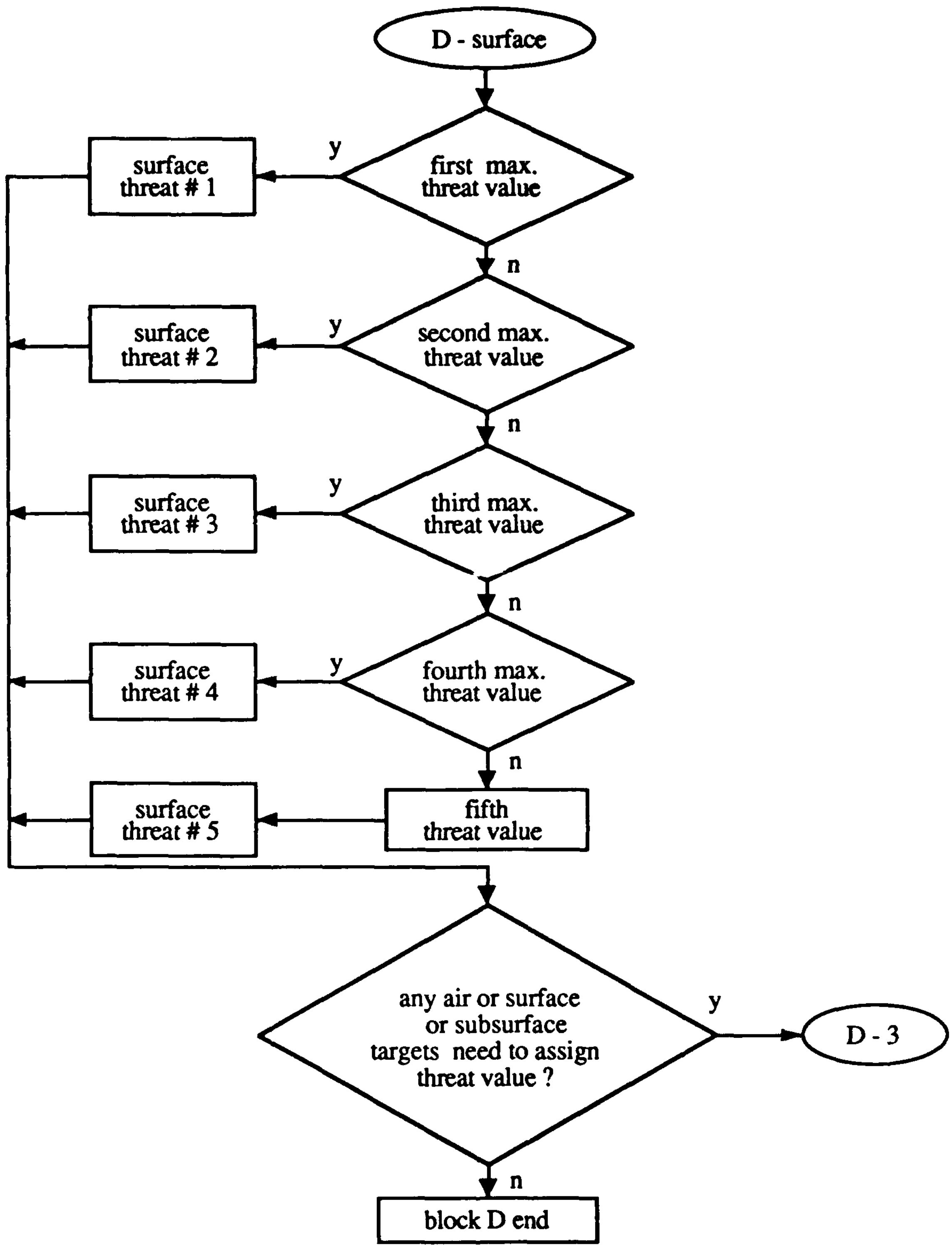

Figure A-8. Sort Surface Target Threat Class Flow Chart

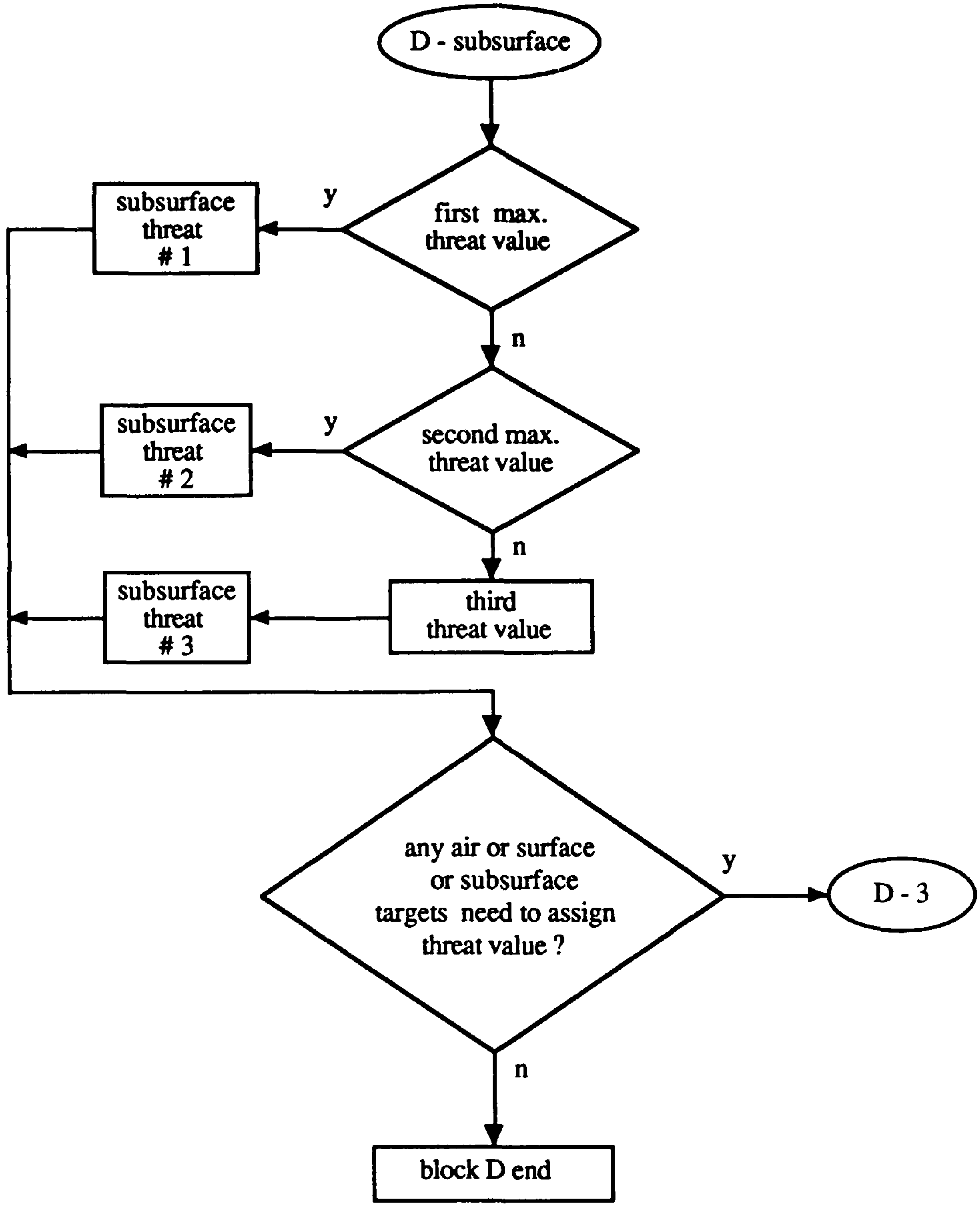

Figure A-9. Sort Subsurface Target Threat Class Flow Chart

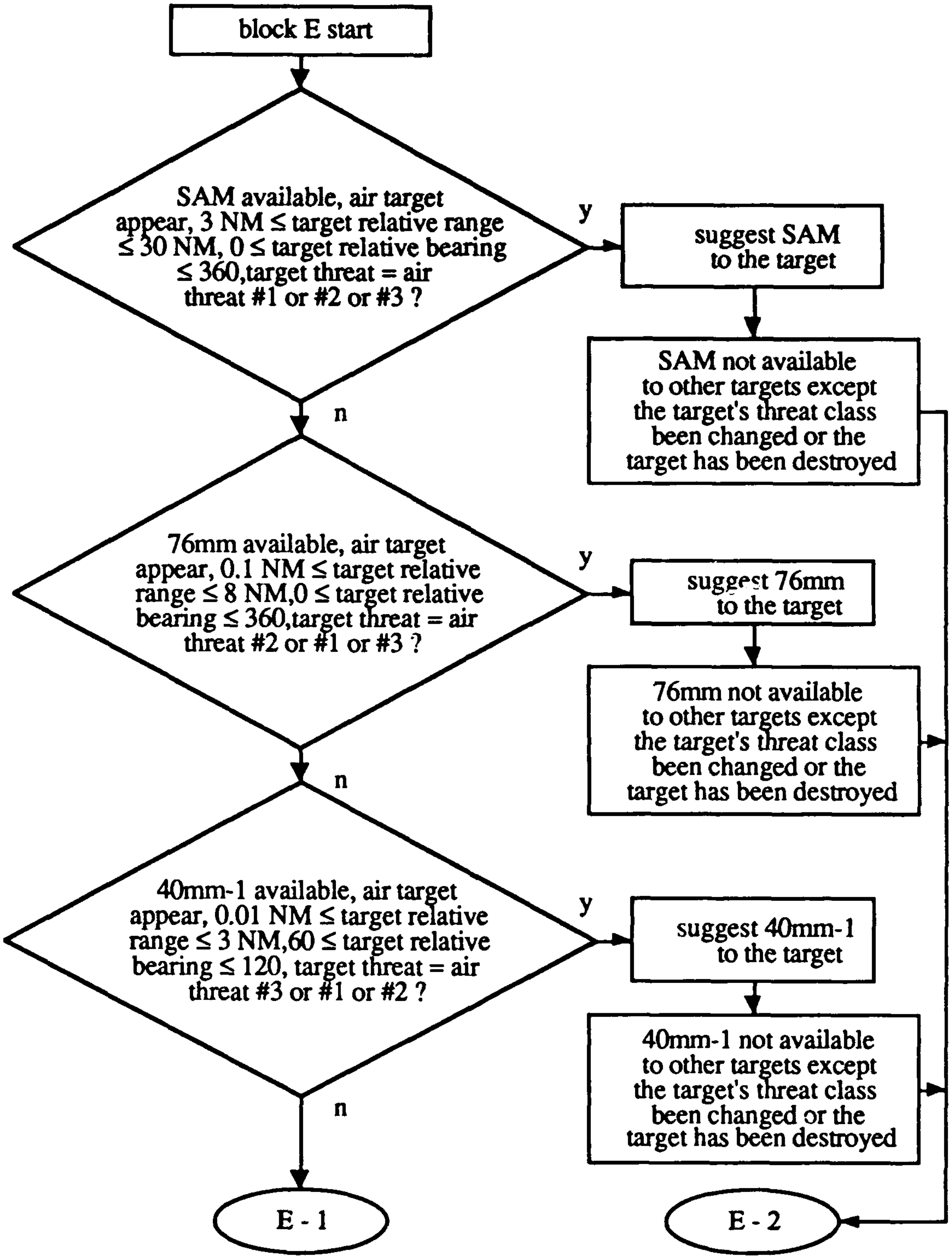

Figure A-10. Against Air Target Weapon Suggestion Flow Chart

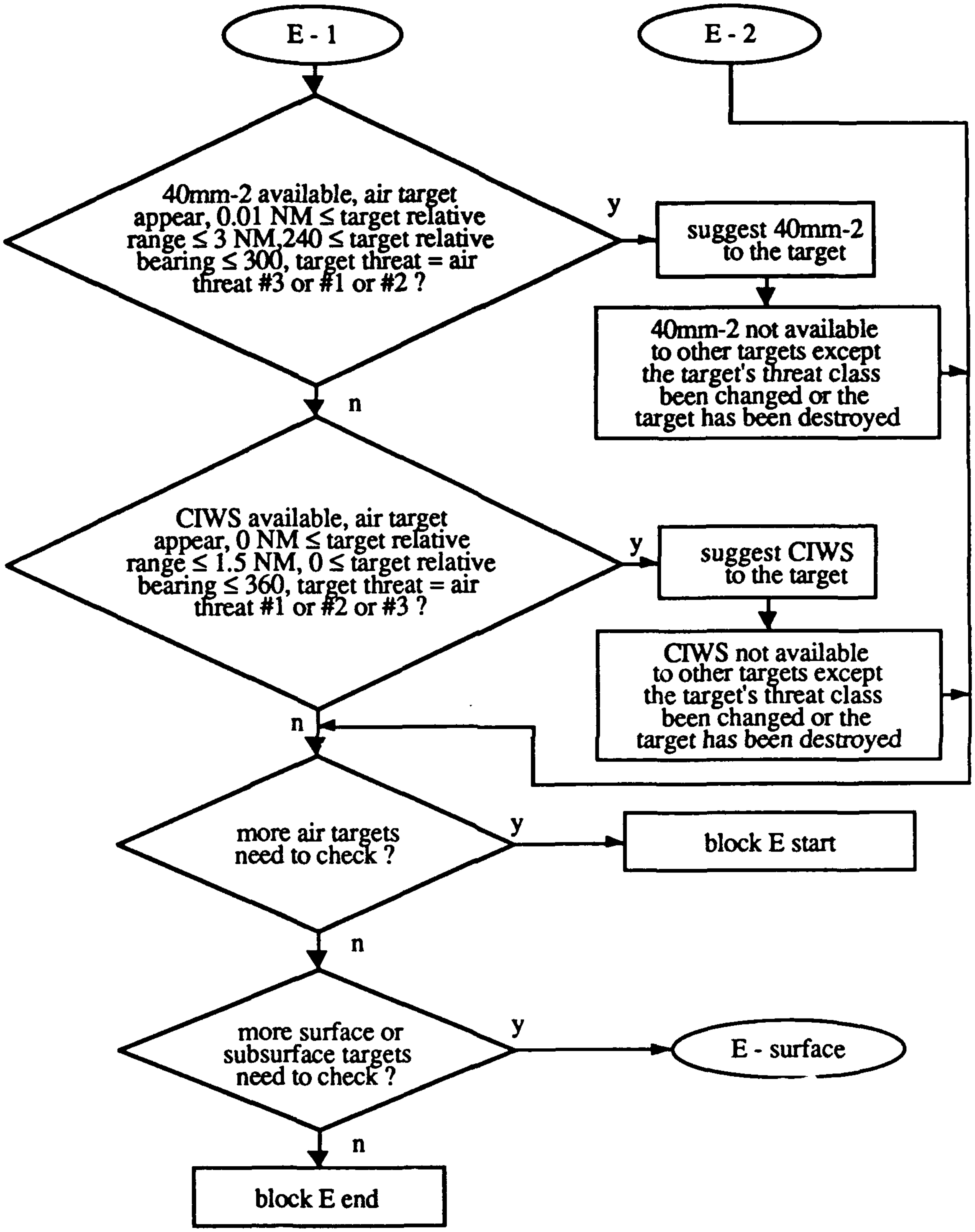

Figure A-11. Against Air Target Weapon Suggestion Flow Chart (continued)

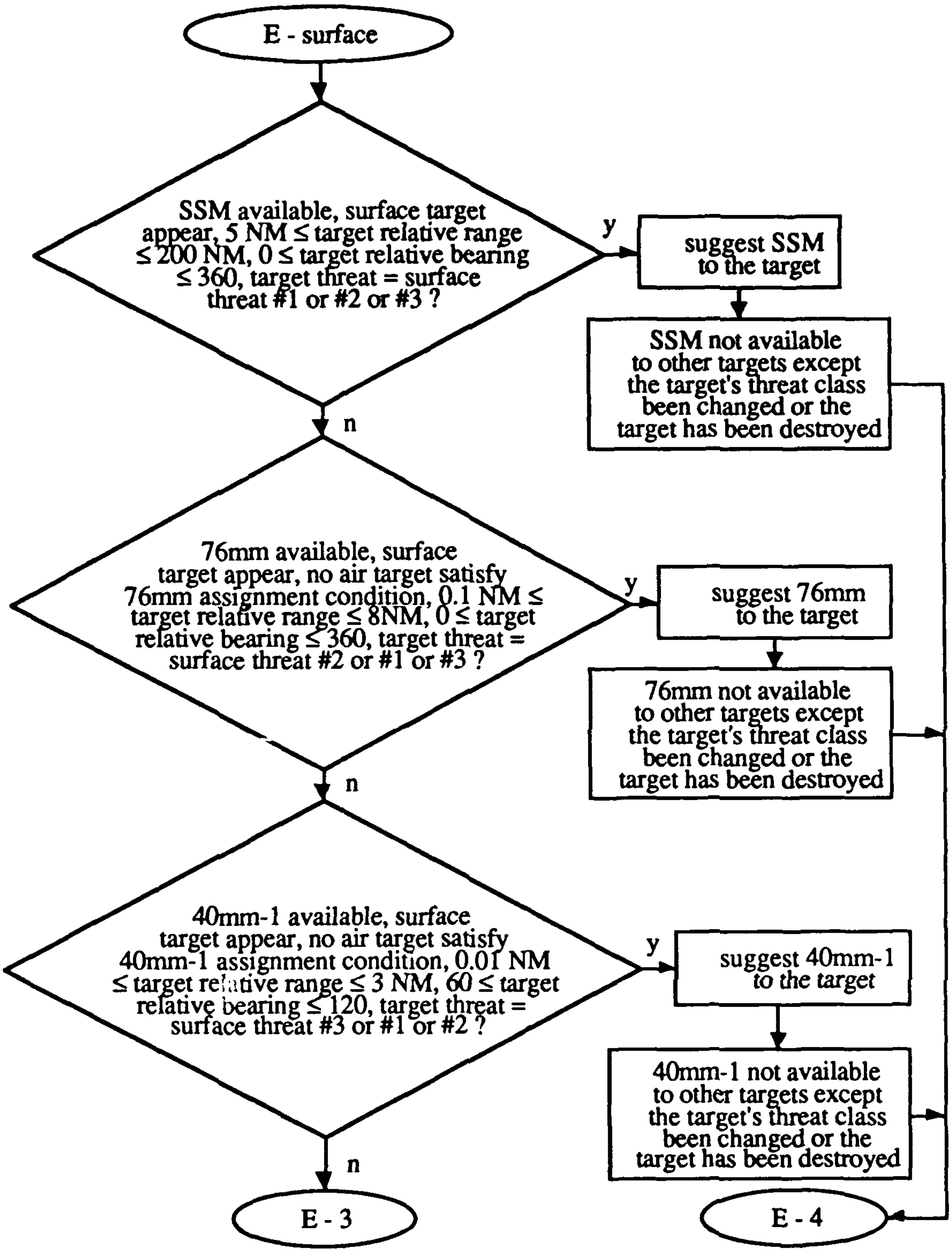

Figure A-12. Against Surface Target Weapon Suggestion Flow Chart

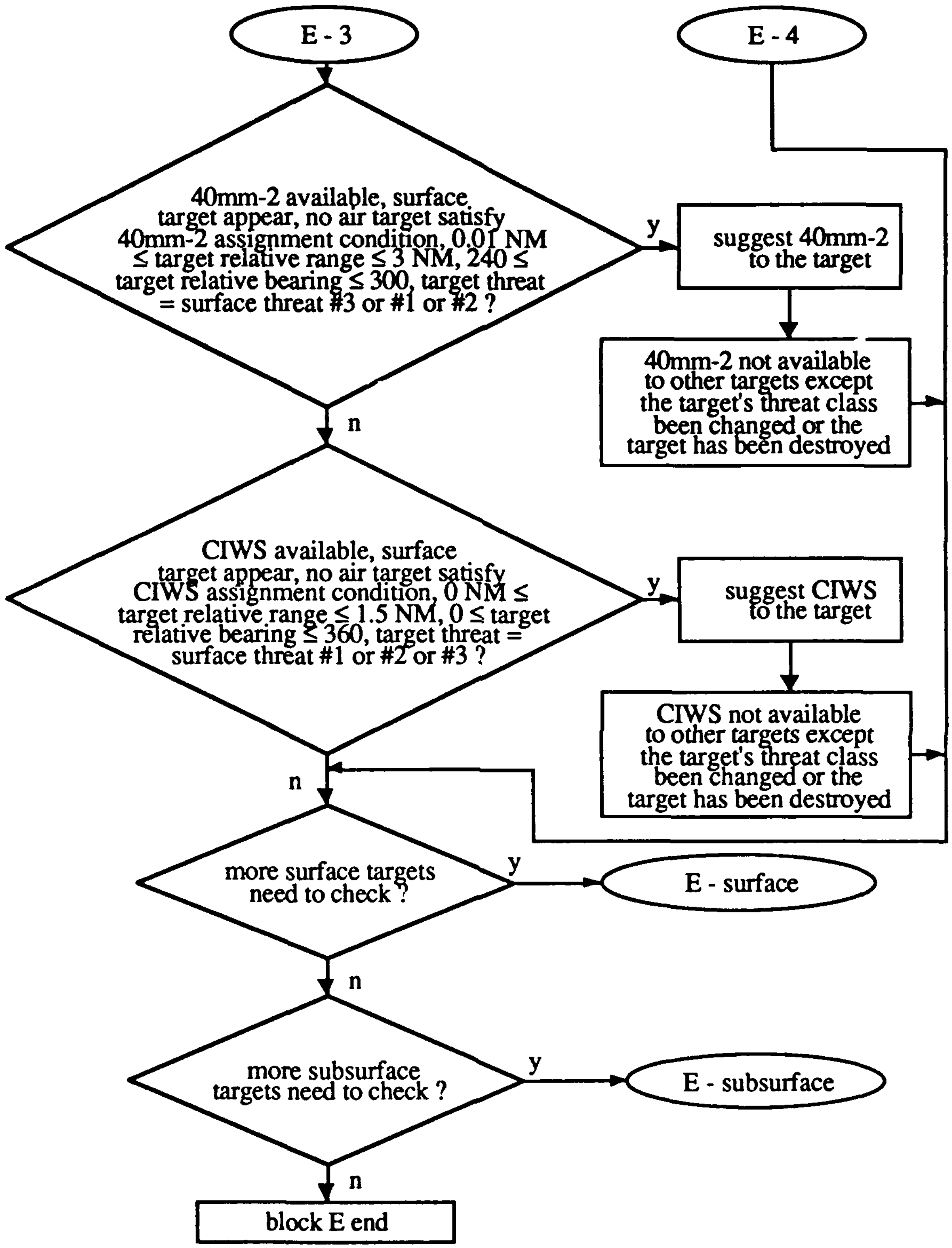

Figure A-13. Against Surface Target Weapon Suggestion Flow Chart (continued)

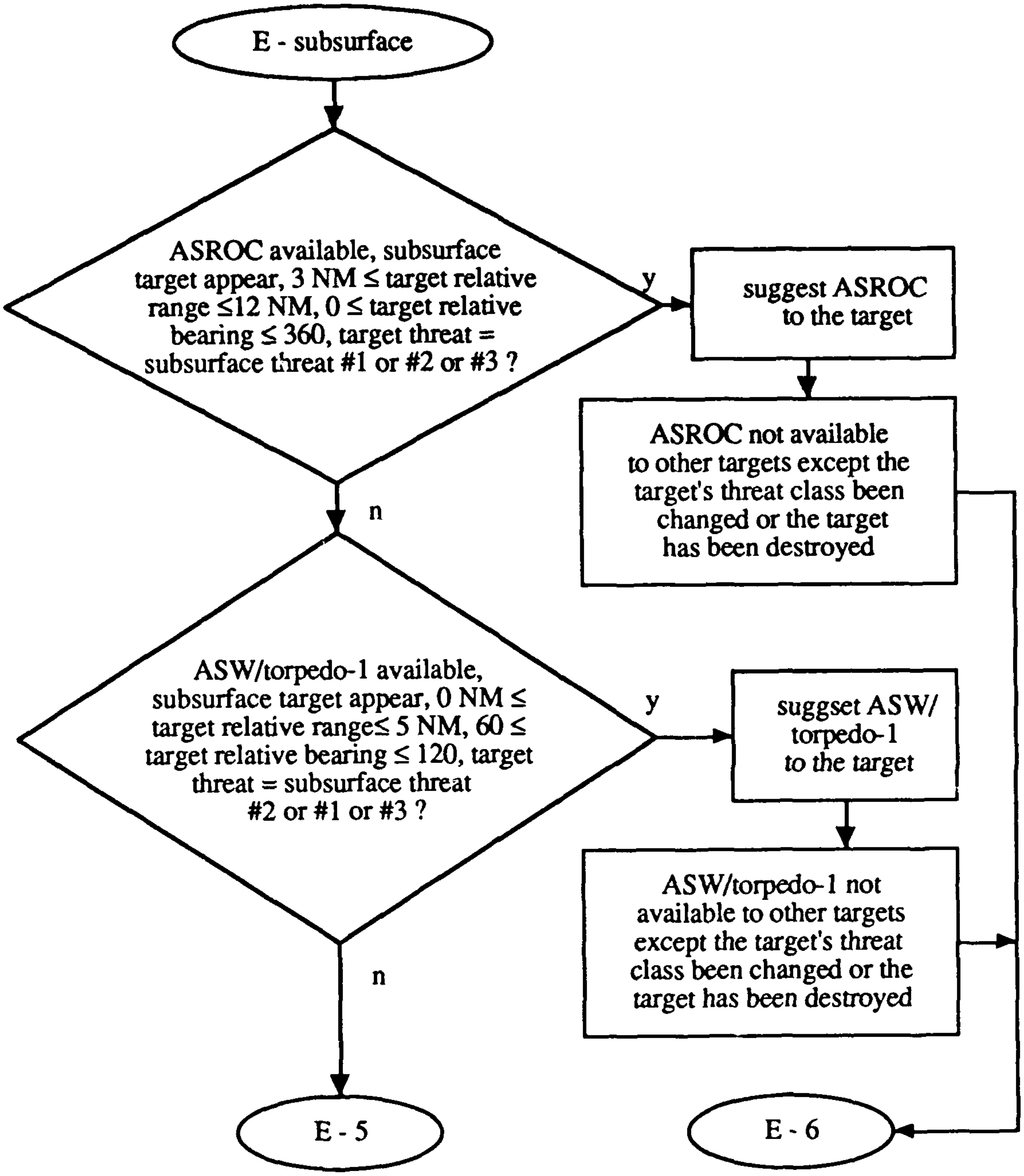

Figure A-14. Against Subsurface Target Weapon Suggestion Flow Chart

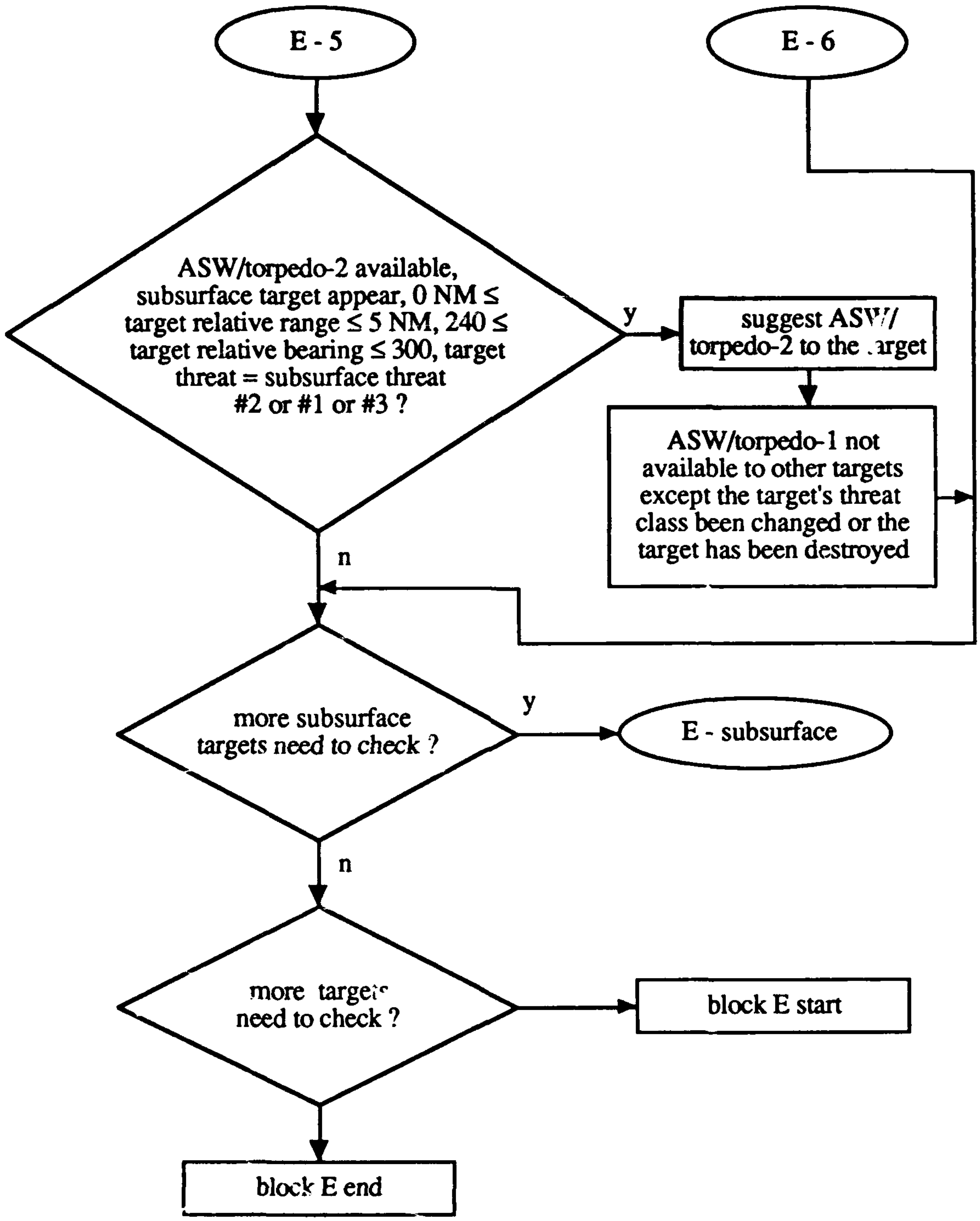

Figure A-15. Against Subsurface Target Weapon Suggestion Flow Chart (continued)

LIST OF REFERENCES

Barr, A., Choen, P. R., and Feigenbaum, E. A., *The Handbook of Artificial Intelligence*, v. 4, Addison-Wesley Publishing Company, 1989.

Bullock, R. A., Geschke, M. J., and Widmaier, L. E., *TAC* II An Expert Knowledge Based System for Tactical Decision Making*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1983.

Clair, W. C., and Danhof R. F., *Adaptation of a Knowledge-Based Decision-Support System in the Tactical Environment*, Master's Thesis, Naval Postgraduate School, Monterey, California, December 1981.

Common LISP, Addison-Wesley Publishing Company, Inc., 1988.

Davis, R., "Expert Systems: Where are we? And where do we go from here?," *AI Magazine*, 1982.

Davis, R., *Human Factors in Automated and Robotic Space Systems*, National Research Council, 1987.

Davis, R., "Expert Systems: How far can they go?," *AI Magazine*, 1989.

Dreyfus, H., and Dreyfus, S., "Why Expert Systems do not Exhibit Expertise," *IEEE Expert*, 1986.

Feigenbaum, E. A., McCorduck, P., and Nill, H. P., *The Rise of Expert System*, New York Times Books, 1989.

Gevarter, W. B., *Expert Systems: Limited but Powerful*, Petrocelli Books, Inc., 1985.

Horn, B. K. P., and Winston, P. H., *LISP, 3rd ed.*, Addison-Wesley Publishing Company, 1989.

Jennings, K. E., *Design Guidelines for a Rule-Based Passive Surveillance System*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1986.

KEE ActiveImages Reference Manual, IntellCorp, Inc., 1989.

KEE Interface Reference Manual, IntellCorp, Inc., 1987.

KEE TellAndAsk Reference Manual, IntellCorp, Inc., 1989.

KEE User's Guide, IntellCorp, Inc.,1988.

Lenat, D. B., Prakash, M., and Shepherd, M. "CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks," *AI Magazine,* 1986.

Lenat, D. B., and others, *Building Expert Systems,* Addison-Wesley Publishing Company, 1983

Lehner, P. E., *Artificial Intelligence and National Defense,* TAB Books, Inc., 1989.

MaCarthy, J., *Some Expert Systems need Common Sense,* paper presented in New York Academy of Sciences Science Week Symposium in Computer Culture, 5-8 April 1983.

Masica, J. M., and Buscemi, T., *A Prototype Production Rule Program for a Decision Support System,* Master's Thesis, Naval Postgraduate School, Monterey, California, June 1979.

Michalki, R. S., and others, "The Multipurpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains," *Proc. AAAI-86,* 1986.

Michie, D., and others, *The First Conference on Artificial Intelligence,* IEEE Computer Society Press, 1984.

Mitchell, J. M., Carbonell, J. G., and Michalski, R. S., *Machine Learning: A Guide to Current Research,* Kluwer Academic Publications, 1986.

Nilsson, N. J., "Two Heads are Better than One," *SIGART Newsletter,* 1982.

Ragsdale, D. J., and Tidd, J. P., *Designing Intelligent Computer Aided Instruction Systems with Integrated Knowledge Representation Schemes,* Master's Thesis, Naval Postgraduate School, Monterey, California, June 1990.

Schank, R. C., "The Current State of AI: One Man's Opinion," *AI Magazine,* 1983.

Touretzky, David S., *Common Lisp: A Gentle Introduction to Symbolic Computation,* The Benjamin/Cummings Publishing Company, Inc., 1990.

Wang, Ming-Hua, *A Rule-Based System for Shipboard Air Defense,* Master's Thesis, Naval Postgraduate School, Monterey, California, December 1989.

Wilensky, R., *Common LISPcraft,* W. W. Norton & Company, Inc., 1986.

Winograd, T., and Flores, F. F., *Understanding Computers and Cognition,* Ablex, 1986.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Professor Yuh-jeng Lee, Code CS/Le Department of Computer Science Naval Postgraduate School Monterey, California 93943	9
4.	Commander B. B. Giannotti, Code CS/Gg Department of Computer Science Naval Postgraduate School Monterey, California 93943	1
5.	Wen-I Weng 83, Lane Shih Wi, Fu Ch'un RD., Fu Liao Li Qs'ao T'un, Taiwan, Republic of China	1